Cryptex: Secure Local Chat Application Using Hybrid Encryption

A Development Project for Information Security

Najam Ul Islam Saeed 22L-7497 BS(DS)
Annas Ali 22L-7480 BS(DS)
Shahmir Iqbal 22L-7471 BS(DS)

Supervisor: Mam Nosheen Manzoor

Department of Computer Science, FAST-NUCES Lahore

December 2025

Abstract

Cryptex is a secure local chat application designed to demonstrate the principles of safe and reliable communication in modern computing environments. The project explores how confidentiality, integrity, and authentication can be ensured even in a local network setting. By combining strong cryptographic techniques with a user-friendly interface, Cryptex provides a practical example of how security concepts can be applied in real-world applications. The system emphasizes privacy, trust, and robustness, and a functional platform for understanding secure messaging practices.

1 Introduction

Secure communication has become essential across modern computing environments. Whether in personal messaging, enterprise systems, or cloud platforms, ensuring confidentiality and integrity is a fundamental requirement. This project, Cryptex, implements a prototype encrypted communication tool designed specifically for secure messaging over a local network (LAN).

The objective of Cryptex is to design a system that demonstrates real-world security mechanisms while remaining lightweight and educational. The system employs well-established cryptographic primitives such as RSA, AES, and HMAC, while using a zero-knowledge server that never accesses plaintext messages. With its graphical interface and modular architecture, the project offers a complete demonstration of both information security concepts and practical software engineering.

2 Motivation

Unsecured communication, especially on local networks, leaves users vulnerable to various attacks such as packet sniffing, replay attacks, message tampering, and impersonation. Tools like Wireshark or ARP spoofing malware can easily intercept unencrypted LAN traffic.Cryptex was motivated by the need to create a compact, demonstrable project applying information security concepts taught in class:

- Understanding and applying symmetric and asymmetric cryptography
- Ensuring message integrity using HMAC
- Demonstrating confidentiality through AES encryption
- Implementing a zero-knowledge server model
- Implementing audit logging for forensic value
- Practical exposure to secure system design

The project also addresses the gap in many student submissions that often skip message integrity checks and server-level security controls.

3 System Methodology

Cryptex follows a modular design consisting of four main components:

1. **Client Application (Tkinter GUI)**
2. **Server Application (Zero-knowledge Router)**
3. **Cryptography Engine (RSA + AES + HMAC)**
4. **Audit Logging Module**

Each module is described below.

3.1 Client Architecture

3.1.1 Graphical User Interface (Tkinter)

The Cryptex client uses Python's Tkinter library to create a WhatsApp-inspired interface. Features include:

- Message display panel
- Input field
- Status indicators
- Online users section
- Broadcast and private message option

Tkinter was chosen due to its minimal resource requirements and cross-platform support.

3.2 Server Architecture

3.2.1 Zero-Knowledge Router

The server in Cryptex is designed as a "blind message router":

- It does **not** decrypt messages.
- It does **not** store plaintext.
- It forwards encrypted payloads to connected clients.

This fulfills the zero-knowledge security principle: the server cannot leak what it cannot read.

3.3 Cryptographic Methods

3.3.1 Hybrid Encryption: RSA-2048 + AES-256

Cryptex uses a hybrid cryptographic approach:

- RSA-2048: Authenticates client identity via public key exchange.
- AES-256-CBC: Encrypts all chat messages with a shared session key

This mirrors the secure key bootstrap techniques used in modern messaging systems.

3.4 Security Mechanisms

3.4.1 Implemented Mechanisms

1. AES-256-CBC Encryption Each message is encrypted using a shared AES key and a **random 16-byte IV**. This ensures:

- identical plaintext results in different ciphertext
- prevention of pattern analysis

2. HMAC-SHA256 Authenticated Encryption An HMAC is generated for every encrypted message. This provides:

- integrity protection
- tamper detection
- replay attack mitigation

3. RSA-2048 Public Key Authentication Verifies client identity and establishes trust.

4. Security Audit Logging Cryptex maintains an append-only security log:

- timestamps
- cryptographic events
- key exchange entries
- detected tampering attempts
- client join/leave events

5. Zero-Knowledge Server All encryption and decryption happen client-side.

4 Security Analysis

4.1 Confidentiality

AES-256 with random IVs guarantees strong confidentiality. Even identical messages produce different ciphertext.

4.2 Integrity

HMAC-SHA256 ensures that:

- messages are untampered
- corrupted packets are rejected
- MITM modifications fail

4.3 Authentication

RSA public-key authentication verifies client identity. The shared AES key is distributed to authorized clients in the trusted LAN environment. Each message uses a random IV to ensure unique ciphertext even with a shared key.

4.4 Replay Attack Prevention

Cryptex detects and blocks replays because:

- IV is random for each message
- HMAC changes on every packet

4.5 Message Security Summary

Cryptex uses:

- AES-256-CBC encryption
- Random IV per message
- HMAC-SHA256 authentication

Future improvements may include:

- ephemeral AES keys
- PFS via automatic key rotation

5 Conclusion

Cryptex demonstrates strong applied information security principles in a functional LAN messaging system. Using hybrid RSA-AES encryption, HMAC integrity protection, and a zero-knowledge server, the system ensures confidentiality and integrity of communication. While simplified for academic demonstration, the design supports future enhancements such as ephemeral per-session keys and Perfect Forward Secrecy. Cryptex shows that secure communication can be both practical and educational.

6 References

1. William Stallings, *Cryptography and Network Security*, 8th Edition.
2. Bruce Schneier, *Applied Cryptography*.
3. RFC 2104 - HMAC: Keyed-Hashing for Message Authentication.
4. NIST SP 800-38A: Recommendation for Block Cipher Modes.
5. OWASP Cryptographic Storage Guidelines.