End-To-End Chat Application

DC Security
Donato Estolano & Connor Kobel
CECS 478

Project Overview

- Client side application is created using Python
- Server is composed of Node.js with NGINX as a reverse proxy
- MongoDB was used for the database
- Our website: https://www.cecs-478.me/

Security Goals

- Confidentiality
  - Use encryption and decryption, JSON
- Integrity
  - Data doesn’t change, key
- Authenticity
  - Making sure that the person sending/receiving is who they say they are, hashed password

Things to Consider

- Assets
  - The data, the actual message
- Stakeholders
  - Us, the users/creators
- Adversaries
  - Server
  - Outsider
  - Active
  - Passive

Attack Surfaces

- Most prevalent
  - Man-in-the-middle
  - Brute force
- Less prevalent
  - DDoS
  - Social Engineering
  - Shoulder surfing
  - Key Loggers

Project Design/Implementation

Messages

- Using Python’s cryptography library
- Encrypted using AES with appropriate padding
- Keys and IV are randomly generated every time
  - Keys are 32 bytes
  - IV is 16 bytes
- HMAC is utilized to generate a tag for the ciphertext
- Overall, using PGP (Pretty Good Privacy)

RSA Private/Public Keys

- RSA key pairs are generated using OpenSSL command line
- Public key exchange will take place offline through USB drives or if users decide to just email each other or use Google Drive

Client/Server

- Passwords are hashed with a randomly generated salt before being stored in the database
- Remote login
- JWT
- Python tkinter library for the GUI

Project Flow

- Registration/Login
- Token Reception
- User specifies who they want to talk to
- Chat is initialized
  - Message is sent
  - Message is received
- Fun ensues

Demo