CIS 535 Computer Networks and System Security

**Project 2: Secure Message System**

**Project Objectives:**

In this project, you will are asked to build a secure message system to simulate the secure email communication across the Internet. You should NOT use any existing crypto liberties, not even the ones that are part of the standard java library. The purpose of this project is to implement a secure facility from scratch, and use it to secure message communication in the simulator.

**Project Requirements:**

1. In the secure facility, you need to implement the three algorithms, i.e, symmetric, hash function, RSA algorithms, and Certificate Authority, as describe in lecture. The secure facility enables 1) Confidentiality, 2) Message Integrity, and 3) Sender authentication and should be used in the System. You may use very simple algorithms. For example,

* CA can be as simple as a dictionary. For example

HashMap<String, BigInteger> CA = new HashMap<>();

* Hash function can be as simple as ***H(x) = x mod 11***
* Symmetric encryption algorithm can be as simple as “***Shift Cipher***” or “***Substitution Cipher***”.
* RSA algorithm, the code will be provided. You may start with manually calculated keys.
  + Sender's public key: Amy+(n, e) = (4127118949, 157) //generated using the provided RSA algorithm
  + Receiver's public key: Bob+(n, e) = (35, 5) ,
  + Receiver’s private key: Bob-(n, d) = (35, 29)

// directly from lecture on page 21

2. The secure message system include four components, 1) Sender, 2) Receiver, 3) Packet, and 4) Network. To be specific,

* Sender should generate a message and process the message using ALL steps listed in Table 1 and prepare the Packet
* Sender send the Packet to Network
* Receiver receive the Packet from Network
* Receiver should decrypt the Packet and check Packet integrity using the steps listed in Table 2.

**Table 1: Sender (Alice)’s Processing Steps:**

|  |  |
| --- | --- |
| Step | Description |
| 1 | |  |  | | --- | --- | | Get message |  | |
| 2 | |  |  | | --- | --- | | Hash message |  | |
| 3 | |  |  | | --- | --- | | Encrypt hashed message to generate signature |  | |
| 4 | |  |  | | --- | --- | | Generate random symmetric key |  | |
| 5 | |  |  | | --- | --- | | Encrypt message with Ks |  | |
| 6 | |  |  | | --- | --- | | Encrypt signature from step 3 |  | |
| 7 | |  |  | | --- | --- | | Concatenate encrypted signature from step 6 with encrypted message from step 5 |  | |
| 8 | |  |  | | --- | --- | | Encrypt key with Bob’s public key |  | |
| 9 | |  |  | | --- | --- | | Concatenate encrypted Ks from step 8 with results from step 7 and make a packet including all the three fields |  | |
| 10 | |  |  | | --- | --- | | Send packet to Bob |  | |

**Table 2: Receiver (Bob)’s Processing Steps:**

|  |  |
| --- | --- |
| Step | Description |
| 1 | |  |  | | --- | --- | | Receive packet from Alice |  | |
| 2 | |  |  | | --- | --- | | Split session key and message |  | |
| 3 | |  |  | | --- | --- | | Decrypt session key with Bob’s private key |  | |
| 4 | |  |  | | --- | --- | | Decrypt message with session key |  | |
| 5 | |  |  | | --- | --- | | Decrypt signature with session key |  | |
| 6 | |  |  | | --- | --- | | Hash message from step 4 |  | |
| 7 | |  |  | | --- | --- | | Decrypt signature from step 5 with Alice’s public key |  | |
| 8 | |  |  | | --- | --- | | Compare the hash results from 6 and 7 |  | |
| 9 | |  |  | | --- | --- | | If hashes are equal, message is OK |  | |

**Sample Output:**

run:

\*\*\*\*\*\*\*\*\* Secure Message System Simulation \*\*\*\*\*\*\*\*\*\*\*

Sender is created

Receiver is created

Network is created

\* Cerficate Authority (CA) is created \*

Sender's public key: Amy(n, e) = (2808768283, 239) now in CA , it is generated from the RSA algorithm

ONLY for INFO: Sender's private key: Amy(n, d) = (2808768283, 493572143) NOT in CA, Sender NEVER gives it out, just for INFORMATION

Receiver's public key:Bob(n, e) = (35, 5) now in CA, it is hard coded,taking from Lecture

ONLY for INFO: Receiver's public key:Bob(n, D) = (35, 29) NOT in CA, Receiver NEVER gives it out, it is hard coded,taking from Lecture

\*\*\* Sender's Operations for secure message communication \*\*\*

---Step 1: Sender generate message and

m = 13

---Step 2: Sender hash message and

H(m) = 2

---Step 3: sign H(m) with sender's private key and

Ka-(H(m)) = 703526442

---Step 4: generate a session key and

Ks = 2

---Step 5: encrypt Ks with receiver's public key using RSA algoirthm and

Kb+(Ks) = 32

Receiver's N is: 35

Receiver's E is: 5

encryptedKsWithReceiverPublicKey is: 32

and Set packet.sessionKey = 32

---Step 6: encrypt message m with session key and symmetric algorithm and

Ks(m) = 15

also set Packet.message = 15

---Step 7: encrypt sender's digital signature Ka-(H(m)) with session key and symmetric algorithm and

Ks(Ka-(H(m))) = 703526444

also set packet.signature = 703526444

---Step 8: the packet to be sent on to Internet is:

pk.message 'Ks(m)' = 15

pk.signature 'Ks(Ka-(H(m)))' = 703526444

pk.sessionKey 'Kb+(Ks)' =32

------Send packet to Receiver through internet

|||||||||||||||||||||||||||||||||||||||||||||

|||||||||||||||||||||||||||||||||||||||||||||

------Asumme perfect Internet with no error-----

|||||||||||||||||||||||||||||||||||||||||||||

|||||||||||||||||||||||||||||||||||||||||||||

\*\*\* Receiver's Operations for secure message communication \*\*\*

------Receiver get packet through internet

---Step 1: Receive the packet from Internet rcvPacket

---Step 2: Split the packet

Receiver: received cipher from the network is:

pk.message 'Ks(m)' = 15

pk.signature 'Ks(Ka-(H(m)))' = 703526444

pk.sessionKey 'Kb+(Ks)' =32

---Step 3: Decrypt pk.sessionkey, i.e., Kb+(Ks) with Receiver's private key 'Kb-'

Receiver: the encryptedKs 'Kb+(Ks)' is: 32

After decryting with receiver's privateky 'Kb-', it get Ks = 2

---Step 4: decrypt pk.message 'Ks(m)' using Ks which is gotten from step 3: 15

After decryption receiver gets message 'm' = 13

---Step 5: Decrypt Ks(Ka-(H(m))) with 'Ks' got from step 4, Ks(Ka-(H(m))) =703526444

and gets the digital signature 'Ka-(H(m))' = 703526442

---Step 6: decrypt 'Ka-(H(m))' from sender's public key 'Ka+()'

Sender N = 2808768283, Sender E = 239

The decrypted message digest 'H(m)' = Ka+(Ka-(H(m))) = 2

---Step 7: hash message m from step 4 'm' = 13

The Hash result H(m) = 2

---Step 8: Compare results from step 6 and step 7, if they match then accept otherwise discards

The packet has passed through the integrity checking and is accepted!

BUILD SUCCESSFUL (total time: 0 seconds)