Guilty PlAESures

Michael Pepsin

Milestone

4/13/20

1. Abstract

Guilty PlAESures is a web app that uses AES encryption to encrypt MP3 files and anonymously emails the ciphertext and decryption key to an input email. Once these data are fed back into the web app, the user receives the original file. This paper will expand on the motivation for this project, some similar available tools, as well as the methodology and a progress report of what is done and what is not.

2. Motivation

This is a serious paper, but my inspiration came from a song from Spongebob Squarepants. I stumbled across this song while letting my Spotify play some recommended tracks. When the song “Gary Come Home” came on, I actually enjoyed it quite a bit and was a little embarrassed to listen to it repeatedly several times in the days to come, even adding it to my main playlist. This project is a way to anonymously share music, requiring only an MP3 file of a song and the email of the intended recipient.

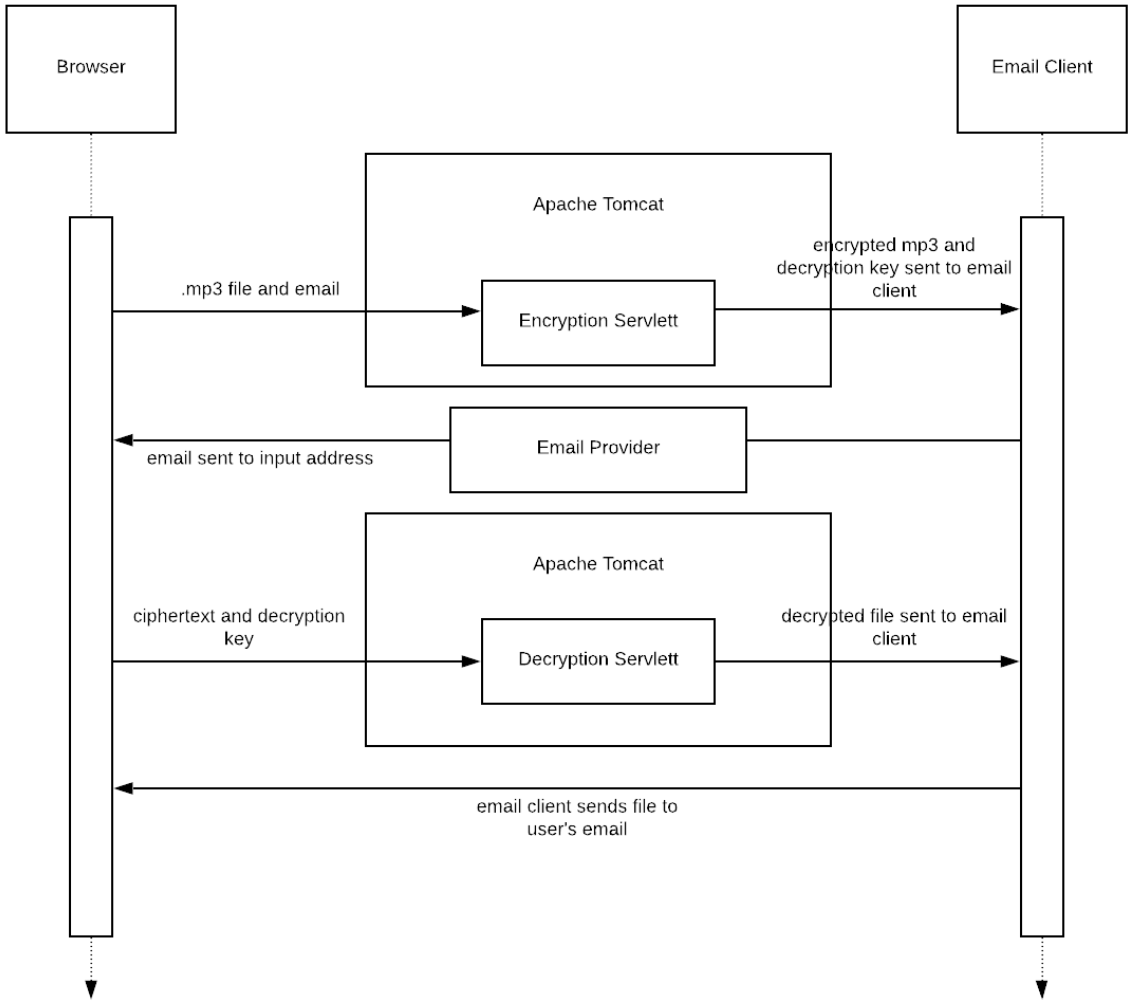
3. Related Work

A quick Google search of the phrase “mp3 encryption” rendered a result for MP3Crypt, a console mode application available on www.compuphase.com. This program is very old and uses H04x0 encryption, which is suited for audio. However, this means using a CD or USB to store the encryption keys and the MP3Crypt itself. There is also a GitHub repository that encrypts MP3 files and uses open ssl for decryption, but by its nature, the keys are stored client-side as plaintext, making it effectively useless against hackers. Guilty PlAESures is more oriented towards sharing music compared to MP3Crypt, and since the key it generates is unique to the MP3 file, it is more secure than the GitHub repo Encrypt/Decrypt.

4. Methodology and Progress

Guilty PlAESures uses AES encryption with keys of size 128. The web app is serviced by a Tomcat servlet, one for encrypting and one for decrypting. These servlets communicate with an email client to send information to users. A user of Guilty PlAESures supplies the recipient’s email and an MP3 file, sends it through the encryption servlet, which sends the decryption key

and ciphertext to the recipient's email. The recipient can then input these data into the app to be processed by the decryption servlet, which then provides the file encrypted by the first user.



Currently, the Tomcat servlets exist and are connected to the HTML page for the web app. The email client and encrypt/decrypt functionality are not yet implemented. I will continue, first by sending simple emails through the Tomcat servlet when the encrypt form is submitted, and then the encrypt and decrypt functions will follow once basic functionality and communications between clients is properly working.

Sources:

https://www.compuphase.com/mp3/mp3crypt.htm

https://gist.github.com/mmcc/5862775