Steganosa

Alexis Allen (laetaku@cs.washington.edu)
Zhengyang Gao (gaoz6@cs.washington.edu)

Problem:
Security has become increasingly important in today's world; often times, people need a secure way of sending encrypted messages without the encrypted message drawing too much attention itself.

Vision:
For those keeping an eye out on network traffic, an encrypted file can attract interest into both the sender and recipient -- an alternative is steganography, which is the practice of concealing a message/file within another message/file. However, most steganography applications on the market today, such as *Steganographia* and *Hide It In*, use password-based security to control access to the messages/files in the image. Passwords have repeatedly been proven to not be secure enough for everything. Therefore, our vision is to provide an easy, fun, and secure way for the more security-minded people to encrypt messages and files in a way that does not draw as much attention.
Two of the most important differentiators of this application is that it uses asymmetric encryption and the image containing the encrypted message/file is only intended for one recipient, thus increasing its security level.
In regards to scope, we try to make this project reasonable for 10 weeks by focusing on the security and embedding of the information.

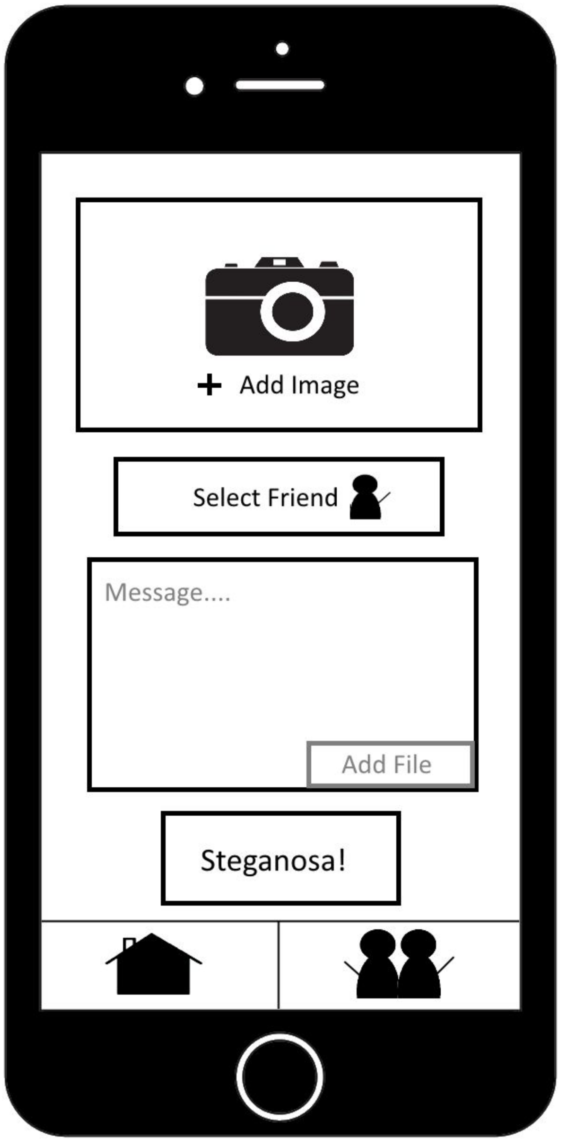
Software Architecture:
We expect that there is going to be three major modules that we need to implement in order to develop this application:

1). The asymmetrical encryption/decryption process of the message/file
2). Embed the encrypted message/file into the image
3). Construct a user and relationship database

As asymmetrical encryption requires the recipient's public key, the application would offer a friends list to supply this value for the user. This application would encrypt the message asymmetrically, pulling the recipient's public key from the user database or using an entered key, and embed this into the user-supplied image. This encrypted message/file can only be decrypted by the one specified recipient who holds the private key.
We intend to use MySQL for the database, a known, trusted algorithm for the asymmetrical encryption, and likely Java (Android) for the steganography section. The language can also be

changed from Java to Swift (iOS), or a multi-platform option like Unity, depending on the group's phone preferences. The app could also offer a nice learning experience for those new to the phone app development scene. Learning to implement steganography can also be exciting.

Challenges & Risks:
Embedding encrypted messages/files in images of different sizes, dealing with MySQL-specific syntax, and communicating correctly with the database can challenge the group in finishing on schedule. To minimize risks, we plan to use the more common and documented libraries/apis and try to avoid languages that no group members have experience with. As a side note, a more well-rounded application could have additional features but we felt these would expand the project beyond the allotted time period.


+ Add Image
Select Friend
Message....
Add File
Steganosa!