SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

19ITB302-Cryptography and Network Security

UNIT-2 NUMBER THEORY AND PUBLIC KEY CRYPTOSYSTEMS

Example

- Find the GCD of 270 and 192
- A=270, B=192
- A $\neq$0
- B $\neq$0
- Use long division to find that 270/192 = 1 with a remainder of 78. We can write this as:
- A=B*Q+R
- 270 = 192 * 1 +78

- Find GCD(192,78), since GCD(270,192)=GCD(192,78)
- A=192, B=78
- A $\neq$0
- B $\neq$0
- Use long division to find that 192/78 = 2 with a remainder of 36. We can write this as:
- A=B*Q+R
- 192 = 78 * 2 + 36

- Find GCD(78,36),
- A=78, B=36
- A $\neq 0$
- B $\neq 0$
- Use long division to find that 78/36 = 2 with a remainder of 6. We can write this as:
- 78 = 36 * 2 + 6

- Find GCD(36,6),
- A=36, B=6
- A $\neq 0$
- B $\neq 0$
- Use long division to find that 36/6 = 6 with a remainder of 0. We can write this as:
- 36 = 6 * 6 + 0

Find GCD(6,0),

- A=6, B=0
- A ≠0
- B =0, GCD(6,0)=6
- **So we have shown:**
- GCD(270,192) = GCD(192,78) = GCD(78,36) = GCD(36,6) = GCD(6,0) = 6

GCD(270,192) = 6

Modular Arithmetic

The Modulus

- If **a** is an integer and **n** is a positive integer, we define **a mod n** to be the **remainder** when **a** is divided by **n.** The integer **n** is called the **modulus.**
- Modular arithmetic is a system of arithmetic for integers, where numbers "wrap around" upon reaching a certain value called **modulus**
- 1:00 and and 13:00 hours are the same(1=13mod12)

Congruence

Integers that leave the same remainder when divided by the modulus m are somehow similar, however, not identical.

Such numbers are called **“congruent”.**

For instance, 1 and 13 and 25 and 37 are congruent mod 12 since they all leave the same remainder when divided by 12.

$a \equiv b \bmod m$

$15 \equiv 3 \pmod{12}$

$23 \equiv 11 \pmod{12}$

$33 \equiv 3 \pmod{10}$

$23 \equiv 3 \pmod{10}$

$38 \equiv 2 \pmod{12}$ q=3

$38 \equiv 14 \pmod{12}$ q=2

Computational Rules

- $((a \bmod m) + (b \bmod m)) \bmod m = (a + b) \bmod m$
- $((a \bmod m) - (b \bmod m)) \bmod m = (a - b) \bmod m$
- $((a \bmod m) * (b \bmod m)) \bmod m = (a * b) \bmod m$

Example

[(15 mod 8) + (11 mod 8)]mod 8=(15+11) mod 8

(7+3) mod 8 =26 mod 8

10 mod 8=26 mod 8

2=2

Properties of Modular Arithmetic

Property	Expression
Commutative Laws	$(a + b) \bmod n = (b + a) \bmod n$ $(a \times b) \bmod n = (b \times a) \bmod n$
Associative Laws	$[(a + b) + c] \bmod n = [a + (b + c)] \bmod n$ $[(a \times b) \times c] \bmod n = [a \times (b \times c)] \bmod n$
Distributive Laws	$[a \times (b + c)] \bmod n = [(a \times b) + (a \times c)] \bmod n$
Identities	$(0 + a) \bmod n = a \bmod n$ $(1 \times a) \bmod n = a \bmod n$
Additive Inverse	For each $a \in Z_n$, there exists a '$-a$' such that $a + (-a) \equiv 0 \bmod n$

RSA Algorithm

RSA Algorithm is based on factorization of large number and modular arithmetic for encrypting and decrypting data. It consists of three main stages:

1. **Key Generation: Creating Public and Private Keys**
2. **Encryption: Sender encrypts the data using Public Key to get cipher text.**
3. **Decryption: Decrypting the cipher text using Private Key to get the original data.**

RSA Algorithm

1. Key Generation

Choose two large prime numbers, say p and q. These prime numbers should be kept secret.

Calculate the product of primes, n = p * q. This product is part of the public as well as the private key.

Calculate Euler Totient Function$\Phi(n)$ as $\Phi(n) = \Phi(p * q) = \Phi(p) * \Phi(q) = (p – 1) * (q – 1)$.

Choose encryption exponent e, such that

$1 < e < \Phi(n)$, and

$gcd(e, \Phi(n)) = 1$, that is e should be co-prime with $\Phi(n)$.

RSA Algorithm

Calculate decryption exponent d, such that

(d * e) ≡ 1 mod Φ(n), that is d is modular multiplicative inverse of e mod Φ(n). Some common methods to calculate multiplicative inverse are: Extended Euclidean Algorithm, Fermat's Little Theorem, etc.

We can have multiple values of d satisfying (d * e) ≡ 1 mod Φ(n) but it does not matter which value we choose as all of them are valid keys and will result into same message on decryption.

Finally, the Public Key = (n, e) and the Private Key = (n, d).