5 Public Key Cryptography

Introduction

All the previous modern cryptographic systems depend on the elementary tool of S-P network. However, public key algorithms are based on mathematical functions and are asymetric in nature, involving the use of 2 keys as opposed to conventional key encryption techniques.

The various misconception about public key cryptography are:

1. Public key encryption is more secure from crypto-analysis view than the digital conventional method. In fact, the security of any system depends upon the key length and computational effort required in breaking down cipher.

2. Public key encryption supersedes single key encryption. This is unlikely due to the increased processing power required.

3. Key management is trivial with public key cryptography which is not always correct.

Principles of Public Key Cryptography

The very concept of public key evolve from an ~~append~~ attempt to solve two problems: Key distribution and the development of digital signature. In 1976, White ~~field~~ Diffie and Martin Hellman achieved great success in developing the conceptual framework.

→ It is the system that relies on one key for encryption and another key for decryption. Furthermore, the public key cryptography algorithms have the following properties:

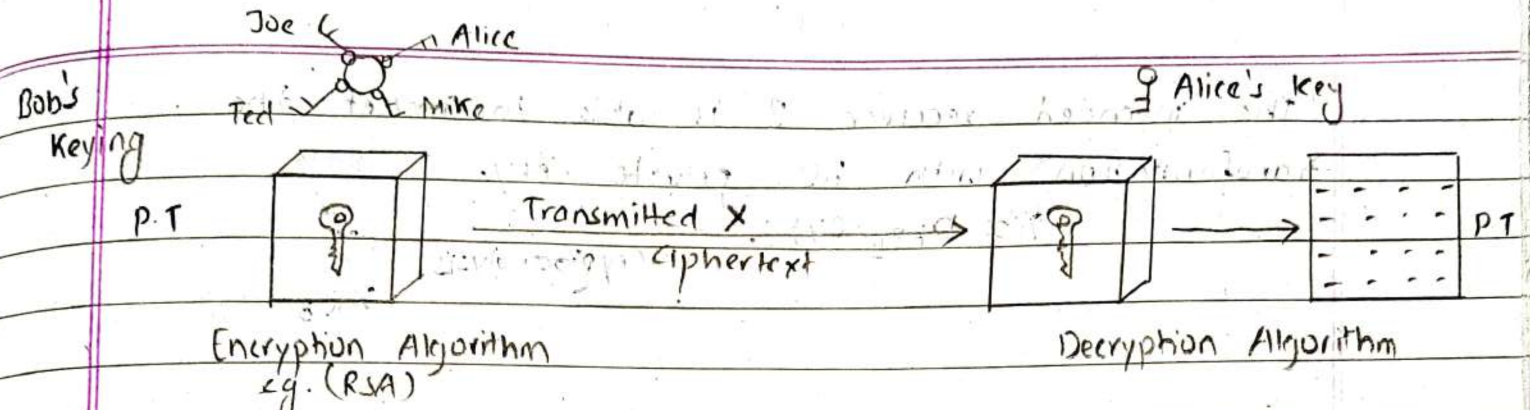
1. It is computationally infeasible to determine the decryption key given only the knowledge of the algorithm and encryption key. In addition, algorithms like RSA has the following characteristics:

a) Either of the two related keys, one can be used for encryption, while the other for decryption.

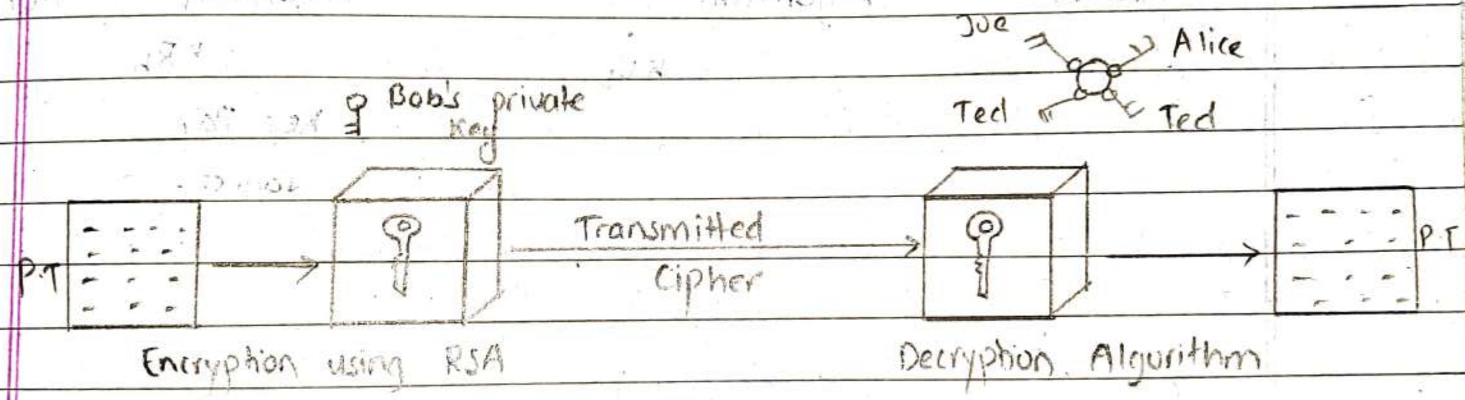
P-K

Steps:

1. Each system generates a pair of keys.
2. Each system publishes its encryption key (public key) keeping its companion key private.
3. If A wishes to send a message to B, it encrypts the message using B's public key.
4. When B receives the message, it decrypts the message using its private key. No one else can decrypt the message as only B knows its private key.



a) Encryption



b) Authentication

Fig : P-K Cryptography

Diffie Hellman Key Exchange

Considering P-K in details, we have a source A that produce PT 'X' destinied for B generates a pair of key KU_b (a public key) and KR_b (a private key) with 'X' and 'KU_b' as input, a cipher text forms as,

$$Y = E_{KU_b}(X)$$

The intended receiver B is able to invert the transformation with his private key.

$$X = D_{KR_b}(Y)$$

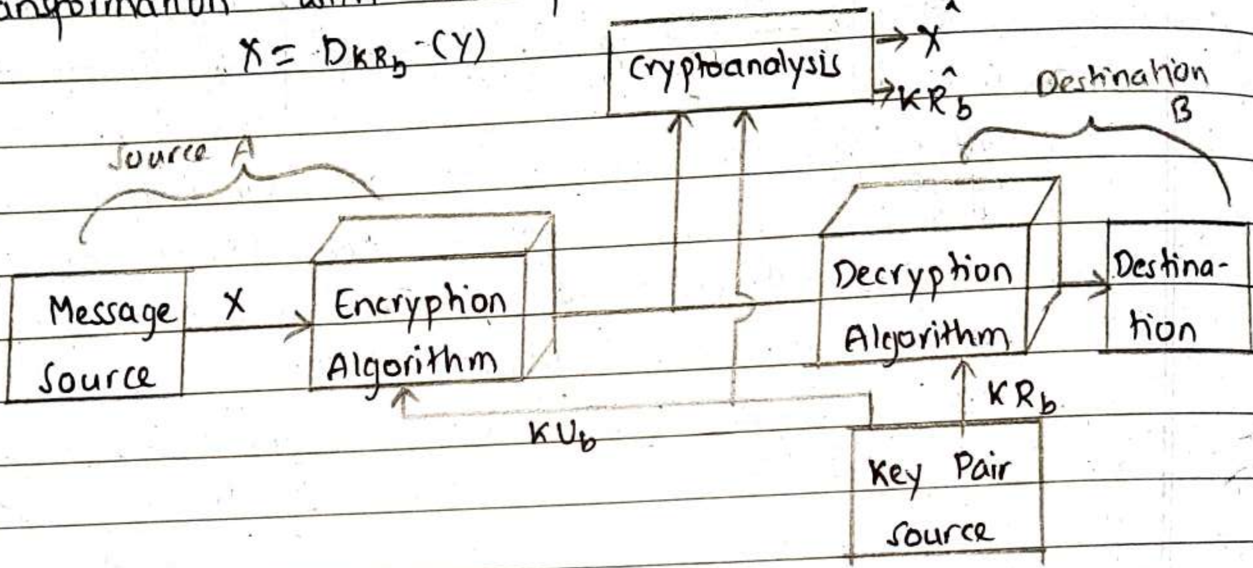


Fig: P-K Cryptography (Secrecy)

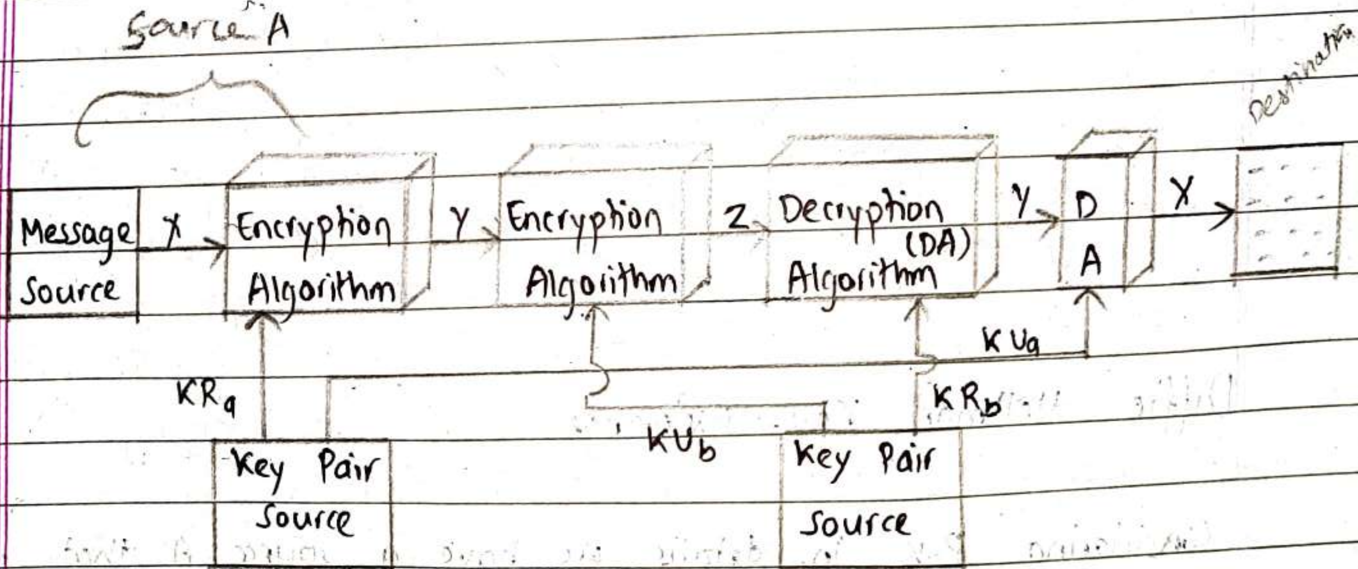


Fig: P-K Cryptography Secrecy & Authentication

Diffie Hellman Key Exchange Algorithm

→ 1st published P-K algorithm by Diffie Hellman

→ Diffie & Hellman first coined the term Public Key Cryptography.

→ It is limited to the secure exchange of a secret key and not of a message.

→ The security of the scheme depends on the difficulties of computing discrete logarithm.

→ The Diffie Hellman key exchange consists of two publically known numbers:

a) a prime number q and

b) an integer 'α' which is a primitive root of q and $\alpha<q$.

let user a and b wish to ~~extends~~ exchange a key. User A selects a random integer X_A which is less than q.

i.e. $X_A < q$

and calculates

$$Y_A = \alpha^{X_A} \bmod q$$

Similarly, user b selects a random integer X_B and computes

$$Y_B = \alpha^{X_B} \bmod q$$

Here, each side keeps X's values private and Y's values are publically known to the other side. Now, the user at side A computes the key as

$$K = (Y_B)^{X_A} \bmod q$$

Similarly, user B calculates the key as: $K = (Y_A)^{X_B} \bmod q$

These two calculation produces identical results and the result is that the two sides have exchanged a secret key. This has been shown mathematically in the section below:

$$K = (Y_B)^{X_A} \bmod q$$
$$= (\alpha^{X_B} \bmod q)^{X_A} \bmod q = (\alpha^{X_B})^{X_A} \bmod q$$
$$= (\alpha^{X_A} \bmod q)^{X_B} \bmod q$$
$$= (Y_A)^{X_B} \bmod q$$

Furthermore, X_A and X_B are private, the hacker or opponent is forced to take a logarithm to determine the key. For eg: for attacking the key of user B, one must compute:

$$X_B = ind_{\alpha,q}(Y_B)$$

where,

$ind_{\alpha,q}$ is the discrete logarithm of index Y_B for base α mod q.

The Diffie Hellman key exchange ~~key~~ algorithm can be summarized as follow:

Global Public Element

$q \leftarrow$ prime number

$\alpha \leftarrow \alpha < q$, α is primitive root of q

$\downarrow$

User A Key Generation

Select Private Key

$X_A \longrightarrow X_A < q$

Calculate public Key

$Y_A \longrightarrow Y_A = \alpha^{X_A} \bmod q$

$\downarrow$

User B Key Generation

Select private Key

$X_B \longrightarrow X_B < q$

Calculate public Key.

$Y_B \rightarrow Y_B = \alpha^{X_B} \bmod q$

$\downarrow$

Generation of secret Key by A

$K = (Y_B)^{X_A} \bmod q$

Generation of secret Key by B

$K = (Y_A)^{X_B} \bmod q$

Q. let $q=11$ and $\alpha=2$ i.e. primitive root $(\alpha)=2$. Now, let the private key of A is 8 i.e. $8<11$. So, $X_A=8$ and the private key of B is 4. i.e. $4<11$. $4<11$.

Sol^n

Here, $q=11$ & $\alpha=2$

& $X_A=8$, $X_B=4$

Now,

$$Y_A = \alpha^{X_A} \bmod q$$
$$= 2^8 \bmod 11$$
$$= 3$$

& $$Y_B = \alpha^{X_B} \bmod q$$
$$= 2^4 \bmod 11$$
$$= 5$$

Then,

$$K = (Y_B)^{X_A} \bmod q$$
$$= 5^8 \bmod 11$$
$$= 4$$

& $$K = (Y_A)^{X_B} \bmod q$$
$$= 3^4 \bmod 11$$
$$= 4$$

RSA Algorithm

→ Developed by Ron Rivest, Adi Shamir and Len Adelman at MIT in 1978.

→ Most widely accepted and implemented general purpose P-K encryption.

→ Block cipher

→ Plaintext and ciphertext are the integers between 0 & n-1 for some n.

→ The scheme makes use of exponentiation. For some plaintext (M) and ciphertext (C), we have

$$\left.\begin{aligned} C &= M^e \bmod n \\ M &= C^d \bmod n \\ &= (M^e \bmod n)^d \bmod n \\ &= M^{ed} \bmod n \end{aligned}\right\} \text{—— (1)}$$

→ Both the senders and receivers know 'n'. The sender knows the value 'e' only and the receiver knows the value 'd' only.

$$\left.\begin{aligned} K_U &= (e, n) \\ K_R &= (d, n) \end{aligned}\right\} \text{—— (2)}$$

For this algorithm to be satisfactory for P-K encryption, the following condition are to be met:

1. It is possible to find e, d, n values such that
$$M^{ed} = M \bmod n \quad \text{for } (m>n)$$
2. It is relatively easy to calculate M^e & C^d for all values of $m>n$.
3. It is infeasible to determine d from given e and n.

Focusing on the first requirement, we need to find the relationship of the form

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

Recalling Euler theorem, which states that

$$\left.\begin{array}{l} a^{\phi(m)} = 1 \pmod m \\ \gcd(a, m) = 1 \end{array}\right\} \text{ --- } (3)$$

There is a corollary to this theorem that can be used to produce the required relationship.

Given two large primes p & q and integer n = p.q an m with 0 < m < n. the following holds

$$M^{\phi(n+1)} = M^{(p-1)(q-1)+1}$$

$$= M \bmod n \text{ --- } (4)$$

If $\gcd(m, n) = (1 \bmod n)^m$

Then, this holds by virtue of this theorem.

If $\gcd(m, q) = 1$,

Then Euler's theorem holds and

$$M^{\phi(q)} = 1 \bmod q$$

But, by Modular Arithmetic,

$$[M^{\phi(q)}]^{\phi q} = 1 \pmod q$$

$\therefore$ There is some integer k such that

$$M^{\phi(n)} = 1 + kn$$

$$\text{or, } M^{\phi(n)} \cdot M = M + MKn$$

$$\text{or, } M^{\phi(n)+1} = M \bmod n.$$

→ A similar reasoning is used for the case in which m is multiple of q so, equation (4) is proved.

→ An alternative form of this collary is relevant:

$$M^{k\phi(n)+1} = ([M^{k\phi(n)}] * m) \pmod n$$

$$= [1^k * m] \pmod n \quad \text{[By Euler's Theorem]}$$

$$= M \bmod n.$$

→ Now, we can state the RSA scheme.

→ The ingredients are the following:

p & q : two primes (privately chosen)

e : $\gcd(\phi(n), e) = 1$ & $1 < e < \phi(n)$, publicly chosen

d : $d \equiv e^{-1} \bmod \phi(n)$. (private calculation)

Private Key : (d, n)

Public Key : (e, n)

Encryption:

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

Decryption:

$$M = C^d \pmod n$$

RSA Algorithm Key Generation

1. Select two large primes p & q
2. Calculate $n = pq$
3. Calculate $\phi(n) = (p-1).(q-1)$
4. Select an integer e $\begin{bmatrix} \gcd(\phi(n), e) = 1 \\ \& \ 1 < e < \phi(n) \end{bmatrix}$
5. Calculate $d = e^{-1} \bmod \phi(n)$
6. Public key : $K_u = (e, n)$
7. Private key : $K_R = (d, n)$

Q. 1. Take $p = 11$ & $q = 17$.

Sol^n.

Here $p = 11$ & $q = 17$

so, $n = pq = 11 \times 17$

$= 187$

And, $\phi(n) = (p-1).(q-1)$

$= (11-1).(17-1)$

$= 160$

To find e, $\gcd(\phi(n), e) = 1$; $1 < e < \phi(n)$

ie $\gcd(160, e) = 1$ ie. $1 < e < 160$

$\therefore e = 7$ (chosen)

To find d, $d = e^{-1} \bmod \phi(n)$ $7d \equiv 1 \bmod 160$

$= 7^{-1} \bmod 160$

$= 23$

160	160
e = 7	1
7×22	1×22
160 − 7×22 = 6	160 − 1×22 = 138

160/7 = 22.85

7	1
6	138
7 − 1×6 = 1	1 − 1×138 = −137
	−137 + 160 = 23

∴ d = 23

calculation for d.

∴ Public key : (e, n) = (7, 187)
& Private key : (d, n) = (23, 187)

For encryption & decryption,

Enc: let M = 5

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

$= 5^7 \bmod 187$

$= 146$

Dec:

$M = C^d \bmod n$

$= 146^{23} \bmod 187$

$= 5$

$2^{11} \cdot 2$

$185 \cdot 185 \cdot 185 \cdot ... \quad 185^{10} \cdot 146$

$= (185^2)^5 \cdot 146$

$45 \cdot 146$

$89 \cdot 146$

91

2. Take $p = 89$, $q = 107$

Sol^n =

$n = pq = 89 \times 107 = 9523$

$\phi(n) = (p-1) * (q-1) = (89-1) * (107-1) = 9328$

To find e, $\gcd(\phi(n), e) = 1$; $1 < e < \phi(n)$

ie $\gcd(9328, e) = 1$

$\therefore e = 3$ (chosen)

$x \cdot 9328$	9328
3	1
$x - 3 \times 3109$	$x - 1 \times 3109$
$= 1$	6219

To find d, $d = e^{-1} \bmod \phi(n)$

$= 3^{-1} \bmod 9328$

$= 6219$

$\therefore$ Public Key : $(e, n) = (3, 9523)$

Private Key : $(d, n) = (6219, 9523)$

For encryption:

$C = M^e \bmod n$

For decryption:

$M = C^d \bmod n$

Security of RSA

To break the security of RSA, there may be three possible approaches.

1. Brute Force Attack

Try all the possible key since there exits a large key space. So, the longer the values 'e' and 'd' are, we may have the following possibilities:

	5 years ago	Today
Casual use	384 bits	768 bits
Commercial use	512 bits	1024 bits
Military Specification	1024 bits	4096 bits

2. Mathematical Attack

Factor the value n into two primes enabling the calculation of $\phi(n)$ and the prime key $e \equiv d^{-1} \bmod \phi(n)$. The best known algorithm used in factorizing the integer n takes time proportional to ~~[illegible]~~

$$u_n = e^{\sqrt{\ln(n)\ln(\ln(n))}}$$

i.e. for ~~1~~200 digits, this would take 1000 years approx. on a larger machine.

3. Timing Attack

These attacks depend on the running time of the algorithm.