Chapter 10

Asymmetric - key Cryptography

We actually believe that the Symmetric key Cryptography & Asymmetric key cryptography are compliment of each other But they exist in parallel.

The conceptual differences between the two Sytem are based on how these system keep a "Secret".

In Symmetric - key cryptography, the "Secret" must be Shared between two person.

In assymmetric - key cryptography the "secret" is personal or unshared, each person creates and keeps his or her own secret.

In a community of n people, $n(n-1)/2$ Shared "Secrets" are needed for Symmetric key Cryptography.

But only n personal secrets are needed in asymmetric - key cryptography.

In Symmetric-key cryptography, the plaintext and ciphertext are thought of as a combination of symbols. Encryption and decryption permute these symbols or substitute a symbol for another.

In Asymmetric-key cryptography, the plaintext and ciphertext are numbers. Encryption & Decryption are mathematical functions that are applied to numbers to create other numbers.

Keys :- Asymmetric key cryptography uses two separate keys: one private and one public.

If Encryption and decryption are thought of as locking & unlocking padlocks with keys, then the padlock is locked with the public key & can be unlocked only with the corresponding private key.

General Idea :- Unlike Symmetric key cryptography there are distinctive keys in ansymmetric-key-cryptography: a private key & a public key.

We believe that the nature of the secret key used in Symmetric-key cryptography is different from the nature of the ~~asy~~ private key used in asymmetric key cryptography.

~~Because the first is~~

Because the secrete key is in Symmetric key cryptography is Symbols (string)
But in Asymmetric key cryptography the Private key is Numbers.

Some important facts about Asymmetric key cryptography :-

(1) The Burden of providing security is mostly on the shoulders of the receiver.
Receiver needs to create two keys : one private and one public.
Receiver is responsible for distributing the public key to the community.
This can be done through a "public key distribution channel". Although this channel is not required to provide secrecy But it is Required to provide authentication and integrity. So that the attacker do not change the public key.

(2) Asymmetric key cryptography means that Sender & receiver cannot use the same set of keys for two way communication.
Each entity in the community should create its own private and public keys.
It means that Sender(Alice) can use receivers(Bob) Public key for Encryption.
But if the receiver(Bob) wants to respond then the Sender (Alice) needs to creates his public & private key.

(8) Asymmetric-key cryptography means that Bob (receiver) needs only one private key to receive all correspondence from anyone in the community.
But Alice (sender) needs n public keys to communicate with n entities in the community. one public key for each entity.

Plaintext / ciphertext :- Plaintext and ciphertext are treated as integers in asymmetric key cryptography.

The message must be encoded as an integer before encryption.
The integer must be decoded into the message after decryption.
Asymmetric key cryptography is normally used to encrypt or decrypt small pieces of information.

In other words, asymmetric key cryptography normally is used for auxillary goals (like the cipher key for symmetric key cryptography) instead of message encipherment.

Encryption / Decryption:-

Encryption & Decryption in asymmetric key cryptography are mathematical functions applied over the numbers representing the plaintext and ciphertext.

The ciphertext can be thought of as

$$C = f(K_{public}, P)$$

The plaintext can be thought of as

$$P = g(K_{private}, C)$$

Need of Both Symmetric & Asymmetric Encipherment:-

The ~~dd~~ advent of asymmetric key (Public key) cryptography does not eliminate the need for Symmetric key (Secret-key) cryptography.

The reson is that ~~asymmetr~~ asymmetric-key cryptography, which uses mathematical functions ~~for~~ encryption & decryption is much slower then the Symmetric-key cryptography.

for Encipherment of large message Symmetric key cryptography is still needed.

On the other hand Because of slow speed we can't eliminate the need of Asymmetric key cryptography.

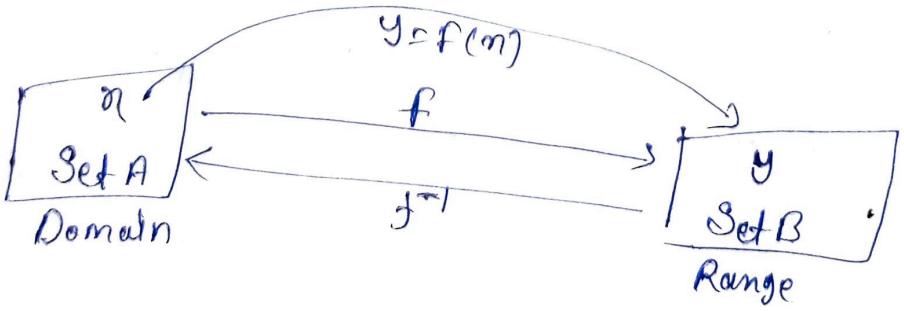
A Asymmetric key cryptography is still needed for authentication., digital signatures, Secret key Exchanges.

This means that to be able to all aspects of today, we need both Symmetric & Asymmetric key cryptography

Trapdoor One-way function :- The main idea behind asymmetric-key cryptography is the concept of the "trapdoor one way function."

function :- A function is a rule that associates (maps) one element in Set A, called the "Domain" to one element in set B called the range.

A invertible function is a function that associates each element in the range with exactly one element in domain.



One-way-Function :- A one-way function is a function that satisfies the following two properties:

(1) f is easy to compute

(2) f^{-1} is difficult to compute.

Trapdoor One-way-function :-

It is a function with third property :-

(3) Given y & a trapdoor (secret), x can be computed easily.

Knapsack Cryptosystem

The first brilliant idea of Public-key cryptography came from Merkle and Hellman, in their knapsack cryptosystem. Although this system was found to be insecure with today's system.

The key Idea of cryptosystem is :-

~~Suppose~~

It is based on the knapsack problem, a well known computational problem in computer science.

→ Suppose we have set of integers (weights of item). If it is given that which numbers are included in the knapsack. ~~And ask~~ And someone ask for sum, it is easy to calculate.

→ But given the total weight, identifying which specific numbers were used to produce that sum is difficult. This is the crux of the Problem.

Definition :- Suppose we are given two k-tuples

$a = [a_1, a_2 \ldots a_k]$ and $n = [n_1, n_2 \ldots n_k]$

the first tuple is predefined set, the second tuple in which n_i is only 0 or 1 defines which element of 'a' are to be dropped in the knapsack.

The sum of elements in the knapsack is:-

$$S = \text{knapsackSum}(a, n) = n_1 a_1 + n_2 a_2 \ldots n_k a_k$$

Given a & n it is easy to calculate S. However, given S & a it is difficult to find n.

In otherwords $S = \text{knapsackSum}(n, a)$ is easy to calculate, but $n = \text{inv\_knapsackSum}(S, a)$ is difficult

The function ~~knapsackSum is a one-way~~ function if

Superincreasing Tuple :- Given a and n it is easy to calculate s

Superincreasing Tuple :- It is easy to compute KnapsackSum and inv_KnapsackSum If the k-tuple a is superincreasing.

In a Superincreasing tuple, $a_i \geq a_1 + a_2 \cdots a_{i-1}$.
In other word each element is greator than or equal to the sum of all prevous Elements.

In this case we can calculate KnapsackSum and inv_KnapsackSum.

The algorithm inv_KnapsackSum Starts from the largest element and proceed to smallest one. In each iteration, It checks whether an element is in the knapsack.

Ex:- a= [17, 25, 46, 94, 201, 400] and s=272

i	a_i	s	$s \geq a_i$	x_i	$s \leftarrow s - a_i * x_i$
6	400	272	false	0	272
5	201	272	true	1	71
4	94	71	false	0	71
3	46	71	true	1	25
2	25	25	true	1	0
1	17	0	false	0	0

In this case x= [0, 1, 1, 0, 1, 0] means 25, 46, 201 are in knapsack.

Secret Communication with Knapsacks :-

Steps in this

→ Key Generation :-

(a) Create a superincreasing k-tuple
$b = [b_1, b_2 \ldots b_K]$

(b) Choose a modulus n, such that
$n > b_1 + b_2 + \ldots + b_K$

(c) Select a random integer r that is relatively prime with n & $1 \leq r \leq n-1$

(d) Create a temporary k-tuple $t = [t_1, t_2, \ldots t_k]$ in which $t_i = r \times b_i \bmod n$

(e) Select a permutation of K objects and find a new tuple $a = \text{permute}(t)$

(f) The public key is k tuple a,
The private key is n, r and the k-tuple b.

→ Encryption :-

Suppose Alice needs to sends a message to Bob

(a) ~~Alice Alice needs to send a message to Bob~~

(a) Alice converts her message to a k-tuple $x = [x_1, x_2 \ldots x_K]$ in which x_i is either 0 or 1. x is a plaintext.

(b) Alice uses the knapsackSum routine to calculate 's'. She then sends the value of s as the ciphertext

Decryption :-

Bob receives the ciphertext S.

(a) Bob calculates $s' = r^{-1} \times s \bmod n$

(b) Bob uses inv_knapsackSum to create a'.

(c) Bob permutes a' to find a. The Tuple x is recovered plaintext.

Example :-

① key generation :-

(a) Bob creates the superincreasing tuple

$b = [7, 11, 19, 39, 79, 157, 313]$

(b) Bob chooses the modulus $n = 900$ & $r = 37$ and $[4, 2, 5, 3, 1, 7, 6]$ as permutation table.

(c) Now Bob calculate a tuple t

$t = [259 \quad 407 \quad 703 \quad 543 \quad 223 \quad 409 \quad 781]$

(d) Bob calculates the tuple $a = \text{permute}(t)$

$a = [543, 407 \quad 223 \quad 703 \quad 259 \quad 781 \quad 409]$

(e) Bob publicaly announces a, he keeps n, r, b secret.

(8) Suppose Alice wants to send a single character 'g' to Bob.

(a) She uses the 7-bit ASCII of value of g: (1100111). & creates a tuple.

$x = [1, 1, 0, 0, 1, 1, 1]$ this is a Plain text.

(b) Now Alice calculates $s = knapsackSum(a, n) = 2165$ This is the ciphertext sent to Bob

(9) Bob can decrypt the ciphertext $s = 2165$

(a) Bob calculate $s' = s \times r^{-1} \bmod n$

$s' = 2165 \times 37^{-1} \bmod 900 = 527$

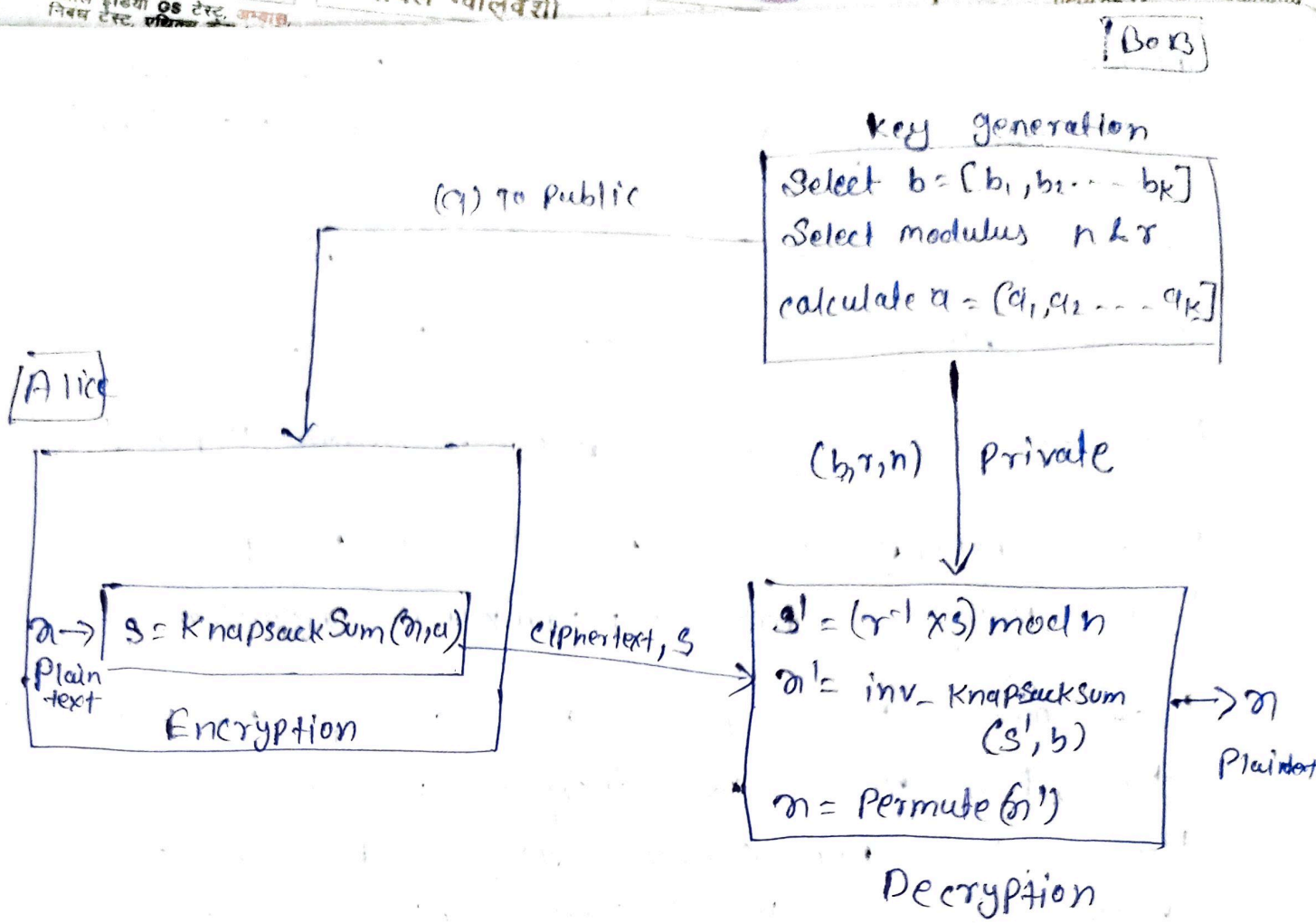
(b) Bob calculates $x' = Inv\_Knapsack\_Sum(s', b)$

$= [1, 1, 0, 1, 0, 1, 1]$

(c) Bob calculates $x = permute(x')$

$x = [1, 1, 0, 0, 1, 1, 1]$.

He interprets the string (1100111) as character 'g'



RSA Cryptosystem

The most common public-key algorithm is the RSA cryptosystem, named for its inventors (Rivest, Shamir & Adleman).

Introduction :- RSA uses two exponents, e & d where e is public and d is private. Suppose P is plaintext & C is a ciphertext. Alice uses $C = P^e \bmod n$ to create ciphertext C & Bob uses $P = C^d \bmod n$ to create plaintext P from ciphertext

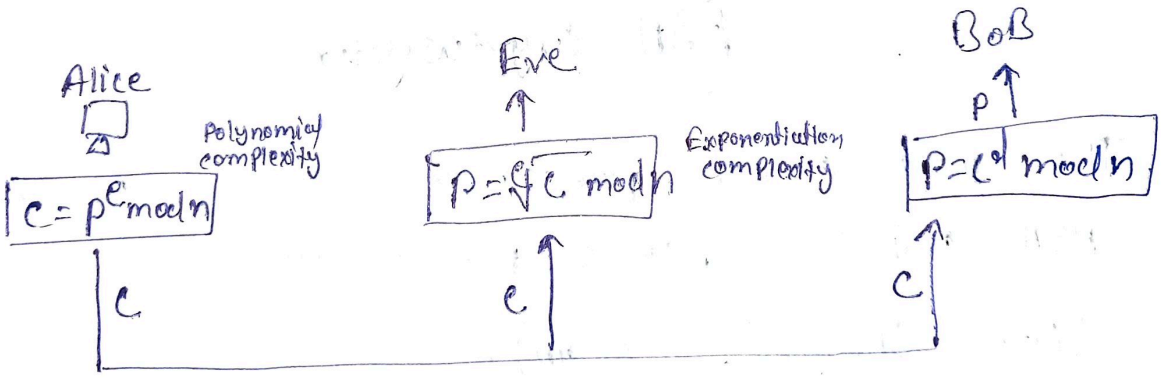
~~the~~ this is possible because e & d are inverse of each other in modulo $\phi(n)$

The modulus n, a very large number, is created during the generation process.

Encryption & Decryption use modular exponentiation

Modular exponentiation is feasible in polynomial time using a fast exponentiation algorithm.

This means that Alice can encrypt in polynomial time, Bob also can decrypt in polynomial time, but Eve cannot decrypt because she would have to calculate e^{th} root of C using modular arithmetic.



Because the Alice uses trapdoor one-way function, know to Bob.
Eve who does not know the trapdoor, cannot decrypt the message.

If some day, a polynomial algorithm for e^{th} root modulo n calculation is found, modular exponentiation is not a one-way function any more.

Steps in RSA :-

① **Key Generation :-** Bob uses ~~the~~ this step to create Public & Private key.

① Select ~~to~~ two large Primes p and q such that $p \neq q$.

② $n = p \times q$

③ ~~$\phi(n) = (p-1) \times (q-1)$~~

~~$\phi(n) = \phi(p-1) \times \phi(q-1) = p$~~

③ $\phi(n) = \phi(p) \times \phi(q) = (p-1) \times (q-1)$

④ Select e such that $1 < e < \phi(n)$ & e is coprime to $\phi(n)$, means inverse is possible $de = 1 \bmod \phi(n)$

⑤ $d = e^{-1} \bmod \phi(n)$

⑥ Public key $= (e, n)$

⑦ Private key $= \{d, n\}$

Encryption :- Alice uses Bob's public key (e, n) for encryption

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

Decryption :- Bob uses It's Private key to decrypt a ciphertext C

$$P = C^d \bmod n$$

this is possible because e & d are inverse of each other in modulo $\phi(n)$ which is unknown to the Eve.

Prove =

$$P = C^d \bmod n$$

$$P = (C \bmod n)^d$$

$$P = (P^e \bmod n)^d$$

$$P = P^{exd} \bmod n$$

$$P = P \bmod n$$

$$P = P$$

Example :-

Bob chooses 7 & 11 as p & q

→ $n = 77$

→ $\phi(n) = 60$

Select e & d in Z_{60}^*

→ let $e = 13$

→ then inverse of e in Z_{60}^* is 37

→ $d = 37$

~~Alice~~ → ~~(37, 77) Public~~ (13, 77) public key ~~Encryption~~

→ (37) is private key

Encryption :-

$P = 5$

then $C = 5^{13} \bmod 77 = 26$

Decryption :-

$P = 26^{37} \bmod 77 = 5$

Attacks on RSA :-

No devastating attacks on RSA have been yet discovered.

Several attacks have been predicted Based on the weak plaintext, weak parameter selection or inappropriate implementation.