공개키 암호

(Public Key Cryptography)

Q. RSA의 개인키의 크기를 줄이시오.

Answer. Recall that Chinese Remainder Theorem (CRT) :

Chinese Remainder Theorem (CRT) - Special Case

Theorem. *Consider a system of two linear congruences:*

$$\begin{aligned} x &\equiv a_1 \pmod{p} \\ x &\equiv a_2 \pmod{q} \end{aligned}$$

where p, q are coprime. Let $N = pq$. Then, the unique solution of the system of congruences is give by

$$x = a_1 q q_p^{-1} + a_2 p p_q^{-1} \mod N$$

where $q_p^{-1} = q^{-1} \mod p$ and $p_q^{-1} = p^{-1} \mod q$.

Recall that Bézout's identity : $a, b \in \mathbb{Z} \implies \exists x, y \in \mathbb{Z} : \gcd(a,b) = ax + by$. Especially,

$$p, q \text{ are coprime} \implies \exists x, y \in \mathbb{Z} : px + qy = 1.$$

Let p, q are coprime. Then $\exists x, y \in \mathbb{Z} : px + qy = 1$ and so

$$px = (-y)q + 1 \quad \rightsquigarrow \quad px \equiv 1 \pmod{q} \quad \rightsquigarrow \quad x = p^{-1} \mod q.$$

Similarly, $y = q^{-1} \mod p$. Thus we have $px + qy = 1 \rightsquigarrow pp_q^{-1} + qq_p^{-1} = 1$. Consequently,

$$\begin{aligned} x = a_1 q q_p^{-1} + a_2 p p_q^{-1} \mod N &\leftrightsquigarrow x = a_1 q q_p^{-1} + a_2 (1 - q q_p^{-1}) \mod N \\ &\leftrightsquigarrow x = (a_1 - a_2) q q_p^{-1} + a_2 \mod N \end{aligned}$$

continue on next page.

and that RSA-CRT algorithm :

Algorithm 1: RSA-CRT Algorithm

Data: The security parameter k, a public key (N, e), a ciphertext $\mathcal{C}$.
Result: The plaintext message $\mathcal{M}$ corresponding to the ciphertext $\mathcal{C}$.

/* Key Generation */
Function KeyGen (k)**:**
- $p, q \leftarrow$ random prime numbers of $k/2$ bits each ; // Generate two primes
- $N \leftarrow pq$; // Compute modulus
- $\phi(N) \leftarrow (p-1)(q-1)$; // Compute Euler's phi function
- $e \leftarrow$ integer s.t. $1 < e < \phi(n) \wedge \gcd(e, \phi(n)) = 1$; // Choose encryption exponent
- $d_p \leftarrow e^{-1} \mod p-1$; // Compute decryption exponent for p
- $d_q \leftarrow e^{-1} \mod q-1$; // Compute decryption exponent for q
- $q_{inv} \leftarrow q^{-1} \mod p$; // Compute q inverse modulo p
- Set the RSA public key as (N, e);
- Set the RSA secret key as $(p, q, d_p, d_q, q_{inv})$;

End Function

/* Encryption */
Function Enc $(N, e, \mathcal{M})$**:**
- $\mathcal{C} \leftarrow \mathcal{M}^e \mod N$; // Encrypt with e and N

End Function

/* Decryption */
Function Dec $(\mathcal{C})$**:**
- $m_1 \leftarrow \mathcal{C}^{d_p} \mod p$; // Decrypt with d_p and p
- $m_2 \leftarrow \mathcal{C}^{d_q} \mod q$; // Decrypt with d_q and q
- $t \leftarrow q_{inv}(m_1 - m_2)$; // Reconstruct the message using CRT
- $m \leftarrow m_2 + qt \mod N$; // $m = (m_1 - m_2)qq_{inv} + m_2 \mod N$

End Function

return m;

위 알고리즘을 통해 d_p, d_q, p 및 q를 사용하여 공개 키 (N, e)로 암호화된 메시지를 복호화할 수 있다는 걸 알 수 있습니다. d_p와 d_q는 전체 d보다 훨씬 작기 때문에 RSA 개인 키의 길이가 줄어듭니다.

따라서 RSA-CRT는 복호화 지수 d를 두 부분으로 분할하여 RSA 개인 키의 길이를 줄이며 RSA의 복호화 지수를 더 쉽게 계산할 수 있도록 합니다. □