[CS309] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy
Scribed by: Kunj Thakkar (202251142)

Autumn 2024-2025
Lecture (Week 8)

1 RSA Cryptosystem

There are some facts which should be pre-known before understanding RSA. Here are they

1.1 Facts

1. If gcd(a,m) = 1, then $a^{\phi(m)} \equiv 1 \pmod{m}$
2. $a^{p-1} \equiv 1 \pmod{m}$, if p is prime and p does not divides a.

1.2 Components of RSA

1. n = pq where p and q are large prime numbers
2. Plaintext space: $\mathbb{Z}_n$
   Ciphertext space: $\mathbb{Z}_n$
3. Key space: $\{ K = (n, p, q, e, d) \mid ed \equiv 1 \pmod{\phi(n)} \}$
4. Encryption:
$$E(x, K) = c$$
$$c = E(x, K) = x^e \pmod{n}$$
5. Decryption:
$$\text{Dec}(c, K) = x$$
$$c = \text{Dec}(c, K) = c^d \pmod{n}$$

2 Correctness of RSA

Starting with the fact that $ed \equiv 1 \pmod{\phi(m)}$,

$$\implies ed - 1 = t\phi(m)$$
$$\implies 1 = ed + t_1\phi(m)$$
$$1 = gcd(e, \phi(m)) = ed + t_1\phi(m)$$

Using encryption function, which is,

$$c = x^e \pmod{n}$$

also decryption is,

$$c = c^d \pmod{n}$$

$$c^d = (x^e)^d \pmod{n}$$

using the fact that $1 = ed + t_1\phi(m)$,

$$c^d = x^{1+t\phi(m)} \pmod{n}$$

$$c^d = x * x^{t\phi(m)} \pmod{n}$$

since p and q are primes and n = pq, then $\phi(n)$ = (p-1)(q-1)

$$c^d = x \cdot x^{t[(p-1)(q-1)]} \pmod{n}$$

Finally,

$$c^d = x \cdot x^{t[(p-1)(q-1)]} \pmod{pq}$$

Now, let us simplify the part $x^{t[(p-1)(q-1)]} \pmod{pq}$, where $x \in \mathbb{Z}$:

We check $x^{t[(p-1)(q-1)]} \pmod{p}$:

$$\equiv (x^{p-1})^{t(q-1)} \pmod{p}$$

$$\equiv 1 \pmod{p}$$

(As $x^{p-1} \equiv 1 \pmod{p}$)

Now we check $x^{t[(p-1)(q-1)]} \pmod{q}$:

$$\equiv (x^{q-1})^{t(p-1)} \pmod{q}$$

$$\equiv 1 \pmod{q}$$

(As $x^{q-1} \equiv 1 \pmod{q}$) We finally have:

$$x^{t[(p-1)(q-1)]} \equiv 1 \pmod{p}$$

$$x^{t[(p-1)(q-1)]} \equiv 1 \pmod{q}$$

$$\Rightarrow x^{t[(p-1)(q-1)]} \equiv 1 \pmod{pq}$$

Substituting the above result:

$$c^d = x \cdot x^{t[(p-1)(q-1)]} \pmod{pq}$$

$$c^d = x \cdot 1 \pmod{pq}$$

$$c^d = x \pmod{pq}$$

Hence, our decryption is successful.

3 Chinese Remainder Theorem

Suppose $m_1, m_2, \ldots, m_r$ are pairwise relatively prime positive integers, and let $a_1, a_2, \ldots, a_r$ be any integers. Then the system of congruences

$$x \equiv a_i \pmod{m_i} \quad (1 \leq i \leq r)$$

has a unique solution modulo $M = m_1 \times m_2 \times \cdots \times m_r$, which is given by

$$x = \sum_{i=1}^{r} a_i M_i y_i \pmod{M},$$

where

$$M_i = \frac{M}{m_i} \quad \text{and} \quad y_i = M_i^{-1} \pmod{m_i}$$

for $1 \leq i \leq r$.

4 Digital Signature Algorithm (DSA)

The Digital Signature Algorithm (DSA) is a cryptographic tool used to create digital signatures, which verify the origin of messages and help prevent tampering. The DSA structure includes five main components:

- P: the space of possible plaintext messages,
- S: the set of possible signatures,
- K: the set of parameters used,
- **Sign**: the algorithm for generating signatures, and
- V: the validation algorithm.

DSA relies on **two types of keys**:

1. A private key, kept secure by the sender.
2. A public key, distributed to recipients for verification.

The main signature generation process in DSA is represented as:

$$S = m^d \pmod{n}$$

where m is the message, d is the private key, and n is the modulus.

4.1 Comparing RSA and DSA

In RSA encryption, messages are encrypted using:

$$c_1 = m_1^e \mod n \quad \text{and} \quad c_2 = m_2^e \mod n,$$

and combined as:

$$c_1 \times c_2 = (m_1 \times m_2)^e \mod n.$$

In contrast, DSA utilizes hash functions, and the encrypted values do not combine multiplicatively:

$$s_1 = (h(m_1))^d \mod n \quad \text{and} \quad s_2 = (h(m_2))^d \mod n,$$

so that

$$s_1 \times s_2 \neq (h(m_1 \times m_2))^d \mod n.$$

4.2 Signature Generation Steps

To generate a digital signature:

1. Choose two distinct prime numbers p and q.
2. Compute $n = pq$ and the totient $\varphi(n) = (p-1)(q-1)$.
3. Select an integer e such that $1 < e < \varphi(n)$ and e is relatively prime to $\varphi(n)$.
4. Calculate d (the private key) so that $ed \equiv 1 \pmod{\varphi(n)}$.
5. Publish (n, e) as the public key and keep (n, d) as the private key.

4.3 Signature Validation

To encrypt a message m with the public key (n, e), generate the signature S using:

$$S = m^d \mod n.$$

Then, to verify the original message with the private key (n, d), calculate:

$$V = S^e \mod n.$$

In DSA, the validation step ensures security by confirming:

$$S_1 \times S_2 \neq (h(m_1 \times m_2))^d \mod n,$$

where $h()$ is a hash function. This approach ensures both the integrity and authenticity of digital communications.