COMP90043 Cryptography & Security Assignment 2

Emmanuel Macario - 831659

Semster 2, 2019

Question 1

a. Lianglu could recover the message without knowing Ge's or Jiajia's private key by considering the following:

Since different e are relatively prime, then $gcd(e_G, e_J) = 1$. Hence, $\exists x, y \in \mathbb{Z}$ such that $xe_G + ye_J = 1$. We can use XGCD to find x, y. Then, we can use x, y to find M as follows:

$$C_G^x C_J^y = (M^{e_G})^x (M^{e_J})^y = M^{xe_G + ye_J} = M^1 = M \pmod{n}$$

b. Lianglu can recover Zhuohan's private key using the following strategy.

Firstly, since we know d_G, we can factor n. Then, we can use XGCD to obtain d_Z.

In order to factor n, let $k = d_G e - 1$. Since $ed - 1 = 0 \bmod \phi(n)$, then k is a multiple of $\phi(n)$. We also know that k must be even. Hence, let $k = 2^t r$ with r odd and $t \geq 1$. Now, we pick a random generator g, where $1 < g < N$, and compute the sequence $g^{k/2}, g^{k/4}, \ldots, g^{k/2^t}$. We determine the first sequence element $x \neq \pm 1$, where $gcd(x - 1, n) > 1$.

If no such element exists, then choose another g and try again.

Otherwise, let $p = gcd(x - 1, n)$ and $q = n/p$.

Now that we have p and q, we can compute $\phi(n) = (p - 1)(q - 1)$. Using XGCD, we can now find $d_Z = e_Z^{-1} \bmod \phi(n)$.

Question 2

a. Encryption function for Ge:

$$E_G(M) = M^{k_1} \bmod n$$

Decryption function for Jiajia:

$$D_J(C) = C^{k_2 k_3} \bmod n$$

b. Encryption function for Jiajia:

$$E_{J\to X}(M) = \begin{cases} M^{k_3} \bmod n & \text{if } X = \text{Ge} \\ M^{k_2} \bmod n & \text{if } X = \text{Zhuohan} \end{cases}$$

Decryption function for Ge:

$$D_G(C) = C^{k_1 k_2} \bmod n$$

Decryption function for Zhuohan:

$$D_Z(C) = C^{k_1 k_3} \bmod n$$

Question 3

a. (1) A sends a request message for a new session key to B, which includes the identity of A, and a unique nonce N_a encrypted with their secret key K_a

(2) B sends A's original message, plus their own message mirroring A's, which includes the identity of B, and a unique nonce N_b encrypted with their secret key K_b, to the KDC

(3) The KDC sends B a response message. The message includes two components, one meant for A, and the other meant for B. Each component is encrypted with the intended recipient's secret key, and includes:

- The one time session key, K_s, to be used for the session
- The identity of the opposite communicating user (e.g. ID_a for B)
- The original nonce, to enable the users to match the response with the corresponding request

(4) B forwards A's component, which they received from the KDC, to A. Now, both A and B have access to the session key.

b. To estimate the storage requirements of both the KDC and users, some assumptions must be made. We assume that a session key is discarded after use, and that the KDC does not store session keys. For each user, there is a master key that they share with the KDC. Hence, if there are a total of N unique users, then the KDC is required to persistently store N master keys, one for each user.

In contrast, each user must only store their own master key. Session keys must also be kept for the duration of the corresponding session.

c. This scheme does not ensure the authenticity of both A and B. The authenticity of B can be compromised if either B or an attacker C sends the KDC the message $ID_c \| E(K_c, N_c)$ (or any message consisting of a matching identity and encrypted nonce pair). This means that when A receives a response, the identity of the responder will not be who they believed it to be.

d. A replay attack can be detected in this scheme, and can be identified in steps (2), (3), and (4). In step (2), if the KDC is sent a message $ID_x \| E(K_y, N_y)$, where $x \neq y$, then the KDC may be suspicious of a replay attack, since they cannot decrypt the encrypted nonce. In both steps (3) and (4), both B and A can detect a replay attack if the nonce that they receive is not equivalent to their original nonce.

Question 4

a. Yes. Each user must have access to the other user's public key. The procedure can be summarised in the following three steps:

$$\text{(1) Alice} \rightarrow \text{Bob: } g^x$$
$$\text{(2) Alice} \leftarrow \text{Bob: } g^y, E_{g^{xy}}(S_B(g^y, g^x))$$
$$\text{(3) Alice} \rightarrow \text{Bob: } E_{g^{xy}}(S_A(g^x, g^y))$$

b. No. An attacker can simply change the data stream and recompute the hash value, before sending the message.

c. Yes. The procedure can be summarised in the following four steps:

$$\text{(1) Alice} \rightarrow \text{Bob: } g^x, C(K_{AB}, g^x)$$
$$\text{(2) Alice} \leftarrow \text{Bob: } g^y, C(K_{AB}, g^y)$$
$$\text{(3) Alice: Test if received MAC matches calculated MAC}$$
$$\text{(4) Bob: Test if received MAC matches calculated MAC}$$

Question 5

a. Yes. If the last block M_i has length less than the fixed size, then it is possible to pad it to the specified fixed length.

b. Yes, since the output is always mod n.

c. No, since a message containing many blocks may require numerous expensive exponentiation computations.

d. Yes, since the hash function is based on the RSA problem.

e. No.

f. No.