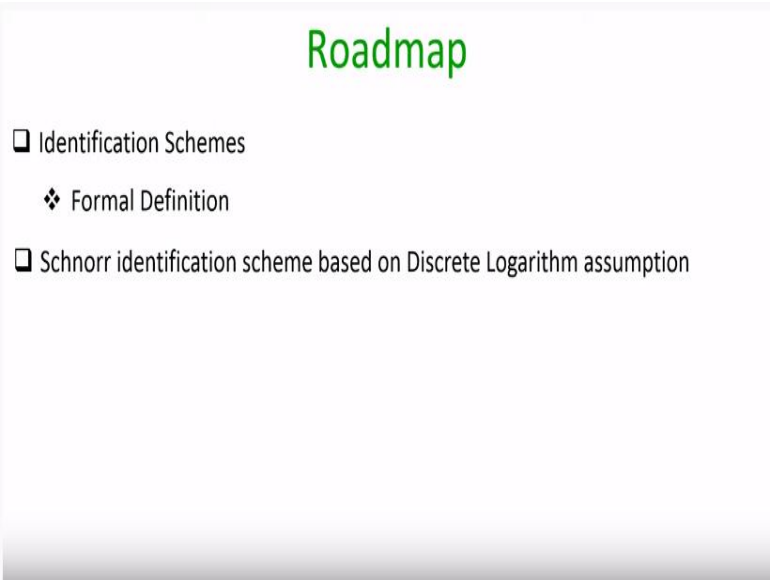
Foundations of Cryptography
Dr. Ashish Choudhury
Department of Computer Science
International Institute of Information Technology – Bangalore

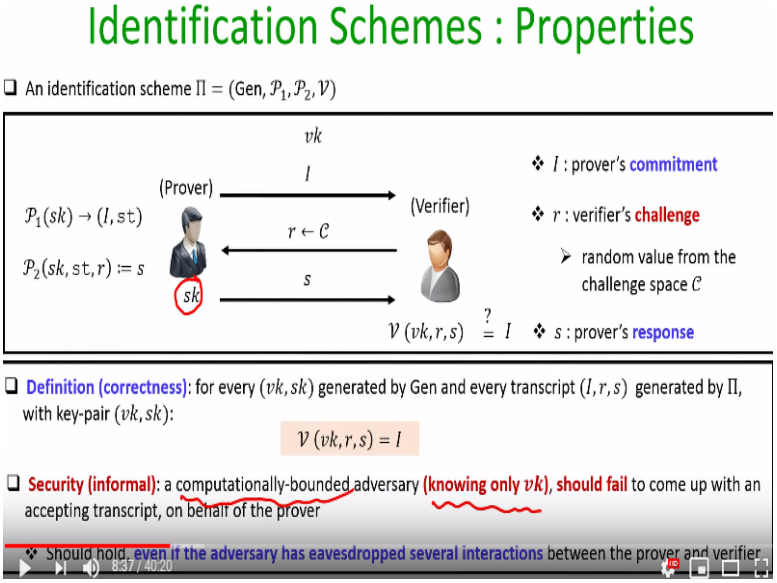
Lecture – 53
Identification Schemes

(Refer Slide Time: 00:39)



Hello everyone, welcome to this lecture. The plan for this lecture is as follows. In this lecture, we will introduce the definition of identification scheme, which is a well-known crytographic primitive and we will see an instantiation of identification scheme, namely we will see the description of Schnorr identification scheme based on the Discrete log assumption. In the next lecture, we will see that how using the Fiatt-Shamir heuristic, we can convert the Schnorr identification scheme into an instantiation of digital signature scheme based on discrete log assumption.

(Refer Slide Time: 01:01)



So let us try to understand the motivation behind the identification scheme. So, again let me take an example, which I stated during our first lecture. So, this is a well-known Sunderkand episode in Ramayana. So, Ram is in India and he is very disappointed. He is missing mother Sita, so he instructs his messenger that please go and pass my message to Sita that I am missing her.

The messenger namely Lord Hanuman says as you wish my lord and then Ram says that since you are going to meet Sita for the first time, she may ask you to prove your identity. So you can take this ring as a proof because only Sita knows about this ring and the messenger takes the ring and he goes to Lanka and then he starts the interaction with mother Sita saying that I am the messenger of Lord Ram and have a message for you.

But Sita is so scared there, so she is not willing to believe Hanuman, then she asks how can I trust you and prove your identity and Hanuman proves his identity by showing the ring which Lord Ram has given to Hanuman and once Mother Sita sees the ring, she accepts the identity of the messenger. So, in this example, the ring serves as a proof and confirms Hanuman's identity and proof.

The proof that Hanuman gave is shown in clear, but it turns out, that it is extremely dangerous to show proof in clear in the current age of Kalyuga where proof is very volatile and people may not trust each other. So, identification scheme is a cryptographic primitive,

basically it is a n-interactive protocol between two entities, which allows a prover to prove its identity.

In this example, Hanuman without revealing the secret credentials namely the ring. So, let us go into the formal details of identification scheme and we are interested in the constructing of identification scheme in the public key setting, and more specifically we will focus on three round or commit challenge response identification scheme, but it is not necessary that your identification scheme should have three rounds.

But we are interested in only studying the identification scheme consisting of three rounds of interaction, because later on we will see how we can construct signature schemes from three-round identification schemes. So an identification scheme consists of four protocols. So we have key generation algorithm, and we will have two algorithms, P_1 and P_2 for the prover who wants to prove his identity. And we will have an protocol or an algorithm for the verifier using which the verifier can verify the identity of the prover. So, the way we use an identification scheme is as follows. So, we have a prover and a verifier. The key generation algorithm will be run mostly by the prover and it will run the key generation algorithm to obtain a verification key and a secret key.

The verification key will be available in the public domain to verify the identity of the so called prover whereas the secret key will be available only to the prover and using this identification scheme, the goal of the prover is to convince the verifier who is aware of the verification key that indeed the prover knows the corresponding secret key sk associated with this verification key vk and the way this happens is by a 3-round protocol.

So during the first round, prover runs the algorithm P_1, which takes an input secret key and outputs commitment, which we denote by I, and the commitment is given to the verifier. The prover sends the commitment to the verifier and along with that the algorithm P_1 outputs state information, which prover keeps with itself. Now seeing the commitment, the verifier picks a challenge, which I denote by r.

This challenge is selected from a challenge space and the challenge is picked uniformly randomly from the challenge space and on seeing the challenge r from the verifier, the prover has to come up with a response, and the response is denoted by s, which is computed by

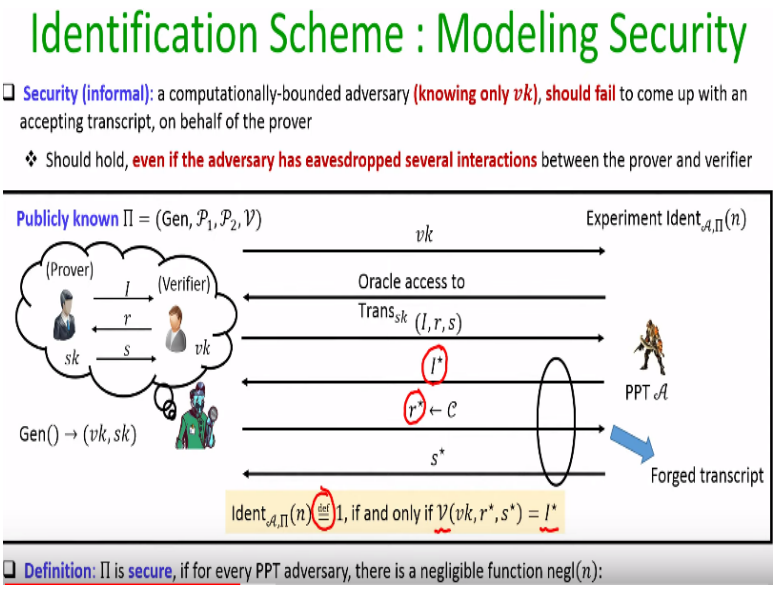
running the algorithm P_2, which takes the secret key sk, the state information and the challenge.

On seeing the response, the verifier has to verify whether the prover has responded correctly in response to the challenge r, with respect to the commitment I, which prover has committed at round 1 and to verify the verifier runs the algorithm V, which takes the verification key, the challenge and the response and the goal of the verifier is to verify whether the output of this algorithm V is equal to the commitment I or not.

If the output matches the commitment I then we say that verifier accepts the identity of the prover, that means verifier is convinced that indeed prover is the person who knows the secret key corresponding to the publicly available verification key, whereas if this test that the output of the algorithm V should be equal to I fails, then the verifier is not convinced that the prover who actually participated in this protocol knows the corresponding secret key sk.

So a successful execution in this protocol implies that the communication happened indeed with the intended prover and not an imposter. More specifically we require the following two properties from an identification scheme. So, the first property is the correctness property, which says that for every pair of key which your key generation algorithm could output and every transcript, which is generated by running an instance of the identification scheme, the following should hold. If the verifier runs the verification algorithm with respect to the verification key and the r and s component of the transcript, it should give the I component of the transcript. That means the verification should be successful at the verifier side. So, that is the correctness property. The security property informally requires that an imposter or an eavesdropper who has eavesdropped an interaction between a prover and verifier polynomial number of the times should not be able to come up with accepting transcript and successfully get it accepted at the verifier side and this should hold if my adversary is computationally bounded. So basically what it means is, if the adversary has seen polynomial number of conversations between an honest prover and an honest verifier then even after seeing a polynomial number of conversations in the absence of the secret key sk and only with the knowledge of the verification key vk, it should not be possible for the computationally bounded eavesdropper to pretend as a prover and come up with an accepting conversation, which gets accepted at the verifier side.

(Refer Slide Time: 08:48)



So, we model this requirement this informal requirement by a security experiment. So in this experiment, which we call as the identification experiment $\text{Ident}_{\mathcal{A},\Pi}(n)$, we have a computationally-bounded adversary and the challenger and the description of the identification scheme is publicly known, so the challenger goes first and it runs the key generation algorithm, keeps the secret key with itself, and sends the verification key to the adversary.

Now, what the adversary can demand is, it can demand for oracle access to the transcription service and this models the fact that in the real world, there might be an adversary who might have seen polynomial number of interactions, or polynomial number of instances of the identification scheme getting executed between an honest prover and an honest verifier. So, that adversary might have seen polynomial number of transcripts under the unknown key sk, which might be only available with the prover.

So, to model that, we give the adversary here Oracle access to the transcript service, so to respond to this Oracle access, what the challenger has to basically do the following. So, it knows the verification key vk, and it knows the secret key sk as well, so it runs the instance of this identification scheme, simulating the role of the prover and the verifier in its mind. So, basically, it just runs the instance of this identification scheme as playing the role of the prover itself.

Playing the role of the verifier itself and generates the corresponding I, r, s and that transcript is given back in response to the Oracle access service that the adversary has asked for and

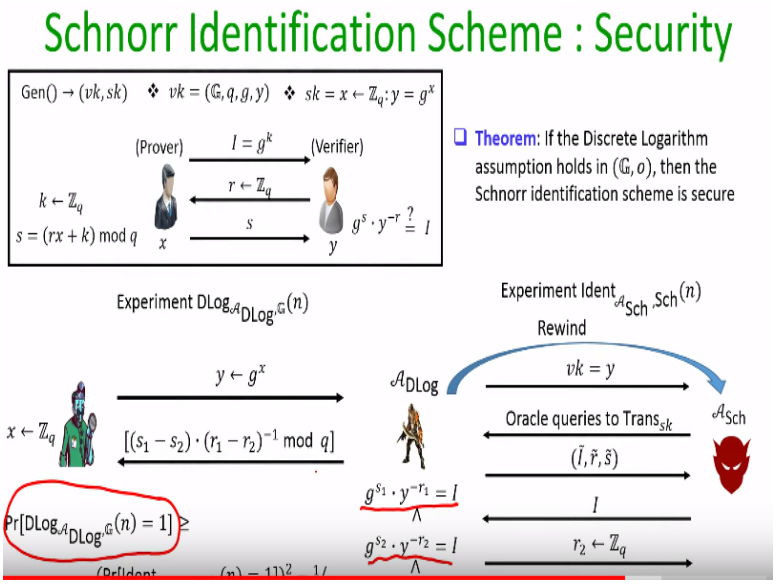
since the adversary could ask for Oracle access to the transcript service for a polynomial number of times, every time such an Oracle access or Oracle request comes, the challenger has to generate a simulated transcript like this and given it to the adversary.

Once the adversary is trained by seeing the polynomial number of transcript, it tries to come up with a forged transcript and try to get it accepted by the challenger. So to do that, it pretends as if it is the prover and tries to come up with the accepted transcript without even knowing the corresponding secret sk, so it submits a commitment, which I denote by I*. In response the challenger submits a challenge, which I denote by r*, and in response to the challenge the adversary submits a response s*.

This triplet I*, r*, and s* is a forged transcript, which the adversary is trying to produce with respect to this entire experiment and the definition of the experiment says that adversary is able to forge a transcript, which is denoted by saying that the output of this experiment is one, if and only if the verification algorithm v, when run by the challenger with respect to the verification key, and r* component, and s* component of this forged transcript indeed gives I*.

That means, I*, r*, s* constitutes a accepting transcript, and our definition of security is we say that an identification scheme is secure if for every poly-time adversary participating in this experiment, the chance that it can win the experiment is upper bounded by some negligible function.

(Refer Slide Time: 12:42)



So, that is our definition of our identification scheme. Now, let us see whether we can come up with an instantiation of such a scheme and there is a well-known identification scheme due to Schnorr, and it is based on the following idea. So, basically prover in this identification scheme tries to prove its identity by saying that it knows the discrete log of a publicly known value y under the base g $\mathrm{DLog}_g(y)$, and to verify the claim of the prover, the verifier basically challenges the prover to show a random linear combination of the discrete log of y, where the random combiners for the linear combination will be selected by the verifier.

So, the idea here is that indeed if prover knows the discrete log of y, then it should be able to produce a random linear combination of the discrete log of y with any other value from the corresponding range of the discrete log, and this whole interaction happens in the zero knowledge fashion in the sense that throughout the interaction, it will be ensured that indeed the prover knows the discrete log of y to the base g, then the discrete log is not learnt by a malicious verifier.

So, the key generation algorithm of this scheme is as follows. It outputs a verification key and the secret key, where the verification key is the description of a cyclic group of order q and a description of the generator and the random element y from the group where y is basically g^x, where x is selected from the set 0 to q - 1 and the verification key namely the discrete log of y, which is x will be available with the prover, whereas the verification key namely y will be available with the verifier.

So, the prover goes first in this identification scheme and it commits a value k from the selected set Z_q by computing g^k, so that is a random value which it gives to the verifier and the challenge picked by the verifier is a random value r, selected from the set Z_q and to respond to the challenge, basically the prover has to come up with a linear combination of the discrete log x of the y.

The value k which it has selected in the round 1 and the random linear combination here is (r * x + k) modulo q and to verify whether the response of the prover is correct or not, verifier has to verify whether $g^s * y^{-r} = I$ or not, which should actually be the case if indeed prover knows x and it has sent g^k during the first round.

So, before going into the analysis of this identification scheme, let us see a definition here, we say a triple I, r, s where I is a group element, and r and s are elements of Z_q is an accepting transcript if $g^s * y^{-r} = I$ holds and the correctness property of the identification scheme are of Schnorr follows from the fact that indeed prover and verifier are honest and prover knows that discrete log of y.

Namely it knows x, then the transcript generated by running an instance of the Schnorr identification scheme will indeed be an accepting transcript. So the verification at the verifier's end will be successful. So that proves the correctness property. Now, let us try to understand the security property here. So, we first consider an eavesdropper here and imagine prover and verifier are honest and there is an eavesdropper who has monitored polynomial number of executions of the Schnorr Identification scheme.

So imagine it has eavesdropped upon one transcript, which is I, r, s, and I claim here that by seeing the transcript not learn anything about the secret key sk, namely the discrete log of y to the base g, which is x, and this is because if you see the distribution of the commitment namely the I, it is independent of x because the commitment I is g^k, where k is picked independent of x.

In the same way, the r component of the transcript, it is completely independent of x and it is picked by the verifier, so it also does not reveal anything about the secret x. However, if you see the value s, then the value is (r * s + k), so one might feel that by seeing s, the eavesdropper might learn something about x, but that is not the case here because the distribution of s here is independent of x because the k which is used in the linear combination compute s is independently and randomly picked by the prover, and if the prover is honest, then the value k, which is used in the linear combination, will be uniformly random and unknown to the adversary. That means just by seeing the adversary, again cannot figure out anything about x and that means an eavesdropper who sees an accepting transcript I, r, s, will not learn anything about the underlying secret x.

What the adversary or the eavesdropper will learn just that the transcript I, r, s is an accepting transcript and its distribution is independent of x. So based on this observation, we can make a very strong claim here that, we can say any eavesdropper can simulate an accepting

transcript based on the knowledge of verification key itself. That means, it is as good as saying that even if no interaction happens between the prover and the verifier the eavesdropper could well ahead come up with a probability distribution of the transcript, which would be seen by eavesdropping real conversation between the prover and the verifier and how can this be possible, here is the way the adversary or the eavesdropper could come up with a simulated transcript on its own without even eavesdropping the conversation between the prover and the verifier.

So, what the eavesdropper could do is, it could randomly pick an r value and s value from Z_q and then once it picks the r value and s value, it could set the I value as g^s value it has picked, multiplied by y^{-r} value that it has picked, and it turns out that if you compare the probability distribution of the real transcripts, and by real transcripts, I mean the transcripts, which are actually generated by real execution of this Schnorr Identification scheme where an honest prover and an honest verifier participates in the protocol.

If we consider the probability distribution of the simulated transcripts, whereby simulated transcripts, I mean, the transcripts that are generated by the eavesdropper by this method, where it does not see the real execution of the protocol, but it comes up with the values of I, r, s using the method, which I have discussed just now. So if I consider the probability distribution of these two transcripts, they are exactly identical.

This is because if you see probability distribution of the r value in the real transcripts and the probability distribution of the r value under simulated transcripts they are identical. In a real execution, r will be randomly picked from the set Z_q, and in the simulated transcript also r values are also picked randomly from the set Z_q. In the same way in the real transcript s is going to be a uniformly random value from Z_q because k would have been chosen uniformly randomly by the prover.

The same is true for the s value in the simulated transcripts and if you see the probability distribution of the I value in the real transcripts as well as in the simulated transcripts in both the cases $I = (g^s * y^{-r})$ parts of the transcripts and this is true both for the real transcript as well as for the simulated transcript. So if you see the distribution wise, the way the transcripts would have been generated in a real execution of the protocol and the way the transcripts are generated by the adversary in its mind without actually seeing any conversation have exactly

the same distribution. That means that just eavesdropping the communication between a honest prover and a verifier is not going to help the adversary to learn anything about the underlying secret key x. Now here is a food for thought for you.

Since I am claiming here that if eavesdropper could simulate and come up with an accepting transcript on its own without even knowing the secret key sk, does that mean that using the strategy, which the simulator is using or the eavesdropper is using to come up with the simulated transcript, it could forge an accepting transcript and participate in an instance of the Schnorr identification scheme and end up convincing an honest verifier that indeed it knows the x value, which is not exactly the case.

It turns out that is not the case because the reason for that is if you see the simulation strategy here, the way adversary has come up with simulated transcript, it is fixing the r value and s value to begin with. That means it is guessing in its mind that this could be the r value or the challenge value, which the verifier would pick and only after fixing the r value and s value, it is coming up with its commitment I.

So, if with this strategy tries to participate in an execution of the Schnorr Identification scheme with an honest verifier, then the probability that indeed he is able to come up with an accepting transcript which gets accepted by the verifier is same as the challenge r̃, which the adversary is thinking well ahead in its mind matches exactly the transcript which an honest verifier is indeed going to pick up during the real execution of the Schnorr Identification Scheme.

But the probability that the simulated r value, which adversary is guessing in its mind well ahead matches the exact value of the challenge r which is going to be picked by the verifier is one upon the size of Z_q set, namely its 1/q and if q is sufficiently large, then this is a very small quantity a negligible function in the security parameter. So, that means this simulation strategy is not going to help the eavesdropper to win or forge or break this identification scheme.

So what we have proved till now is that eavesdropping is definitely not going to help the adversary to break the security of the Schnorr Identification Scheme, so only way it could attack this identification scheme is follows, without knowing the secret key is that it has to

interact with a honest verifier as follows. So it has to come up with some commitment with respect to some strategy that the adversary has in its mind, and once the verifier throws a challenge, it has to come up with a response s, such that $g^s * y^{-r}$ indeed equal to I and if we want the adversary should be able to break the security of the Schnorr Identification Scheme with high probability, then it should be the case that this adversary who is trying to break the security should be able to come up with its response s irrespective of what value of r is used as a challenge by the verifier.

That means, what I am trying to say here is that once adversary has committed some value and submitted its commitment I, and if indeed that adversary is able to break the security of this identification scheme with very high probability, then it does not matter what exactly is the challenge. It could be r_1, it could be r_2, it could be anything. For any value of challenge, it should be possible for the adversary to come up with the responding response s such that g^s multiplied by y^{-r} should be equal to the commitment of the adversary.

That means, once the adversary has submitted its commitment, it does not matter whether the challenge is r_1, corresponding to the r_1 the adversary should be able to come up with s_1 such that this property or this verification is successful or it does not matter whether the verifier throws the challenger as r_2, still it should be possible for the adversary that for the same commitment I and the challenge r_2 the adversary should be able to come up with the corresponding response s_2, such that the verification is successful at the verifier's end because if the adversary knows only to come up with the response for some specific values of the challenge, then that is not a good adversary strategy. The probability that the adversary is able to break the security of the identification scheme is not significantly high.

Its significantly high only when irrespective of what is the challenge my adversary should be able to come up with the corresponding valid response, but if you see closely here, if adversary is able to come up with successful response s_i irrespective what is the value of r_i that means if for the same commitment I but for different challenges r_1 and r_2, my adversary is able to come up with the corresponding accepting responses s_1 and s_2, then by solving these two equations $(g^{s_1}, y^{-r_1}) = I = (g^{s_2}, y^{-r_2})$, we know that adversary actually knows how to compute the discrete log of the value y because if it indeed the commitment I, r_1, and s_1 is a accepting transcript. So is the case for the transcript I, r_2, and s_2, then it means the discrete log of y is nothing but $[(s_1 - s_2).(r_1 - r_2)^{-1}]$ and since s_1, s_2 and r_1, and r_2 are all known to the

adversary, that means the adversary actually knows to compute a discrete log of y, which goes against the assumption that discrete log of problems is difficult to solve in my underlying group.

That means, if I make the assumption that discrete log problem is difficult to solve in my underlying group, then it is very difficult for any adversary to respond with s_2 corresponding to the challenge r_2, as well as to respond with the response s_1 corresponding to the challenge r_1. So, now let us try to formalize this argument through a concrete security reduction proof, so the theorem which we want to prove here is that if the discrete problem log is difficult to solve in the underlying cyclic group, then the Schnorr Identification Scheme is secure, and assume on the contrary, we have an adversary who can break the security of the Schnorr Identification Scheme with significant probability, then we can use that adversary to come up with another adversary or algorithm, which can win an instance of discrete log problem, which can solve an instance of discrete log problem, with significant probability and here is how the reduction goes.

So, the adversary which we want to construct is $\mathcal{A}_{\text{DLog}}$. It participates in an instance of the DLog experiment, where it is thrown g^x (unknown x) as a challenge and its goal is come up with the x. It invokes the adversary who can break the security of the identification scheme by the setting y the verification key of the identification scheme, so now this adversary could ask for oracle access to the transcript service as per the identification scheme, and the DLog adversary does not know the secret key sk because the secret key is x, but x is not known to this discrete log adversary.

So you might be wondering that how exactly it is possible for this DLog adversary to respond to this oracle queries to the transcript service of the identification scheme. But that is possible because in the couple of slides back, we had discussed that any eavesdropper could come up with a simulated transcript say Ĩ, r̃, s̃, whose distribution will be exactly the same as an accepting transcript, which an honest prover and an honest verifier would have generated by participating in a real instance of the Schnorr Identification Scheme. So, what our DLog adversary could do is in response to this transcript oracle service, it could just come up with such simulated transcripts and respond back to the adversary. Now once adversary is

sufficiently trained, it will try to come up with a forged transcript, so it submits its commitment I, and then in response these DLog adversary throws a challenge r_1.

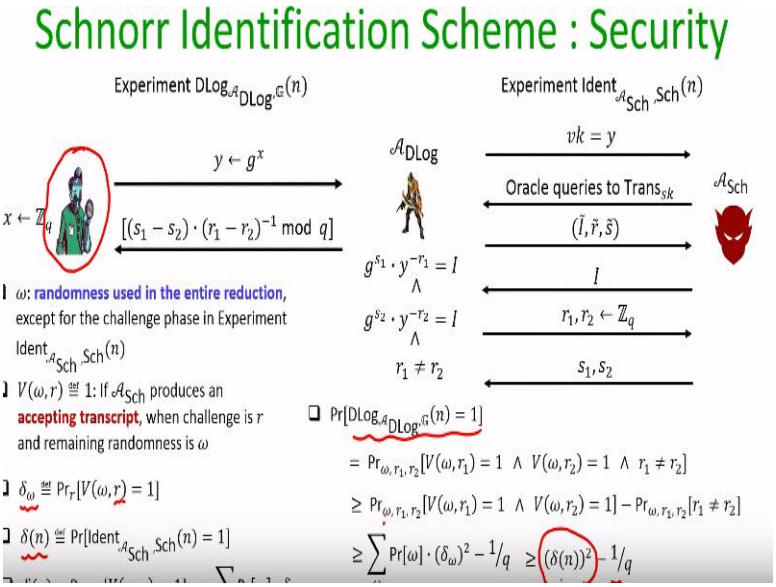
It sees the response of the adversary against the identification scheme say s_1, now what the DLog adversary is going to do is, it is going to rewind the adversary who can break the security of your identification scheme, namely it asks the adversary that you please go two steps back to the step where you have submitted the commitment I. Now I want to test you with a new challenge say r_2, which is randomly chosen from the set Z_q, and this adversary against the identification scheme has to now respond corresponding to the challenge r_2.

So it responds with s_2, and now what does DLog adversary is going to do is, it checks whether I, r_1, s_1, is an accepting transcript namely it checks whether the first condition here holds and then it checks whether I, r_2, s_2 is an accepting transcript namely the second condition holds, and finally it checks whether the two challenges, which has picked randomly are different or not.

If all these three conditions hold simultaneously for the DLog adversary, then it submits or it computes the discrete log of y to be the product of s_1 minus s_2 and the multiplicative inverse of r_1 minus r_2 and since r_1 and r_2 are different, r_1 minus r_2 will be not zero, hence this multiplicative inverse modulo q will exist. Now, I claim that in this whole reduction the advantage of the discrete log solver that we have constructed namely the probability that it can solve or win the discrete log experiment is at least the square of the probability that the adversary against the identification scheme breaks the security of the identification scheme minus 1/q.

If I am assuming that the discrete log assumption or the discrete log problem is difficult to solve in my underlying group, then I know that the probability in my left hand side here of the inequality is some negligible function and if q is also some exponential function in the security parameter, then I know that 1/q is also some negligible function. So, if my left hand side is negligible function and 1/q is also a negligible function, then it automatically implies that the advantage of the adversary against the identification scheme also has to be negligible.

(Refer Slide Time: 33:10)



So, now let us try to prove this claim formally. So for this I introduce some notation. So, let ω denote the randomness used in this entire reduction except the challenge values r_1, r_2 which are picked by the discrete log solver. So, the ω denotes the randomness used by the challenger in the discrete log experiment. It denotes the randomness used by the adversary against the identification scheme, and it also denotes the randomness used by the discrete log solver except for the randomness used to pick up the challenges r_1 and r_2 in this whole experiment.

Now, I use this quantity V(ω, r), and I say that this function V(ω, r) = 1, if corresponding to the randomness ω and corresponding to the challenge r, the adversary against the identification scheme could come up with an accepting transcript. If that is the case, then I say the output of the function V(ω, r) is 1, otherwise it is zero and with respect to a fixed randomness ω, this quantity δ_ω is defined to be the probability that the output of the function V with respect to the fixed randomness ω over all possible challenge randomness r = 1.

That is my quantity δ_ω, so it is basically the probability of the adversary against the identification scheme coming up with an accepting transcript with respect to a fixed randomness ω over all possible challenge randomness r, and then let me call this function δ(n) to be the probability with which this adversary against the identification scheme can win the security game against the identification scheme in this reduction.

So, as per our notations that we have introduced till now this δ(n) is nothing but probability of all randomness ω used in this entire reduction and the probability over all challenge

randomness r, the probability that $V(\omega, r) = 1$, and if we expand it further, it is nothing but summation of all randomness ω, that what is the probability that ω is the randomness used in this entire reduction except for the challenge randomness and with respect to that fixed randomness ω, what is the probability of δ_ω.

So that is the way we expand our function δ(n). Now, if you see this entire reduction, our discrete log solver can successfully extract out discrete log x only if these three conditions hold namely I, r_1, s_1 is an accepting transcript, and I, r_2, s_2 is an accepting transcript and the challenges r_1 and r_2 are different where the challenges r_1 and r_2 are randomly chosen by the discrete log solver.

So, we can formally state that the probability that our discrete log solver can solve the discrete log is the probability that if you take the probability over all randomness ω and all challenge randomness r_1 and r_2, $V(\omega, r_1) = 1$. This captures the fact that I, r_1, and s_1 should be accepting transcript and $V(\omega, r_2)$ should be 1. This captures the fact that I, r_2 and s_2 should be accepting transcript and r_1 should be different from r_2. This captures the third condition.

So now what we have to do is just basically we expand this probability expression because this probability are over all candidate randomness ω, challenge randomness r_1 and challenge randomness r_2. So, by using the rules of probability if I solve, then this quantity r_1 not equal to r_2, I can replace by the probability r_1. I can take this probability of ω, r_1 and r_2 inside and then I can substitute this AND condition by this subtraction condition and what I can do is I know that this thing that probability that my challenge randomness r_1 and r_2 are different is 1/q because both r_1 and r_2 are picked randomly from the set Z_q and now what I can do is, I can expand this first quantity and the second quantity with respect to r_1 and r_2 at fixed ω values, and once I fix ω, these two events of $V(\omega, r_1)$ should be 1 and $V(\omega, r_2)$ should be 1, they are independent of each other, because r_1 and r_2 are picked independently by the discrete log solver.

So, if I fix the randomness ω, and take the ω inside and try to expand with respect to the randomness r_1 and r_2, the inequality here then basically I get that the above inequality turns out to this thing and now I can apply the well-known Jensen's inequality, which I am not stating here. You can use any standard reference to find out formula for Jensen's inequality,

what I can do is I can take the square here inside the expression probability of randomness ω to happen here.

If you now see that this entire big bracket here, the square of this big bracket is nothing but δ(n), that is what the definition of δ(n), δ(n) is nothing but the probability with which the adversary against the identification scheme can win the game here in the reduction and that is what is the claim we wanted to prove. We have proved that the advantage of the discrete log solver here is greater than or equal to the square of the advantage of the adversary of the identification scheme minus 1/q.

If q is some exponential function in the security parameter, then this 1/q is negligible and as per the assumption that is advantage of the discrete log solver should be some negligible function that proves that the square of the advantage of the adversary against the identification scheme also should be negligible. So, that brings me to the end of this lecture. Just to summarize, in this lecture we have introduced identification scheme, which is a well-known cryptographic primitive and we have seen an instantiation of identification scheme based on discrete log assumption namely the Schnorr Identification Scheme. Thank you.