

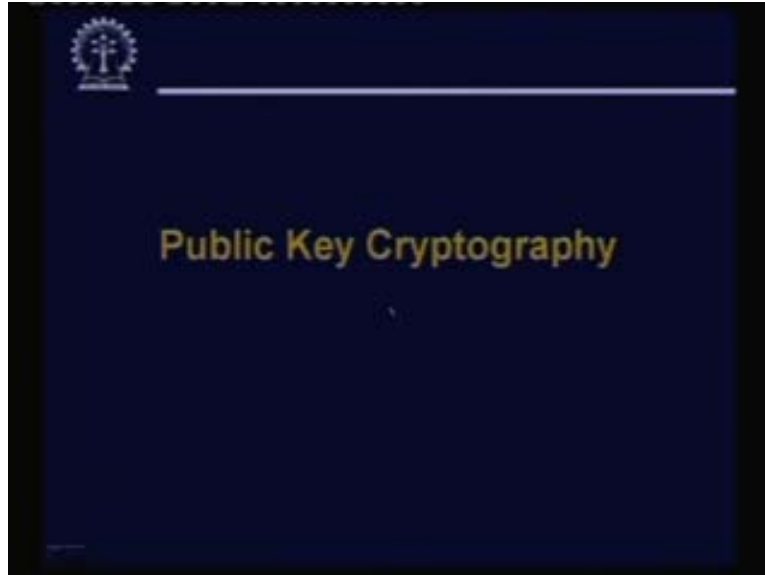
**Internet Technology**  
**Prof. Indranil Sengupta**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Kharagpur**  
**Lecture No #33**  
**Basic cryptographic concepts – Part 2**

In our last lecture we were talking about some of the network security issues and some of the cryptographic concept. In particular we talked about the symmetric key encryption decryption techniques. We looked at some of the classical algorithms like Caesar cipher, mono alphabetic cipher, transposition cipher and you also looked at some of the practical schemes that people used like DES, triple DES and you also mentioned about the standard people which use nowadays AES. So, continuing with the discussion.

(Refer Slide Time: 01:44)



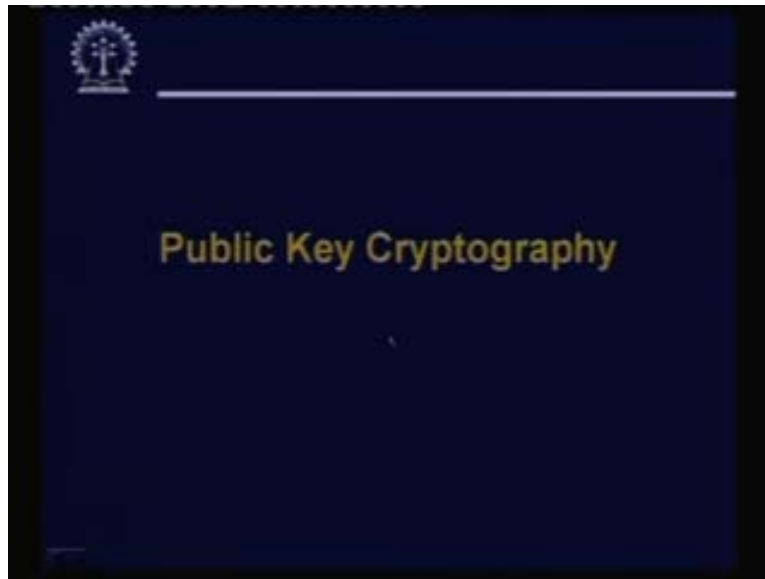
(Refer Slide Time: 01:50)



Today, we shall be first looking at public key cryptography. Let us first try to understand the motivation before trying to explain what public key cryptography is. See symmetry key cryptography methods like DES, triple DES, AES all are fine. They are pretty strong algorithms. They are very difficult to break secrecy of the key is central point in guaranteeing the secrecy of information. But you try to think about one thing try to answer one question that how does this technology help me in the present day internet scenario? I am not communicating with somebody who is just beside me or in the same office. I may be communicating with somebody which is who is far off, who is located in a geographically remote location with respect to myself. So we can use methods like DES, triple DES.

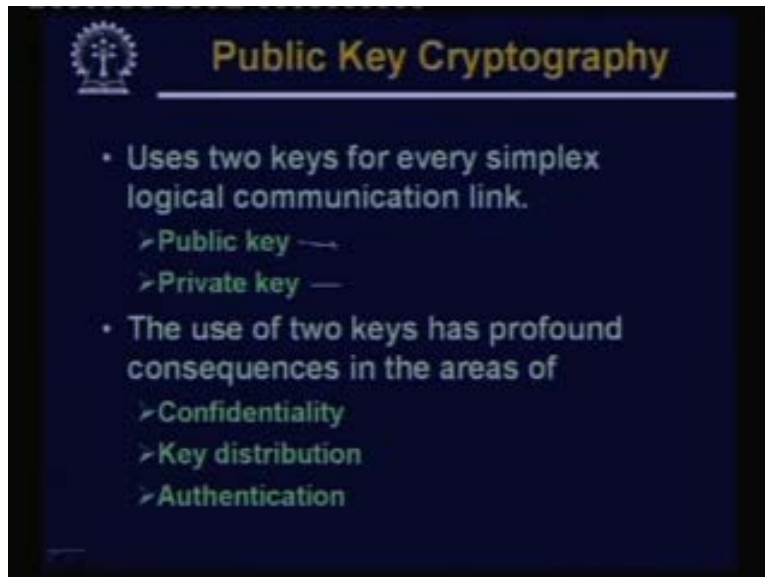
Alright, but now the question arises how do we agree upon the common secret key because unless both of us have this same key we cannot start using either DES or AES? In the internal scenario this is the biggest problem that this kind of symmetric encryption schemes faces. Let me go one step ahead and tell you that the symmetric encryption schemes without any other support are virtually unusable. Unless you have a very reliable courier whom you can write the key on a piece of paper, write a sealed letter envelope. That envelope will get delivered to the other person other person opens it and then you can start using that secret key. That is of course one solution but that has very limited utility. You cannot use it in general. Now this public key cryptography comes to the rescue.

(Refer Slide Time: 03:55)



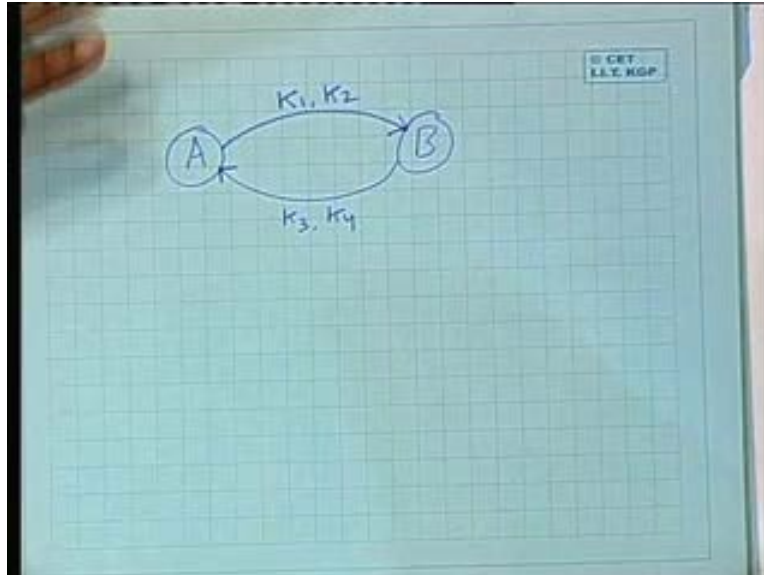
Let us try to understand the basic concept behind public key cryptography.

(Refer Slide Time: 04:01)



Here we are using two keys for every simplex logical communication link. Simplex means a link in a particular direction.

(Refer Slide Time: 04:16)



Suppose when I am saying there is a node A; there is another node B. A is sending some data to B. This is considered to be a simplex channel and on this simplex channel I am saying that there are two keys, say  $K_1, K_2$ . Similarly if we look at the communication in the other direction there can be a pair of the keys  $K_3, K_4$  say. So for every simplex links there are two keys. This is one of the main concepts in public key cryptography.

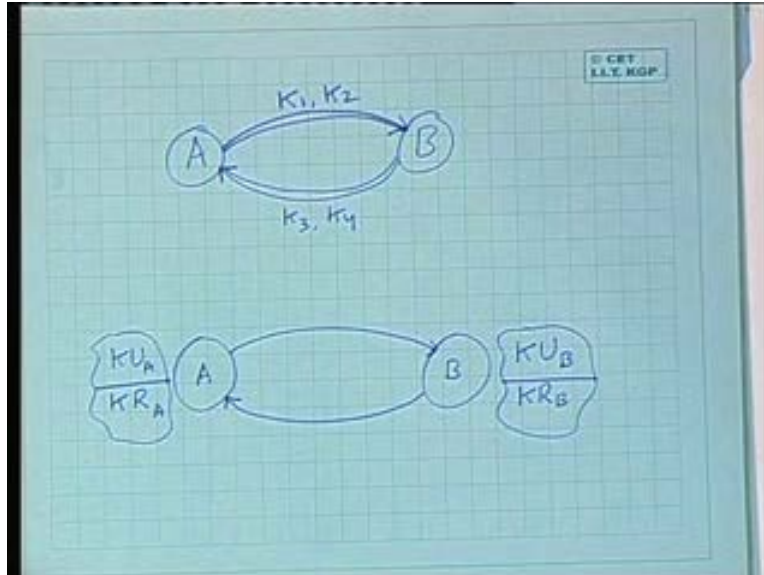
(Refer Slide Time: 04:49)

**Public Key Cryptography**

- Uses two keys for every simplex logical communication link.
  - Public key —
  - Private key —
- The use of two keys has profound consequences in the areas of
  - Confidentiality
  - Key distribution
  - Authentication

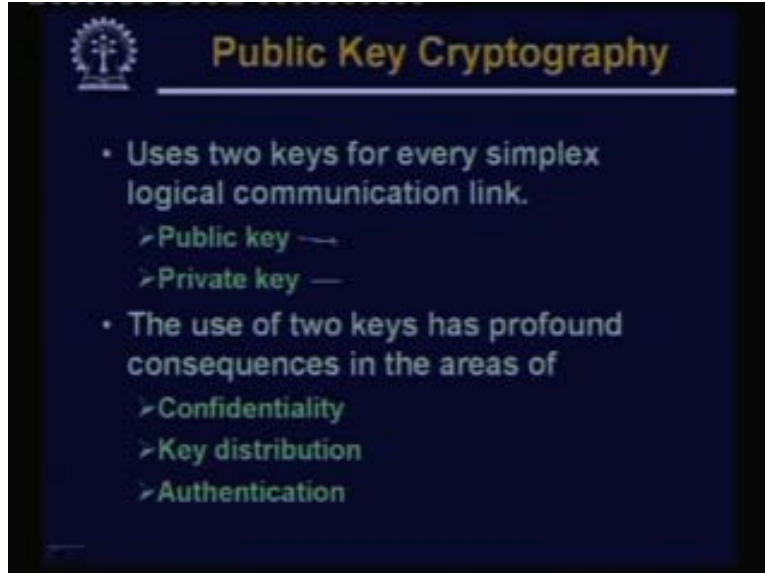
And these two keys are traditionally called public key and the private key. Now these two keys are typically generated by every node.

(Refer Slide Time: 05:15)



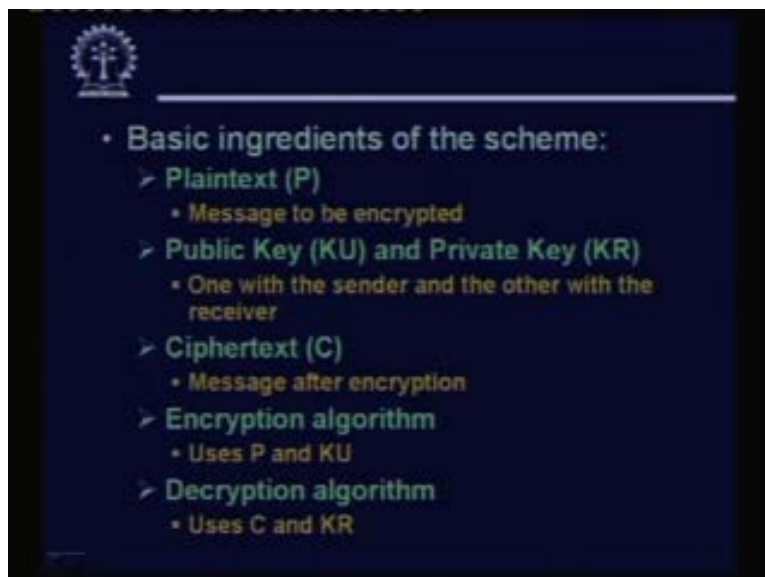
Like again if you look back at this diagram, I am saying that on this particular, there are two keys key1, key2. On this particular link there are two keys  $K_3, K_4$ . But I can also consider the keys in a slightly different way I can say B has two keys one we are calling public key of B. This is the notation we use  $KUB$  and private key of P. Similarly A will be having public key of A, private key of A. So now you can suitably define the simplex communication links by suitably telling by which pair of keys will be used for which communication link. Now as such when these pairs of keys are generated with respect to a particular node for A there is a pair of keys, for B there is a pair of keys. So when a communication takes place, say suppose I am A; and you are B when I want to send something to you I will be using one of my two keys and you will be using one of some keys. So some information available to both of us we can use for the communication. We shall see this more detail.

(Refer Slide Time: 06:43)



Now we shall see that this public key cryptography can be used in a number of different ways. In particular, confidentiality, key distribution, authentication. These are the three important areas where we can use this technology. Confidentially means secreta communication which symmetric key systems can also do. Key distribution is new thing which is coming up which I have said that key distribution is a problem in internet scenario. Now here I am claiming public key cryptography can help solve this problem and authentication is another problem which also can be solved. Let us how.

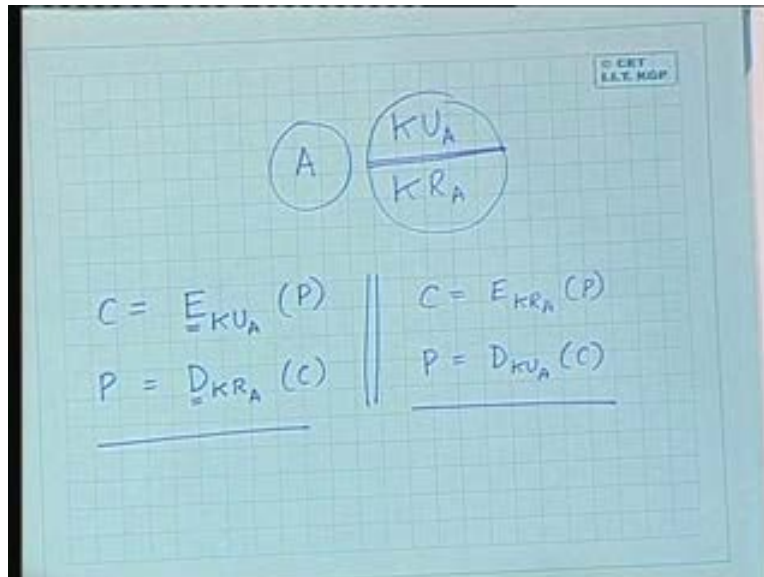
(Refer Slide Time: 07:26)



See before going further see what are the basic ingredients or the components of this particular scheme. As usual the message to be encrypted is referred as plaintext denoted by P. As I said with every node we associate a pair of keys; public key, private key. Now as the name implies see we are calling public and private. Why do we call them? Suppose a pair of keys public key and private key. Private key; I am supposed to keep in my pocket. I will not show it to anybody but public key is something which I am supposed to tell everybody that this is my public key, this is true for everybody.

Public key is supposed to be a key which is broadcast to all the parties with whom you are expected to communicate. But private key is something which you keep secret with you only and to nobody else. Private keys reside in only one place; public key can reside in multiple places. Of course you have this ciphertext after encryption while here there can be some variations for encryption algorithm, you use the plaintext. You can use the public key for decryption. You can use the ciphertext using the private key. Now here I want to make a point. This pair of keys which are generated I am telling you like what I am saying.

(Refer Slide Time: 09:15)

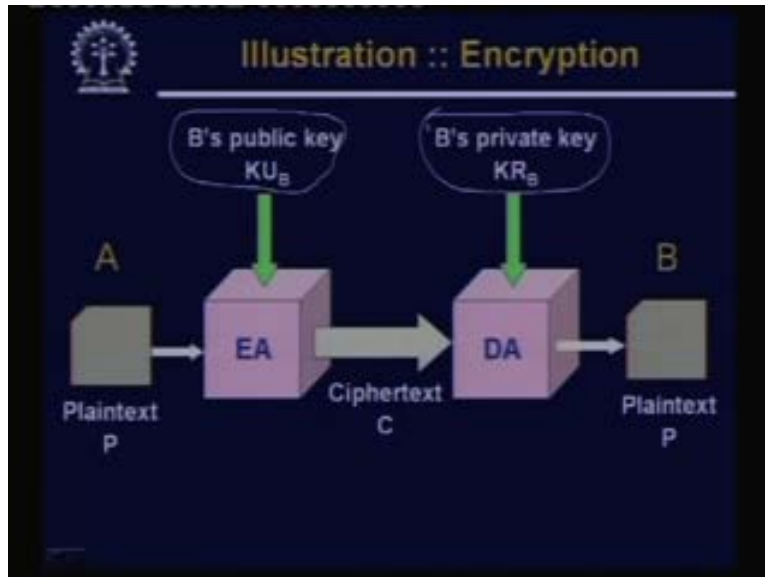


For any particular node A, I am saying that there are a pair of keys which are generated public key of A and private key of A. Now these pair of keys they share a very unique relationship between themselves. The relationship is like this. Suppose if I use, if I encrypt using K<sub>U<sub>A</sub></sub>, a plaintext I call it this ciphertext. Then I can decrypt it back by using the decryption algorithm by using the private key. This means that if I encrypt it using the public key I can decrypt it using the private key. Now my encryption and decryption algorithms can be slightly different.

Now another point to notice is that the public key and the private key are interchangeable in terms of the function which means, that I can encrypt also using the private key and I can decrypt it back using the public key. So these two things you should keep in mind.

Based on this property this basic public key cryptography scheme can be used or utilized in a number of different very interesting ways. Let us see.

(Refer Slide Time: 11:00)



The first illustration, we show here is that of conventional encryption or secret communication. Here you try to understand A is the sender B is the receiver; A want to send to B. The way things progress you see encryption algorithm encrypts the plaintext using the public key of B, public key of the receiver. So ciphertext flows through the network. On the other side decryption is carried out using the private key of B. Just now we had said that if you encrypt using the public key you can decrypt back using the private key. Now in this scheme suppose you want to send some message to me then you will be encrypting using my public key which I have told everybody.

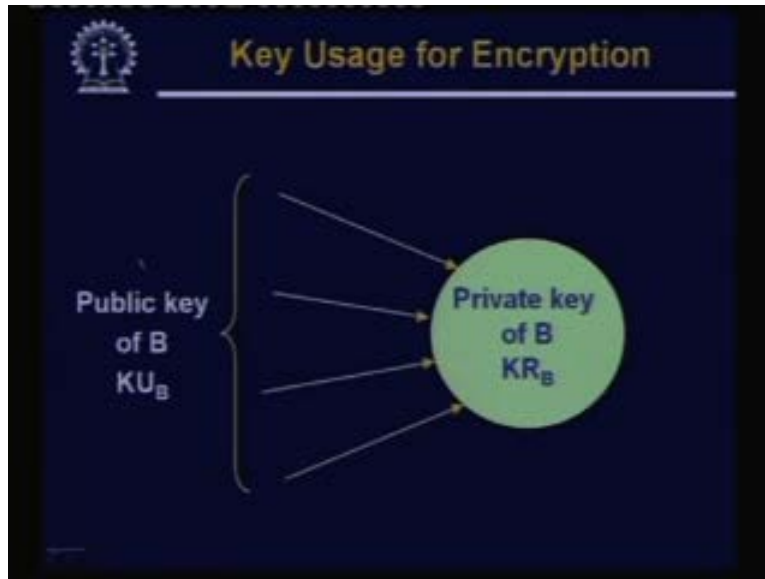
So in fact everybody can send message to me if they want. But only I can decrypt the message because only I have the private key and nobody else. So the idea is very simple. Anyone can send the message to me, but other than myself; no one can decrypt. This kind of a thing was not possible for symmetric key cryptography because both the parties were having the same key. So now you see now you have one method which will enable you to send some data to a remote location very secretly like some website. Suppose you have a website and you want to send some very secret information to the website. The website can give you that will this is my public key you encrypt that secret information using a public key and send it to me.

Well of course I must first be sure that whether the website that means wherever talking to be that indeed is to website. So some authentication issue is there. But after it is done I can simply encrypt my secret information that can be my credit number also. I can encrypt it using other parties, public key and I can send it. Now even if an intruder intercepts the packet in between that packet cannot be decrypted because that packet can be decrypted only by that website. Because that website only has the private key the



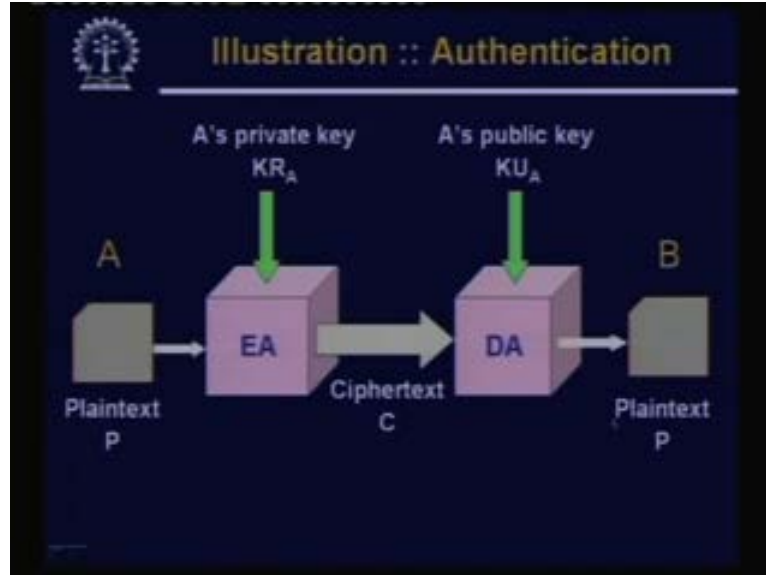
corresponding private key. So this is the basic idea that over the internet you can even carry out very secure transmission or transaction using this basic concept of public key cryptography.

(Refer Slide Time: 14:06)



So in this model as I said pictorially this can be represented like this. This is the person to whom you are trying to send. So the private key is lying only here but the public key is in many different places. Now the only problem is that since the public key is available with everybody, so everybody can send me messages. But well that I am allowing that I will have to leave with it. So I may not like everyone sending me messages. But this scheme allows one to do so because public key if it is truly public then anyone who has access to the public key can send a legitimate message to me.

(Refer Slide Time: 15:00)



Now with slight modification we can also use this method for authentication. Like authentication is the other part. So sending something securely is one thing. The other part is that I must be sure that the person with whom I am talking is indeed the person who I believe that fellow is. So authentication is also a very big problem. Whenever I talk to somebody, on the other side I am communicating. I cannot see that person, but I am believing that person is x. So how to verify that person is x? So this public key cryptography can again come to the rescue. See this is done like this. Here we are thinking of a scenario where the two parties again A and B; and A wants to authenticate itself to B.

Suppose I am A, you are B. I want to tell you that will please trust me I am A. This is what we are trying to do. Now the mechanism here is like this. A generates any arbitrary plaintext which is a plain English plaintext that can be as simple as hello, good morning. That is some legitimate English text anything. The sender or A will be encrypting this plaintext with its own private key. You see now the usage of the key has been reversed for secure transmission. Let us go back we were using the public key of B for transmission encryption. But here we are using the private key of A for encryption and you are sending. On the other side B is using the public key of A for decryption.

Since both are A's keys, so B will be able to successfully decrypt. So now you try to understand what is happening here. Now if I had sent a message hello good morning to you. You would be able to decode finally and get back a legitimate English message. Now if you find that what you get back after decryption is a legitimate message then you will know that the message is actually coming from me because you think. Otherwise it is not possible someone else sending you a message and you are decrypting using my public key getting back a legitimate English message the probability is very **very very very** small. So here you are not talking about message secrecy.

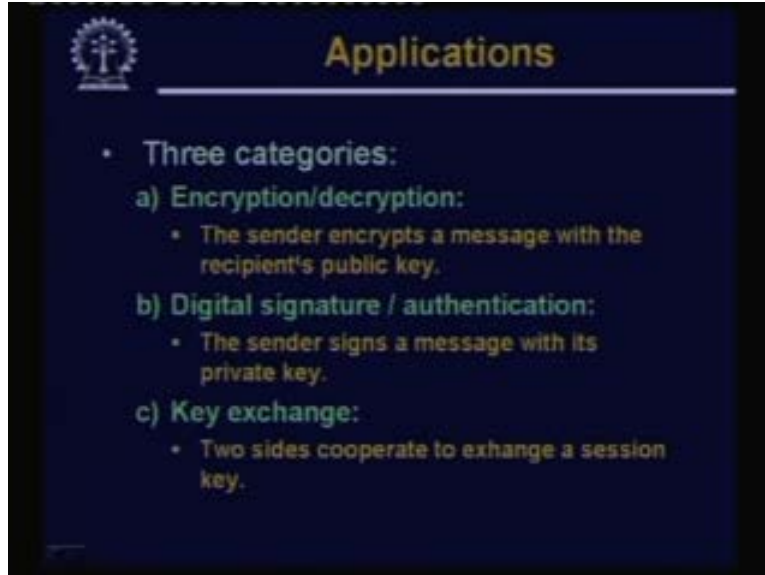
This hello good morning, anyone who is having my public key can get my packet decodes it and gets that hello good morning. But hello good morning is nothing secret. This is just a mechanism for me to authenticate to the other end. So the other end can be sure that it is actually coming from me. See if the other end is able to decode a decrypted successfully into a legitimate message, then the only situation which can lead to such a thing is the message was encrypted by my private key. And since my private key is only in my pocket, no one else is means no one else will be able to do such a thing. So it must have been done by me and me only. So this is how authentication can be carried out using public key system.

(Refer Slide Time: 18:56)



Now here diagrammatically again the usage is different. We are sending by using the public using the private key of the sender and decrypting is carried out using the public key of the receiver. Now as you can see by changing the way we are using the keys we have suddenly some very interesting new application which has come up. We can send something securely over the internet. We can verify the identity of a person or so over the internet. Internet is an anonymous communication medium. We cannot see the other end. But this public key cryptography helps us immensely in using or means in using this unanimous infrastructure to our advantage so that we can also carry out some very serious transactions. Today we talk about so many transactions ecommerce monetary transactions. So, many kinds of information flow in the net which are very confidential in nature. Now these kinds of technologies allow us to do so.

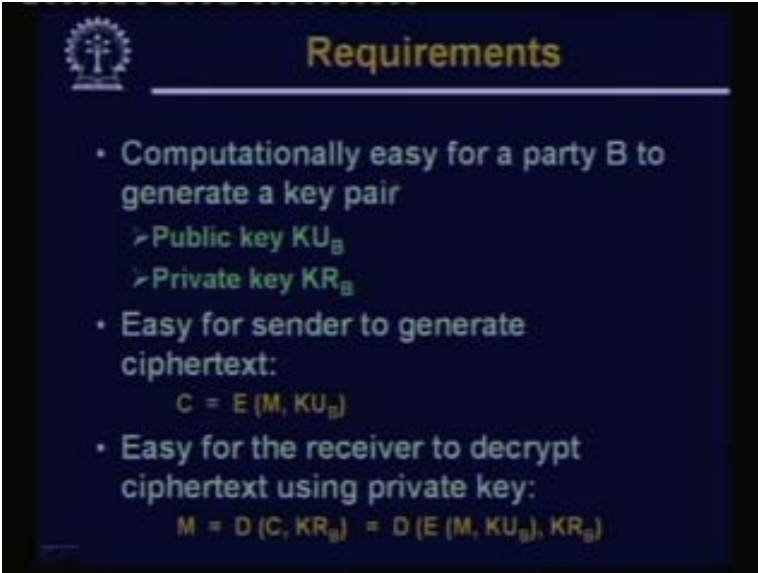
(Refer Slide Time: 20:12)



As we have said talking about applications of public key systems, there are broadly three categories of applications you can categorize. One is simple encryption decryption. This we have already seen here. The sender can encrypt a message with the public key of the recipient and the recipient can accordingly decrypt it. Second application is authentication or some kind of a digital signature you can say. The sender signs a message with its private key. Now this message is not something private and lastly key exchange this we shall see later. The two sides can also cooperate to exchange a session key. See if I am using public key cryptography I can share key with my remote friend.

Like suppose we want to use symmetric key cryptography. Suppose but we do not have any easy mechanism to share a common key, so what we say is that, let us agree upon a public key system. But I generate a public key and a private key I give my public key to you. So you generate a key randomly, you generate a random key yourself, do not tell anybody else. You will encrypt it using my public key and send it to me. I will be able to decrypt it. No one else will be able to do so. Now both of us can have the same shared key. So, public key cryptography can also be used for sharing or sending receiving secret keys.

(Refer Slide Time: 21:55)



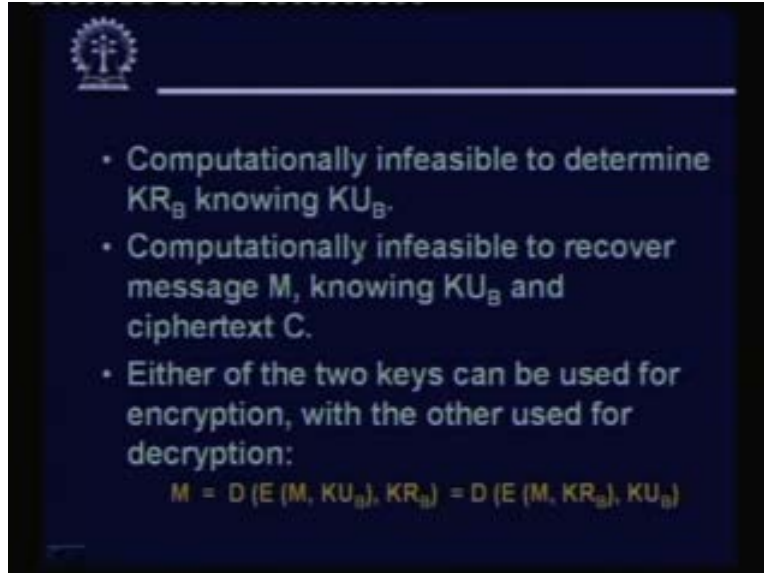
**Requirements**

- Computationally easy for a party B to generate a key pair
  - Public key  $KU_B$
  - Private key  $KR_B$
- Easy for sender to generate ciphertext:  
 $C = E(M, KU_B)$
- Easy for the receiver to decrypt ciphertext using private key:  
 $M = D(C, KR_B) = D(E(M, KU_B), KR_B)$

Now some of the requirements of a public key cryptography algorithm, firstly it should be computationally easy for a party to generate a key pair the public key and private key. Well this easy difficult they are related terms, it should not be computationally too time consuming for generating these pair of keys. Easy for the sender to generate ciphertext, easy for the receiver to decrypt ciphertext, again this easy is a very subjective term. Now we shall see later with some quantitative figures. But let me tell you right now that public key cryptosystems are very good.

They are very attractive but the only problem is that they are very slow to operate as compared to symmetric key cryptosystems. So, symmetric key cryptosystems also has its own uses. Suppose I have a very big file, I have a one gigabyte file. I want to send it to this is a confidential file I wanted to send it to you. Public key infrastructure is there, but I cannot possibly use public key cryptography to encrypt it because it will take me possibly hours to encrypt such a big file. But what is the faster method? Faster method is I can use public key cryptography to share a secret key between us and use that secret key for symmetric key encryption decryption which will be much faster.

(Refer Slide Time: 23:45)



• Computationally infeasible to determine  $KR_B$  knowing  $KU_B$ .

• Computationally infeasible to recover message  $M$ , knowing  $KU_B$  and ciphertext  $C$ .

• Either of the two keys can be used for encryption, with the other used for decryption:

$$M = D(E(M, KU_B), KR_B) = D(E(M, KR_B), KU_B)$$

So some other requirements are as follows. It should be computationally infeasible to determine one key knowing the other. See public key is known to everybody. But knowing the public key should not be very easy; should not be computationally feasible for anybody to derive the private key. Similarly knowing the public key and the ciphertext no one should be able to decode an encrypted message that should also be computationally very hard. The last point I have already said that the two keys are interchangeable in terms of the encryption decryption capabilities. So you can use any one of them for encryption. The other one has to be used for decryption.

(Refer Slide Time: 24:37)

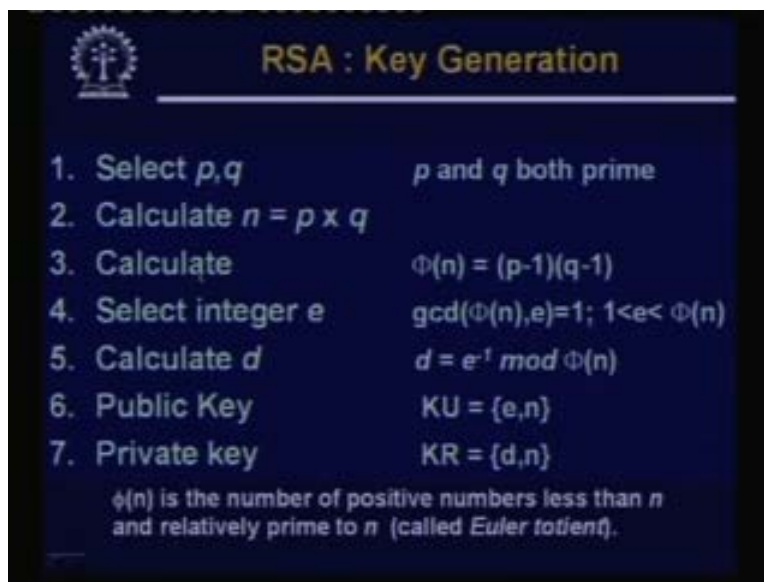


### The RSA Public Key Algorithm

- RSA Algorithm
  - Developed by Ron Rivest, Adi Shamir and Len Adleman at MIT, in 1977.
  - A block cipher.
  - The most widely implemented.

Now this RSA is perhaps the most widely used or I can even say the only algorithm used for public key cryptosystem. Nowadays because of its flexibility, because of its power and up to date no weakness of the RSA algorithm has been found. This is not a very new algorithm. This algorithm was proposed as early as in 1977. The name RSA comes from the first letters of its developers Rivest, Shamir and Adleman. RSA is also a block cipher, but the size of the block is not fixed. The size of the block is up to the user it is up to you whether you use an 8 bit block, 100 bit block, 200 bit block or 1000 that is entirely up to you. How many bit blocks are using, that is entirely a design decision from the point of view of the user. And as I said as the public key algorithm, this is the most widely implemented.

(Refer Slide Time: 25:44)



**RSA : Key Generation**

1. Select  $p, q$                        $p$  and  $q$  both prime
2. Calculate  $n = p \times q$
3. Calculate  $\phi$                        $\Phi(n) = (p-1)(q-1)$
4. Select integer  $e$                  $\text{gcd}(\Phi(n), e) = 1; 1 < e < \Phi(n)$
5. Calculate  $d$                        $d = e^{-1} \text{ mod } \Phi(n)$
6. Public Key                         $KU = \{e, n\}$
7. Private key                         $KR = \{d, n\}$

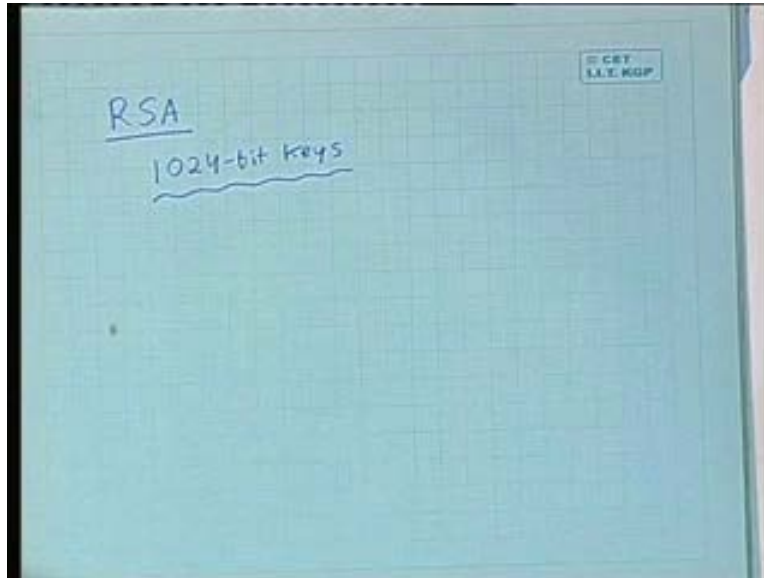
$\phi(n)$  is the number of positive numbers less than  $n$  and relatively prime to  $n$  (called Euler totient).

First let us look at how the public key and private key pairs are generated. See this algorithm which will be generating pair of the keys. It is important to know and understand this algorithm because this will help you in also understanding the way encryption and decryption works much better. The way key generation proceeds is like this. Select two numbers  $p$  and  $q$ . Both are prime numbers, calculate their product  $p$  into  $q$  call it  $n$ . Then you calculate a number  $p$  minus 1 into  $q$  minus 1 that you denote as  $\phi n$ ;  $\phi n$  has a significance this is called the Euler totient. This actually indicates the number of positive numbers less than  $n$  and relatively prime to,  $n$ . This you can easily verify this from this product.

The definition follows after you have computed the product of  $p$  minus one  $q$  minus 1. You select an integer  $e$  which is relatively prime to  $\phi n$  which means GCD of  $\phi n$  and  $e$  will be 1 while  $e$  must be less than  $\phi n$  is a positive numbers greater than 1 you try to find out such  $e$ . After you have got such  $e$ , you try to find out  $d$  such that  $d$  and  $e$  are inverse. Which means  $d$  into  $e$  will be 1 modulo  $\phi n$  or  $d$  equal to  $e$  inverse mod  $\phi n$ . Now you can define public key as the combination of  $e$  and  $n$  and private key as  $d$  and  $n$ .

Now one thing you understand here that the method apparently simple. But one thing you remember typically RSA algorithm today the secure version of RSA.

(Refer Slide Time: 28:11)



They work with key sizes of 1024 bits or more keys or even more. So the keys are so big which means this number  $p$  and  $q$  the random numbers which you are generating. They are very large random numbers their product will be even larger. So here, all the processing you have carried out, you are working on a very large number in terms of decimal. It will be something in the tune of 300 to 350 digits; so big numbers.

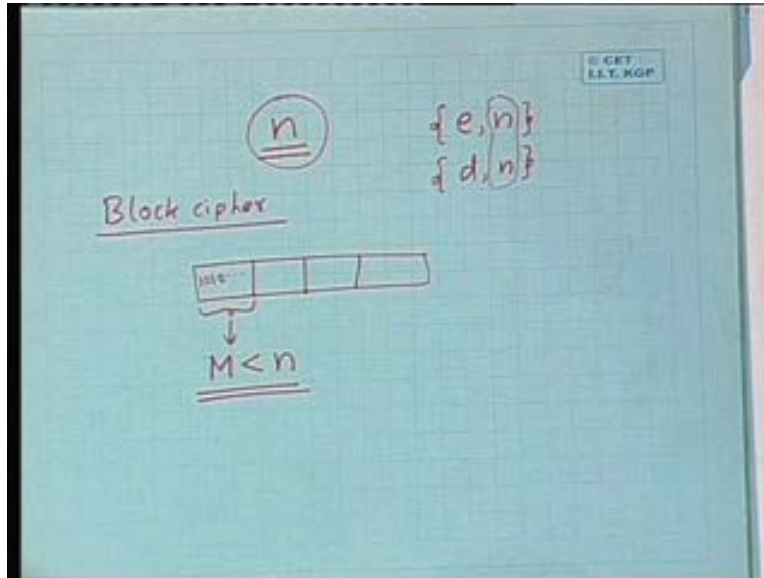
(Refer Slide Time: 38:56)





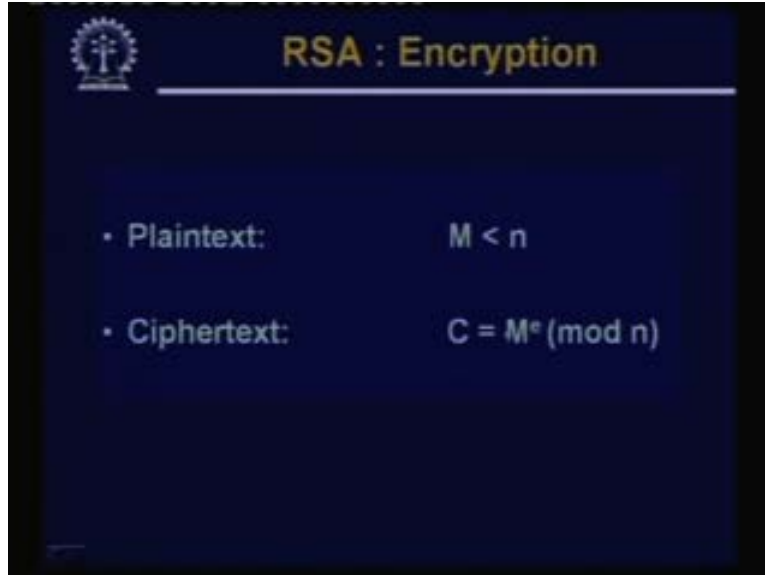
Now let us look at the process of encryption and decryption. Here encryption decryption conceptually is very simple. Now  $n$  is something which you already know.

(Refer Slide Time: 29:11)



I am trying to explain. Say  $n$  is part of the key, as I said in one key you take  $e$  and  $n$ , in the other key you take  $d$  and  $n$ . Out of these two anyone you can call as the public key, the other you can call as the private key. So actually  $n$  is known the other part is secret for the private key. Now this  $n$  is a number. This is the block cipher I told you. So whenever you have some data to be encrypted, you divided it up into blocks. Each of the blocks you treat it as a number. This will be a stream of zeroes and ones you treat it as a number call it  $M$  the constraint is  $M$  must be less than  $n$ . So this defines what is the maximum block size you can have? The choice of  $n$  will constraint the maximum block size you can have.

(Refer Slide Time: 30:24)

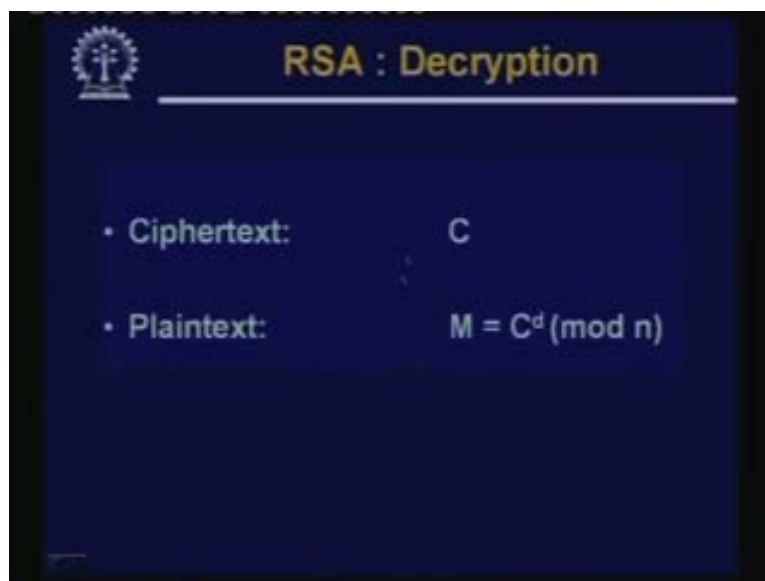


The slide is titled "RSA : Encryption" in yellow text at the top. It features a logo in the top left corner. The main content is a list of two items:

- Plaintext:  $M < n$
- Ciphertext:  $C = M^e \pmod{n}$

So the plaintext is this  $M$ . Processor encryption is conceptually very simple. You raise,  $M$  to the power  $e$  modulo  $n$  just that is your  $C$ . See as an equation it is very simple to see simple to understand. But one thing you should understand both  $C$  and  $E$  are very large numbers. You are trying to raise a very large number into the power of another very large number. So there has to be some efficient methods to calculate the power. We shall actually talk about one such method later. But say to computing,  $M$  to the power  $e$  does not really mean that you are multiplying  $M$  to itself  $e$  minus one time. There are more efficient ways for doing that. So this is encryption.

(Refer Slide Time: 31:21)

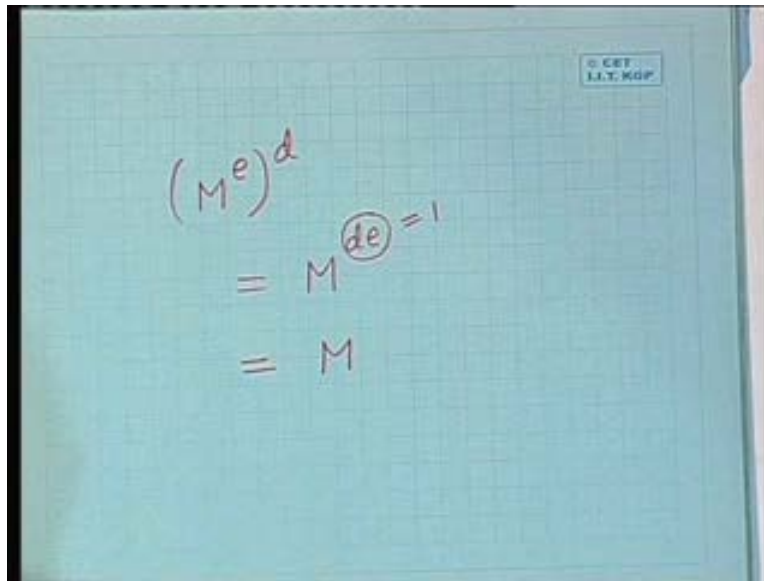


The slide is titled "RSA : Decryption" in yellow text at the top. It features a logo in the top left corner. The main content is a list of two items:

- Ciphertext:  $C$
- Plaintext:  $M = C^d \pmod{n}$

Decryption is very similar you take C raise C to the power d.

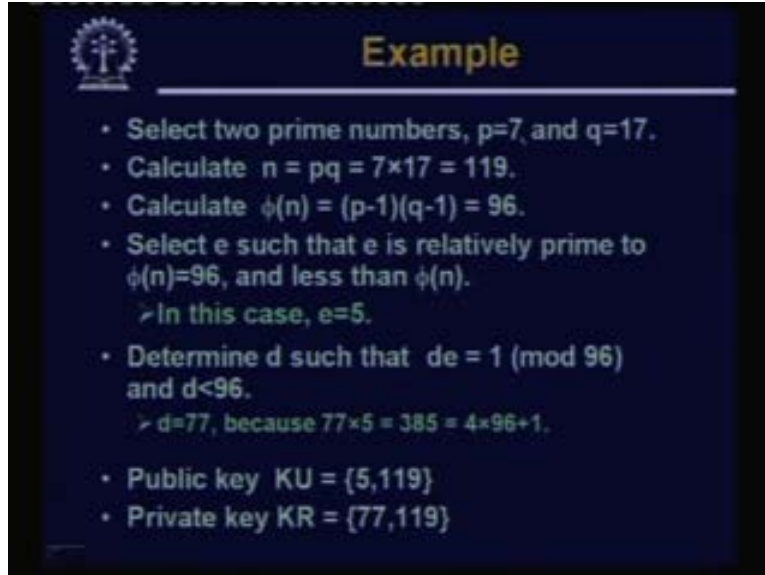
(Refer Slide Time: 31:30)



The image shows a handwritten derivation on a grid background. The text is written in red ink. It starts with the expression  $(M^e)^d$ , followed by an equals sign and  $M^{de} = 1$ , where 'de' is circled. The final line shows an equals sign and  $M$ . In the top right corner of the grid, there is a small blue stamp that reads '© CET, I.I.T. KGP'.

So you see for encryption what you do? Given the plaintext  $M$  you are raising it to the power  $e$ . For decryption you are taking the cipher text, you are raising it to the power  $d$ . So actually what you are getting is  $M$  to the power  $d e \bmod n$ . Of course now  $d e$  has been chosen so that one is the inverse of other. So  $d e$  is 1. So what you get back is the original message  $M$ . This is the basic idea behind encryption decryption. Now since the product of  $d e$  is one you can easily understand that whether you take to the power  $e$  first or the power  $d$  first does not matter. So the public key and private key the way you do encryption and decryption, they are interchangeable.

(Refer Slide Time: 32:18)

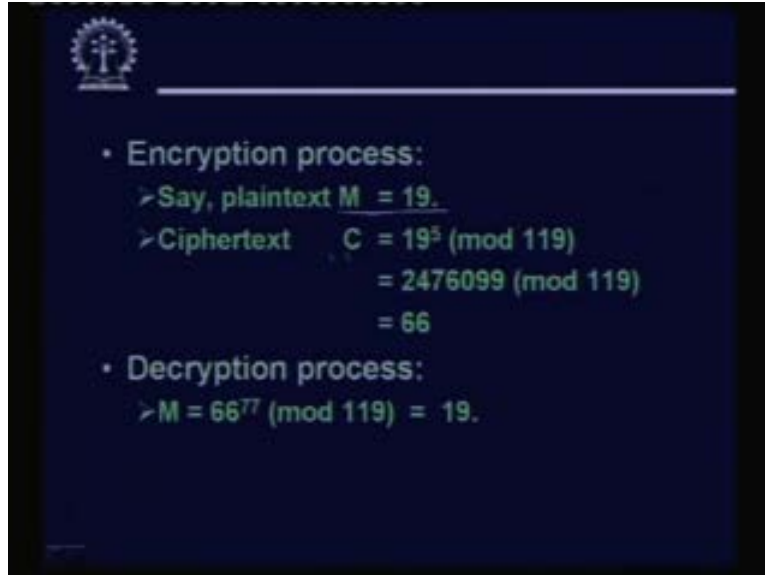


The slide is titled "Example" and contains a list of steps for RSA key generation. It starts with selecting two prime numbers, p=7 and q=17. Then it calculates n = pq = 7\*17 = 119. Next, it calculates the Euler totient function phi(n) = (p-1)(q-1) = 96. It then selects an encryption exponent e such that e is relatively prime to phi(n) and less than phi(n). In this case, e=5. Then it determines a decryption exponent d such that de = 1 (mod 96) and d < 96. In this case, d=77, because 77\*5 = 385 = 4\*96 + 1. Finally, it lists the public key KU = {5, 119} and the private key KR = {77, 119}.

- Select two prime numbers,  $p=7$ , and  $q=17$ .
- Calculate  $n = pq = 7 \times 17 = 119$ .
- Calculate  $\phi(n) = (p-1)(q-1) = 96$ .
- Select  $e$  such that  $e$  is relatively prime to  $\phi(n)=96$ , and less than  $\phi(n)$ .
  - In this case,  $e=5$ .
- Determine  $d$  such that  $de = 1 \pmod{96}$  and  $d < 96$ .
  - $d=77$ , because  $77 \times 5 = 385 = 4 \times 96 + 1$ .
- Public key  $KU = \{5, 119\}$
- Private key  $KR = \{77, 119\}$

So here is a small example to illustrate the encryption and decryption process. Let us take we choose two prime numbers  $p$  and  $q$  7 and 17. First step is to compute the product in it comes to 119. Then the Euler Totient  $p$  minus 1 into  $q$  minus 1, it comes to 96. You select some  $e$  such that  $e$  is relatively prime to 96. By trial and error you can find out that five is one such value that you can have a value of 5 which is relatively prime to 96. Next step is to find a value  $d$  such that  $d$  and  $e$  are inverses or  $de$  equal to 1 mod 96. So, again through trial there are again some heuristic to speed up the search. You can find out a value of  $d$ ,  $d$  equal to 77. Now this you can verify  $d$  into  $e$  is 385 which if you take mod  $n$  the remainder is 1. So it satisfies this  $de$  equal to one property. So now your public key will be 5 and 119  $n$  is 119 private key will be 77 and 119.

(Refer Slide Time: 33:50)



• Encryption process:

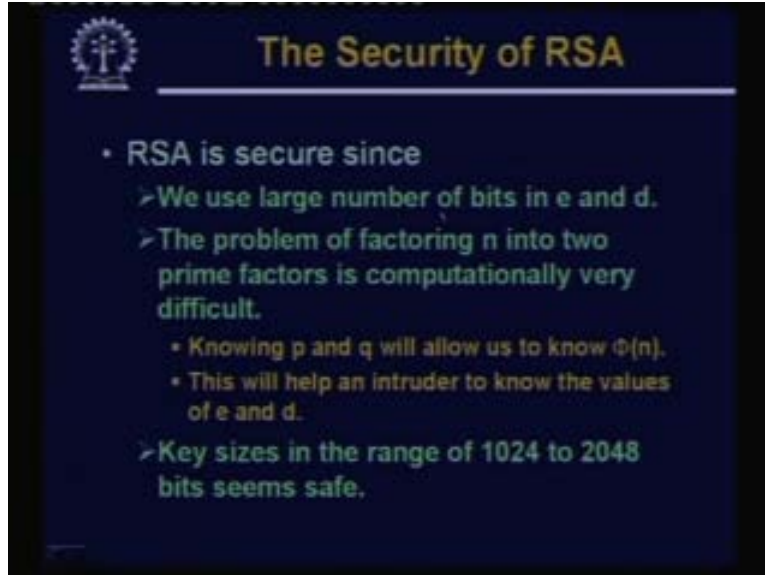
- > Say, plaintext  $M = 19$ .
- > Ciphertext  $C = 19^5 \pmod{119}$   
 $= 2476099 \pmod{119}$   
 $= 66$

• Decryption process:

- >  $M = 66^{77} \pmod{119} = 19$ .

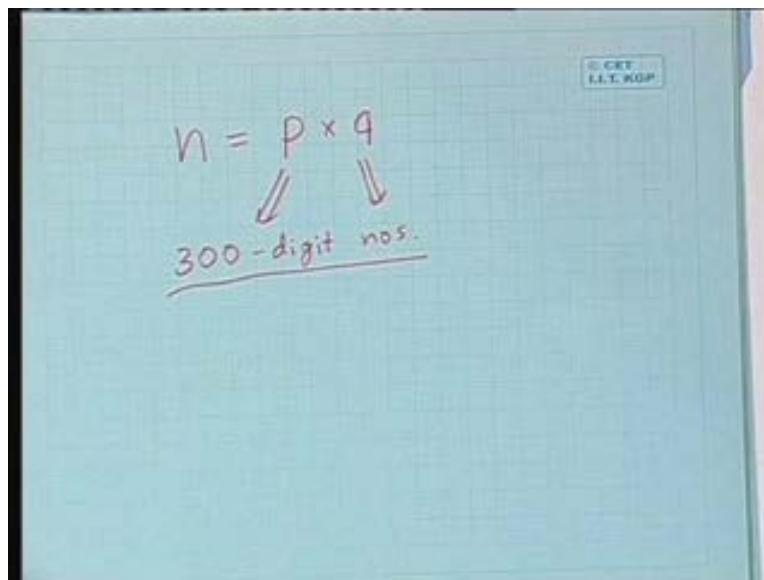
Next let us look at encryption. Say suppose the plaintext is a number 19 say. So ciphertext will be 19 to the power  $e$ ;  $e$  is 5 mod 119. So 19 to the power 5 is this number, mod 119 which comes to 66. During decryption I am only showing the result 66 to the power 77 mod 119. You will in fact get back 19. But how do you calculate 66 to the power 77? The result will be a huge number in fact you maintain. This huge number while RSA computation. Here you use something called multiple precision arithmetic libraries to store to manipulate to perform operations on very large numbers. They can be 100s of digits long. So this is the central idea behind this RSA. This is the reason why I said that RSA algorithm is much slower as compared to DES. Because in DES or AES there is no such very complicated, very high precision or very large numbers you need to handle in terms of computational. For DES, AES the key sizes are of the order of 256 maximum. But here we are talking of 1024 bit key. Key sizes are also much bigger.

(Refer Slide Time: 35:31)



Now let us have a quick look into the security features of RSA. Why do we consider that this RSA is secure? Now RSA is secure because we have chosen very large numbers p and q. As a result the values of e and d which will be getting finally they will also be pretty large and the security of RSA is primarily hinged on this property which is still unsolved. The problem of factoring n into two large prime factors is computationally very difficult. You try to understand I have the product of two numbers n. Now n is part of the public key. So it is known to the public. But what is n?

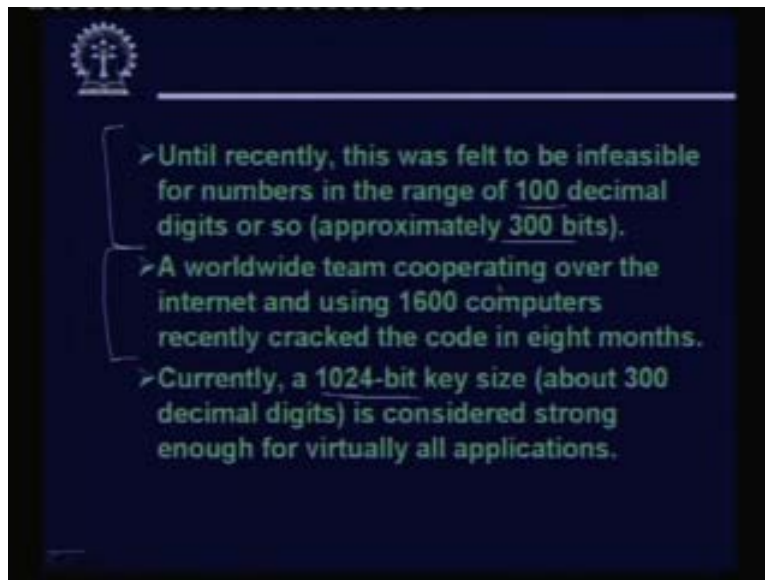
(Refer Slide Time: 36:34)



$n$  is the product of  $p$  and  $q$ ; both  $p$  and  $q$  can be of the order of 300 digit big numbers. So the product will be almost double the sizes. Now the point is that if you have such a big number  $n$ , how do you factorize that number into two prime factors? This problem is still considered to be computationally interactive. Now some people say or they confuse this with another thing. See sometime back there were some research group at IIT Kanpur they cracked a relative they cracked a related problem. Given a number, they could find out in resembled time whether that number is a prime number or not. That can be very large number also but here the problem is different. We are not trying to check whether the number is prime or not. We are trying to find out the prime factors of a very big product number. So this problem is still safe and you can still use RSA without any worry.

So if someone is able to factor  $n$  into the prime factor. So actually they will know  $p$  and  $q$  and knowing  $p$  and  $q$  will allow us to know  $\phi n$ . Knowing  $\phi n$  you can know  $e$  and  $d$  very easily because one of  $e$  and  $d$  will be part of the public key. So the private key it can also be deduced from there. If you know  $\phi n$  because you recall  $d$ ,  $e$  is one mod  $\phi n$ . As I said key sizes in the range 1024 to 2048 bits, today are considered to be very safe. Most of the financial transactions that you carry out over your internet today they use 1024 bit RSA. But if you are even fussy about the security level you can erase it even further by increasing the size of the key. So the size of the key is entirely up to you. Only thing is that the time taken for encryption decryption will be much longer.

(Refer Slide Time: 39:03)



So this RSA this 1024 bit, how does this figure come? See until about few years back it is felt that the keys which represented numbers of about 100 decimal digits which approximately translates to 300 bits. They are quite safe, but actually for the purpose of demonstration a worldwide research team who worked on cryptanalysis. They actually cooperated over the internet and they used 1600 computers in the distributed fashion and cracked the code 300 bit code in 8 months. This is for 300 bit and for every added bit

complexity gets doubled. So 1024 bit is way beyond 300. So far as my knowledge goes up to 480 bit RSA key sizes have been cracked or people to trying to crack them. So 1024 is still a generation ahead of that complexity. So 1024 bit keys which translate to about 300 decimal digits are considered to be strong enough for virtually all applications at least for the next 10 years.

(Refer Slide Time: 40:34)



Now in this context let us look at another related method. This is primarily used for sharing a secret key between two parties, but not using RSA. Using RSA also you can do. But this method uses some other very interesting techniques. Let us see this method was proposed in 1976 and again after the person who proposed them this method came to known as Diffie Hellman key exchange algorithm. Now Diffie Hellman is used as part of many security solutions. Many applications they use Diffie Hellman as one of the steps where keys are distributed. Here the idea is very simple. There is no concept of encryption you allow group of users to agree on a secret key over insecure channel. The idea is see if you talk about RSA, using RSA also we can communicate a key to the other end to the other party.

But here we are actually doing an encryption for sending the encrypted version of the key. But in Diffie Hellman key exchange, the interesting thing is that we are not doing encryption at all. We are using the public channel, the internet to send some numbers back and forth. Intruder may be listening to those numbers. But the beauty of this method is that even by knowing the numbers. What are flowing through intruder; will not be able to decode the key value. That will be finally be used by the two parties. So let us see how it happens. But Diffie Hellman key exchange as I said it can be used for key generation or key agreement. But cannot be used for encryption and decryption. This method again. The goodness of this method depends on a computationally difficult problem. That is, that the problem of computing discrete logarithm. We will come to this later.



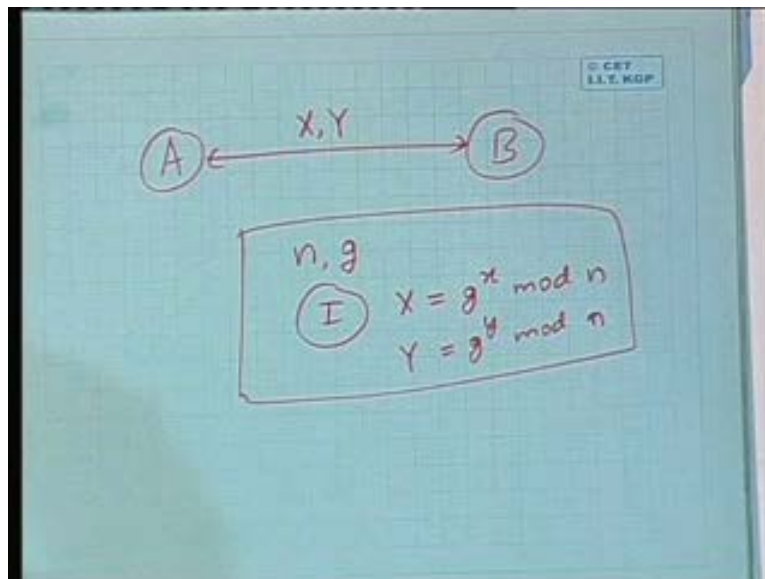
(Refer Slide Time: 42:55)

**D-H Algorithm**

- A and B want to agree on secret key.
  - They agree on two large numbers  $n$  and  $g$ , such that  $1 < g < n$ .
  - A choose random  $x$ , computes  $X = g^x \text{ mod } n$ , and sends  $X$  to B.
  - B chooses random  $y$ , computes  $Y = g^y \text{ mod } n$ , and sends  $Y$  to A.
  - A computes  $k_1 = Y^x \text{ mod } n$ .
  - B computes  $k_2 = X^y \text{ mod } n$ .
- Note:  $k_1 = k_2 = g^{xy} \text{ mod } n$ .

The method goes like this. Suppose there are two parties A and B who want to agree on a shared secret key? At the first step over the public channel, they agree on two large numbers  $n$  and  $g$  where  $g$  is less than  $n$ . So an intruder silently listening to the communication will also come to know about  $g$  and  $n$  fine. So the idea is this.

(Refer Slide Time: 43:28)



There is this party A; here this party B. They are trying to exchange some data and an intruder sitting here silently gets hold of the values of  $n$  and  $g$ . Now in this method A chooses a random number  $X$ . Similarly B also chooses a random number  $Y$ . Now this selection of the random numbers is done at the two different ends. A selects a random

number, B selects some other number. What A will do? It will compute  $g$  to the power  $x$   $g$  is known  $g$  to the power  $x$  modulo  $n$ ,  $g$  and  $n$  are known and let us call it capital X. Similarly B will compute a similar quantity  $g$  to the power  $y$ ,  $y$  is the number B have generated. Let us call it capital Y and this capital X and capital Y are transmitted. So what happens is that over the channel this capital X and capital Y are going. So intruder will also know the values of capital X and capital Y. These are the things which are available with the intruder  $n$   $g$  capital X and capital Y.

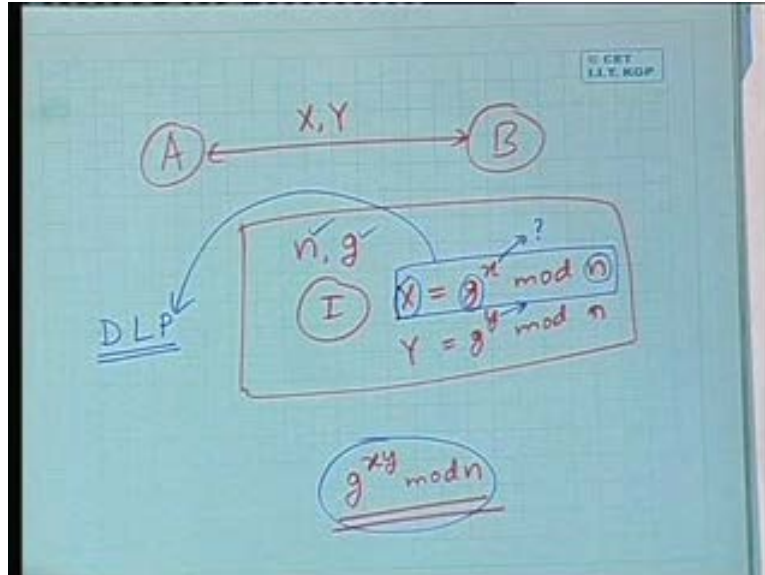
(Refer Slide Time: 44:57)

**D-H Algorithm**

- A and B want to agree on secret key.
  - They agree on two large numbers  $n$  and  $g$ , such that  $1 < g < n$ .
  - A choose random  $x$ , computes  $X = g^x \bmod n$ , and sends  $X$  to B.
  - B chooses random  $y$ , computes  $Y = g^y \bmod n$ , and sends  $Y$  to A.
  - A computes  $k_1 = Y^x \bmod n$ .
  - B computes  $k_2 = X^y \bmod n$ .
- Note:  $k_1 = k_2 = g^{xy} \bmod n$ .

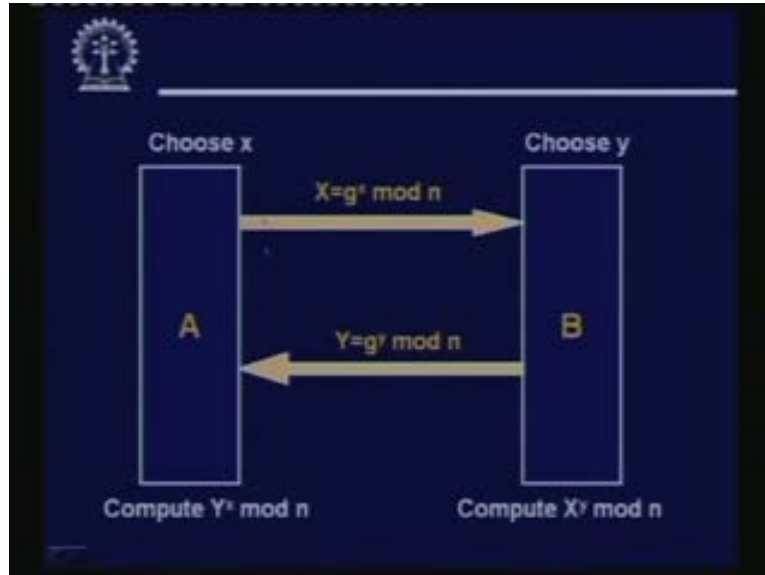
Next step what will happen, A whatever it receives from B capital Y it raises it again to the power of that  $x$  which it had generated earlier. B similarly raises whatever it has received from X to the power  $y$ . So now what will happen after this step? Both A and B have computed the same number  $g$  to the power  $x y$  because Y was  $g$  to the power  $y$  to the power  $x$  was  $g$  to the power  $x$  to the power  $y$ . So  $k_1$  and  $k_2$  are equal. So after this step, these numbers  $k_1$  and  $k_2$  can be used as the secret key.

(Refer Slide Time: 45:50)



Now let us come back to this diagram again. So the secret key we are saying is  $g$  to the power  $x$   $y \bmod n$ . Now let us look at the intruder the intruder already has these values. For example intruder has this value  $X$  which is  $g$  to the power  $x \bmod n$ . Now  $n$  is a known quantity  $n$  is known  $g$  is a known quantity and  $X$  is something which the intruder has already got. So how difficult it is to determine  $x$  from here because if the intruder can somehow determine  $x$  from here and determine  $y$  from here then very easily  $g$  to the power  $x$   $y$  can be calculated. Now this problem is known as the discrete logarithm problem and this also is considered to be till today a computationally intractable problem that it cannot be broken in a reasonable amount of time. Given  $g$ , given  $n$ , given the value of the whole expression, it is not computationally feasible to calculate the value of  $x$ . So the goodness of this method depends on the complexity of the DLP problem discrete logarithmic problem.

(Refer Slide Time: 47:11)



So whatever we have mentioned is shown diagrammatically here. So A will be sending to B  $g$  to the power  $x \text{ mod } n$ , B will be sending  $g$  to the power  $y$ . A will be computing this  $y$  to the power  $x$  B will be computing  $X$  to the power  $y$  and finally both will be in position of the same secret key.

(Refer Slide Time: 47:37)

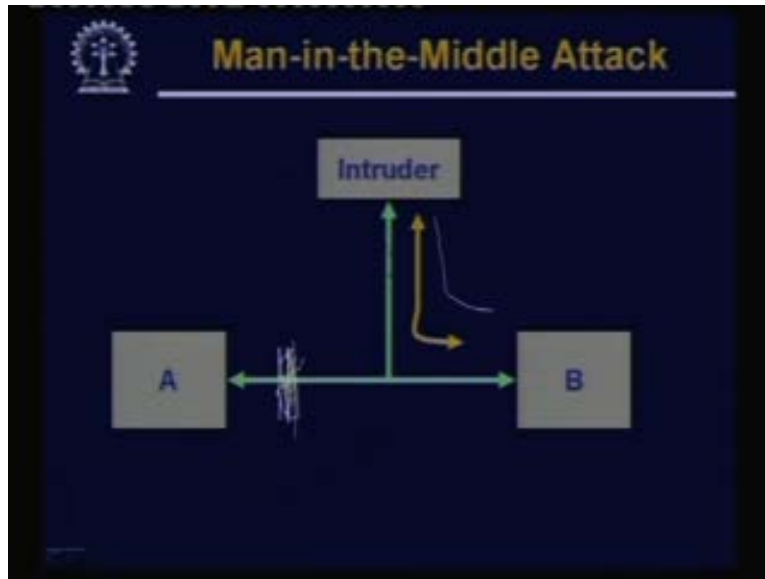
### D-H Algorithm: contd.

- Requires no prior communication between A and B.
- Security depends on difficulty of computing  $x$ , given  $X = g^x \text{ mod } n$ .
- Choices for  $g$  and  $n$  are critical.
  - > Both  $n$  and  $(n-1)/2$  should be prime.
  - >  $n$  should be large.
- Susceptible to intruder-in-the-middle (man-in-the-middle) attack.
  - > Active intruder.

Now some of the points here is that this method requires no prior communication between A and B. Like if you are using public key cryptography there is a need for a prior communication where I am broadcasting the public key then your encrypting and sending the key back. So here there is no prior communication needed and as I said

security depends on the discrete logarithmic problem, just now I have mentioned. This is the second point and the choice of  $g$  and  $n$  are critical because it has been found that for some values of  $g$  and  $n$  attacking is relatively easier. So it has been seen that if you choose a value of  $n$ , such that  $n$  and  $n - 1$  by 2, both are primes and of course  $n$  should be large then it is really difficult for the intruder to break. But this method has one drawback. This is susceptible to the so called intruder in the middle or the man in the middle attack, a so called active intruder.

(Refer Slide Time: 48:39)



Pictorially it is depicted like this. See A and B wants to exchange the key to agree upon a key. Somehow there is a router or some node between intruders has hacked that node and some intruder has cut A of the network. So A is here no longer here on the network. Now intruder and B can communicate intruder creates packets and sends to B. B will believe that actually the packet is coming from A and intruder will know the Diffie Hellman algorithm. So intruder can guess that random number  $x$  himself and they can also proceed in the same way. The same sequence of steps they can follow and intruder and B will finally agree on a common key. Now B does not know whether the other side is intruder or A. So B in good faith will believe that the other part is A, and after agreeing on the key B can actually start sending some confidential information. So this man in the middle attack is possible in case of Diffie element key exchange.

(Refer Slide Time: 49:58)

**A Comparison**

- Symmetric encryption/decryption is much faster than asymmetric encryption/decryption:
  - RSA: kilobits/second*
  - DES: megabits/second*

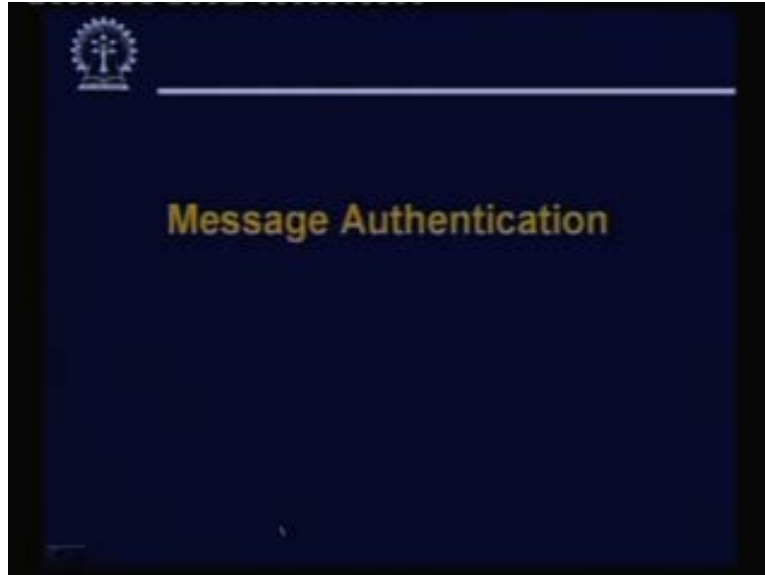
↓

*DES is about 100 times faster than RSA*

- Key size:
  - > RSA: selected by user
  - > DES: 56 bits

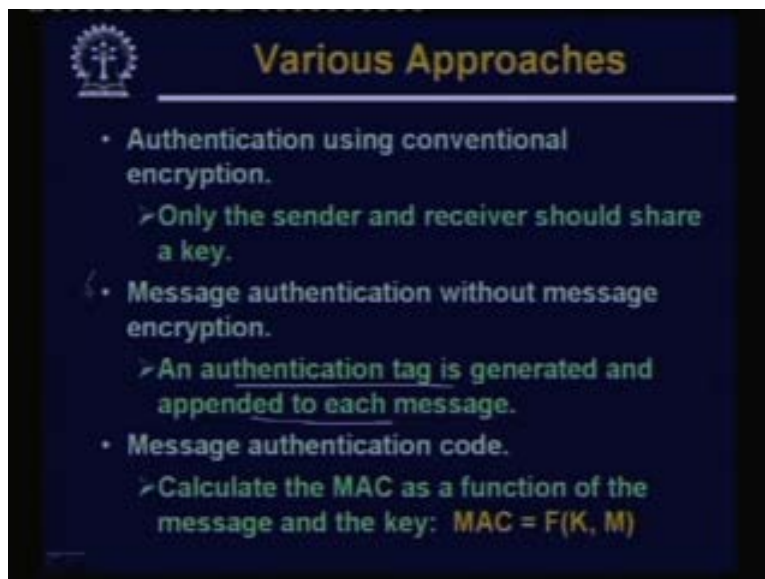
Now this is what I have said. Here I have some figures I had said symmetric encryption decryption scheme is much faster than asymmetric schemes. Some figures RSA work at in the order of kilobits per second DES works at megabits per second. DES is at least 100 times faster. Sometimes it can be about 1000 times faster also. So as you can see the symmetric encryptions will be at least 100 times faster as compared to the public key schemes. So public key schemes should be used sparingly and only in cases where very small volumes of data need to be encrypted. If you try to encrypt very huge volumes of data the computation time may become large and other issue regarding key size. For DES or AES we do not have much choice about the key sizes. But RSA it uses selectively. In DES or AES it is the standard which tells you these are the key sizes you can possibly use. But in RSA you can you can even use 69 bits of key. No one stops you from doing that.

(Refer Slide Time: 51:18)



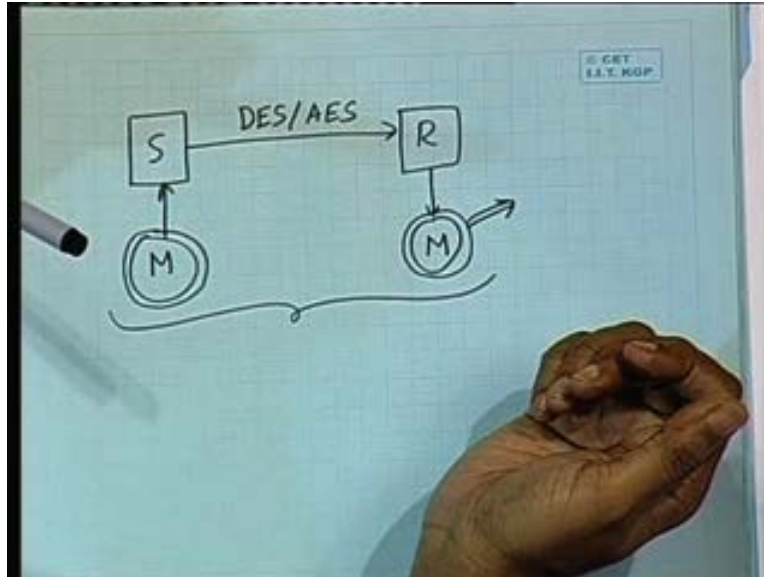
We now look at a related issue this is also a very important problem. Of course we have said that we can use public key cryptography for authentication. But in all environments it is not suitable. Message authentication in general I am sending a message to you. How do you authenticate that it is actually coming from me?

(Refer Slide Time: 51:43)



So broadly there are several methods. First one is authentication using conventional encryption, where only the sender and the receiver should share a key. Here the idea is like this.

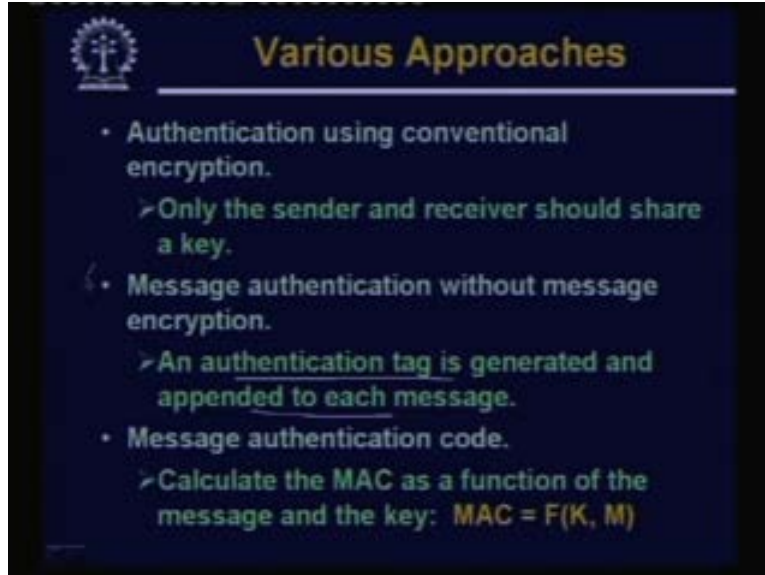
(Refer Slide Time: 52:04)



Here is the sender. Sender takes a message to be transmitted encrypts it, sends to the receiver the receiver decrypts back the message. Now here you are using conventional or symmetric encryption schemes like DES or AES. Now here we are calling this. An authentication scheme because whatever you are saying that this message itself can be a can be an English text message for example. So the receiver after decrypting, if the receiver finds that what is obtained back is a meaningful message, then the receiver will be able to know that in addition to receiving the message it would be able to know that it is actually coming from the person from whom it is expecting to come. Because both of them are sharing the same key. Someone else is not expected to know the value of the key. So it is not possible.

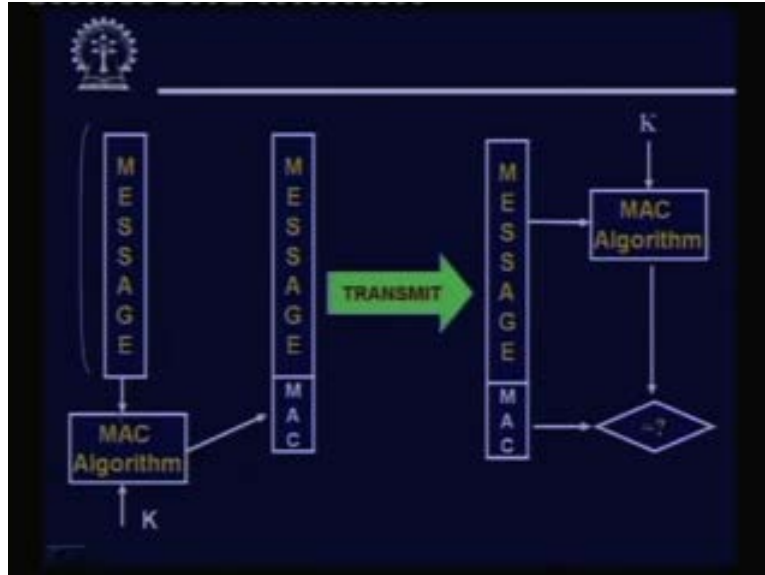


(Refer Slide Time: 53:19)



So the second method that we shall see this is called message authentication without message encryption. Because, sometimes we need only authentication. There is no necessity for message encryption because the message can be pretty large. But I only want to know that from whom the message is coming, maybe I am downloading some program texts from somewhere. I do not worry that whether someone is reading whatever I am downloading. I do not care but what I must be sure I must be sure of the identity of the origin. So this is authentication without message encryption. Here, an authentication tag can be generated to which message, we will see this with the help of an example in the next slide and thirdly this is similar. These two are similar in fact a message authentication code. Now this authentication tag and message authentication could conceptually. They are similar there is message authentication code which is generated which is a function of the key and the message which is transmitted as part of the message. So let us see how it happens.

(Refer Slide Time: 54:32)



Suppose here is the message we want to send here is the message. There is a message authentication code generation algorithm which takes a secret key. It generates the MAC. MAC is appended to the message and transmitted. So the receiver gets back message as well as the MAC. Receiver again computes the message authentication code from the message using the same value of the key and it compares whether the two MACs match. If they match then they will know the message is actually coming from the sender. Because only the sender and the receiver is knowing the same key and any intruder cannot change the message or the MAC in such a way here you will be getting a perfect match. So in fact there are many algorithms which are existing in this family.

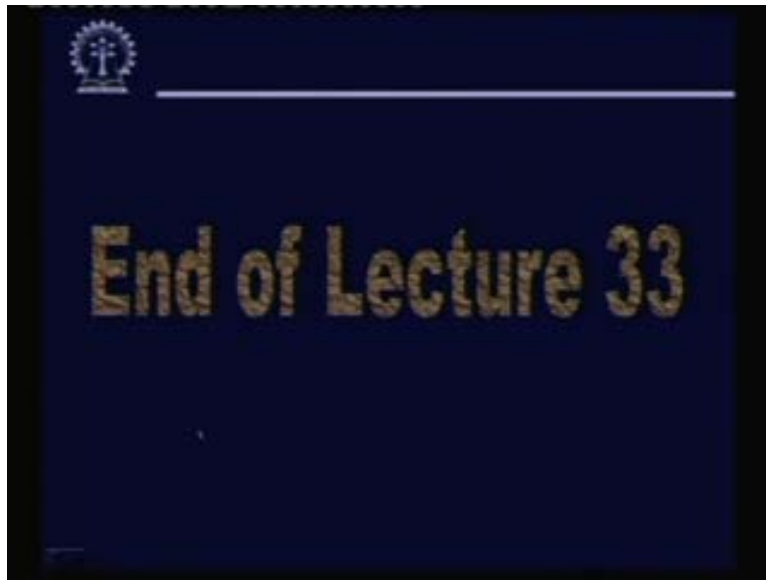
(Refer Slide Time: 55:31)

### Commonly Used Schemes

- The MD family
  - MD2, MD4 and MD5 (128-bit hash).
- The SHA family
  - SHA-1 (160-bit), SHA-256 (256-bit), SHA-384 (384-bit) and SHA-512 (512-bit).
- RIPEMD-128 (128-bit), RIPEMD-160 (160-bit).

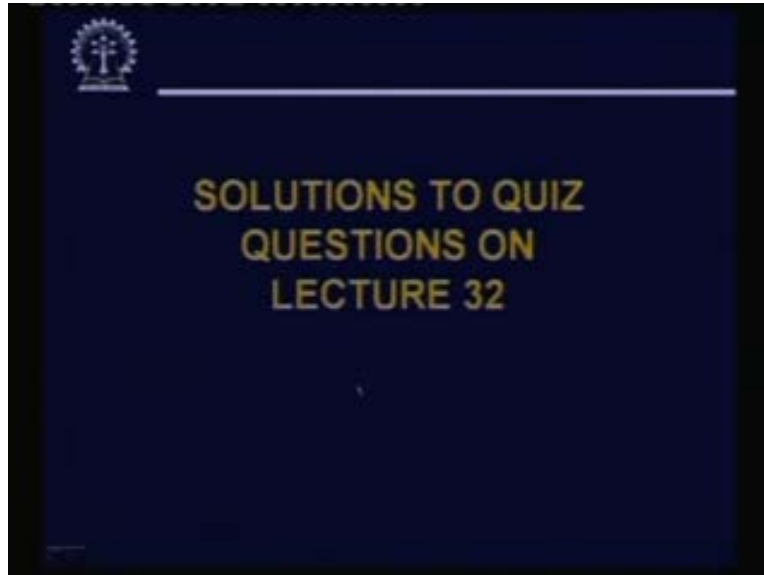
There is MD family; these are all authentication algorithms. In the MD family there are a number of defined variations. MD5 was the most popular 128 hash bit code it generates. Then SHA family S H A again, there are a number of variations depending on how many bit signature or the authentication code it generates. Then RIPEMD is a version again. There are two variations 128 bit, 160 bit. We are not going in the detail of this algorithm. If you are interested, there are so many excellent books on cryptography. There are also lots of materials available on the net. You can always go and have a look at the details of this algorithm.

(Refer Slide Time: 56:18)



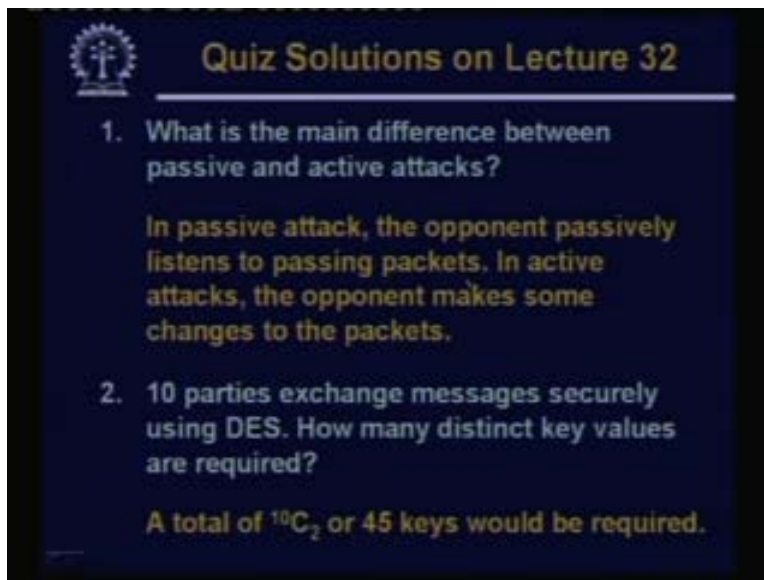
So with this we come to the end of this lecture.

(Refer Slide Time: 56:26)



Let us now first look at the solutions to the previous classes problems.

(Refer Slide Time: 56:28)



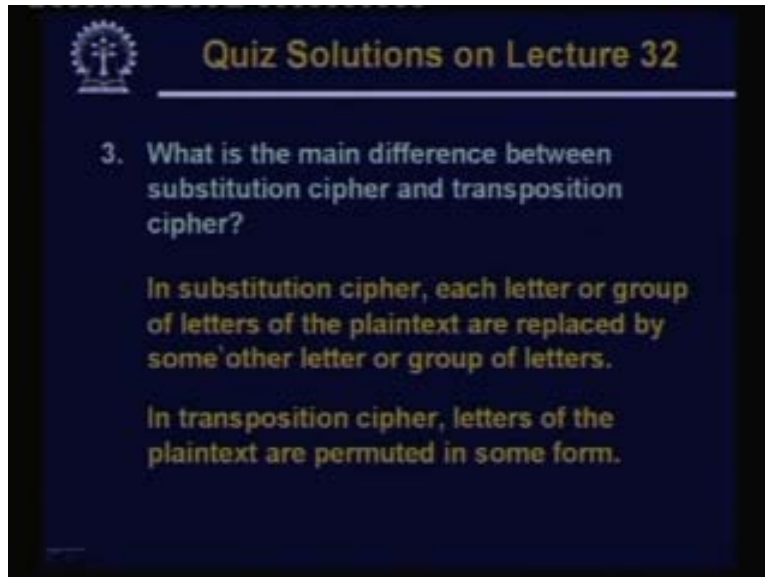
What is the main difference between passive and active attacks?

In passive attack the opponent passively listens to the passing packets. In active attacks the opponent also makes some modifications to the packets.

10 parties exchange messages securely using DES. How many distinct key values are required?

While we said that there will be  $^{10}C_2$  number of possible keys required which comes to 45.

(Refer Slide Time: 57:03)



The slide features a dark blue background with a white logo in the top left corner. The title "Quiz Solutions on Lecture 32" is written in yellow at the top. Below the title, a question is posed in white text, followed by two explanatory paragraphs in yellow text.

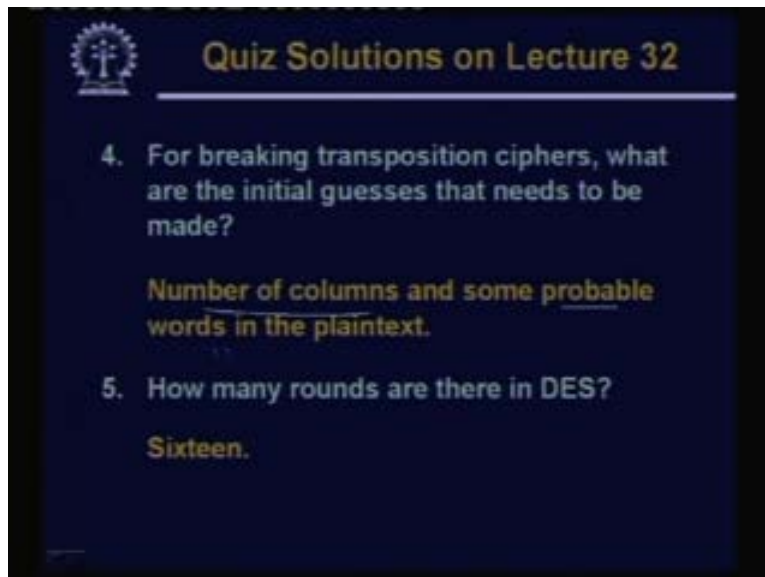
**3. What is the main difference between substitution cipher and transposition cipher?**

**In substitution cipher, each letter or group of letters of the plaintext are replaced by some other letter or group of letters.**

**In transposition cipher, letters of the plaintext are permuted in some form.**

What is the main difference between substitution cipher and transposition cipher?  
In substitution cipher each letter or group of letters are replaced by some other letter or group of letters. So there is a replacement or substitution whereas in transposition there is no replacement rather the letters are jumbled or permuted.

(Refer Slide Time: 57:27)



The slide features a dark blue background with a white logo in the top left corner. The title "Quiz Solutions on Lecture 32" is written in yellow at the top. Below the title, two questions are posed in white text, followed by their respective answers in yellow text.

**4. For breaking transposition ciphers, what are the initial guesses that needs to be made?**

**Number of columns and some probable words in the plaintext.**

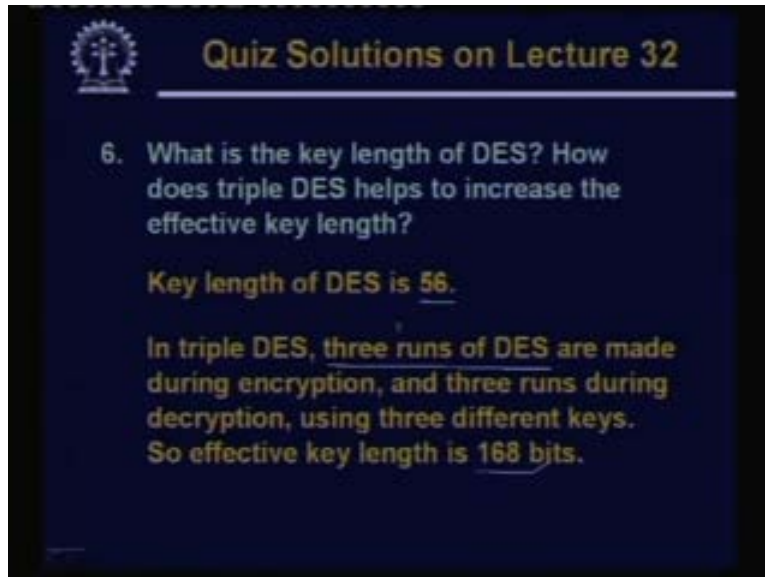
**5. How many rounds are there in DES?**

**Sixteen.**

For breaking transposition ciphers what are the initial guesses that needs to be made?  
We said number of columns is the first guesses and some probable words in the plaintext, is second guess.

How many rounds are there in DES?  
16.

(Refer Slide Time: 57:42)



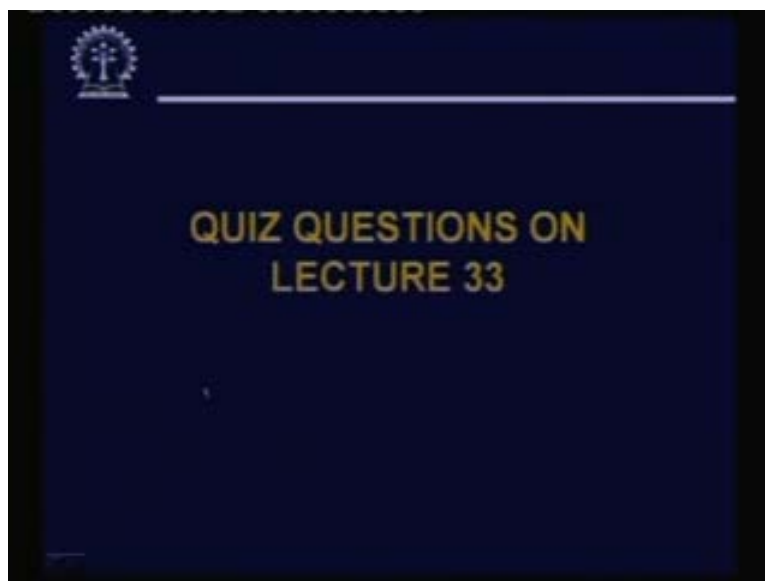
What is the key length of DES?

It is 56.

How does triple DES help to increase the effective key length?

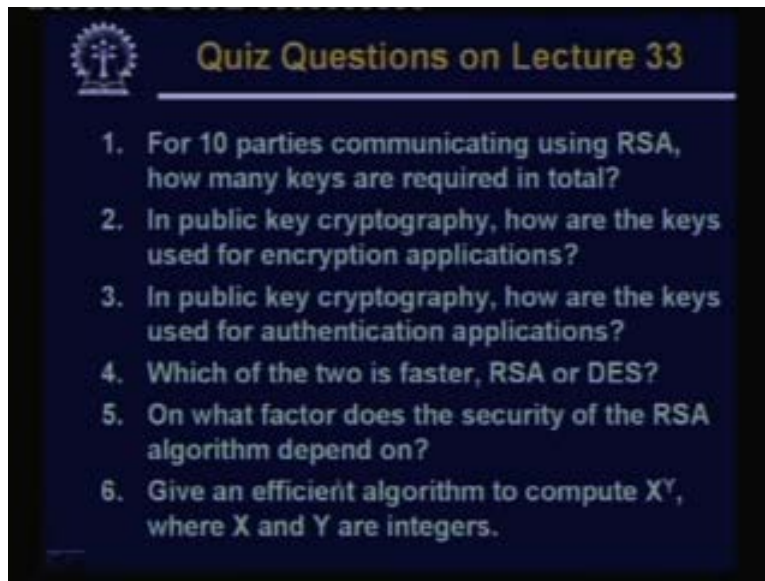
Here we are using three runs of DES. So effectively the key size becomes 168 bits, 56 into 3.

(Refer Slide Time: 58:01)



So now questions from today's class.

(Refer Slide Time: 58:04)



For ten parties communicating using RSA how many keys are required in total?

In public key cryptography how are the keys used for encryption applications?

Again how are the keys used for authentication applications?

Which of the two is faster RSA or DES?

On what factor does the security of the RSA algorithm depend on?

Give an efficient algorithm to compute  $X^Y$  where X and Y are integers.

Well I am leaving the last question as an assignment for you. You can think over this problem I will give a hint. You treat Y as a binary number. You look at the binary representation of Y and try to come up with an efficient algorithm. So with this we come to the end of this lecture. Thank you.