INF-2301: Computer communication and security

File sharing with encryption

Camilla Stormoen

10.10.2014

Introduction

This assignment is about implementing and analyze a file-sharing solution using encryption. The implementation should require use of both symmetric- and asymmetric encryption. There should be used AES for symmetric encryption and RSA for asymmetric encryption.

Technical background

AES

AES stands for Advanced Encryption Standard. The same key is used for both encryption and decryption in AES. The attacker who eavesdrop can not decrypt the message without the key.
Another way to make a symmetric cryptosystem is to use a public-key encryption. In this encryption, the one person has two keys, one secret key and one public key. The person sends the public key and the receiver can encrypt the message. AES is a fast and secure encryption form.

RSA

RSA is an cryptoalgorithm based on a public key. This asymmetric encryption, needs two separate keys, one secret and one public. In asymmetric key can everyone encrypt a message with the public key, but the holder of private key is the only one to decrypt the message. The attacker, or no one else can therefor not decrypt the message without the private key.

Design

The implementation for both the server and client is made quite simple, and doesn't include much code or knowledge about implementing a server/client.
The implementation of the RSA and AES is only implemented by the minimum requirements in the assignment.

RSA is implemented in the client-file and the the implementation generate a new RSA-key and a private key. The client then send the public key over to the server.
The server receive the public key and generate an symmetric AES key with the public RSA key. Then

the implementation in the server read the text-file which is to be sent, encrypt the message and send it to client. The server receive the encrypted message from server, decrypt the message and put it in new document. The client has to generate a cipher witch is a algorithm for performing encryption or decryption.

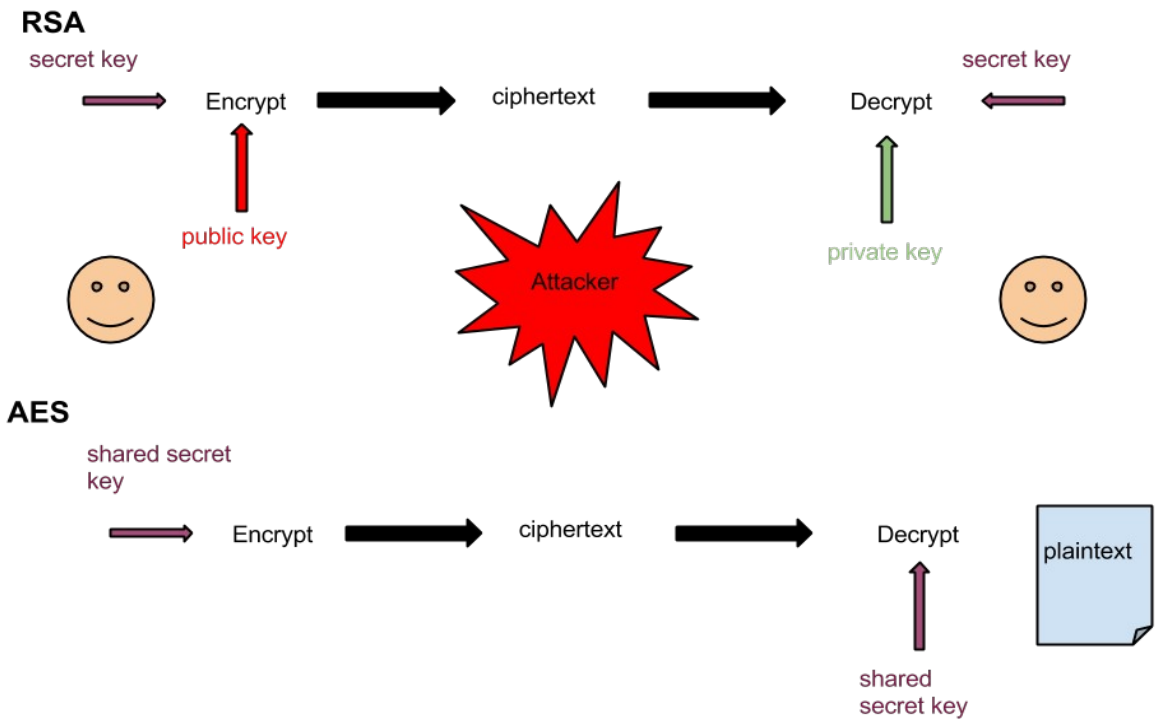

Figure 1: RSA and AES encryption

Implementation

The problem was solved using the programming language Python and the Python Crypto[1] for the encryption. To run the program, you have to have downloaded the Pycrypto library to your computer or else the program won't run.
The server is automatically set to run on 'localhost 8080', but in the client you have to choose the host and port, which will be the same as the servers host and port.

Discussion

C.I.A stands for confidentiality, integrity and availability. *Confidentiality* avoids sensitive information to reaching the wrong persons, but the right people to get it. *Integrity* is about the information has not been changed in an unauthorized way, and the last *Availability* is that the data you need should always be availably to you.

In *Confidentiality* you have access control and physical security. This implementation doesn't have any of those two. The program would be more secure if for example a user-name and password was implemented to access the program. That would strengthen the implementation for attacks and hacks on Confidentiality.

When it comes to *Integrity*, the code should be backed up somewhere else so they could check later if the code has been changed. Hashing is in the section *Integrity* in C.I.A . The purpose is that if someone changed something, you could know what's changed. Hash functions can also be used to implement digital signatures. Hash functions is easy to compute, but hard to invert.

In the *Availability* section, the users who are allowed should always have access yo the data. That would probably mean that there should be some kind of physical protection like “locks” or access control. The code implemented in this assignment has no kind of physical protection. Whoever wants to can modify the code or the message the program is supposed to send.

1 http://pythonhosted.org//pycrypto/, [Online-accessed 08-October-2014]

A.A.A stands for Assurance, Authenticity and Anonymity. *Assurance* is how the trust is provided and managed in computer systems. *Authenticity* is the ability to decide that statement, policies and permissions issued by persons or systems are real. *Anonymity* is that certain records or transactions are not to be attributable to any individual.

Assurance could be implemented by using policies, permissions and protection. Policies are often very long and written by a committee.

The implementation has no guarantee that the client will receive the correct data from the server or that it is the right person who has sent it. To make this more safe, we could add a digital signature. A digital signature can be used with encrypted- or non encrypted messages to make sure the sender's identity and message arrive intact. This will go under the section of *Authenticity* in A.A.A.

Anonymity can be used by having proxies which limit access to certain web pages from a network. A proxy is set up between the server and the client.

The implementation made based on C.I.A and A.A.A is not particular good. The implementation does not regard the protocols and it would be easy for someone to attack both server and client.
The solution is supposed to send encrypted files between servers and clients located on different computers. An attacker could easily pretend to be the server or the client and then tamper with the file. An attacker could also be eavesdropping the private conversation when the keys are sent to one another.

A different way to be hacked is by the "*man-in-the-middle*"-attack which is a form for eavesdropping where the attacker makes independent connection with the people who think they are the only one they talk to. The attacker replaces the keys and messages between them. This "*man-in-the-middle*"-attack will make victims believe that they are talking directly to each other, when the conversation is essentially controlled by the attacker.

To make the sending be more safe and trustful, we can add an certificate authority. To create a certificate authority you need to know the public key of certificate authorities. This would make the people on the client side to rely on the signature made by the private key that correspond to the certified public key. The CA trust relationship is based on a TTP(trusted third party) which is a facility based on both parties trusting the TTP.

In the implementation, the receive is sat to an exactly size, which is here 2048 and 4096. This means that if the client or server sends more than the set size, the other part can't receive everything. That would be a problem. To fix this problem, you could send the actually length of the message, key, iv and everything you will send to the other part. When the other part are receiving, there will always be enough bits to receive.

Evaluation

The program works as expected and the code meets the requirements for the assignment. The Pycrypto library was very useful because you got a lot of help with the implementation of the various encryption methods.
When implementing the encryption, I used the print-function to check what I send from server to client and reverse. It helped me a lot with knowing what was sent and what was received on the other side. When everything was received and sent on both server- and client-side, the minimum requirements had been fulfilled.

Conclusion

This assignment is about implementing and analyze a file-sharing solution using encryption. The implementation should require use of both symmetric- and asymmetric encryption and there should be used AES for symmetric encryption and RSA for asymmetric encryption.
This was a assignment I really liked. It was fun and informative, and the implementation was not so difficult as I thought it would be. One tricky thing was too read the documentation for the crypto library for Python.

References

Wikipedia, “*Rsa (cryptosystem)*” - http://en.wikipedia.org/wiki/RSA_%28cryptosystem%29 [Online accessed 06-October-2014]

Wikipedia, “Public-key *cryptography*” - http://en.wikipedia.org/wiki/Public-key_cryptography [Online accessed 06-October-2014]

“*Introduction to Computer Security, First Edition”* - Michael Goodrich, Roberto Tamassia