Will Wainscott

Software Development 1

Professor Arias

5/4/18

Public and Private Key Encryption

Abstract

My project is a system to encrypt and decrypt messages sent via text files. I am using the RSA public and private key cryptosystem to encrypt the text inside of an external text file. With my program, you generate your own public and private key pair, encrypt the message within a text file using the public key, and then decrypt the encrypted message using the private key. While it is possible to do on just one computer, the keys and messages are all saved to files which can be sent to any computer. This allows for someone to send a encrypted message to someone who has the private key without fear of anyone else being able to decrypt the message. My program implements the different methods in the Java security libraries to create a program that encrypts securely and accurately.

Introduction

I have always been interested in how both big companies and smaller tech businesses keep their information safe when sending sensitive data. Upon further exploration, I found that the most common way is the Public and Private key encryption method. This method consists of a user creating two associated keys, and then giving out the public one such that others can

encrypt their data with it, and the private key is kept secret to ensure only the person who created the keys can decrypt the data sent. I have used this method in a very simple way, but the concepts used can be applied to much bigger projects. I also think that using some of the Java security libraries has helped me to understand what different security measures Java has available to programmers.

Detailed System Description

My system takes the user through the whole process of the creation of the public and private keys, the encryption, and the decryption of the message. My first class, GenerateKeys, just creates the keys and puts them into a folder on the computer. While the keys are actually objects created by implementing different Java security methods, my system saves them to external files that can be moved around or sent to other people. The specific encryption technique I use is the RSA algorithm. This algorithm stands for the names of the algorithm's creators, Ron Rivest, Adi Shamir, and Leonard Adleman. This algorithm is the most common public and private key encryption algorithm used, and I implement a key of size 2048 bits. My system writes the public and private keys into a file by converting them into an array of bytes. The public and private key's primary encoding format is an array of bytes, and this can be written into a file easily. This array Once the keys have been created, and a text file is created, the Encryption class can be run to encrypt the message within the file. This is done using a Cipher, which is an object from the Java security libraries. Once the cipher is initialized, the class reads the key from the file that it has been saved to, and then writes the now encrypted message onto a new file. This file can now be sent to another computer with the private key, or

the Decryption class can be run. This class also uses a Cipher to decrypt the message. Using the private key, it takes the encrypted message, and writes the original message onto a new file.

GenerateKeys
-pairGen: KeyPairGenerator -pair: KeyPair -publicKey: PublicKey -privateKey: PrivateKey
+GenerateKeys() +createKeys(): void +getPublicKey(): PublicKey +getPrivateKey(): PrivateKey +createTextFile(key: byte[], name: String): void +main(): void

Decryption
-cipher: Cipher
+Encryption() +getPrivateKey(keyFileName: String): PrivateKey +decryptFile(startingFile: File, newFileName: String, key: PrivateKey) : void +main(): void

Encryption
-cipher: Cipher
+Encryption() +getPublicKey(keyFileName: String): PublicKey +encryptFile(startingFile: File, newFileName: String, key: PublicKey) : void +main(): void

Requirements

This system is addressing the ever-increasing need for the secure transfer of data. Something as simple as this can help to keep sensitive data such as passwords or personal information safe from those that wish to do harm with the information. With the increasing amount of reported break ins, malware, and other ransomware attacks in the news lately, the more secure your data is the better. It is unnerving to think that there may be some information about us that has already been stolen, and we just don't know it yet. This system, while rather simple, can be applied to many other situations where securing data is imperative to keep people's internet use safe.

Literature Survey

Encryption is one of the most important parts of cybersecurity. There have been many ways that different people and companies have tried to ensure their information could not be stolen. The RSA algorithm is just one of many different ways that data can be encrypted. Another example of encryption is the AES or Advanced Encryption Standard. This method of encryption is so sophisticated, that it is used by the U.S. government for most of their encryption. The AES is considered unbreakable as of right now, and many believe that it will become the standard in the cryptography. As with many encryption methods AES is also available for public use but is often not used because something simpler often works just fine. This means that potential flaws or improvements are often collaborated upon by a larger group of people, therefore strengthening the algorithms. There would always be different ways to

encrypt data, but techniques such as the RSA and AES are some of the most common that are used by the public, tech companies, and the government.

User Manual

To use the system, the first thing that you do is run the GenerateKeys class. This class will create the public and private keys and save them to a folder titled Encryption. These keys can now be sent out and shared. For example, if you wanted someone to send you a message, you would send them the public key. Then to encrypt a message, you have to create a .txt file in the Encryption folder. Once both the public key and the .txt file are in the folder, then the Encryption class can be run. It will create a file titled EncryptedMessage. This file can then be sent to the holder of the private key, or decrypted on the same computer, assuring the private key is still in the Encryption folder. Once the file titled EncryptedMessage and the private key are in the Encryption folder, the Decryption class can be run, and it will produce a file titled DecryptedMessage. This message is the same as the original .txt file that was created. After every class runs successfully, there is a message assuring that the program ran without any problems.

Conclusion

In conclusion, this project has taught me a lot about public and private key encryption. It has helped me to learn how to implement the Java security libraries, and to understand what they do. I think that the end product is something that is not only cool to see in action but is actually useful in keeping information safe. I think that this is something that I can use to have the information I receive and the information I send out always stay secure.