NUMBER THEORY AND PUBLIC KEY CRYPTOGRAPHY

Syllabus: Number Theory: Prime and Relatively Prime Numbers, Modular Arithmetic, Fermat's and Euler's Theorems, The Chinese Remainder theorem, Discrete logarithms.

Public Key Cryptography: Principles, public key cryptography algorithms, RSA Algorithms, Diffie Hellman Key Exchange, Elgamal encryption & decryption, Elliptic Curve Cryptography.

NUMBER THEORY

Prime Numbers: An integer P>1 is a prime number if its only divisors are ±1 and ±p. Prime numbers play a critical role in number theory.

Relatively Prime Numbers: The integers a and b are relatively prime if they have no prime factors in common, that is if their only common factor is 1. This means, a and b are relatively prime if gcd(a,b)=1.

Modular Arithmetic: For any positive integer n and any integer a, if we divide a by n, we get a quotient q and a remainder r:

$$a = q*n+r \qquad 0\leq r<n;\ \text{and}\ q=[a/n]$$

Ex: 11 mod 7=4 for a=11,n=7 11=1*7+4; r=4 and q=1

Two integers a and b are said to be congruent modulo n if(a mod n)=(b mod n). This is like a= b mod n. The mod operator has the following properties.

1. a= b mod n if n|(a-b)
2. (a mod n)=(b mod n) implies b= a mod n
3. a=b mod n implies b=a mod n
4. a=b mod n and b=c mod n implies a=c mod n

Modular arithmetic Operations: The (mod n) operator maps all integers into the set of integers [0,1,2,...(n-1)]. Modular arithmetic exhibits the following properties.

1. [(a mod n) + (b mod n)]mod n=(a+b) mod n
2. [(a mod n) - (b mod n)]mod n=(a-b) mod n
3. [(a mod n) * (b mod n)]mod n=(a*b) mod n
4. (a+b) mod n=(b+a) mod n (commutative)
5. (a*b) mod n=(b*a) mod n (commutative)
6. [(a+b)+c] mod n=[a+(b+c)] mod n (Associative)

7. [(a*b)*c] mod n=[a*(b*c)] mod n (Associative)
8. [a*(b*c)] mod n=[(a*b)*(a*c)] mod n (Distributive)
9. (0+a)mod n=a mod n (identity)
10. (1*a) mod n= a mod n (identity)
11. For each a belongs to Za. There exists a Z such that a+z=0 mod n (additive inverse)

Additive Inverse Modulo:

Ex: Find -15 mod 26=9 (26-9=15)

Multiplicative Inverse Modulo: Given two integers 'a' and 'm', find modular multiplicative inverse of 'a' under modulo 'm'. The modular multiplicative inverse is an integer 'x' such that.

a x ≡ 1 (mod m) ; The value of x should be in {0, 1, 2, … m-1}, i.e., in the ring of integer modulo m. The multiplicative inverse of "a modulo m" exists if and only if a and m are relatively prime (i.e., if gcd(a, m) = 1).

Ex: Find 3^{-1}mod 11

1*3 mod 11 !=1

2*3 mod 11!=1

3*3 mod 11 !=1

4*3 mod 11 =1

So, 3^{-1}mod 11=4

EX: Find 5^{-1}mod 96

1*5 mod 96 !=1

2*5 mod 96 !=1

3*5 mod 96 !=1

4*5 mod 96!=1

…..

77*5 mod 96=1

S0, 5^{-1}mod 96=77

Reciprocal Modulo: if a and n are relatively prime numbers, then it is possible to find the reciprocal modulo using Euclidian algorithm, that is 1/a mod n can be identified if gcd(a,n)=1

Ex: 1/12 mod 41

Gcd(12,41)=1, so

1=41*5-12*17 by reducing this ,

-12*17=1 mod 41

Therefore 1/12 mod 41=-17

That is -17 mod 41=24

Ex: Compute 4/9 mod 11

First Find 1/9 mod 11,

Gcd(9,11)=1, so

1=5*9-11*4 by reducing this,

5*9=1 mod 11

➔1/9 mod 11=5

So, 4/9 mod 11=4*1/9 mod 11=4(1/9 mod 11)=4*5=20=20 mod 11=9

Exponentiation Modulo: a^{b} mod c = ((a mod c)^{b}) mod C Often we calculate a^b mod c for large values of b. Unfortunately, a^b becomes very large for even modest sized values for b.

Ex: 2^{90}mod 13= ((2 mod 13)^{90}) mod 13 (or)

2^90 mod 13 = (2^50 * 2^40) mod 13

2^90 mod 13 = (2^50 mod 13 * 2^40 mod 13) mod 13

2^90 mod 13 = (4 * 3) mod 13

2^90 mod 13 = 12 mod 13

2^90 mod 13 = 12

Fermat's Theorem: Fermat's theorem plays a vital role in cryptography. There are two versions of Fermat's theorem.

First Version: If P is prime and a is a positive integer which is not divisible by P, then

a^{P-1}=1 mod pØØØ

Proof: As we know, if all the elements of Z_P are multiplied by a, modulo p, then the result contains all the elements of Z_P in some order. Furthermore ax0=0 mod p. therefore,

1. The (p-1) numbers [a mod p,2a mod p,... (p-1)a mod p] are just the numbers between [1,2,...(p-1)] in some order.
2. Multiply these numbers together

   (ax2ax....x(p-1)a)=[(a mod p)x(2a mod p)x...x((p-1)a mod p)] mod p =(p-1)! mod p

   But, (ax2ax3ax...x(p-1)a) = (p-1)!a^{p-1}

   Therefore, (p-1)!a^{p-1}=(p-1)! mod p , cancel (p-1)! Term because it is relatively prime to p.

   which yields a^{p-1}= 1 mod p

Second Version: This version removes the condition on a. it says that if P is prime and a is an integer then a^{P}=a mod P

Applications:

1. **Exponentiation:** It is very helpful to do fast exponentiation.

   Ex: 6^{10} mod 11 =1(since 10=11-1 , using a^{p-1}= 1 mod p)

   3^{12}mod 11 =(3^{11}x3) mod 11=(3^{11}mod 11)x(3 mod 11)

   = (3 x 3) mod 11

   =9 mod 11=9

2. **Multiplicative Inverses**: A very interesting application of Fermat's theorem is in finding some multiplicative inverses quickly if the modulus is a prime. If p is prime, and a is an integer such that p does not divide a, then a^{-1}mod p=a^{p-2}mod p

   This can be easily proved if we multiply both sides with a and the first version of Fermat's theorem can be used.

   axa^{-1} mod p = axa^{p-2}mod p

   →1 mod p = a^{p-1}mod p

   ➔ 1=a^{p-1}mod p

   Ex: 8^{-1} mod 17 =8^{17-2} mod 17=8^{15}mod 17= 15 mod 17=15

   5^{-1}mod 23 = 5^{23-2}mod 23 = 5 ^{21}mod 23 =14 mod 23=14

Euler's Totient Function: Another important result of number theory that is useful in understanding public key systems is Euler‟s theorem. Before going into details of statement and examples, we introduce totient function.

Given an integer n > 1, the number of integers less than n and prime to n is called Euler‟s totient function denoted by $\phi(n)$. $\phi(1)$ is taken to be 1, $\phi(p)$ = p-1 and

If n= p × q (p, q are primes) then $\phi(n) = \phi(p) \times \phi(q) = (p-1) \times (q-1)$

Ex: n=37 (prime number). As n is prime all numbers less than n (36 in number) are prime to n thus $\phi(37)$ =37-1= 36. For any prime p, $\phi(p)$ = p-1

Ex: $\phi(35)$ = 24.

This can be written as $\phi(7x5)= \phi(7)\ x\phi(5)$

=(7-1)x(5-1)

= 6x4

= 24

This can also be written as the numbers less than 35 and prime to 35 are 1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34 and there are 24 numbers in this list.

Eulers Theorem: The modulus in the Fermat theorem is a prime, whereas here it is an integer. There are two versions of this algorithm.

First Version: The first version of Eulers theorem is similar to the first version of the Fermat's theorem.

If a and n are co-prime, then $a^{Ø(n)}$= 1 mod n

Ex: Let a=3, n=10; $\phi(10) = 4$, 34 =81 ≡ 1 mod 10

Alternative form of this theorem is as follows: For any two integers a, n we have a $^{\phi(n)+1}$ ≡ a mod n. (Note that a and n need not be relatively prime.

Ex: Let a=3, n=10; $\phi(10) = 4$, a ϕ(n)+1 = 3 5 = 243 ≡ 3 mod 10 = a mod n

In this example a and n are relatively prime.

Ex: a=3, n=6; $\phi(6)=2$, a ϕ(n)+1 = 3 3 = 27 ≡ 3 mod 6 = a mod n

In this example a and n are not relatively prime.

Applications:

1. **Exponentiation:** Helpful for quickly finding a solution to some exponentiation.

   Ex: 6^{24}mod 35

   Since $\phi(35)= \phi(7x5)= \phi(7)x\phi(5)=(7-1)x(5-1)=6x4=24$

   According Euler's first version $a^{Ø(n)}$= 1 mod n, 6^{24}mod 35=1

   Ex: 20^{62}mod 77

   Since $\phi(77)= \phi(11x7)= \phi(11)x\phi(7)=(11-1)x(7-1)=10x6=60$

   Therefore according to Euler's second version a $^{\phi(n)+1}$ ≡ a mod n,

   a $^{\phi(n)+1}$ ≡ a $^{\phi(n)}$x a^1 = a $^{\phi(77)}$ x a^1

   =20x20 mod 77

   =400 mod 77

   =15

2. **Multiplicative Inverse**: Euler's theorem can be used to find multiplicative inverses modulo a prime. Euler's theorem can be used to find multiplicative inverses modulo a composite. If the integers a and n are co-prime, then a^{-1} mod n= a $^{\phi(n)-1}$ mod n

   Ex: 5^{-1} mod 96 =77

   4x96+1/5=0

   Therefore, the quotient value is 77.

Prime Number Generation Algorithms: For many cryptographic algorithms we need large prime numbers. So, we generate a random number and then test if it is prime. Deterministic algorithm for prime number test is complex (check if any number in the range of 2 to √n divides n; if so it is not prime). Miller (1975) and Rabin (1980) developed efficient algorithm that almost certainly determines if n is prime. More results on number theory are needed to understand Miller Rabin method. Before discussing the algorithm that tests a given number to be prime, we state some results on prime numbers supported by examples.

Any odd positive number n >2 can be expressed as n-1=2^kq where k>0 and q is odd

Ex: Let n=23 (prime), n-1 = 22 = 2 x 11 (k=1 and q=11)

Ex: Let n=35 (non prime), n-1 = 34 = 2 x 17 (k=1 and q=17)

Primality Testing

1. If p is prime and a is any positive integer then a^2 mod p =1 iff either a mod p = 1 or a mod p = -1(p-1)

   Ex: Let a=4, p=3; a^2 mod p =16 mod 3 =1; Also a mod p =1

   Let a=4, p=2; a^2 mod p =16 mod 2 = 0; Neither 4 mod 2 = 1 nor 4 mod 2 = -1

   Let a=6, p=7; a ^{2} mod p = 36 mod 7 = 1 and also a mod p =-1

2. Let p be prime > 2. Recall p-1 = 2^k q where k >0, q odd. Let a be any integer such that 1 < a < p-1 then one of the following conditions is true. i. a^q mod p =1 ii. One of a^q , a^{2q}, ... $a^{(2\wedge(k-1)*q)}$ is congruent to -1 mod p. That is there is some number j ∍ 1 < j < k so that a^j = -1 mod p.

   Ex: Suppose p=5, a=3; p-1= 4 and k = 2, q=1. a^q mod p = 3 mod 5 is not 1 But a^{2q} mod p = 9 mod 5 = 4 = -1 mod 5

   Ex: Suppose p= 7, a=4; p-1=6 and k=1, q=3. a^q mod p = 64 mod 7 =1

Miller Robin Algorithm: We can use the preceding property to devise a test for primality. The property 2 implies that if n is prime then first number in list of residues (modulo) (a^q , a ^{2q}, ... $a^{(2\wedge(k-1))*q}$ $^)$ mod n is 1 or some element in the list is -1. On the other hand if the condition is met it does not imply that the number n is prime.

Ex: Let n=2047. n is not prime since n = 23 × 89. Now n-1 = 2046 = 2 × 1023; that is k=1 and q=1023. For a=2 we see that a^q = 21023 = 1 mod 2047 = 1 mod n

The algorithm for testing primality of n is given here.

```
TEST (n)
1. Find integers k, q with k>0, q odd so that n-1=2k × q
2. Select a random integer a ∍ 1 < a < n-1.
3. If aq mod n =1 then return “inconclusive”.
4. For j= 0 to k-1 do 123
5. If a(2^j)*q is -1 mod n then return “inconclusive”
 End for
 6. Return “composite”
```

The algorithm above returns “composite” implies that n is definitely not a prime and if it returns “inconclusive” then it implies that n may or may not be prime. The algorithm is executed with several random a‟s. If the result is “inconclusive” for all a‟s then n is prime with high probability.

Ex: n=29 (prime). The algorithm returns “inconclusive” for all a‟s from 1 to 28. This is consistent with n being prime.

n=221 = 13 x 17, a composite number

n-1 = 220 = 22 x 55 ; Thus q=2, k=55 For a= 21 you get “inconclusive” since 21$^{55\text{^}2}$ is -1 mod 22

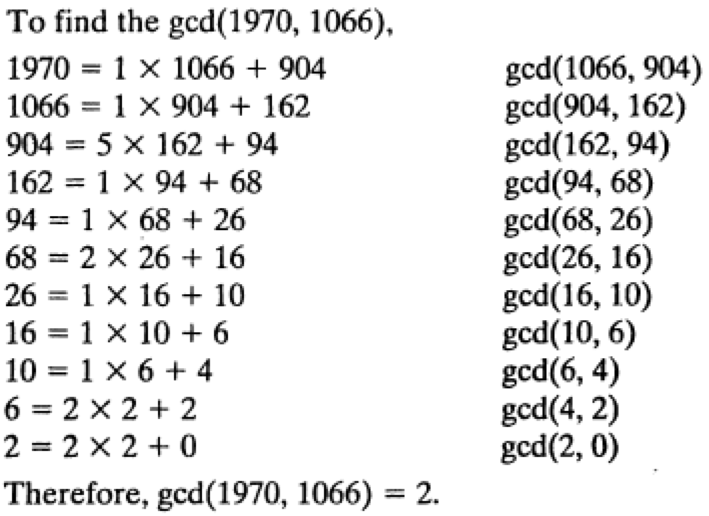
This means that n may be prime (according to the algorithm) but we know that n is definitely composite. You can verify that only for few a‟s (21, 47, 174, 200) the algorithm returns “inconclusive”. For a = 5 the algorithm returns “composite”. Thus running the algorithm for several a‟s will correctly determine composite number. If it returns “inconclusive” for several a‟s it is highly likely that it is prime and we assume n is prime.

Euclidean Theorem: This is one of the basic technique used in number theory to find out the greatest common divisor of two positive integers when both integers are relatively prime. It is based on the following statement that for any non-negative integer a and a positive integer b,

gcd(a,b)=gcd(b,a mod b)

gcd(55,22)=gcd(22, 55 mod 22))=gcd(22,11)=11

Ex:



To find the gcd(1970, 1066),

$1970 = 1 \times 1066 + 904$	gcd(1066, 904)
$1066 = 1 \times 904 + 162$	gcd(904, 162)
$904 = 5 \times 162 + 94$	gcd(162, 94)
$162 = 1 \times 94 + 68$	gcd(94, 68)
$94 = 1 \times 68 + 26$	gcd(68, 26)
$68 = 2 \times 26 + 16$	gcd(26, 16)
$26 = 1 \times 16 + 10$	gcd(16, 10)
$16 = 1 \times 10 + 6$	gcd(10, 6)
$10 = 1 \times 6 + 4$	gcd(6, 4)
$6 = 2 \times 2 + 2$	gcd(4, 2)
$2 = 2 \times 2 + 0$	gcd(2, 0)

Therefore, gcd(1970, 1066) = 2.

Chinese Remainder Theorem: Chinese Remainder Theorem is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime is defined as:

X=a1 mod1

X=a2 mod2

X=a3 mod3

....

Xn=an modn.

The Chinese remainder theorem states that the above equations have a unique solution if the moduli are relatively prime

Proof:

1. Denote M = m1m2 · · · mk
2. Mi = M/mi (i = 1, 2, . . . , k). Since the mi's are pairwise coprime
3. Find GCD(M1, M2, . . . , Mk) = 1 and GCD(mi , Mi) = 1 (i = 1, 2, . . . , k).
4. The following procedure produces a solution x (if there is one!), and also shows that the solution is unique modulo M:

Ex: Find the value of x where x=2 mod 3, x=3 mod 5, x=2 mod 7.

The answer to this set of equations is x = 23. This value satisfies all equations: 23 ≡ 2 (mod 3), 23 ≡ 3 (mod 5), and 23 ≡ 2 (mod 7). The solution to this set of equations is below.

1. Find M = m1 × m2 × ... × mk. This is the common modulus.

2. Find M1 = M/m1, M2 = M/m2, ..., Mk = M/mk.

3. Find the multiplicative inverse of M1, M2, ..., Mk using the corresponding moduli (m1, m2, ..., mk). Call the inverses as $M1^{-1}$, $M2^{-1}$, ..., Mk^{-1}.

4. The solution to the simultaneous equations is

$$X=a_1xM_1xM_1^{-1} +a2xM2xM2^{-1}+a3xM3xM3^{-1}+...+akxMkxMk^{-1})\text{mod } M$$

The solution to the above example is:

1. M = 3 × 5 × 7 = 105
2. M1 = 105 / 3 = 35, M2 = 105 / 5 = 21, M3 = 105 / 7 = 15
3. The inverses are M1−1 = 2, M2−1 = 1, M3 −1 = 1
4. x = (2 × 35 × 2 + 3 × 21 × 1 + 2 × 15 × 1) mod 105 = 23 mod 105

Discrete Logarithms: The Discrete logarithm is fundamental to many public key encoding systems including Diffie Hellman key exchange algorithms and digital signatures. Recall Euler totient function ϕ(n) and the Euler's theorem, which states that $a^{\phi(n)} \equiv 1$ mod n if a, n are relatively prime.

Now consider more general expression $a^m = 1 \bmod n$. If a and n are relatively prime there is at least one integer ($\phi(n)$) that satisfies the general expression. The least positive integer m which satisfies the equation $a^m = 1 \bmod n$ is called

(i) order of a mod n

(ii) exponent to which a belongs to mod n

(iii) length of the period generated by a

Ex: Consider a=7, n= 19 $7^1 = 7 \bmod 19$

$7^2 = 11 \bmod 19$

$7^3 = 1 \bmod 19$

$7^4 = 7 \bmod 19$ … The sequence repeats like this. Therefore the period is 3.

This is nothing but the smallest integer m for which $a^m = 1 \bmod n$

For a=2, one can check that $(a, a^2, a^3, \ldots a^{\phi(19)}) \bmod 19$ are

2, 4, 8, 16, 13, 7, 14, 9, 18, 17, 15, 11, 3, 6, 12, 5, 10, 1

All numbers 1 to 18 (the value of $\phi(19)$) have appeared.

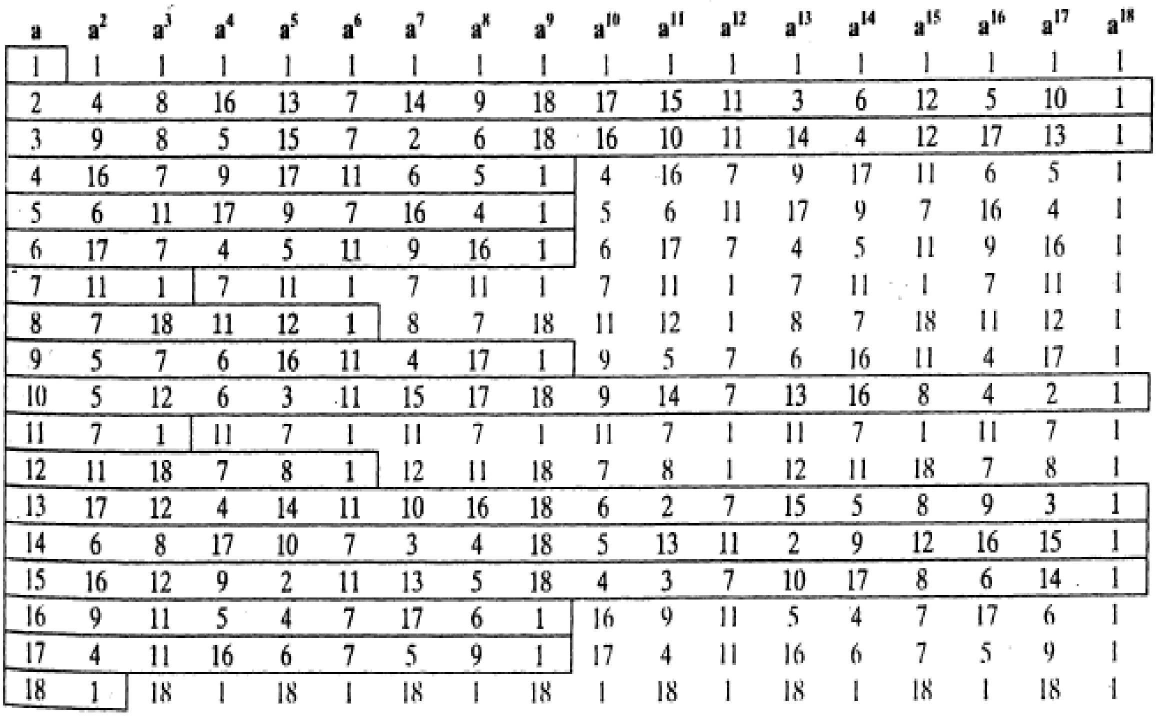
The period is 18, full period. Similarly it can be verified that 3, 10, 13, 14, 15 all have full period.

Primitive Roots: In general the highest possible exponent to whom a number can belong (mod n) is $\phi(n)$. If a number is of this order it is referred to as a primitive root of n. Alternatively if a is a primitive root of n then a, a2 , a3 , … aϕ(n) are all distinct (mod n) and are all prime to n. In particular for a prime number p, if a is a primitive root of p then a, a2 , a3 , … ap-1 are distinct (mod p) and prime to p. from the preceding example we see that, primitive roots of 19 are 2, 3, 10, 13, 14 and 15.

The following table shows all the powers of a, modulo 19 for all positive a < 19. The length of the sequence for each base value is indicated by shading. Note the following:

1. All sequences end in 1. This is consistent with the reasoning of the preceding few paragraphs.
2. The length of a sequence divides $\phi(19) = 18$. That is, an integral number of sequences occur in each row of the table. The length of the sequence for a = 1 is 1. For a = 2 and 3 it is 18. For a = 4 it is 9 and so on. Note that all these lengths divide $\phi(n)$ (=18).

Powers of Integers, Modulo 19



a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Indicies: with ordinary positive real numbers, the logarithm function is the inverse of exponentiation. An analogous function exist in modular arithmetic also. The logarithm of a number is to be defined as a power to which some positive base (except 1) must be raised in order to equal the number. That is, for base x and for a value y

$$y = x^{\log_x(y)}$$

The properties of logarithms include the following

$$\log_x(1) = 0$$

$$\log_x(x) = 1$$

$$\log_x(yz) = \log_x(y) + \log_x(Z)$$

$$\log_x(y^r) = r \times \log_x(y)$$

Consider a primitive root a for a prime number p. then the powers of a from 1thorugh (p-1) produce all the integers from 1 to p-1 exactly once. Any integer b can be expressed in the form by the defintion of modular arithmetic.

$$b \equiv r \bmod p \qquad \text{where } 0 \leq r \leq (p-1)$$

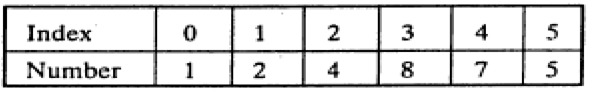
It follows that for any integer b, and a primitive roopt a of prime p, we can find the unique exponent i such that

$$b \equiv a^i \bmod p \qquad \text{where } 0 \leq i \leq (p-1)$$

This exponent i is referred to as the index of number b and nase a (mod p). this is represented as: $\text{Ind}_{a,p}(b)$

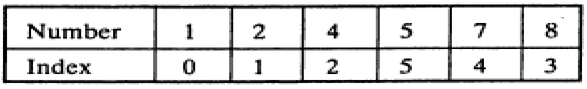
$$\text{ind}_{a,p}(1) = 0, \text{ because } a^0 \bmod p = 1 \bmod p = 1$$
$$\text{ind}_{a,p}(a) = 1, \text{ because } a^1 \bmod p = a$$

This gives us the following table of numbers with given indices (mod 9) for the root a=2



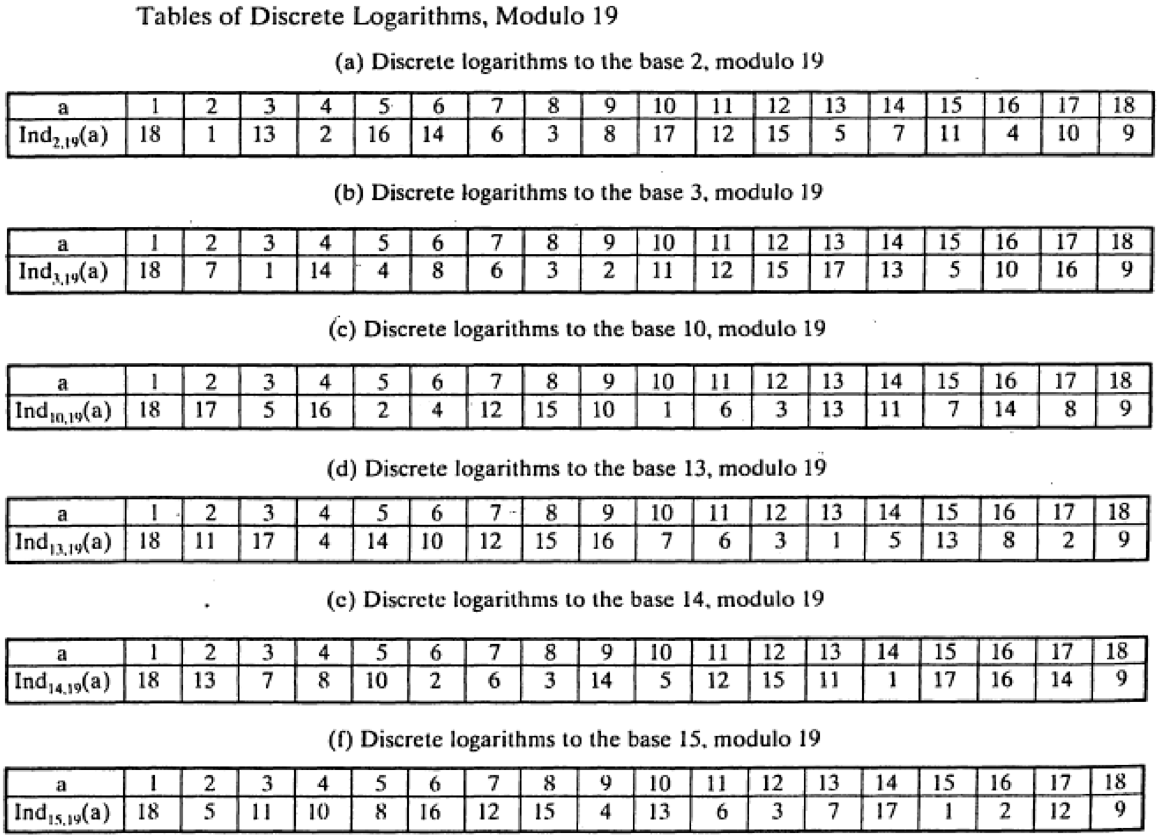
Index	0	1	2	3	4	5
Number	1	2	4	8	7	5

To obtan the index of a given number, we can rearrange the table to make the remainders relatively prime to 9 the primary entry.



Number	1	2	4	5	7	8
Index	0	1	2	5	4	3

The following table shows the sets of discrete logarithms that are defined for mod 19



Tables of Discrete Logarithms, Modulo 19

(a) Discrete logarithms to the base 2, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\text{Ind}_{2,19}(a)$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9

(b) Discrete logarithms to the base 3, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\text{Ind}_{3,19}(a)$	18	7	1	14	4	8	6	3	2	11	12	15	17	13	5	10	16	9

(c) Discrete logarithms to the base 10, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\text{Ind}_{10,19}(a)$	18	17	5	16	2	4	12	15	10	1	6	3	13	11	7	14	8	9

(d) Discrete logarithms to the base 13, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\text{Ind}_{13,19}(a)$	18	11	17	4	14	10	12	15	16	7	6	3	1	5	13	8	2	9

(c) Discrete logarithms to the base 14, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\text{Ind}_{14,19}(a)$	18	13	7	8	10	2	6	3	14	5	12	15	11	1	17	16	14	9

(f) Discrete logarithms to the base 15, modulo 19

a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\text{Ind}_{15,19}(a)$	18	5	11	10	8	16	12	15	4	13	6	3	7	17	1	2	12	9

PUBLIC KEY CRYPTOGRAPHY

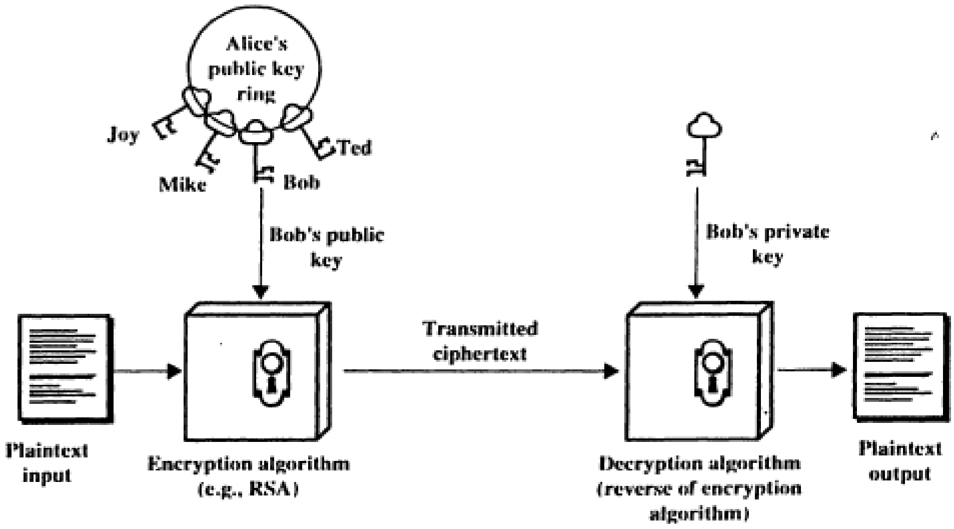
The development of public key cryptography is the greatest revolution in the history of cryptography. Public key algorithms are based on mathematical functions rather than on substitution and permutation. The more important, public key cryptography is asymmetric involving the use of two separate keys. The security of any encryption scheme depends on the length of the key and the computational work involved in breaking a cipher.

Principles of Public Key Cryptosystems: Public key algorithms rely on one key for encryption and a different but related key for decryption. These algorithms have the following important characteristics:

1. It is computationally infeasible to determine the decryption key given only the knowledge of encryption algorithm and encryption key.
2. Either of the two keys are used for encryption, with the other used for decryption.

Public key encryption provide both confidentiality and authentication services. The essential steps in public key encryption process are:

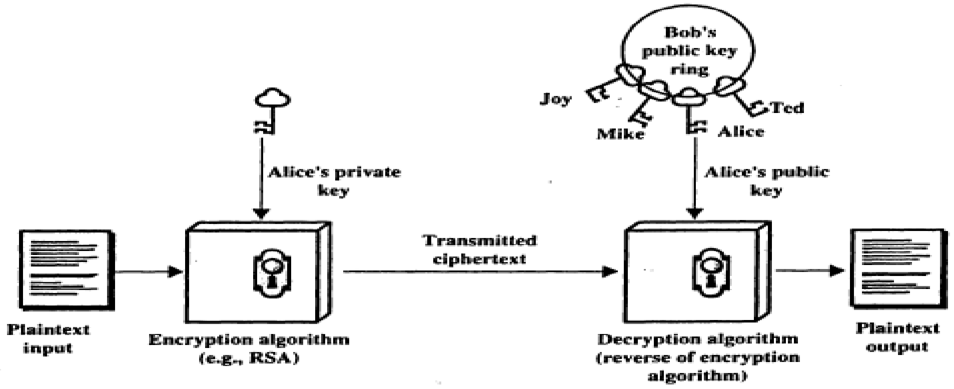
1. Each user generates a pair of keys to be used for encryption and decryption of messages that it will receive and send
2. Each system publishes its encryption key by placing it in a public register or file. This is the public key. The companion key is kept as private.
3. If A wishes to send a message to B, it encrypts the message using B’s public key
4. When B receives the message, B decrypts it using B’s private key. No other recipient can decrypt the message because only B knows B’s private key.



Encryption

Public key encryption to provide authentication service:

1. Each user generates a pair of keys to be used for encryption and decryption of messages that it will receive and send
2. Each system publishes its encryption key by placing it in a public register or file. This is the public key. The companion key is kept as private.
3. If A wishes to send a message to B, it encrypts the message using A's private key
4. When B receives the message, B decrypts it using A's public key. If it is decrypted by A's public key, then receiver accepts this message.

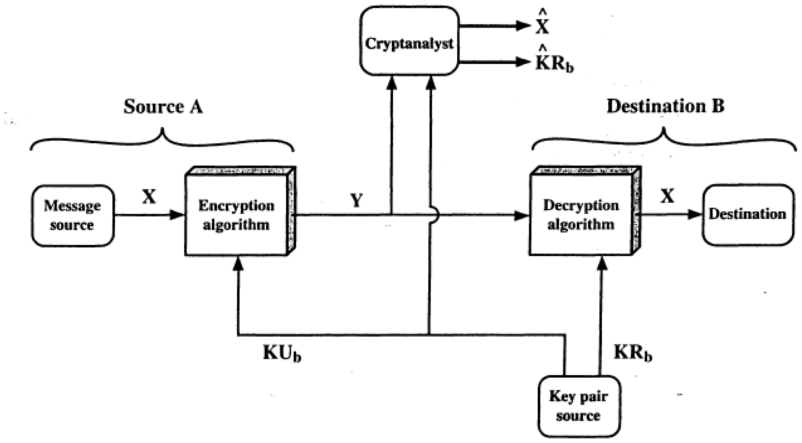


Authentication

Let us assume there is some source A who has a message X=[X1,X2,X3...Xn]. The n elements of X are letters in some finite alphabet set. The message is intended for B. B generates a related pair of keys: a public key KU_a, and a private key KR_a. KR_a is known only to B, whereas KU_b is publicly available and therefore accessible by A.

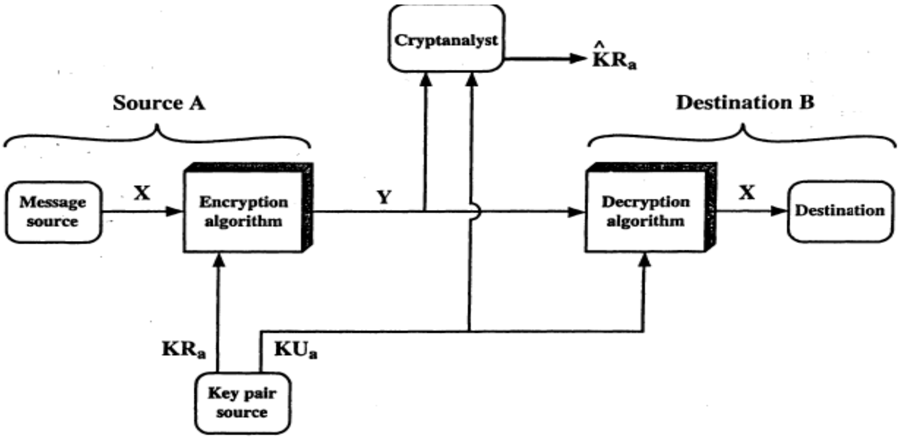
With the message X and the encryption key KUb as input, A forms the cipher text Y=[Y1,Y2,...,Yn]. $Y=E_{KUb}(X)$

The intended receiver, in possession of the matching private key, is able to decrypt the cipher text. $X=D_{KUb}(Y)$



To provide authentication: $Y=E_{KRa}(X)$

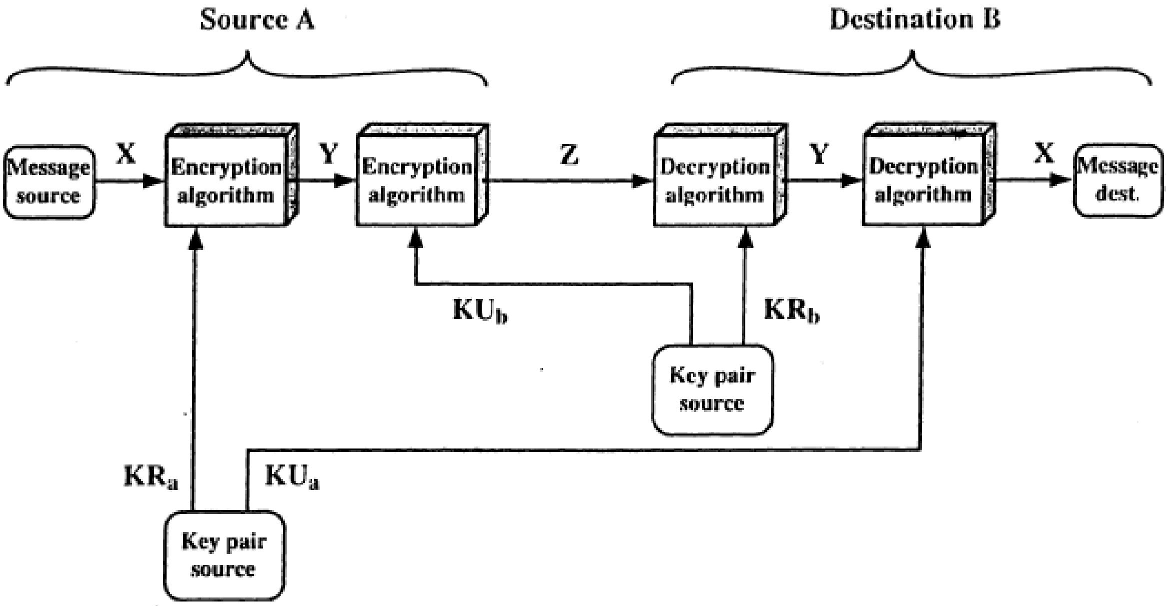
$X=D_{KUa}(Y)$



So far we provided either the confidentiality or authentication service. It is however, possible to provide both the authentication function and confidentiality by a double us of public key scheme.

$$Z = E_{KUb}[E_{KRa}(X)]$$
$$X = D_{KUa}[D_{KRb}(Z)]$$

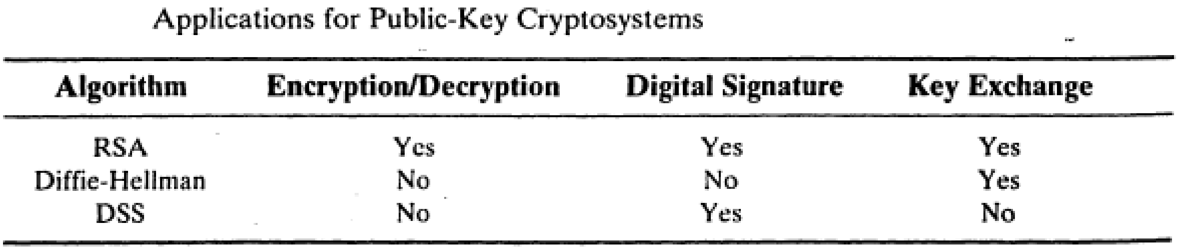
In this case,we first encyrpt a message, using the sender's private key. This provides digital signature. Next, we encrypt again using the receiver's public key. The final cipher text can only be decrypted by the intended receiver. Thus confidentiality is provided to the message.



Applications:

1. Encryption/Decryption: The sender encrypts the message with recipients public key
2. Digital Signature: The sender signs the message with his private key. Signing is achieved by a cryptographic algorithm applied to the message or to a small block of data that is a function of the message.
3. Key Exchange: Two sides cooperate to exchange a session key. Several different approaches are possible involving the private key of one or both parties.

Some algorithms are suitable for all three applications, whereas others can be used only for one or two of these applications:



Applications for Public-Key Cryptosystems

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Requirements of public key cryptography:

1. It is computationally easy for a party B to generate a pair of keys KUb and KRb
2. It is computationally easy for sender A, knowing the public key and a message to be encrypted M, to generate corresponding cipher text.

   C=EKUb(M)

3. It is computationally easy for receiver B to decrypt the resulting cipher text using his own private key

   M=DKRb(C) =DKRb[EKUb(M)]

4. It is computationally infeasible for an opponent, knowing the public key KUb, to determine the private key KRb
5. It is computationally infeasible for an opponent, knowing the public key KUb and a cihpher text C to recover the original message, M.
6. The encryption and decryption functions can be applied in either order(reversible functions).

   M=EKUb[DKRb(M)]

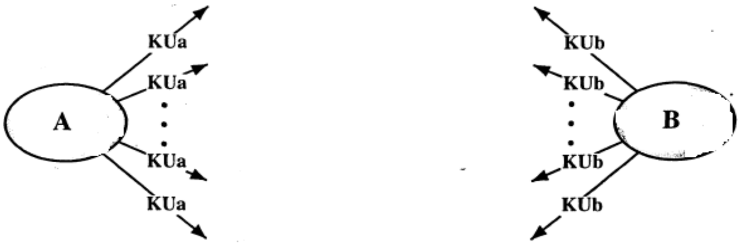
Key Management: One of the major roles of public key encryption has been to address the problems of key distribution. There are two distinct aspects to the use of public key encryption:

- The distribution of public keys
- The use of public key encryption to distribute secret keys

Distribution of Public Keys: Several techniques have been proposed for the distribution of public keys.

- Public Announcement
- Publicly Available Directory
- Public Key Authority
- Public Key Certificates

1. Public Announcement: All participants broadcasts their public keys to other recipients.



Uncontrolled Public-Key Distribution.

This approach is convinient but it has more weakness. Anybody can forge a public announcement. That is some user can prented like a and can broadcast his public key as someone's public key.

2.Publicly Avaialable Directory: A great degree of security can be achieved by using this method. Maintenance and distribution of the public directory would have to be the responsibility of some trusted entity or organization.

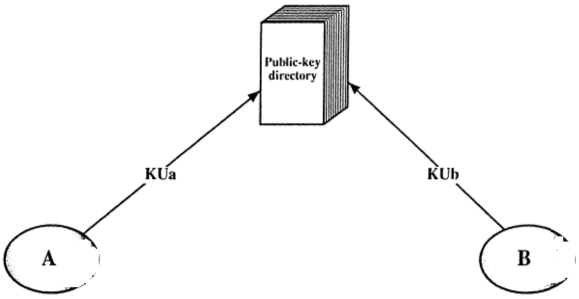
1. The authority maintains a directory with an entry for each participant

2. Each participant registers a public key with the directory authority. Registration would have to be in person or by some form of secure autnetication communication.

3. A participant may replace the existing key with a new one at any time, either because of the desire to replace a public key that has already been used for a large amount of data or because the corresponding private key ahs been compromised in some way.

4. Periodically, the authority publishes the entire directory or updates to the directory.

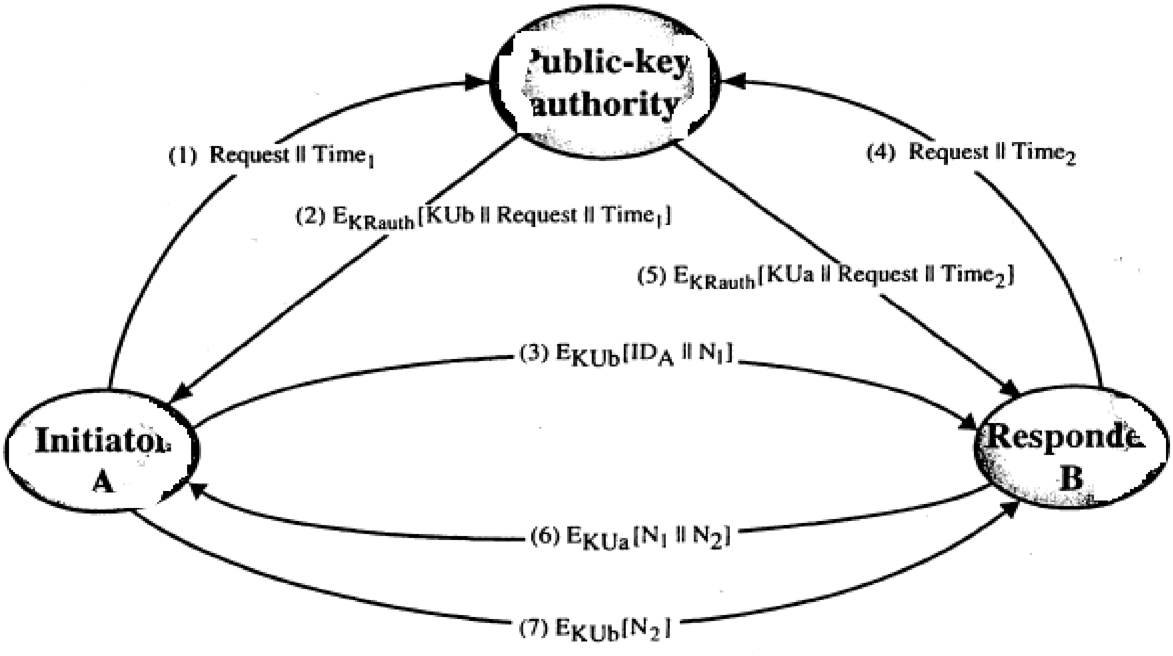
5. Participants could also access the directory electronically.



Public Key Authority: Stronger security for public key distribution can be achieved.

1. A sends a timestamped message to the public key authority containing a request for he current public key of B.
2. The authority responds with a message that is encrypted usign the authority's private key, EKRauth. Thus, A is able to decrypt the message using authority's public key. The message includes the following:
   a. B's public key KUb, which can be used by A to encryot messages destined for B
   b. The original request, to ensure that this reply is for the fresh request made from A
   c. The original timestamp, so a cand etermine that his is not an old message from the authority containing a key other than B's current public key.
3. A stores b's public key and uses it to encrypt a message to B containing an identifier of A Ida and a nonce N1, which is used to identify this transmission uniquely.

4,5. B retrieves A's public key from the authority in the same manner as A received B's public key.



Public-Key Distribution Scenario.

6. B sends a message to A by encrypting with KUa and containing A's nonce (N1) and a new nonce generated by B (N2). Becaue only B has a decrypted message (3) , the presence of N1 in message (6) assures A that the corresponding user is B.

7. A returns N2, by encrypting it with B's public key, to assure B that the corresponidng user is A

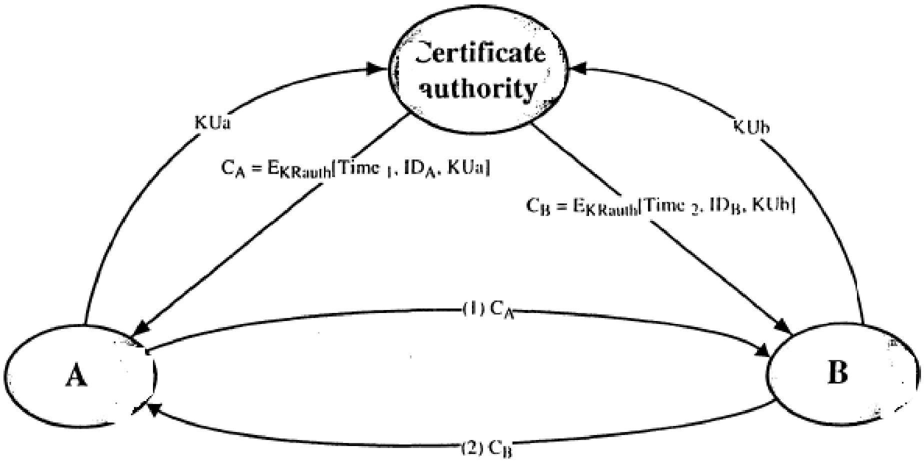
Public Key Certificates: These certificates are generated public key authorities. Once the authority generates public key certificates, these can be used by participant to exchange secret keys. Each certificate contains a public key and other information created by the authoritative server. The following are the necessary requirements to generate a certificate:

1. Any participant can read a public key certificate to know the name and public key of the owner.
2. Any participant can verify that the cerficate was orignated by Certification authority and is not counterfeit.
3. Only the certificate authrotiy can create and update certificates.

Process:

1. Each participant request certification authority(CA) to issue a public key certificate. A requests CA to issue a public key certiifcate C_A by sending his public key KUa.
2. The authority responds with a message that is encrypted using the authority's private key, EKRauth. Thus, A is able to decrypt the message using authority's public key. The message includes the following:
   a. Time stamp (time1)
   b. Identitiy of A (Ida)
   c. A's public key KUa

3,4. B also requests a public key certificate in the same manner as mentioned for A in step 1 and 2.


Certificate authority
KUa
KUb
$C_A = E_{KRauth}[Time_1, ID_A, KUa]$
$C_B = E_{KRauth}[Time_2, ID_B, KUb]$
(1) C_A
(2) C_B
A
B

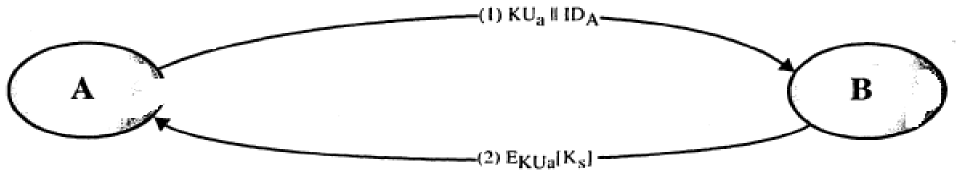
1. Once they get public key certiifactes, Now, to establish a secure connection with B A forwards his public key certificate $C_A$to B
2. Upon receing A's certificate, B verfies it and he also forwards his public key certificate to A

Public Key distribution of Secret Keys: once public keys are distributed, secure communications that thwarts eavesdropping or tampering is possible. However, some users will wish to make exclusive use of public key encryption for communication. Therefore , public key encryption is more a vehicle for the distribtuion of secret keys to be used for conventional(symmetric) encryption.

a) **Simple Secret Key Distribution:** If A wants to have a secure connection with B, he performs the followng steps:

   i) A generate a public/private key pair[KUa,KRa] and transmits a message to B which contains his Id Ida and public key KUa

   ii) B generates a secret key Ks and confidentially transmits it to A, by encrypting with A's public key KUa



Simple Use of Public-Key Encryption to Establish a Session Key.

- A and B can now securely communicate usign symmetric encryption and the session key Ks. At the completion of session both A and B discard Ks. Despite its simplicity, this is an attractive protocol. No keys exist before the start of communication and none exists after the completion of coomunication. At the same time it is secure from eavesdropping.

But this protocol is vulnerable to an active attack. If an opponent E, has control of the intervenning communciation channel , then E can compromise the communicaiton in the following way

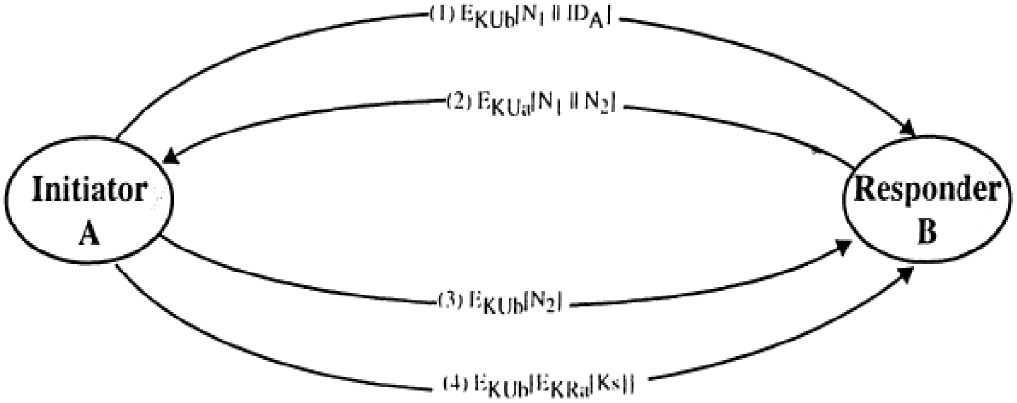
- A generates a public/private key pair [KUa,KRa] and transmits a message containing KUa and the indentity of a Ida.
- E intercepts this message, creates his own public /private key pair [KUc,KRc] and transmits KUc||Ida to B
- B generates a secret key Ks and transmits EKUc[Ks]
- Again E, intercepts this messages and learns Ks, by decrypting DKRc[Ks]
- E transmits Ks by encrytping EKUa[Ks] to A

Thereofre, bot A and A know Ks and are unaware that Ks has also revealed to E. so A, B exchange messages with Ks and E simple eavesdrops everything.

b) Secret key distribution with confidentiality and authentication

Assume that A and B have exchanged public keys using one of the public key distribtuion methods. Then the following steps occur:

1. A uses B's public key to encrypt a message B which cotnains the identifier of A Ida and a nonce N1 which is used to identify the trasaction uniquely.
2. B sends a message to A ecnrypted with KUa which contains A's nonce N1 and nonce generated by B N2. N1 assures that A's request was received by B only.
3. A returns N2 by encrypting it with B's pulbic key. This is to assure B that B's reply was received by A.
4. A selects a secret key Ks and sends as EKUb[EKRa[Ks]]. It provides both authentication and confidentiality.
5. B computes DKUa[DKRb[Ks]]


(1) $E_{KUb}[N_1 \| ID_A]$
(2) $E_{KUa}[N_1 \| N_2]$
Initiator A
Responder B
(3) $E_{KUb}[N_2]$
(4) $E_{KUb}[E_{KRa}[K_s]]$

Public-Key Distribution of Secret Keys.

RSA Algorithm:

RSA was deceloped by Ron Rivest, Adi Shamir and Len Adleman makes use of an exoression with exponentials. The ingredients of RSA alogrithm are:

1 p,q twp prime numebrs (private, chosen)

2 n=pq (public, calculated)

3 e, with gcd(e, ϕ(n)=1;1<e<ϕ(n) (public, chosen)

4 d=e^{-1}mod ϕ(n) (private, calculated)

Algorithm:

1. Select two largest prime numbers p and q
2. Calculate n=pxq
3. Calculate ϕ(n)=(p-1)x(q-1)

4. Select an integer ‘e’, such that gcd(e, ϕ(n)=1 where 1<e<ϕ(n) and make as public key or encryption key
5. Calculate d using d=e^{-1}mod ϕ(n) d is private key or decryption key

Encryption:

6. Encrypt message M using C=(M)^{e}mod n

Decryption

7. Decrypt cipher text C using M=(C)^{d}mod n

Ex: 1. Select two prime numbers p=7 and q=17

2.calculate N=pxq=7x17=119

3. calculate ϕ(n)=(p-1)x(q-1)=6x16=96

4. select an integer e such that e is relatively prime to ϕ(n), in other words gcd(e,ϕ(n))=1, so e=5

5. calculate d= 5^{-1}mod 96 which is 77

Encryption: To encrypt a message ”cd” where a=1,b=2 c=3,...

For character ‘c’:

C=(M)^{e}mod n=(3)5 mod 119 = 5

Therefore Cipher Text C=E

For character ‘d;”

C=(M)^{e}mod n=(4)5 mod 119 = 72

72 is more than 26(since we are following numerical encoding)

72 mod 26=20

Therefore Cipher Text=”T”

“cd” is encrypted as “ET”

Decryption:

“ET”

For character ‘E’:

M=(C)^{d}mod n=(4)^{77} mod 119 = 3

Therefore Plain Text C=c

For character ‘T;”

M=(20)^{77}mod n=4

Therefore Plain Text=d

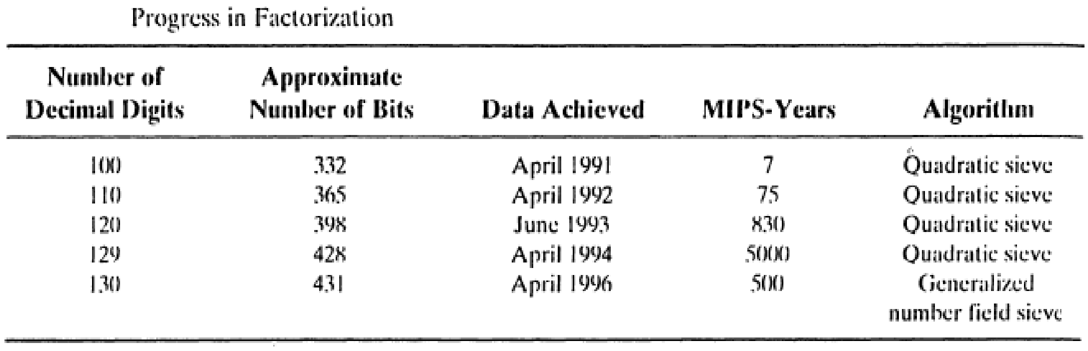
“ET” is decrypted as “cd”

Security of RSA:

Three possible approaches are there to attack RSA algorithm

1. **Brute Force**: Trying with all possible private keys
2. **Mathematical Attacks**: there are several approaches to attack RSA mathematically like factoring n, determining ϕ(n), determining D directly.

   Complexity in Factorization:



Progress in Factorization

Number of Decimal Digits	Approximate Number of Bits	Data Achieved	MIPS-Years	Algorithm
100	332	April 1991	7	Quadratic sieve
110	365	April 1992	75	Quadratic sieve
120	398	June 1993	830	Quadratic sieve
129	428	April 1994	5000	Quadratic sieve
130	431	April 1996	500	Generalized number field sieve

3. **Timing attacks**: These attacks depend on the running time of the decryption algorithm

Although the timing attack is a serious threat, there are simple countermeasures that can be used in the following way:

a. Constant Exponentiation Time: Ensues that all exponentials take the same amount of time before returning a result. This is a simple fix, but degrades performance.

b. Random Delay: better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack.

c. Blinding: Multiply the cipher text by a random number before performing exponentiation. This process prevents the attacker from knowing cipher text.

Diffie-Hellman Key Exchange:

This is the first published public-key algorithm by Diffie-Hellman in 1976. The purpose o algorithm is to enable two users to exchange a key securely that can then be used for subsequent encryption of messages. The algorithm itself is limited to key exchange not for encryption.

The Diffie-Hellman algorithm depends on the effectiveness of discrete logarithm.

Algorithm:

Generation of Global Public Key parameters:

1. Select a prime number q
2. Select an integer α, which is a primitive root of q where α<q

Generation of Public and Private Key pair:

3. User A selects a random integer X_A <q as private key
4. and A computes his public key as $Y_A = (\alpha)^{XA}$mod q
5. User B also selects a random integer X_B<q as private key
6. and B computes his public key as $Y_B = (\alpha)^{XB}$mod q

Generation of Session Key or Secret key:

7. User A computes the key as $K=(Y_B)^{XA}$mod q
8. User B computes the key as $K=(Y_A)^{XB}$mod q

   The calculation of step 7 and 8 produce similar results. To prove that these two results are identical:

Proof:

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

Thus, the two sides have exchanged a secret key. Furthermore, because Xa and Xb are private an opponent only has following ingredients to work: q, α, Ya and Yb. Thus the opponent is forced to use a discrete logarithm to determine the private key.

$$X_B = \mathrm{ind}_{\alpha,q}(Y_B)$$

The opponent can then calculate key in the same way as B calculated. The security of Diffie-Hellman lies in discrete logarithm. It is very difficult to calculate discrete logarithms.

Ex:

1. Select a prime number=97

2. select a primitive root of 97 that is 5

3. A selects his secret key Xa=36

4. A calculates his public key $Ya=5^{36}$mod 97= 50 mod 97=50

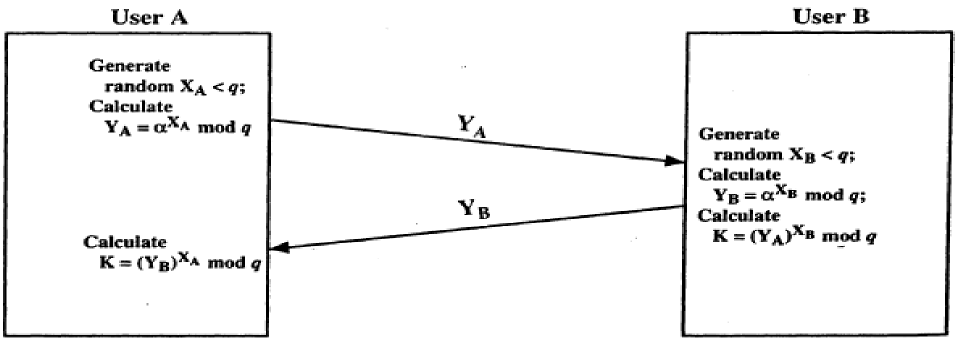
5. B selects a secret key Xb=58

6. B calculates his public key $Yb=5^{58}$mod 97=44 mod 97=44

After exchanging public keys, each one can compute the common secret keys:

$K=(Y_B)^{XA}$mod $97=44^{36}$mod 97=75 mod 97=75

$K=(Y_A)^{XB}$mod $97=50^{58}$mod 97=75 mod 97=75


User A
Generate random $X_A < q$;
Calculate $Y_A = \alpha^{X_A} \bmod q$
Calculate $K = (Y_B)^{X_A} \bmod q$
Y_A
Y_B
User B
Generate random $X_B < q$;
Calculate $Y_B = \alpha^{X_B} \bmod q$;
Calculate $K = (Y_A)^{X_B} \bmod q$

Diffie-Hellman Key Exchange.

Elliptic Curve Cryptography: Elliptic is not an ellipse like an "oval circle" and curve is also a misleading term it is like a cloud of points in the field. These points are not connected together. This is a cloud of points which full fill the "curve equation ". They are so named because they are described in cubic equation. In a finite field; the elliptic group mod p where p is a prime number. This can be defined as:

Choose two positive integers a and b less than p which satisfy 4a3 +27 b2 mod p !=0

Then Ep(a,b) denotes the elliptic group mod p whose elements are (x,y) are pairs of positive integers less than p satisfying y2 = x3 +ax+b mod p together with point at infinity.

The rules for addition over Ep(a,b) for all points x,y belongs to Ep(a,b):

1. P+O=P

2. If p=(x1,y1) and q=(x2,y2) to compute R=(x3,y3)=p+q then compute

$$x_3 \equiv \lambda^2 - x_1 - x_2 (\text{mod } p)$$
$$y_3 \equiv \lambda(x_1 - x_3) - y_1 (\text{mod } p)$$

where

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1} & \text{if } P = Q \end{cases}$$

3.Again multiplication is defined as repeated addition for example 4P=P+P+P+P

Algorithm:

Generation of Global Public Key parameters:

1. Select a largest prime number in the range of 2^{180}
2. Select two non negative integers a and b which satisfy the equation 4a3 +27 b2 mod p !=0. This defines the elliptic group Ep(a,b).

3. Pick a generator point G=(x,y) in Ep(a,b) such that the smallest value of n for which nxG=0 Ep(a,b) and G are global public key parameters known to all the participants.

Secret/Session Key Exchange between Participants:

4. A selects an integer n_A<n . This is A's private key A then generates a public key P_A= n_Ax G
5. Similarly B selects an integer n_B<n . This is A's private key A then generates a public key P_B= n_Bx G.
6. A computes his secret key K= n_AxP_B and B also computes his secret key K= n_BxP_A

   These two calculations in step 6 produce the same result because

   K= n_AxP_B= n_Ax(n_Bx G)= n_Bx(n_Ax G) = n_BxP_A

ECC Encryption/Decryption

7. To encrypt and send a message Pm to B, A chooses a random positive integer k and produces the cipher text Cm consisting of the pair of points.

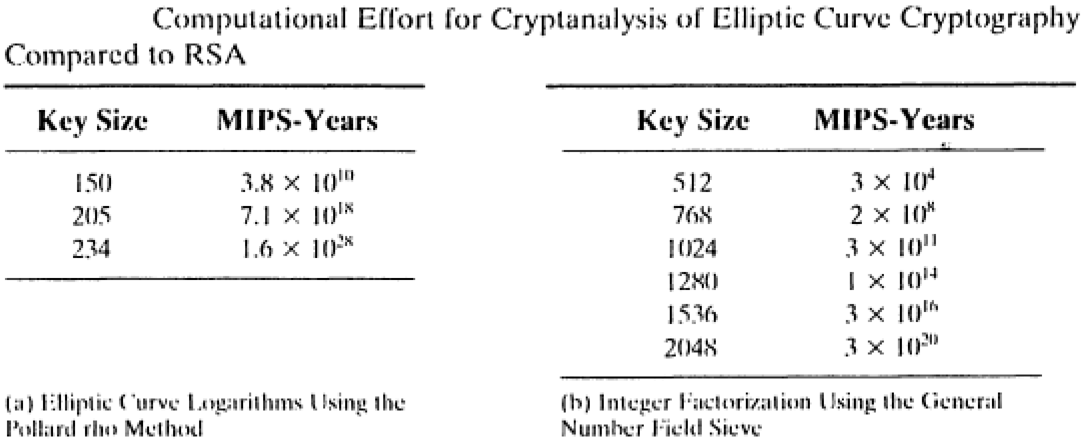
$$Cm=\{ kxG , Pm+kP_B\}$$

8. Since A has used B's public key P_B to encrypt, now B decrypts this by using his private key n_B. B multiplies the first point in the pair by his private key n_B and subtracts the result from the second point. That is

$$Pm+kP_B – kxn_BxG = Pm+kP_B – kxn_BxG = Pm$$

Security of ECC:

The security of ECC depends on how difficult it is to find k given kP ande P. This is referred to as the elliptic curve logarithm problem. The fastest known technique for taking the elliptic curve logarithm is known as Pollard rho method. The following table presents the efficiency of this method with factoring a number into two primes using the general number field sieve algorithm.



Computational Effort for Cryptanalysis of Elliptic Curve Cryptography Compared to RSA

Key Size	MIPS-Years
150	3.8×10^{10}
205	7.1×10^{18}
234	1.6×10^{28}

(a) Elliptic Curve Logarithms Using the Pollard rho Method

Key Size	MIPS-Years
512	3×10^{4}
768	2×10^{8}
1024	3×10^{11}
1280	1×10^{14}
1536	3×10^{16}
2048	3×10^{20}

(b) Integer Factorization Using the General Number Field Sieve

El Gammal Key Exchange:

This public-key cryptosystem is related to Diffie and Hellman. It uses exponentiation in a finite (Galois) field with security based difficulty of computing discrete logarithms, as in Difie and Hellman. Th algorithm is described as follows

Global Public Key Generation

1. Select a largest prime number of length 2^{180}
2. Select an integer 'a' which is a primitive root of q. Now q and a are global public key parameters.

Generation of Public and Private key pair:

3. Each user (eg. A) generates their key and A chooses a private key (number): $1 < Xa < q - 1$ and compute their public key: $Ya = a^{Xa} \bmod q$
4. B also generates his key pair by choosing his private key as $1<Xb<q-1$ and computes his public key $Yb=a^{Xb} \bmod q$

Elgammal Encyrption/Decryption and session/secret key :

5. A encrypt a message to send to B. A represents the message M in a range of $0 <= M <= q – 1$
6. Longer messages must be sent as blocks. A chooses random integer k with $1 <= k <= q - 1$
7. A computes one-time key $K =(Yb)^k \bmod q$
8. A encrypts M as a pair of integers (C1,C2)

   where $C1 = a^k \bmod q$;

   $C2 = KM \bmod q$
9. B recovers plain text message by recovering the key K as

   $K = (C1)^{xb} \bmod q$

   $M = C2K^{-1} \bmod q$