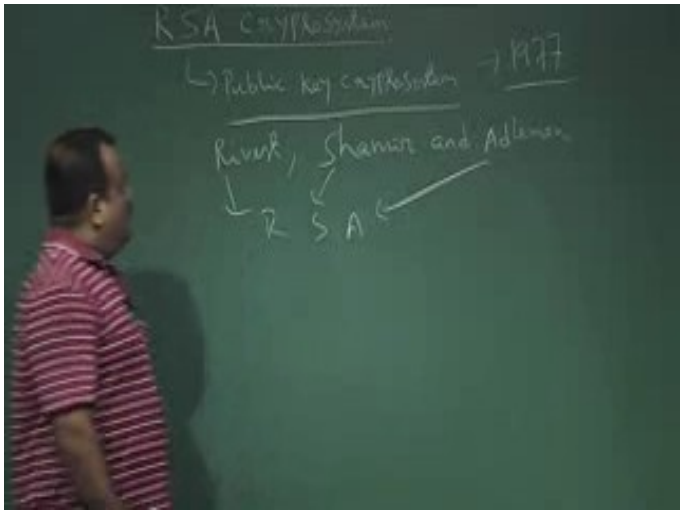
Internetwork Security
Prof. Sourav Mukhopadyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

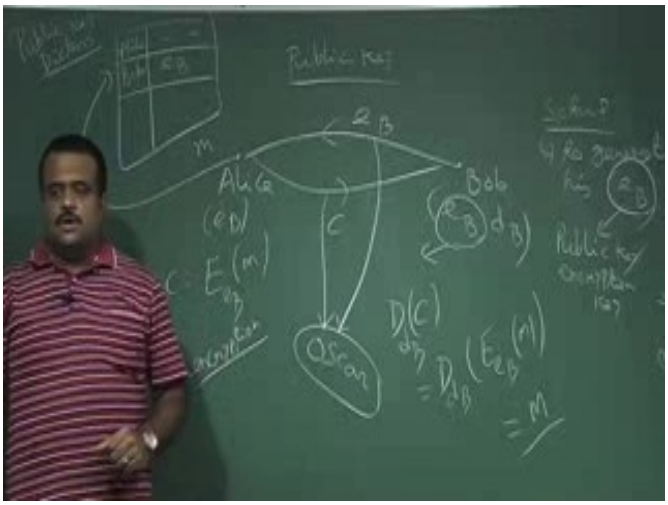
Lecture – 25
RSA Cryptosystem

In this we will talk about RSA cryptosystem which is a public key cryptosystem. It was invented by three cryptographers at MIT – Rivest, Shamir and Adlemen. It is named as RSA based on their name.

(Refer Slide Time: 00:24)



(Refer Slide Time: 01:48)

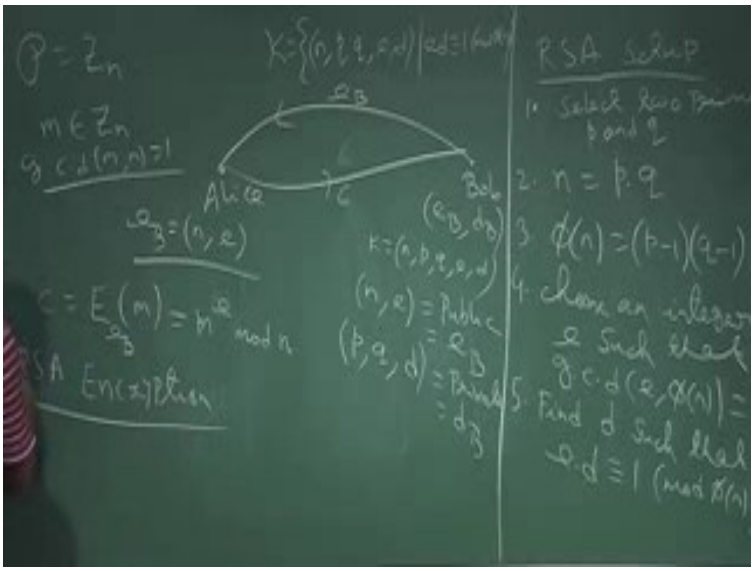


So, we have two party - Alice and Bob. In public key setup, if Alice wants to send a message to Bob then Alice needs to get Bob's public key. Bob has to run the setup phase to generate his public key and private key pair. Then Bob has to make his public key public. Bob can send this public key over public channel to Alice or if there is a public key directory, he can store the public key there.

Alice will encrypt this message using the public key of Bob and generate the ciphertext c and send this to Bob. Bob will apply the decryption algorithm using his secret key. The key should be chosen in such a way that by seeing the public key one should not able to guess the secret key.

So, in public key setup getting the decryption key from public key should be computationally hard. Now we will talk about RSA cryptosystem which is a public key cryptosystem.

(Refer Slide Time: 06:04)



Again suppose Alice wants to send a message to Bob. So RSA setup has to be done by Bob. There are few steps - first step is Bob will choose select two prime number primes p and q and these have to be large primes. Then Bob computes the product p*q and then Bob compute this ϕ_n which is basically (p-1)*(q-1), and now Bob will choose an integer e such that gcd of e and ϕ_n should be 1.

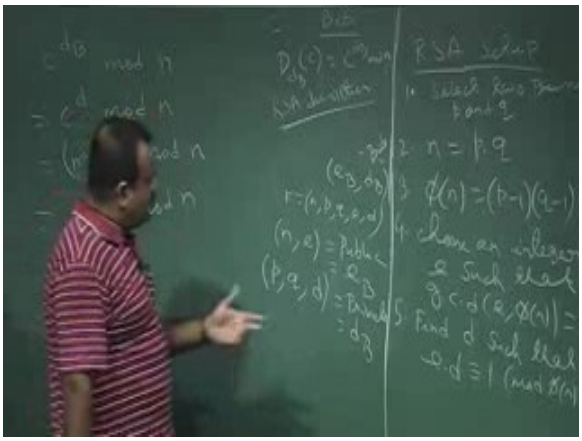
These we need because we want to get the inverse of e under mod ϕ_n. So, if they are relatively prime then the inverse will exist. So, in order to get the inverse of e this condition is

required gcd(e,ϕ_n) should be 1. d is the inverse of e such that e*d is congruent to 1 mod ϕ_n. Key basically consists of n, p, q, e, d.

So, each key k is this tuple. So, among these n and e are public and private is p, q, d. Let us denote this by e_B, d_B respectively. Bob sends e_B to Alice. Now the message space is basically Z_n, so Alice choose a message m from Z_n such that it is relatively prime to n.

Encryption of m using Bob public key is basically m to the power e mod n. So, this is the ciphertext Alice is sending to Bob.

(Refer Slide Time: 12:16)



Bob is having his own secret key. Bob will compute c to the power d_B mod m. Bob will compute c to the power d_B mod m and this is supposed to give us m. We need to verify that whether we are getting m or not. So, this is the decryption and now the question is why this is equal to m. Since e and d are congruent to 1 mod ϕ_n, ed-1 is a multiple of ϕ_n. Also m^{p-1} is congruent to 1 mod p if p is a prime from Fermat's Little Theorem. So the proof is,

$$\begin{aligned}(m^e)^d &= m^{ed} \\ &= m^{ed-1}m \\ &= m^{h(p-1)(q-1)}m \\ &= (m^{q-1})^{h(p-1)}m \\ &= 1^{h(p-1)}m = m \bmod q.\end{aligned}$$

(Refer Slide Time: 14:49)

So, this is the way we can decrypt the RSA.

(Refer Slide Time: 19:45)

We can take an example quick example on this RSA.

a) Select p=7 ,q=17

b) Calculate n = pq = 7 × 17 = 119

c) Calculate φ(n) = (p − 1)(q − 1) = 96.

d) Select e, relatively prime to and less than φ(n), say e = 5.

e) Determine d such that de = 1 (mod 96) and d < 96.

f) The correct value for d is 77 because 77 × 5 = 385 = 4 × 96 + 1 (can be calculated using the extended version of Euclid's algorithm).

g) The resulting public key is KU= {5, 119} and private key is KR= {77}. Say the plaintext is M = 19. For encryption 19 is raised to the 5th power, yielding 2, 476, 099. Upon division by 119, the remainder is 66, hence ciphertext sent is 66. For decryption it is determined using KR that 66 77 ≡ 19 (mod 119) so the recovered plaintext is 19.

(Refer Slide Time: 25:46)

Now the question is there are many issues here. Computational aspect that is how can we calculate such a^b where a and b are two big numbers? Another issue is how to get the primes, because for RSA we need two primes which should be large.

If you choose two simple primes say p is equal to 3 and q is equal to 7. So, then product is basically 21. By seeing 21 we can guess that p is 3, q is 7. Therefore we need to choose two large prime.

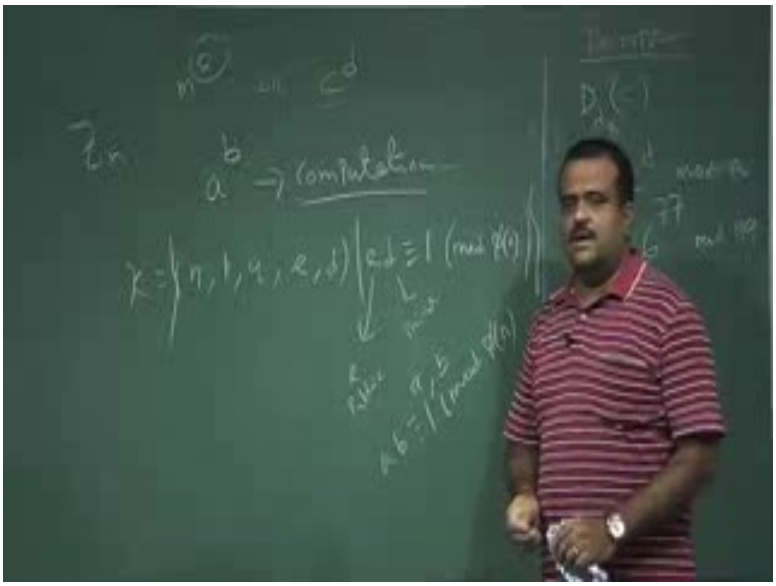
Now the question is, is there any algorithm for which we can give the length of the bit and it should give us the prime number with that number of bits. There is no such algorithm which can do that. So, we can do the primality testing. We can choose an odd integer of these many bits and check whether this is a prime or not.

(Refer Slide Time: 30:20)

There are some probabilistic algorithm and there are some recent breakthroughs which are determinist. Miller-Rabin is the probabilistic algorithm. There is a deterministic algorithm by Prof. Manindra Agarwal from Kanpur IIT and his group. They have designed AKS algorithm. They have given a deterministic pseudo code which can take a n bit input which is taking an odd integer and we can test whether this is prime or not in a deterministic way.

We will talk about some primality test in a later stage. Now another issue in RSA is the computing a^b.

(Refer Slide Time: 31:27)



Usually these are big numbers. m is coming from Z_n. n is very big number. So how can we have this exponentiation with two big numbers? We will talk about this in later lecture.

We choose e as a public key and d as a private key or the symmetric key. Another property of RSA is this role can be interchanged. This is one of the advantage of RSA cryptosystem we can make use of.

Thank you.