Ethical Hacking
Prof. Indranil Sengupta
Department of Computer science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 29
Public - Key Cryptography (Part I)

In this lecture we start with some discussion on Public-Key Cryptography. The title of this lecture is public key cryptography, first part of it. Now earlier we have seen symmetric or private key cryptography, where the same key was shared by the two end systems, sender and receiver, but you think of the internet scenario; in the internet scenario we are already habituated with sending and receiving messages between parties who are widely separated geographically.

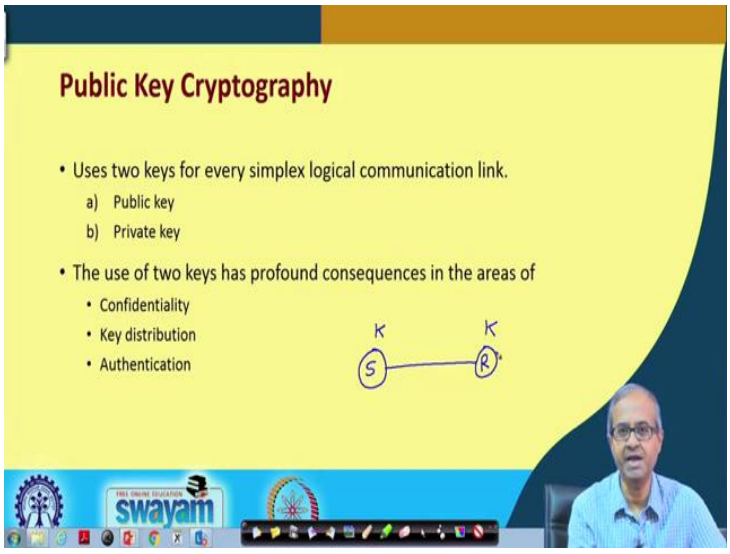
So, the natural question arises if I request secrecy, if I require privacy of my data, suppose I am sending some packets, a message to my friend, who is sitting 1000 kilometres away from here, first thing is that I can use these symmetry key algorithms for encryption, but how do we share the key? Shall we call each other and tell the key over telephone, but telephones can be trapped?

So, there should be some secure mechanism for sharing these keys also. In general, in the internet scenario there should be some technique where people who are sitting at a large distance from each other, not being able to communicate safely or securely using conventional medium can also carry out and enforce some kind of security in terms of the data communication. Let us see this.

(Refer Slide Time: 01:57)

So, in this lecture we shall be talking about public key cryptography as I said and you will see that this public key cryptography can be used for encryption as well as authentication purpose and one of the most popularly used algorithm, the RSA that we shall be discussing.

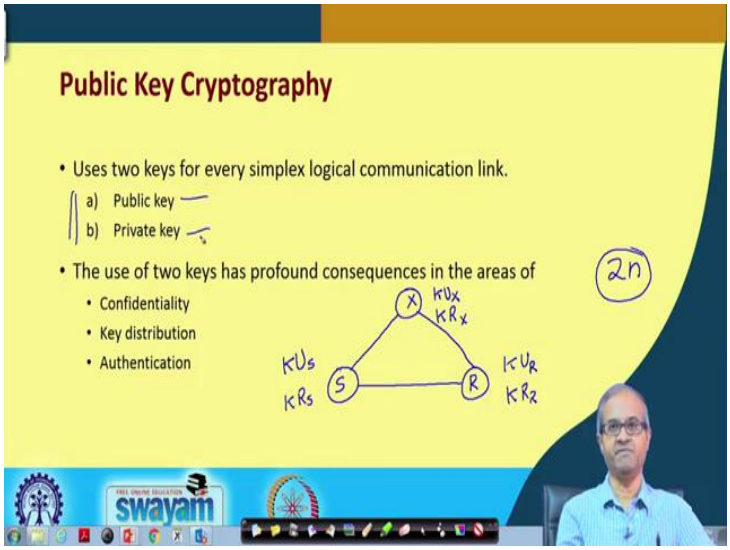
(Refer Slide Time: 02:20)



Now, talking about public key cryptography, here the concept is slightly different from symmetric key cryptography. Now in symmetric key cryptography what is happening? There was a sender; there was a receiver; there was a communication channel.

So, you define a secret key which was shared by both the parties, this was what was happening in symmetric key systems.

(Refer Slide Time: 02:54)



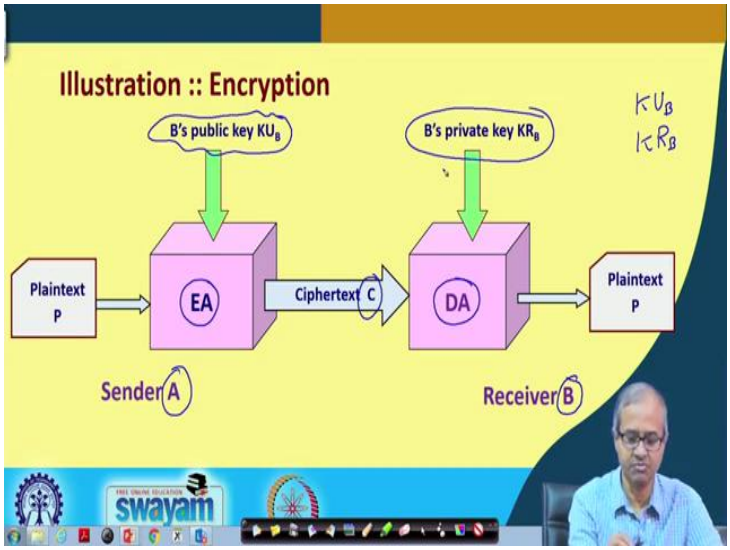
But in public key system what we are saying, it is something different. I have a sender; I have a receiver; there is a communication channel; now I am saying that I am not sharing any key across 2 parties. Let us say, there can be a third party, also X. There are three parties. So, now, what I am saying is that, every party will be having 2 keys. A pair of keys, one is called a public key; one is called private key. Notationally I can denote the public key by KU, KU_S and the private key by KR_S. So, here there would be KU_R and KR_R and here KU_X KR_X.

So, every party will be having 2 keys, the first thing you see that the number of keys are getting drastically reduced here. If there are n number of parties, there will be only 2^n different keys. In the earlier case we discussed, there were $n \times \frac{n-1}{2}$ keys which was the close to $\frac{n^2}{2}$, quite large. Here the name is going to say, public and private. Say everybody is having 2 keys. I am also having 2 keys.

So, one of my keys is a public key; one is a private key. The idea is that public key I am free to tell everybody. So, I can announce that this is my public key to anybody I want to, but my private key I will keep secret with myself. I will not tell anybody in this world. I

will keep it in my pocket. I will keep or maintain the secrecy of my private key. This is the idea of public key cryptography systems. Now let us see how it works.

(Refer Slide Time: 05:03)



So, again let us explain it pictorially, this is the schematic of a public key encryption system. Let us say there is a sender A which wants to securely send some data to a receiver B. So, I said the receiver B will be having a public key KU_B and we will be having a private KR_B, right.

The idea is that when somebody, in this case A wants to send a message to B through encryption, what it does? There will be an encryption algorithm which will be using Bs public key. B's public key is supposed to be known by everybody. So, A also knows that. A knows B's public key and using B's public key this plain text is getting encrypted into the cipher text.

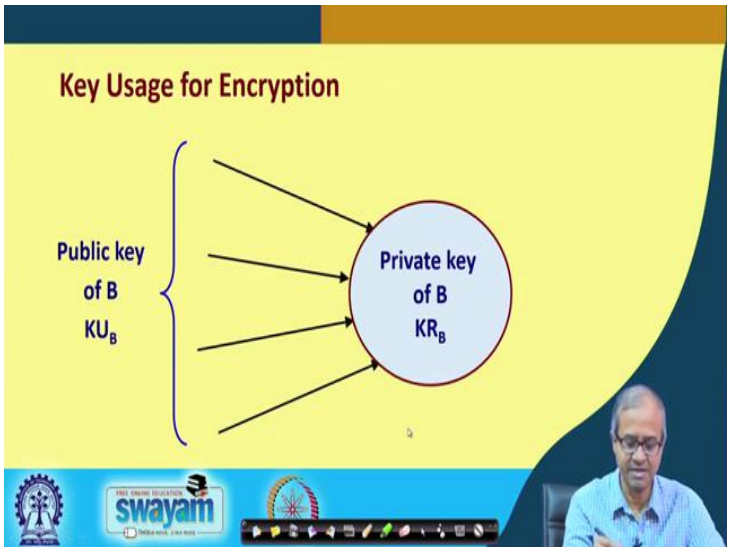
What the receiver is doing? The receiver is running a decryption algorithm, but now this decryption algorithm is using B's private key which is only present with B. This algorithm is such, designed in such a way that if you encrypt using public key, then you can decrypt using private key or also the reverse.

If you encrypt using private key, you can decrypt using the public key. Now in this case for encryption we are saying we will be encrypting using public key which is known to everybody, but it can be decrypted only by the private key so that I can receive message

from many people, because my public is known to everybody, but no one else will be able to decode that messages other than myself, because only I am having the private key, right.

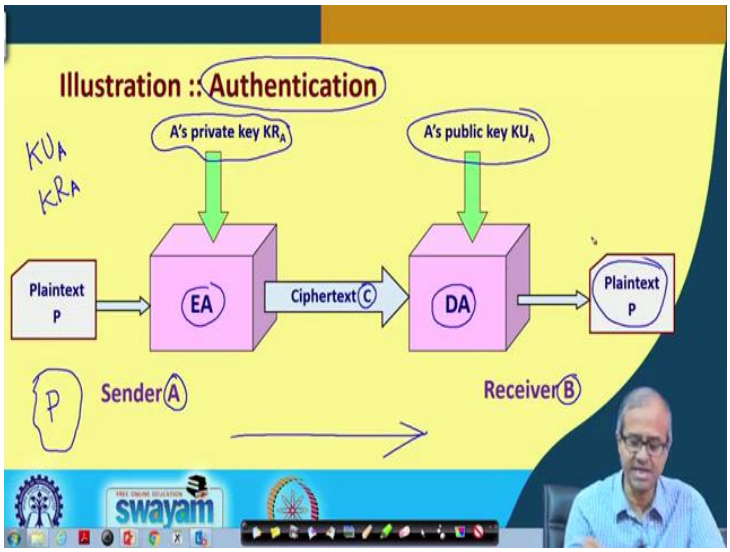
This is how this encryption occurs in public key system. Now just by slightly changing the configuration or the order of these operations. We can also allow some kind of authentication process to take place using the same kind of process.

(Refer Slide Time: 07:26)



First thing is that this is what we are saying encryption. I just now told public key is known to everybody. Private key is only available with B. So, many parties can send the message, but only B can decode.

(Refer Slide Time: 07:43)



Now, let us talk about another application of public key cryptography namely, authentication. Here also let us say A is sending something to B; A sending something to B, but for authentication application we will be using the keys of the sender. A is having a public key KU_A, A is having a private key KR_A. Now you see the purpose of authentication is not to send a message securely, but to make the other person identify or believe that I am the correct person, whom I am claiming to be, right.

Well let us imagine a situation. Suppose A wants to authenticate itself with B. So, what A will do? Let us say A chooses a random text P. This P can be a Shakespeare's poetry, a Shakespeare's text let us say. So, anything which is legible, which someone can read and understand you can select any such plaintext and you encrypt it using your own private key.

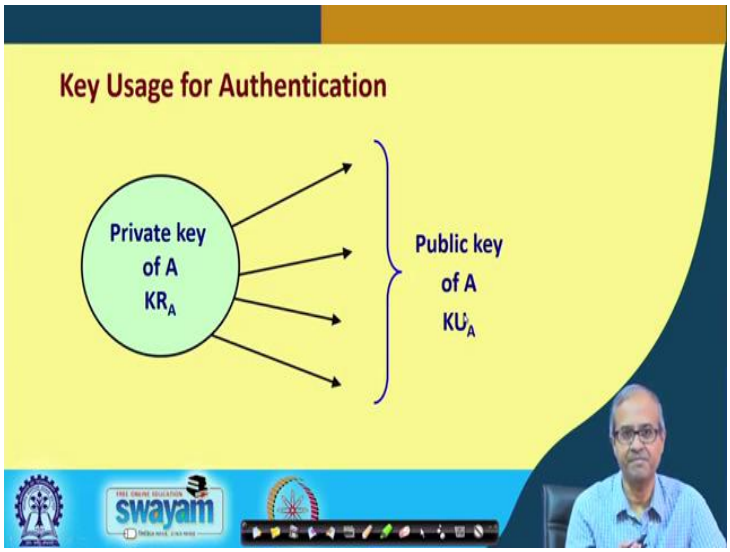
Well, I am trying to authenticate to you. I encrypt it with my private key. I send it over to you. The cipher text is generated. It is sent to the receiver. What the receiver will do? Receiver will try to decode it using the sender's public key which is of course, known.

So, as I said, this public key and private key encryption/decryption are interchangeable. You can use one of them for encryption, the other for decryption. So, what the receiver will see that if I use A's public key for decryption, I am getting back a message which I can read and understand which means I am using the correct public key; that means, it must have been encrypted by A, using A's private key.

So, I can uniquely identify A, right. This is how authentication works in this case. So, just by changing the; but to say here I am saying, I am not talking about the secrets of messages.

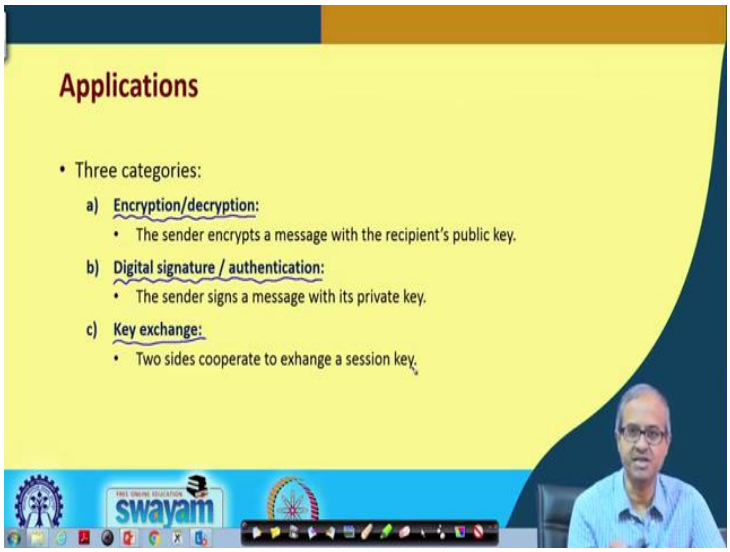
This A is encrypting it using its own private key, the entire world knows my, suppose I am A, the entire world knows my public key. So, anyone can decrypt it, let them decrypt. I do not care, I am only concerned with the receiver B, B should be able to decrypt it and identify that well I am actually A, but let the others also do the same thing I do not care, fine.

(Refer Slide Time: 10:58)



Now, this is the pictorial schematic, now we encrypt using the private key of the sender and it is send out. Anybody, that anybody can verify the authentication of A by using the public key of A decrypting it.

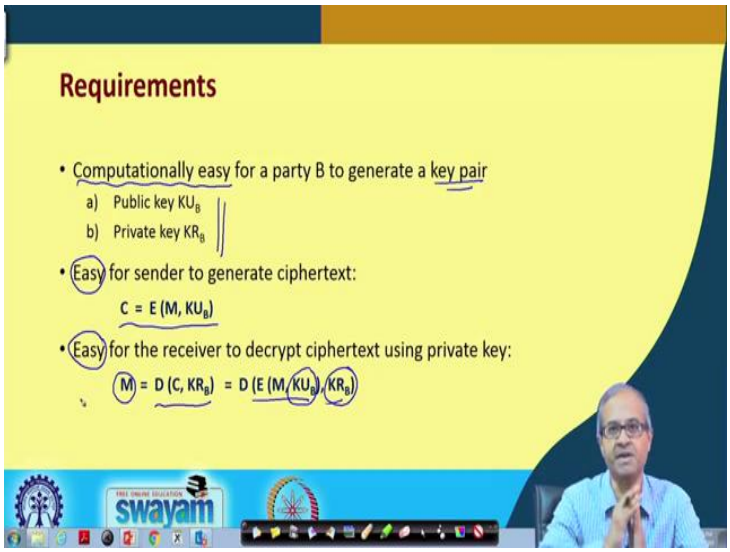
(Refer Slide Time: 11:19)



Now this, application of these public key algorithms encryption/decryption algorithms, broadly you can think of three different applications. First is of course, encryption/decryption as I had said, the sender encrypts the message with the receivers public key, the receiver will decrypt it in its own private key. This is quite possible, but the only problem is, I, we shall be talking about this later, is that these are relatively slow with respect to symmetric key algorithms. These methods will take more time for encryption and decryption. Digital signature is something we shall be discussing later.

So, now, you are aware of this terminology that we conventionally put our ink signature on a paper to authenticate that well, I am the person who read and I am signing it. So, there is an equivalent electronic signature also which is called digital signature that is also becoming quite popular nowadays. That can also be implemented using these kind of public key crypto algorithms and key exchange; 2 persons can cooperate to exchange a key. This we shall see later again.

(Refer Slide Time: 12:45)

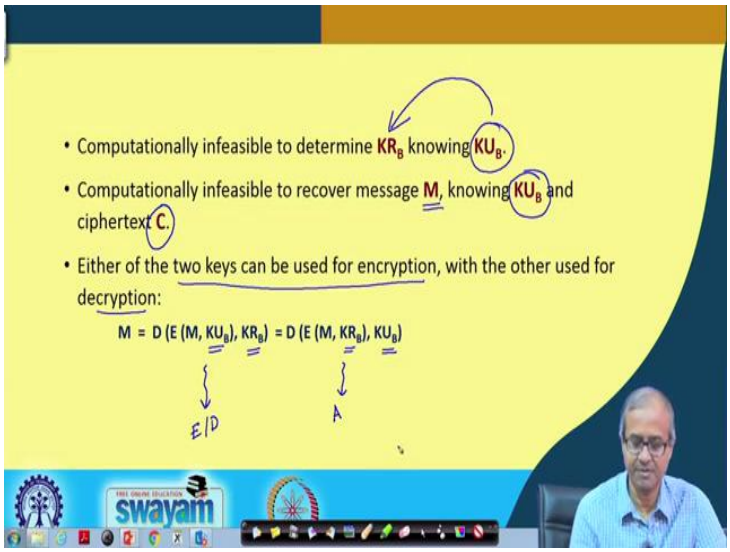


Now for a public key crypto system as I had said every party must be holding a pair of keys, private key and public key. So, there are some essential requirements for this algorithm to use, because in an environment new users can join. So, when a new user joined, the new user should be given a pair of public and private keys new pair. So, it should be computationally not very difficult to generate a key pair, public key and private key.

Somebody should be able to generate that pair of keys relatively easily and these easy are relative terms. So, you should be able to do the encryption process for a message using let us say the public key of B and for the receiver side the reverse process decrypt using the private key of B.

So this encryption/decryption with this will cancel out and you will get back M these are the requirements. So, this algorithm should search that encryption/decryption should be reversible. If you, if you encrypt using the public key and decrypt using private key, you will get back the original thing or even the reverse order you will also get back the same thing. So, these are some of the requirements of public key algorithms.

(Refer Slide Time: 14:21)

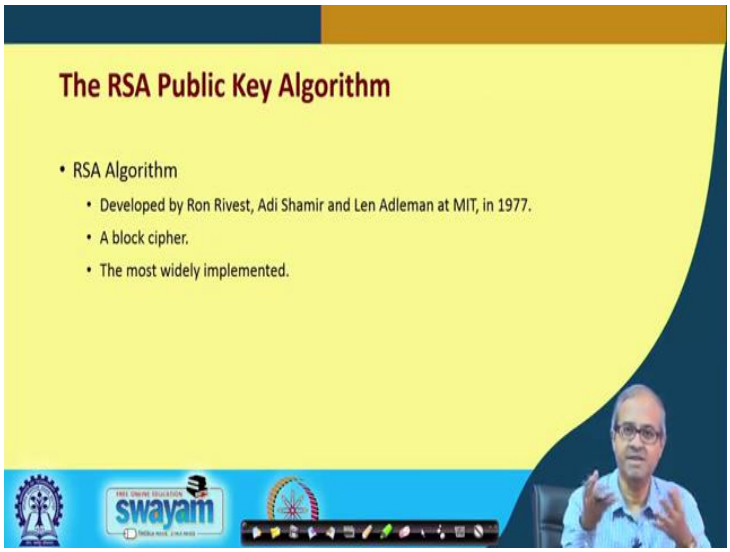


Now, the other things are related to the security of the schemes. Like you see public key is known to everybody, the algorithm should be such or the key generation, key pair, the way they are generated, should be such that, it should not be able to guess or generate the private key from the public key, because my public is known to everybody. So, no one should be able to guess what is my private key, this should be computationally infeasible. It should be extremely hard, mathematically to do this, right.

Similarly, I mean, if I know the ciphertext and if I only know the public key, I do not know the private key of B, I cannot decode it back. I cannot get back the message. This is a requirement and as I had said that the keys can be used in a interchangeable way, either of the 2 keys can be used for encryption and the other for decryption, we will be getting back the message.

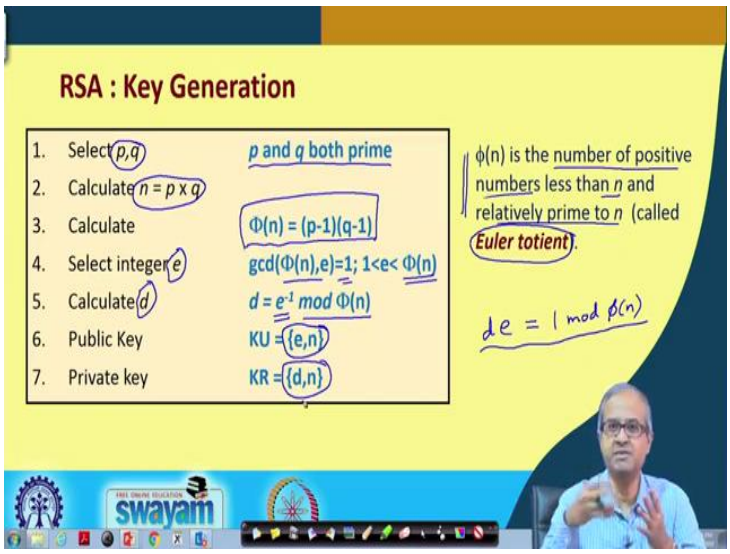
So, either you can encrypt using the public key, decrypt using the private key or you can encrypt using private key, decrypt using public key, but as I had said the first mechanism is used mostly for encryption and decryption and the second mechanism is used mostly for authentication. This already we have discussed.

(Refer Slide Time: 16:01)



Now, this RSA is one of the public key algorithms, which was proposed long back in the 1970s, but it has survived several decades. Still it is considered to be good. In fact, the entire internet is being driven or based on this RSA algorithm as of today. This RSA are the initials of the inventors Rivest, Shamir and Adleman. This is also a block cipher. So, you can take a block of your data of course, the size of the block can vary, not fixed and you can encrypt it, you can send it and this is most widely used.

(Refer Slide Time: 16:51)



I am very briefly telling you how the key pairs are generated in RSA, because you will understand or appreciate that how RSA encryption/decryption works. This is for the key generation, for every party I have to generate a public key and a private key. First thing

is that you select 2 prime numbers p and q. p and q are both prime. Now you must say that well I can choose a prime number 5 and 13; is it good enough?

No, the idea is that these numbers p and q are not small numbers. They are very large numbers. They are of the order of let us say, 300 digit numbers or even more. They are huge numbers. So, in that way there is some kind of computation involved. In this process you are handling very large numbers. Then you calculate the product of these 2 numbers p and q, call it n; then you calculate something called Euler totient and what is defined as $(p-1) \times (q-1)$.

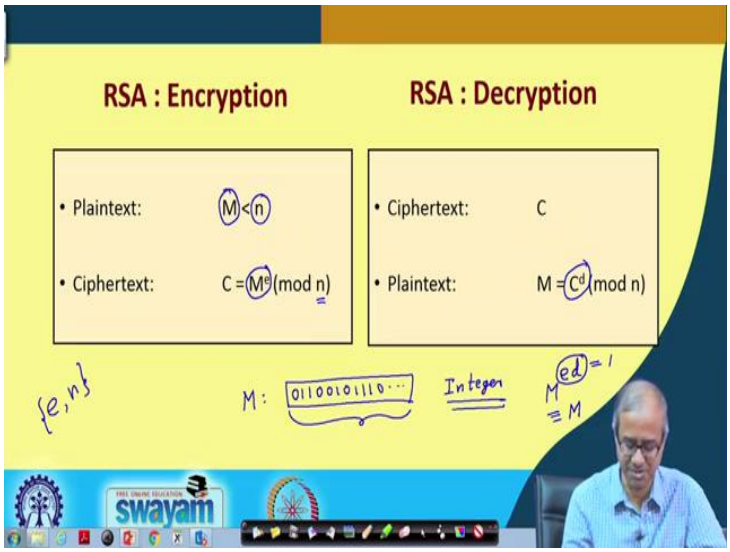
This is actually, if you think, this is the number of positive numbers, less than n and relatively prime to n; n is the product. $(p-1) \times (q-1)$, this cannot factor n. Clearly p and q are both prime numbers.

So, this is the interpretation of Euler totient. In the next step, what you do? You try to select an interior e such that the greatest common division, GCD or HCF. You know of highest common factor, same thing GCD of this Euler totient and this e should be 1 and the other constraint is that the value of e should be less than this phi $\phi(n)$, not greater than that and once you choose a value of e, you determine a value d which will be $e^{-1} \, mod \, \phi(n)$, e^{-1} what is the meaning of this? This means $d \times e = 1 \, mod \, \phi(n)$.

So, I am not going into the details of this number theoretic proofs and derivations, but there are ways to do this. I mean you can find d. Now once you have done this, your key generation is over. So, any pair it is, see (e, n) is one of the keys; d and (d, n) is one of the keys.

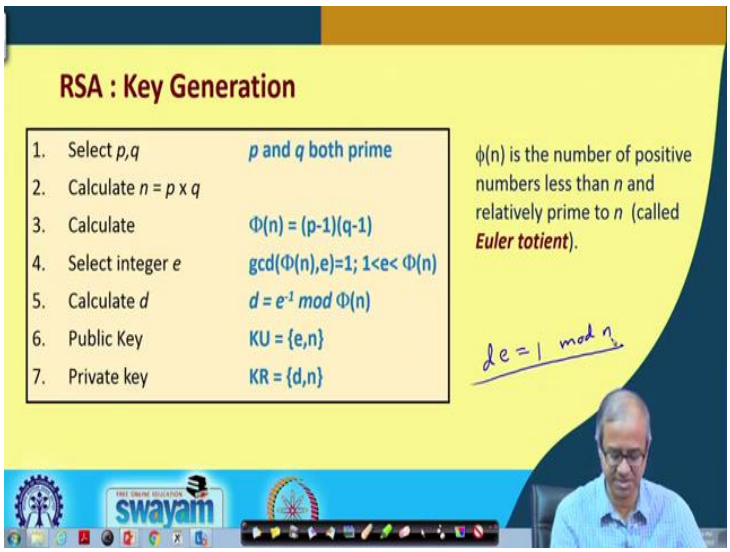
So, any of these pairs you can declare it as the public key, the other pair you can declared as the private key. This is how the key generation takes place. The algorithmic steps appear to be simple, but the problem comes from this size of the numbers. The numbers are very large. In terms of number of bits, they are typically 1024 bits, 2048 bits or even more.

(Refer Slide Time: 20:25)



Now, one thing you just recall here that d and e possess a very interesting property that the product of, sorry.

(Refer Slide Time: 20:42)



$de = 1$. So, I said $de = 1 \; mod \; \phi(n)$, but from number theory you can, it can also be proved that this will be 1 also mod n, So, I am not going into the proof, but just remember this. So, encryption/decryption is very simple. So, how it is done? Let us say I want to encrypt a plain text. Let us say m is the plain text.

Now what is M? The way we look at M is, M is nothing but a stream of zeros and ones. Here we are not talking about alphabets or characters or letters. It is a stream of zeros

and ones anything, any arbitrary stream of zeros and ones. So, this stream of ones, you can regard as a number, you treat your message as an integer. It will, of course, we have very large integer, because your number itself can be quite large. This M can be very large.

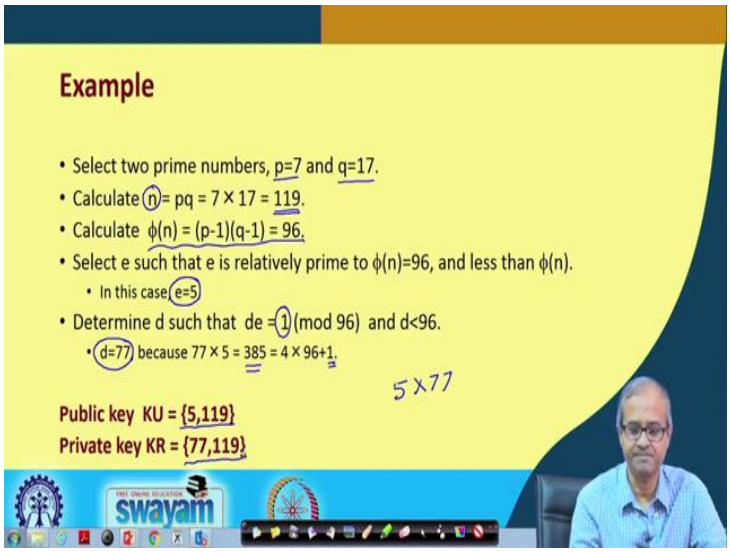
The only constraint is that whatever integer of values represents, this must be less than that value of n that you have selected during the key generation process. This is the only constraint. Your encryption is very simple. You simply M^e, because M is regarded as the integer, integer to the power another integer e, because we had said e and n, this is your public key.

So, using the public key you are doing the encryption. So, you need e. You also need n. So, a large number to the power of another large number, so, it is computationally quite involved. There are efficient algorithms to do this power computation, but still this is much slower as compared to DES or AES encryption ok.

Talking about decryption algorithm, it is extremely simple and very similar, you take the ciphertext, you do C^d. So, whatever C you have got, you raise it to the power d. Now what will happen; C was M^e; C was M^e and you are raising it again to the power d. So, it will become M^{ed}. Now $e\ d = 1$, by definition module n. So, this will be nothing but M. You get back M.

So, conceptually speaking this RSA is very simple, but computationally it is very much complex, because the numbers are very large. You need a of computation to carry out encryption and decryption. If you have message is small its fine, but if your message is very large, you may need a lot of time to do this encryption and decryption.

(Refer Slide Time: 23:58)

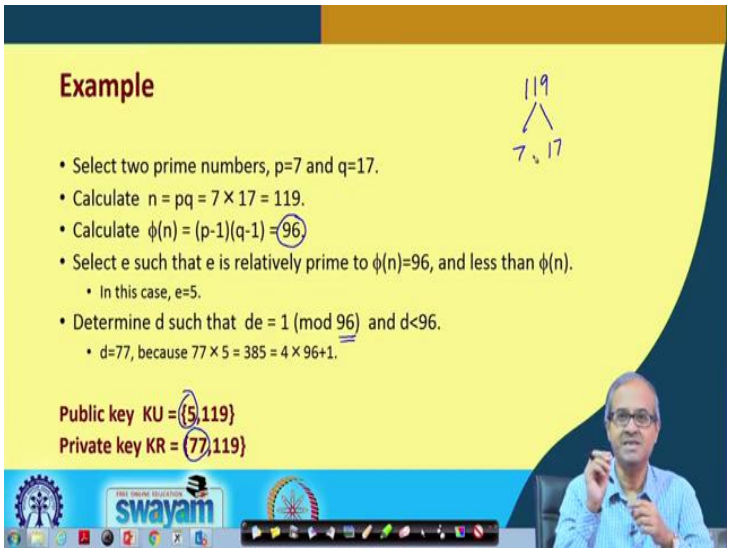


So, let us take a simple example just to illustrate, a very small example. Let us suppose the 2 prime numbers you select are 7 and 17. First you compute the value of n as their product which comes to 119. The Euler totient is coming to 96. Select e such that the $GCD(e, \phi(n)) = 1$, which means relatively prime to $\phi(n)$, that is.

So, they are one of such a value of e, you can take as 5. 5 is one such value. Even verify 5 and 96 are relatively prime their GCD is 1. Determine d such that if you take a product modulo 96 it will be 1.

So, the smallest value of d can be found out to be 77 of course, manually by hand it is very difficult, you need a computer to do this, but you can check 5×77, if you do, it comes to 85, 385 and if you do modular 96 your remainder is 1, divided by 96 remained here. So, d is nothing but 1. So, your public key becomes 5 and 119. Your private key becomes 77 and 119. Now you think one thing, these are publicly known, ok.

(Refer Slide Time: 25:36)

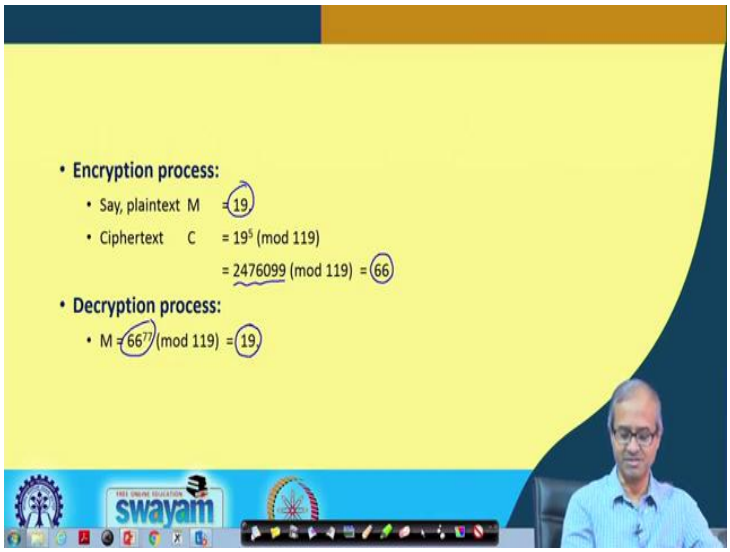


Suppose everyone knows about 119 and everyone also knows that what is RSA algorithm, because this algorithm is not private it is publicly known. So, if it so happened that everybody will know that 119 is what? It is just a product of 2 prime numbers.

So, if I can factor it out, if I can generate the 2 factors, then I can generate also the Euler totient, then from this 5, I can also generate 77 in the same process, because I know 96, So, once you factor this number n, your RSA is broken. Now the cache is factoring two large prime number, the product of two large prime numbers is still today known to be computationally difficult.

So, the security of RSA is based on this assumption that factoring the product of 2 prime numbers is currently not possible, large prime numbers of course. The day it becomes possible, RSA will be immediately broken and you cannot use RSA anymore right.

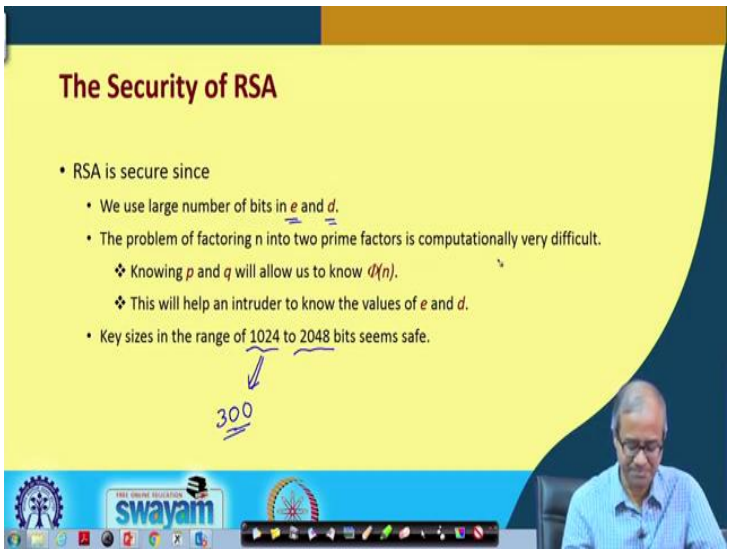
(Refer Slide Time: 27:04)



So encryption process; another simple example, suppose your plain text is 19. So, when you do an encryption, $19^5 mod\ 119$. So, if you calculate it comes to this modulo 119 is 66 divide, take the remainder. For decryption process you see even for this small numbers 66^{77}. This is a very huge thing.

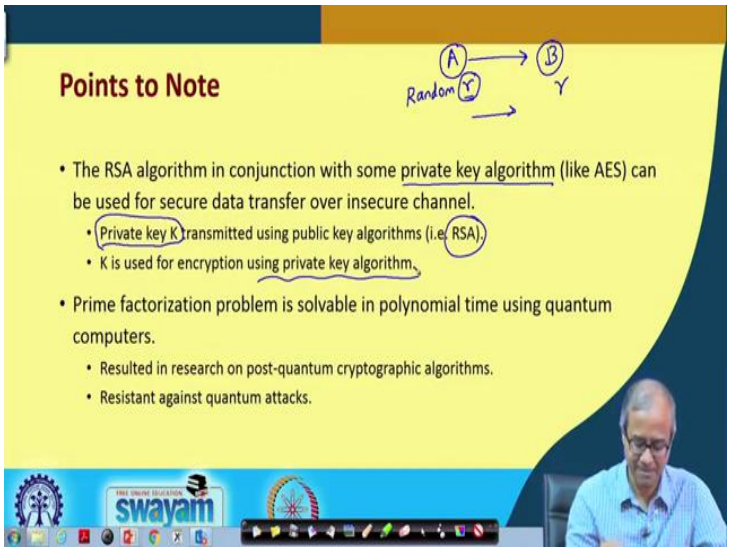
Well, there are efficient methods to calculate modulo power of one number to the power other. If you do this, you will see that you will get back the original 19, this is how RSA works basically.

(Refer Slide Time: 27:52)



Now I just now mentioned that RSA is secure. People use RSA, because we use very large number of bits in e and d. They are on the order of 10242, 2048, even more number of bits say 1024 bits that equivalent to approximately 300 digits numbers, 2048 means about 600 digits, more than 600 digit number. They are huge numbers. The problem of factoring n into 2 prime factors is computationally very difficult. So, the security of RSA is based on that assumption, right.

(Refer Slide Time: 28:45)



Now, there are two points I want to highlight here that RSA algorithm can be used in conjunction with some private key algorithms, because you see RSA algorithm is good over the internet, because you can take the public key of the other party wherever he is, encrypt it and send it to that person. That person can decrypt it using his or her own private key, but the trouble is that this public algorithmic RSA is very slow as compared to DES or AES.

So, what do you do? Whenever you want to send a, let us say large message to some parties sitting somewhere far apart in a secure way, well you can do it in a 2 step process. Like let us say I am giving an example, let us say a party A wants to send a message in a secure way to a party B.

First step, A will be generating some random number. Let us say it generates a random integer, a large integer r ok.

Then this r is regarded as the private key k for symmetric encryption, and this r is being sent to B by encrypting using RSA. This is feasible because this is very small. Encrypting a small numbering using RSA is still not that slow.

Now once you do it, the other party can decrypt it back using RSA, using its own private key and get back the same random r. This same number r will be shared by the 2 parties. That can be used as a symmetric key and you can use some algorithm like AES for encryption and sending now, because both parts are having r that will be your AES key, ok.

So, you are sharing the key, in the first step using public key algorithm, in this second step you are using a symmetric key algorithm for actual encryption. This is what happens in the internet today.

You do internet banking. You do secure http; you just log into Gmail. So, whatever you do over the internet, whatever transaction you do online, this kind of a thing happens in the backend. So, you are relying on RSA for the initial exchange of the key and once the key is exchanged securely, you can communicate, because it s a one time key. You do not have to remember this key for long. Once the session is over, you can forget the key. Now the problem is this prime factorization problem on which RSA is based, this can be solved using quantum computers.

Now there has been lots of research going on in quantum computers, small scale quantum computers have already been built by big giants like Google, IBM, Microsoft and there are several other companies now in India. Also there have been some efforts which are going on for building indigenous quantum computers.

So, once a large enough quantum computer has been built, you will be able to factor this large prime products and RSA will be broken, but of course, you are rest assured. It will not happen tomorrow or within a few years; it will take much longer time to built a quantum computer of that size, but the way things are progressing the day is not that far ahead when you will be having a quantum computer that we will be able to break this problem; mathematical problem, break RSA and your entire internet infrastructure will crash down; come crashing down. You cannot use any kind of online transactions as we use today because everything can be broken.

So, there is lot of research on something called post-quantum cryptographic algorithms, which are supposed to be resistant against attacks using a quantum computer, quantum attacks. So, such things will become, you see some of the companies are already come up with some protection against this kind of quantum attacks.

So, they are already upgrading their encryption/decryption infrastructure using this kind of methods, So, with this we come to the end of this lecture where we discussed about public key cryptography in particular the RSA algorithm and finally we talked about some of the possible future witnesses of RSA, once a reasonably sized quantum computer become available.

Thank you.