Secure WebSocket Chat Application

Author: Curtis Quan-Tran
Date: 3/23/2025

GitHub Link

---

1. Introduction

CurtisConnect is a secure, real-time chat application designed to provide encrypted and reliable communication over the Internet. This application ensures user privacy and data integrity by exclusively supporting secure WebSocket (**WSS**) connections while actively blocking any insecure (**ws://**) traffic. Utilizing robust encryption protocols, including AES-256-GCM for message encryption and RSA-4096 with OAEP for secure key exchanges, CurtisConnect protects messages and files both in transit and at rest. Coupled with secure authentication, intuitive interfaces, and enhanced security features,

Technology Used

- **Frontend**: JavaScript, Markdown Support, Emoji Mart
- **Backend**: Python (Flask, Flask-SocketIO, Flask-Sessions, Gevent, Gevent-Websocket)
- **Security**:
    - SSL: Let's Encrypt (Certbot)
    - Encryption: AES-256-GCM, RSA-4096 (OAEP)
    - Password hashing: bcrypt
- **Database**: JSON-based user storage

---

2. Changes from Previous Version

- **Removed Excessive SSL Warnings:** Suppressed unnecessary SSL error logs during HTTP connection attempts.
- **Changed SSL certificate:** Replaced previous self-signed SSL certificate with a trusted Let’s Encrypt SSL certificate using CertBot.

- **Changed Localhost Deployment:** Transitioned from running locally to hosting on a Raspberry Pi server accessible via the domain **curtisqt.com**
- **Changed Password Hashing:** upgraded to a more secure bcrypt hashing with salting.
- **Changed session timeouts:** changed from 40 seconds to 30 minutes
- **Removed Warning Notifications:** Remove unnecessary inactivity warning notifications..
- **Changed message Limits:** changes from 50 characters to 150 characters.

---

3. User Guide

Running the Application

Open your default web browser and enter the URL.

```
https://curtisqt.com/
```

Application Usage Guide

Register/Login

- **New users:** Click **register**, and enter a username & password.
- **Existing users:** Log in with your **credentials**.

Join the Chat

- Send encrypted real-time messages.
- Message Limit: 150 characters.
- Rate limit: 1 message per second.

Room Management

- **Sidebar** to manage chatrooms.
- **Create or join** rooms easily using unique Room IDs.

File Sharing

- **Supported Formats:** TXT, PDF, PNG, JPG, JPEG, GIF

- **Maximum Size:** 8 MB
- Files are securely **encrypted** using AES-256-GCM in both transit and at rest.

Text & Emoji

- **Markdown support** for rich text formatting.
- Integrated **unicode emoji picker** (Emoji Mart).

Logout

- Click **“Log Out”** to log out and get redirected to the log-in page.
- Alternatively, exit out the tab/browse

---

4. Key Security Features

Encryption

- **AES-256-GCM:** Data encryption for messages and files both in transit and at rest. Secure IV generation and base64 encoded data.
- **RSA-4096 with OAEP**: For AES key exchange.

Authentication & Authorization

- Secure **bcrypt** hashing (with salt) for storing user credentials.
- Server-side authentication only.

Brute Force Protection

- **IP address** blocked for 5 minutes after 3 failed login attempts.
- Logging failed attempts with timestamps and IP address.

Logging System

- Secure, clear **chat logging** per chat session.
- Includes client IP, timestamps, and enhanced formatting for **readability**.

SSL Certificates

- Trusted certificates via **Let's Encrypt (Certbot)** replacing previous self-signed certificates.

---

5. Hosting & Infrastructure

Server Setup

- Hosted on **Raspberry Pi** Web Server.
- Domain: **curtisqt.com** via **Cloudflare.**

Cloudflare Security

- **Geo-blocking** high-risk regions.
- **Bot Protection**: Enabled for malicious bots.
- **Security challenges** for suspicious traffic.

SSH Security Hardening

- SSH on custom port **2222**.
- Disabled root password login.
- **SSH key authentication** enforced.

Firewall & Security Tools

- UFW: Strict incoming/outgoing rules.
- Automatic Security Updates.
- **Fail2ban**: Preventing brute force attacks.
- **Rkhunter**: Detecting rootkits and system threats.

---

6. Key Features Summary

- User-friendly GUI replacing terminal-based interactions.
- Secure, encrypted file transfers.
- Robust text formatting and emoji support.

- Advanced security enhancements and encryption.
- Effective session and room management.

---

7. Future Improvements

- User input validation and sanitization.
- Better activity monitoring.
- Scalability with chat room limits.
- Room passwords.
- Improvements to the front-end interface.
- Add a script or method for server administrators to decrypt the encrypted chat logs.

---

8. AI Assistance & Contributions

- **Frontend Development:** The dashboard.html layout was created with the help of AI, providing an understanding of how layout and styling with CSS works as well as identifying bugs in my client.js file.
- **Backend Development:** AI was used primarily to identify issues with my code, particularly with key management and session management.
- **Security:** AI provided suggestions to improve the quality of security mechanisms such as using OAEP for RSA encryption as well as advice on how to harden my raspberry Pi 5 effectively, to prevent risk to my home network.
- **Documentation:** AI was used to help structure this document and wording to maintain a professional tone.