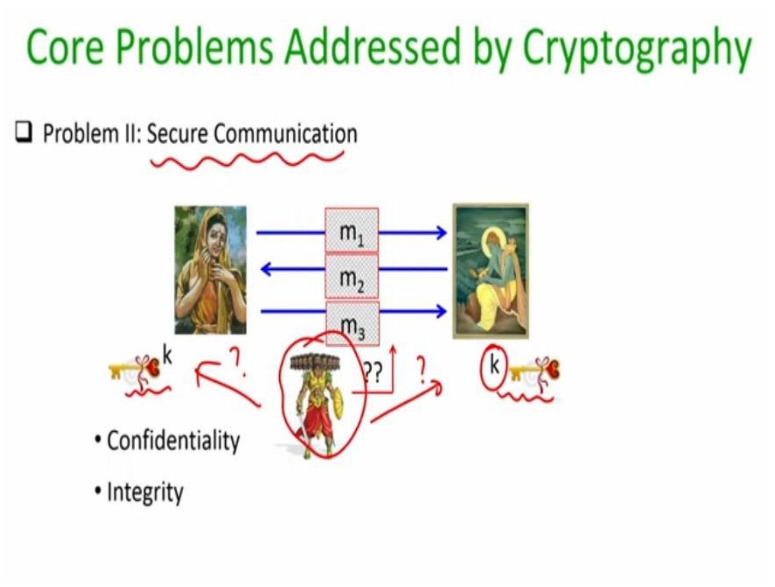


## Core Problems Addressed by Cryptography

- Problem II: Secure Communication

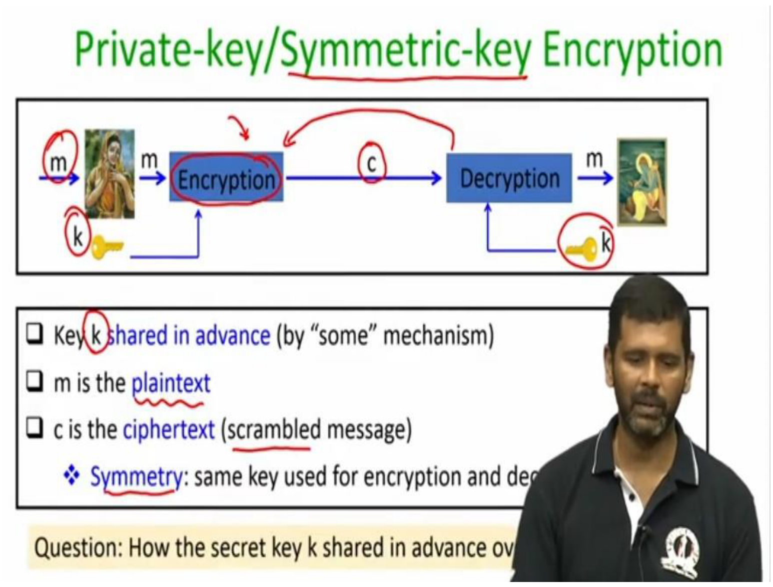
- Confidentiality
- Integrity

And assuming that the key agreement has been achieved, the second problem that is addressed by the cryptography, the second core problem, I should stress here, it is not the case that secure communication is the only problem, the second core problem addressed by cryptography startup secure communication. So, the setting here is the following, we will assume that Sita and Ram has already executed the key agreement protocol over the internet, and they have agreed upon a common key.

And now using this common key, we would require Sita and Ram to come, we would require some algorithms which are publicly known, according to which Sita can convert or encrypt her message into some garbled text into some garbage and communicate to Ram and Ram should be able to convert back those garbage or scrambled text back to the original contents using the same key, k which Sita has. So, namely we want to come up with algorithms which should help me to do secure communication.

And by secure communication here I mean that, if there is a third party or Ravana, who knows the public description of your algorithm but does not know the value of key then even after observing the communication happening between Sita and Ram and even after knowing the full protocol description according to which these messages have been computed, the Ravana should not be able to come up with the values of $m_1$, $m_2$, $m_3$ and so on. So that is the second problem addressed by cryptography.

**(Refer Slide Time: 24:23)**



## Private-key/Symmetric-key Encryption

- Key k shared in advance (by "some" mechanism)
- m is the plaintext
- c is the ciphertext (scrambled message)
  - Symmetry: same key used for encryption and de

Question: How the secret key k shared in advance ov

So, it turns out that there are two kinds of, two classes of cryptographic algorithms which we use. The first category is that of private key or symmetric key encryption. In the symmetric key encryption, the setting is the following. It will be ensured that a common key is already shared between Sita and Ram by some mechanism, say, by running a key agreement protocol and no one else apart from Sita and Ram knows the value of that key.

Now, if that is the case assuming this setup has been done, the way symmetric encryption works is as follows. So, imagine Sita has some message, it could be an email, it could be just a hi message, it could be anything, it could be her net banking password. So, she has some message which is abstracted as a binary string, we call her message as plain text. We want to design an algorithm which we call as an encryption algorithm which takes a message m and the key k both of which are binary strings.

And it should produce another binary string which we call a ciphertext. And this ciphertext will be the scrambled message because it will have absolutely no meaning, in the lose sense and Sita will compute this ciphertext and communicate it over the internet and send it to Ram. Now, once Ram obtains this scrambled message, he will have a decryption algorithm, he will have in a sense, he will know that Sita has used an encryption algorithm whose details are publicly known and the corresponding matching decryption algorithm also will be publicly known.

So, Ram will use the corresponding decryption algorithm. And the inputs for the decryption algorithm will be the ciphertext that he has received and the same key which has been used by

Sita to produce the scrambled text. And this decryption algorithm will magically produce back the same message m which Sita has used or wanted to communicate.

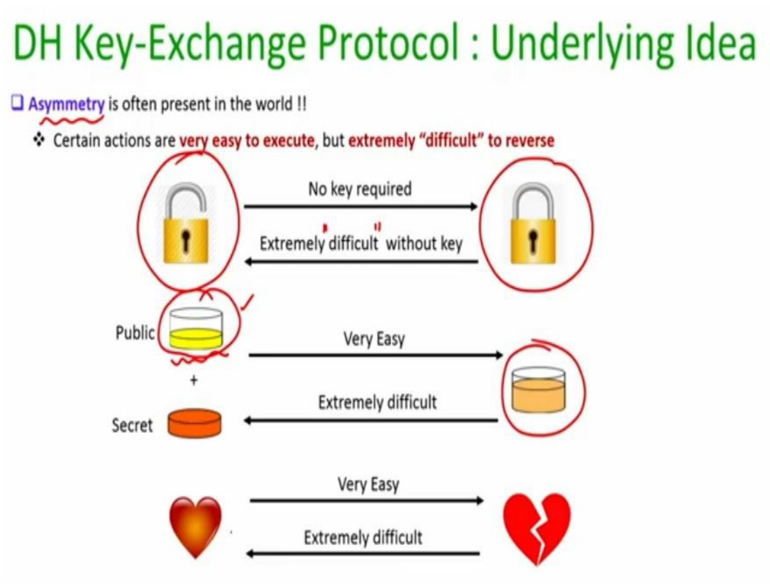
So, the reason it is called symmetric key encryption is because of the symmetry, namely, the same key is used both for encrypting the message as well as for decrypting the message. Now, the system might look very neat, very clean, just you encrypt your message and send a message, encrypted message, Ram receives encrypted message and decrypt and recover back the message.

So, the analogy could be that, assume Sita and Ram have already exchanged a key for a physical lock. If Sita has a message, what she can do is, she can take a box, keep her message written in a paper inside the box and close the box with a lock and using the key that she has. And now she can send this lock box by a courier or anything. So, if there is a third person who does not have the key for opening the lock of the box, he would not be able to do that.

Now, once the courier is delivered to Ram, since Ram also have the same key, he can use it, unlock the lock and see what exactly is the content kept inside that box. So, the same message which Sita wanted to send will be delivered to Ram. So that is the analogy. But the system will work if both Sita and Ram have already agreed upon this common key for the lock. How at the first place they can do that?

Because everything will be now happening over a public channel because it is not the case that Sita and Ram knew beforehand in advance. It is like saying the following, if I want to do a transaction over the internet; Amazon may not be knowing well in advance that a person called Ashish Chowdhury, would like to do a transaction with Amazon. So, I will be doing my transaction at a run time, how at the first place I establish a secure key with Amazon? And that too, by communicating over the internet, so that is a big question. How at the first-place key agreement has taken place?

**(Refer Slide Time: 28:24)**



## DH Key-Exchange Protocol : Underlying Idea

- Asymmetry is often present in the world !!
  - Certain actions are very easy to execute, but extremely "difficult" to reverse

So, it was a folklore belief that it is not possible to agree upon a common key by interacting over a public channel. But the Turing Award winner, Diffie and Hellman, proved this belief to be incorrect, by coming up with their seminal key exchange protocol. So, I would not be going into the full details of security proof and other details of the key exchange protocol, I will just try to give you the underlying idea.

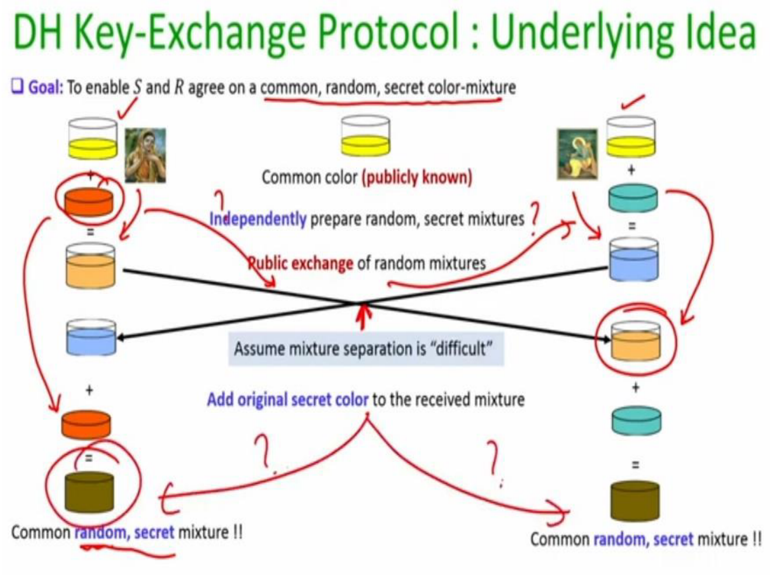
So, the main idea used in their key exchange protocol is the following. They observed that there are plenty of tasks in this universe which are asymmetric, they are asymmetric in the sense, they are very easy to compute in one direction. That means, it is very easy to go from one state to another state but extremely difficult to reverse back the effect of that action. So, for instance, if I take a padlock in an open state then it is very easy to lock the padlock, I just have to press it I do not need any key.

But now, once I go to closed state of this padlock and if I asked you that, can you open it, until and unless you do not have the key it will be extremely difficult for you. So, I am saying it is extremely difficult to open it without a key, there might be some other mechanisms to open it as well. You might have a Jugaad method but that will be extremely difficult, very time consuming. I am not saying it is impossible. So, there is a difference between extremely difficult and impossible.

In the same way, consider this task; I take a publicly known color. And then I prepare a secret mixture. In the sense I do the public color, I add a secret color. And then once the mixture is prepared, I give it to you. So, the mixture preparation is very easy. And now if I ask you that, okay, I give you this mixture, I also tell you the public color with which I started with, can you tell me what exactly was the secret color that I added?

Again, it is not an impossible task, you yourself can take a tumbler with a publicly known color and keep on adding various colors which you can add and see whether that gives you the same secret mixture that I have. But that might be a very time-consuming affair. And most importantly, it is very easy to break someone's heart by saying very bad words, but it is very extremely difficult to win the love and confidence of that person back.

**(Refer Slide Time: 30:56)**



## DH Key-Exchange Protocol : Underlying Idea

- Goal: To enable *S* and *R* agree on a common, random, secret color-mixture

Common color (publicly known)
Independently prepare random, secret mixtures
Public exchange of random mixtures
Assume mixture separation is "difficult"
Add original secret color to the received mixture
Common random, secret mixture !!
Common random, secret mixture !!

So, based on this idea that asymmetry is there in lots of tasks. This is the underlying idea of Diffie Hellman key exchange protocol. So, I will be first explaining the protocol assuming that Sita and Ram want to agree upon a common secret mixture which should be random at least should be decided and no one else should be able to learn what exactly is the secret mixture. So, to begin with, both Sita and Ram will be starting with some common publicly known color.

And now, what they will be doing is the following. They will prepare independently some secret mixtures. So, Sita will prepare her secret mixture independently and Ram will be preparing his secret mixture independently, by adding a secret color, individually and then they will publicly exchange their mixtures. So, Sita will send her a copy of the mixture to Ram, Ram will send his copy of the mixture to Sita.

And here I am assuming that mixture separation is an extremely difficult task that means, if there is a third party who is observing the communication here, who knows the entire process according to which Sita and Ram are acting, so, he knows that both Sita and Ram started with a secret colour. He also knows that, Sita has added a secret component but the exact value of that secret component is not known to this third party.

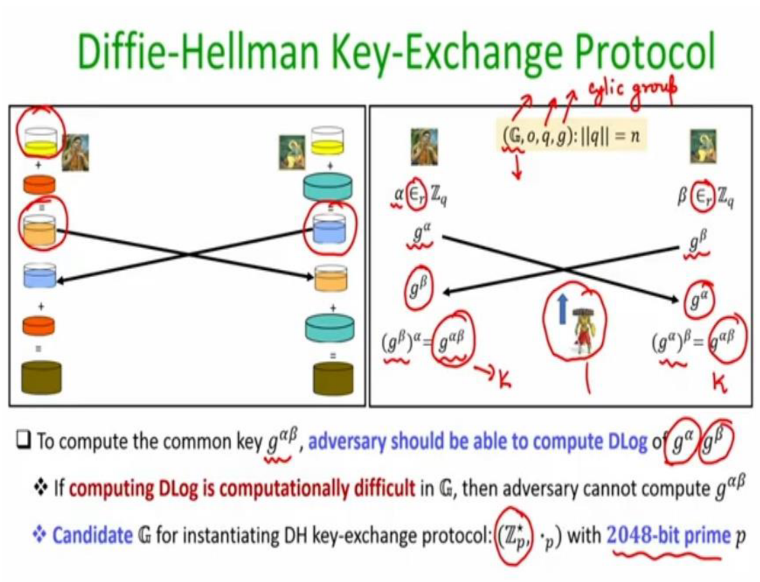
In the same way, he knows that Ram has added a secret component but he does not know what exactly is that secret component? And now of course, he is seeing the public mixtures being exchanged. Now, what is the goal? The goal for Sita and Ram is to come up or agree upon a common mixture which should be known only to them. So, what they can do is, they can individually add the secret component that they have added, to the copy of the mixture that they are receiving from the other party.

So, whatever Ram's mixture that Sita has received, she takes that and to that she adds whatever components she has added to prepare her secret mixture. And same task is done by Ram. He takes Sita's mixture and to that he had the secret color that he added to prepare his copy of the secret mixture. And what this will give? This will give both of them a common mixture because it does not matter in what order you add the 3 colors finally, it will give you the same mixture.

And it will be random, it will be random in the sense, next time Sita and Ram again runs the same protocol, they will start with the same copy of the public mixture. But now the secret components with Sita and Ram are going to add might be different because every time every execution of the protocol they will be preparing independent mixture. So that is why, the output at the end of each instance of this protocol will be a random mixture. And why it will be secret?

It will be secret because any third party who is monitoring the communication, he would not be able to separate out the secret contribution of Sita and Ram. So that is why he would not be knowing what exactly is the final mixture that Sita and Ram have obtained. Now, we have to convert this whole process, this whole color exchanging idea into a concrete algorithm, mathematical algorithm and protocol.

**(Refer Slide Time: 34:13)**



Diffie-Hellman Key-Exchange Protocol

cylic group
$(\mathbb{G}, o, q, g): ||q|| = n$
$\alpha \in_r \mathbb{Z}_q$
$\beta \in_r \mathbb{Z}_q$
$g^{\alpha}$
$g^{\beta}$
$g^{\beta}$
$g^{\alpha}$
$(g^{\beta})^{\alpha} = g^{\alpha\beta}$
K
$(g^{\alpha})^{\beta} = g^{\alpha\beta}$
K

❑ To compute the common key $g^{\alpha\beta}$, adversary should be able to compute DLog of $g^{\alpha}$ $g^{\beta}$

❖ If computing DLog is computationally difficult in $\mathbb{G}$, then adversary cannot compute $g^{\alpha\beta}$

❖ Candidate $\mathbb{G}$ for instantiating DH key-exchange protocol: $(\mathbb{Z}_p^{*}, \cdot_p)$ with 2048-bit prime $p$

So, on your left hand side, I have returned the blueprint of the color based key exchange protocol. And now, I will instantiate each and every step by concrete mathematical step. So, both Sita and Ram started with some public information. That public information is the description of a cyclic group, its order and the description of the generator. So that is a public information.

So, Sita and Ram both knows that, okay, they are going to use this cyclic group and every third party who wants to derive the key that Sita and Ram have agreed, are going to agree upon will also know the description of the cyclic group. So, it is like saying the following, if Sita is considered as a user and Ram is considered as a Amazon and if Sita wants to do a transaction with Ram and want to agree upon a common key then, a third party who wants to break the communication knows what exactly are the description of the cyclic group that Sita and Ram are going to use.

So now, the first step of the protocol was that, Sita and Ram prepare some random secret mixtures independently. The corresponding instantiation of that step is that Sita randomly picks a group element and Ram randomly picks a group element, how they can do that?

So, Sita picks a random $\alpha \in_r \mathbb{Z}_q$ in the range 0 to q - 1. So, this notation means that $\alpha$ is randomly chosen, this r denotes that it is randomly chosen. So, she picks the α randomly from the set 0 to q - 1 and computes $g^{\alpha}$. And independently Ram picks a random $\beta$ in the range 0 to q - 1 and computes $g^{\beta}$. So that is their independent secret mixtures which they now communicate.

And now, what was the final step? So, Sita upon receiving Ram's mixture, she adds her own secret component, her secret component was α. So, adding here will mean that, she will take Ram's mixture namely $g^{\beta}$ and raise the whole thing to α which will give her $g^{\alpha \cdot \beta}$ . And what Ram is going to do? He will take Sita's mixture, namely $g^{\alpha}$ and to that he will add his own contribution.

Again adding in this context means, raise it to the power $\beta$ which will result in $g^{\alpha \cdot \beta}$. But what about a third person, an attacker, an eavesdropper, who has monitored the communication, will he be able to compute $g^{\alpha \cdot \beta}$ because that is a common key which Sita and Ram are going to agree upon. Well, for the adversary or for the third person to compute $g^{\alpha \cdot \beta}$, he should know α or $\beta$.

Because if any of these 2 values is learned by the attacker, he can easily compute $g^{\alpha \cdot \beta}$. But for learning α or $\beta$ he has to basically solve, either this instance of discrete log or this instance of discrete log, namely, upon saying $g^{\alpha}$, he should be able to compute α in the reasonable amount of time or given $g^{\beta}$ he should be able to compute $\beta$ in a reasonable amount of time.

That means, if I ensure that computing discrete log is extremely time consuming for this attacker and by time consuming means, at least it takes say 10 years or 15 years then, I can say that this protocol is safe because I do not care after 15 years if attacker comes to know, what exactly I communicated 15 years back. Because I will not be interested to keep the privacy of my communication for so long, so, as long I ensure that it is extremely difficult.

How extremely difficult it is? So, in loose tense, it is a order of several years, if it is extremely difficult for an attacker to come, solve an instance of discrete log, then this protocol gives me a mechanism according to by which Sita and Ram can agree upon a common key. So, now, you might be wondering that what should be the choice of the group, how big it should be and so on.

So, it turns out that, if we instantiate this protocol with my group being $\mathbb{Z}_{p^*}$ that means, if I ensure that my group G is the set $\mathbb{Z}_{p^*}$ where, p is some 2048 bit prime number then, using current best computing speed machines for solving a random instance of discrete log, it will take order of several years and hence, an adversary who tries to attack the scheme will fail to do that.

And that ensures that Sita and Ram now, can safely use the key $g^{\alpha\cdot\beta}$ as the common key and run an instance of symmetric key encryption scheme to do the secure communication of their messages. So, now you can see that how exactly the concept that we have seen in the context of cyclic groups are useful to come up with a very important practical solution for a practical problem namely that of key agreement.

So, with that, I conclude today's lecture. Just to summarize, in this lecture, we introduced the problem of discrete log and we saw that in some groups, solving discrete log might be very easy, in some groups, it is conjecture that solving a random instance of discrete log is extremely difficult. And if we work with those groups then, we can design practical algorithms for real world problems like the key exchange problem. Thank you!