2. RSA ALGORITHM

- **In a public key system using RSA, you intercept the cipher text C=10 sent to a user whose public key is e=5, n=35. What is the plain text? Explain the above problem with an algorithm description. (16) (MAY/JUNE 2012)**
- **Explain RSA algorithm with example as: p=11, q=5, e=3 and PT=9 (16)**

(NOV/DEC 2013)

- **Demonstrate encryption and decryption for the RSA algorithm parameters: p=3, q=11, e=7, d=? , M=5. (8) (MAY/JUNE 2014)**
- **Describe the mathematical foundations of RSA algorithm. Perform the encryption and decryption for the following: p=17, q=7, e=5, n=119, message="6". Use Extended Euclid's algorithm to find the private key. (16) (NOV/ DEC 2014)**
- **Explain the RSA algorithm in detail. For the given values, trace the sequence of calculations in RSA. p=7, q=13, e=5 and M=10. (16) (APR/MAY 2015) (MAY/JUNE 2014) (APR/MAY 2011) (NOV/DEC 2011) (MAY/JUNE 2013) (NOV/DEC 2012)**

One of the first successful algorithm to the public key cryptography was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978. The Rivest-Shamir-Adleman (RSA) scheme is the most widely accepted and implemented general-purpose approach to public-key encryption.

The **RSA** scheme is a cipher in which the plaintext and ciphertext are integers between 0 and n - 1 for some n. A typical size for n is 1024 bits, or 309 decimal digits. That is, n is less than 2^{1024}.

Description of the Algorithm

RSA makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to $\log_2(n) + 1$.

Encryption and decryption for some plaintext block M and ciphertext block C are:

$$C = M^e \bmod n$$

$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

Both sender and receiver must know the value of n. The sender knows the value of e, and only the receiver knows the value of d. Thus, this is a public-key encryption algorithm with a public key of $PU = \{e, n\}$ and a private key of $PR = \{d, n\}$.

For this algorithm to be satisfactory for public-key encryption, the following requirements must be met.

1. It is possible to find values of e, d, and n such that $M^{ed} \bmod n = M$ for all $M < n$.

2. It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M < n$.

3. It is infeasible to determine d given e and n.

By the first requirement, there is a need to find a relationship of the form

$$M^{ed} \bmod n = M$$

The preceding relationship holds if e and d are multiplicative inverses modulo $\phi(n)\,\phi(n)$, where $\phi(n)\,\phi(n)$ is the Euler totient function. For any prime numbers p, q, $\phi(pq)\,\phi(pq) = (p - 1)(q - 1)$. The relationship between e and d can be expressed as

$$ed \bmod \phi(n)\,\phi(n) = 1$$

This is equivalent to saying

$$ed \equiv 1 \bmod \phi(n)$$

$$d \equiv e^{-1} \bmod \phi(n)$$

That is, e and d are multiplicative inverses mod f(n). Note that, according to the rules of modular arithmetic, this is true only if d (and therefore e) is relatively prime to $\phi(n)$ $\phi(n)$. Equivalently, gcd($\phi(n)\,\phi(n)$, d) = 1.

The ingredients of the RSA schene are the following:

p, q, two prime numbers	(private, chosen)
$n = pq$	(public, calculated)
e, with gcd($\phi(n)\,\phi(n)$, e) = 1; 1 <e <$\phi(n)\,\phi(n)$	(public, chosen)
$d \equiv e^{-1} \bmod \phi(n)$ $d \equiv e^{-1} \bmod \phi(n)$	(private, calculated)

The private key consists of {d, n} and the public key consists of {e, n}. Suppose that user A has published its public key and that user B wishes to send the message M to A. Then B calculates $C = M^e \bmod n$ and transmits C. On receipt of this ciphertext, user A decrypts by calculating $M = C^d \bmod n$.

Key Generation by Alice	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calcuate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Public Key	
Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Figure: The RSA Algorithm

Example:

The keys were generated as follows.

1.Select two prime numbers, $p = 17$ and $q = 11$.

2.Calculate $n = pq = 17 * 11 = 187$.

3. Calculate $\phi(n)\,\phi(n) = (p - 1)(q - 1) = 16 * 10 = 160$.

4.Select e such that e is relatively prime to $\phi(n)\,\phi(n) = 160$ and less than $\phi(n)\,\phi(n)$; choose $e = 7$.

5. Determine d such that $de \equiv 1\ (mod\ 160)$ $de \equiv 1\ (mod\ 160)$ and $d < 160$. The correct value is $d = 23$, because $23 * 7 = 161 = (1 * 160) + 1$; d can be calculated using the extended Euclid's algorithm.

The resulting keys are public key $PU = \{7, 187\}$ and private key $PR = \{23, 187\}$.

Assume $M = 88$,

For encryption,

Calculate $C = 887 \bmod 187$. Exploiting the properties of modular arithmetic, this can be done as follows.

$88^7 \bmod 187 = [(88^4 \bmod 187) \times (88^2 \bmod 187) \times (88^1 \bmod 187)] \bmod 187$

$88^1 \bmod 187 = 88$

$88^2 \bmod 187 = 7744 \bmod 187 = 77$

$88^4 \bmod 187 = 59{,}969{,}536 \bmod 187 = 132$

$88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894{,}432 \bmod 187 = 11$

For decryption,

Calculate $M = 11^{23} \bmod 187$:

$11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$

$11^1 \bmod 187 = 11$

$11^2 \bmod 187 = 121$

$11^4 \bmod 187 = 14{,}641 \bmod 187 = 55$

$11^8 \bmod 187 = 214{,}358{,}881 \bmod 187 = 33$

$11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187$

$= 79{,}720{,}245 \bmod 187 = 88$

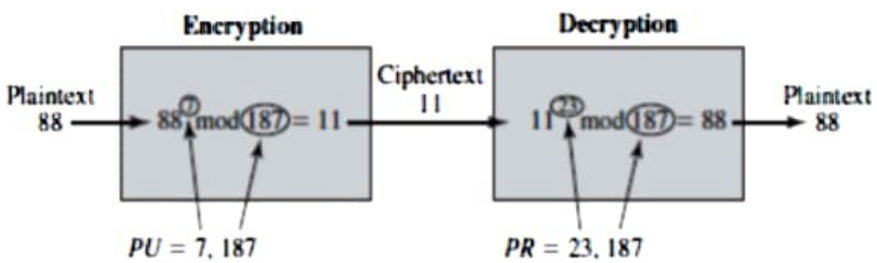

Figure: Example of RSA Algorithm

Computational Aspects

There are actually two issues to consider:

- encryption/decryption and
- key generation.

Exponentiation in Modular Arithmetic

Both encryption and decryption in RSA involve raising an integer to an integer power, mod n. If the exponentiation is done over the integers and then reduced modulo n, the intermediate values would be gargantuan. The property of modular arithmetic is:

$$[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$$

Intermediate results modulo n can be reduced.

Another consideration is the efficiency of exponentiation, because with RSA, we are dealing with potentially large exponents. Consider compute x^{16}. A straightforward approach requires 15 multiplications:

$$x^{16} = x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x$$

More generally, suppose we wish to find the value $a^b \bmod n$ with a, b, and m positive integers. If we express b as a binary number $b_k b_{k-1} \ldots b_0$, then we have

$$B = \sum_{b_i \neq 0} 2^i$$

Therefore,

$$a^b = a^{(\sum_{b_i \neq 0} 2^i)} = \prod_{b_i \neq 0} a^{(2^i)}$$

$$a^b \bmod n = \left[\prod_{b_i \neq 0} a^{(2^i)}\right] \bmod n = \left(\prod_{b_i \neq 0} [a^{(2^i)} \bmod n]\right) \bmod n$$

```
c ← 0; f ← 1

for i ← k down to 0

    do c ← 2 × c

       f ← (f × f) mod n

    if b_i = 1

      then c ← c + 1

           f ← ( f × a) mod n

return f
```

Figure: Algorithm for computing a^b mod n

Key Generation

Before the application of the public-key cryptosystem, each participant must generate a pair of keys. This involves the following tasks.

- Determining two prime numbers, *p* and *q*.
- Selecting either *e* or *d* and calculating the other.

Primes p and q must be chosen from a sufficiently large set. The procedure that is generally used is to pick at random an odd number of the desired order of magnitude and test whether that number is prime. If not, pick successive random numbers until one is found that tests prime.

The procedure for picking a prime number is as follows.

1. Pick an odd integer *n* at random (e.g., using a pseudorandom number generator).
2. Pick an integer $a < n$ at random.
3. Perform the probabilistic primality test, such as Miller-Rabin, with *a* as a parameter. If *n* fails the test, reject the value *n* and go to step 1.
4. If *n* has passed a sufficient number of tests, accept *n*; otherwise, go to step 2.

The Security of RSA (MAY/JUNE 2007)

Five possible approaches to attacking the RSA algorithm are

- **Brute force:** This involves trying all possible private keys.
- **Mathematical attacks:** There are several approaches, all equivalent in effort to factoring the product of two primes.
- **Timing attacks:** These depend on the running time of the decryption algorithm.
- **Hardware fault-based attack:** This involves inducing hardware faults in the processor that is generating digital signatures.
- **Chosen ciphertext attacks:** This type of attack exploits properties of the RSA algorithm.

The Factoring Problem (MAY/JUNE 2012)

Three approaches can be identified to attacking RSA mathematically.

1. Factor n into its two prime factors. This enables calculation of $\phi(n)\,\phi(n)$ = (p - 1) × (q - 1), which in turn enables determination of $d \equiv e^{-1}(mod\ \phi(n))\ d \equiv e^{-1}(mod\ \phi(n))$
2. Determine $\phi(n)\,\phi(n)$ directly, without first determining p and q. Again, this enables determination of $d \equiv e^{-1}(mod\ \phi(n))\ d \equiv e^{-1}(mod\ \phi(n))$
3. Determine d directly, without first determining $\phi(n)\,\phi(n)$.