Hacettepe University

Computer Engineering Department

BBM465 INFORMATION SECURITY - 2022 FALL

Assignment 3

December 7, 2022

Student names:
Yiğit Emir İŞIKÇI
Abdullah Mert DİNÇER

Student Numbers:
2200356028 - 2200356016

1 Problem Definition

In this assignment it is required to design of a licensing framework by utilizing the methods of asymmetric cryptography, MD5, and digital signatures.
Although the methods that take place in symmetric cryptography are strong, they have some shortcomings such as key publication for both receiver and sender. This methods are vulnurable to attacks such as "man in the middle" attack.
On the other hand, asymmetric cryptography presents a different perspective in order to solve these drawbacks with public and private keys which are key pairs.

2 Method - Solution

2.1 Method

We designed these classes according to their abilities such as client can encrypt data with public key, hash, verify but cant decrypt. License manager can decrypt with private key, sign the hash etc.
We choose to create an object of license manager and use its methods but it could be implemented as a static class. Also, its attributes could be private but we didnt do it. Because we thought that doing them private and access them with getter methods and doing them public is same.
We did processes of client and server at client side. Because we thought that the process is spesific for clients.

2.2 Solution

Firstly, we implemented getMacAdress(), getSerialNumbers(), getPublicKey() methods to get Client spesific information. Whenever a client object is created, these methods also executed and gets the information. We got help from stackoverflow to implement these methods (given in references part). We also create an attribute named fullContent to store these information.
We also hard-coded userName and serial number which stated in pdf. ("Yigit" and "1234-5678-9012")
Secondly, we implemented encrypt(), hash(), createLicense(), isLicenseFileBroken(),sendToLicenseManager() and verify() methods in Client class.
Then, we implemented LicenseManager classes methods. Such as getPublickey(), getPrivateKey(),sentToClient(), sign(), hash() and decryptMessage().
The methods getPublicKey and getPrivateKey simply gets key data from "keys" file. The hash() method is same as Client Classes hash() method.

3 Methods of Client Class

3.1 Encrypt() Method

This method is encrypting the full content (plain text) with public key according to RSA algorithm.

```
    public byte[] encrypt(String plainText, PublicKey publicKey) {
        byte[] cipherText = null;
        try {
            Cipher encryptCipher = Cipher.getInstance("RSA");
            encryptCipher.init(Cipher.ENCRYPT_MODE, publicKey);
            cipherText = encryptCipher.doFinal(plainText.getBytes("UTF8"));
        } catch (Exception e) {
            e.printStackTrace();
        }
        return cipherText;
    }
```

3.2 Hash() Method

This method is taking hash of given data with MD5 algorithm

```
    public byte[] hash(String data) {
        byte[] hashedValue = null;
        try {
            MessageDigest messageDigest = MessageDigest.getInstance("MD5");
            messageDigest.reset();
            messageDigest.update(data.getBytes("UTF8"));
            hashedValue = messageDigest.digest();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return hashedValue;
    }
```

3.3 createLicense() Method

This method is creating a file named license.txt and writes the signature in it.

```
    public void createLicense() throws IOException {
        OutputStream os = new FileOutputStream("license.txt");
        os.write(signature);
        os.close();
    }
```

3.4 sendToLicenseManager() Method

This method simply sets the license manager's enctypted data attribute. For abstraction, we named it like that.

```
1     public void send_to_LicenseManager(byte[] encryptedData, LicenseManager
          licenseManager) {
2         licenseManager.setEncryptedData(encryptedData);
3     }
```

3.5 verify() Method

This method first creates signature with public key and hash value and returns true if this signature and the signature that came from license manager are equal.

```
1     public boolean verify() {
2         boolean equal = false;
3         try {
4             Signature signature_verify = Signature.getInstance("SHA256withRSA"
                  );
5             signature_verify.initVerify(publicKey);
6             signature_verify.update(hashedData);
7             return signature_verify.verify(signature);
8         } catch (Exception e) {
9             e.printStackTrace();
10        }
11        return equal;
12    }
```

3.6 isLicenseFileBroken() Method

While we are checking the license file, this method checks if the signature and the data of current user is matching. If matches it returns false, returns false otherwise.

```
1     public boolean isLicenseFileBroken(){
2         hashedData = hash(full_content);
3         signature = readLicenseFile();
4         if (verify()) {
5             return false;
6         } else {
7             return true;
8         }
9     }
```

4 Methods of LicenseManager Class

4.1 decryptMessage() Method

This method decrypts the cipher text with private key according to RSA algorithm.

```
    public String decryptMessage(byte[] cipherText) {
        String plain = "";
        try {
            Cipher decryptCipher = Cipher.getInstance("RSA");
            decryptCipher.init(Cipher.DECRYPT_MODE, privateKey);
            byte[] result = decryptCipher.doFinal(cipherText);
            plain = new String(result, "UTF8");
        } catch (Exception e) {
            e.printStackTrace();
        }
        return plain;
    }
```

4.2 sign() Method

This method signs the hashed value with private key. It uses SHA256WithRSA scheme.

```
    public byte[] sign(byte[] hashedData) {
        byte[] signedData = null;
        try {
            Signature privateSignature = Signature.getInstance("SHA256withRSA"
                );
            privateSignature.initSign(privateKey);
            privateSignature.update(hashedData);
            signedData = privateSignature.sign();
        } catch (Exception e) {
            e.printStackTrace();

        }
        return signedData;
    }
```

4.3 sendToClient() Method

This method simply sends the signature to client because client needs to verify it. For abstraction we named the method like that.

```
    public void send_to_Client(byte[] signature, Client client){
        client.setSignature(signature);
```

```
3       }
```

5 main and the PROCESS()

We want to describe these methods seperately because they include all of the process.
In the main method, firstly it creates the client and license manager objects. Then it checks if license file exist or not.
If license file doesnt exist, process method executes. If exists, it checks this existing license file is broken or not (explained in isLicenseFileBroken method above).
If this existing license file is broken, process method executes and creates NEW license for user. If not broken, simply, program ends with the output "Succeed. The license is correct.".

PROCESS method is for specific user. So it is implemented in Client class.
Firstly, it prints the user spesific data such as mac adress,disk id, motherboard id. Then client encrypts and hashes this data. After this processes, client sends the encrypted data to license manager. License manager firstly decrypts it to have data of user. Then it takes hash of this data to create signature.
As i stated before, license manager signs the hashed data with private key. Then it sends this signature to client. Client verifies this signature with his hashed data and public key. If this verification completed succesfully, it creates license (writes signature to license.txt file).

6 Important Note

You should keep "keys" file and "license.txt" in the same directory with Client and LicenseManager classes.

7 Resources:

- https://stackoverflow.com/questions/6164167/get-mac-address-on-local-machine-
- https://stackoverflow.com/questions/5482947/how-to-get-the-hard-disk-serial-n
- https://www.geeksforgeeks.org/java-program-to-get-system-motherboard-serial-n