CPSC-352 Final Project Report
Secure Internet Poker
Group Members: Gregory Martinez, Troy Lee

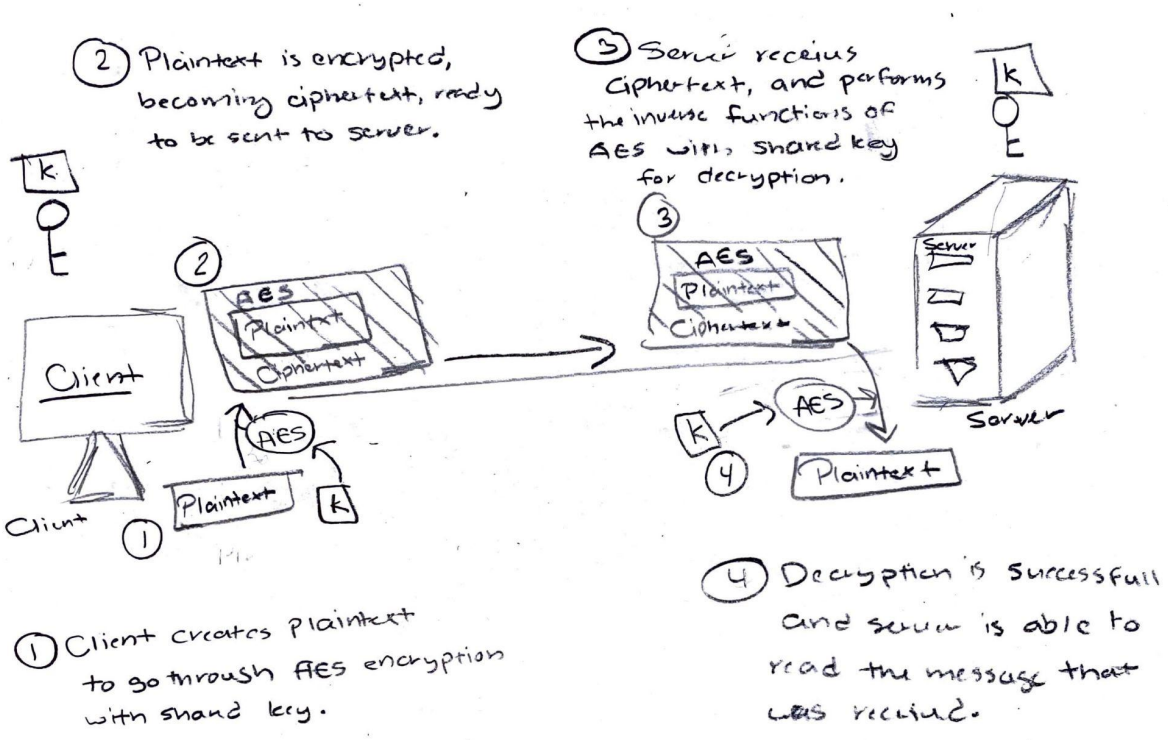
Abstract

This project is intended to allow two players to participate in a poker game in a server-client secure environment with the option of RSA or DSA techniques for message transfer and receiving with the server. Due to trial and error, we have opted to create a simple AES encryption implementation between a client and server socket connection.

Introduction

The intended design of this project would allow a client to achieve a secure connection with a server and retrieve a response in accordance with the input. Unfortunately, we were not successful with a full implementation of this design. We were able to achieve a simple client/server connection in which AES encryption and decryption is utilized through a shared symmetric key. Through this design, we were able to create confidentiality to an extent with the AES method of message cryptography with a shared key between client and server.

Design



Security Protocols

The Client utilizes 128-bit AES keys to provide confidentiality. The Server also utilizes the 128-bit key for decryption. AES is a secure encryption algorithm that provides confidentiality in its practice. The conceptual use of confusion and diffusion allows us to minimize cryptanalysis between key and ciphertext, and plaintext to ciphertext. The user provides a text or phrase that

is utilized as a 16 byte key for AES. In order to mitigate cryptanalysis, the plaintext that the client provides to send to the server is encoded into binary, then padded to be uniformed at 16 bytes. Using Cryptodome Cipher library in python, we are able to utilize AES to create an encryption cipher to encrypt this padded plaintext in byte form. In an inverse method, the server is able to utilize the shared key to create its own AES cipher to decrypt the message and retrieve the padded plaintext. It will then unpad the text to be to decipher the message and see the client message. Upon research and references, we noticed a great design of the confidentiality and authentication of RSA and DSA methods. Asymmetric keys were utilized to generate digital signatures and hashed authentications.

Implementation

A python environment was used for this project as there are libraries such as Cryptodome and Sockets, that allow us to easily use their built-in functions to create confidentiality in our design. Cryptodome allows us to use AES ciphers with a shared key for symmetric encryption and decryption. Python's socket library allows us to easily create a client and server topology for exchanging messages and securely closing them once communication is complete and to allow the server to continuously listen for up to 100 clients until the server is shut down.

Conclusion

With our knowledge of cryptography, we are able to design and implement a secure simplex connection between client and server. The connection only utilizes AES for message encryption, so the security of the design is not fully developed. With more time, we would have hoped to implement an asymmetric use of keys for RSA and DSA methods. The logic of the code also needs to be heightened to mimic the communication of a desired ATM and Bank server. These implementations were difficult to perfect, thus, for efficiency of this project, we decided to submit an AES message encryption design between client and server. This github project that we found while searching for references and implementation aid helped us understand in greater depth of how these methods are used and the creativity that cryptography can be. Greater security for this implementation would be the introduction of nonces for replay attack prevention, RSA and DSA methods for digital signature and hash functions for authenticity.