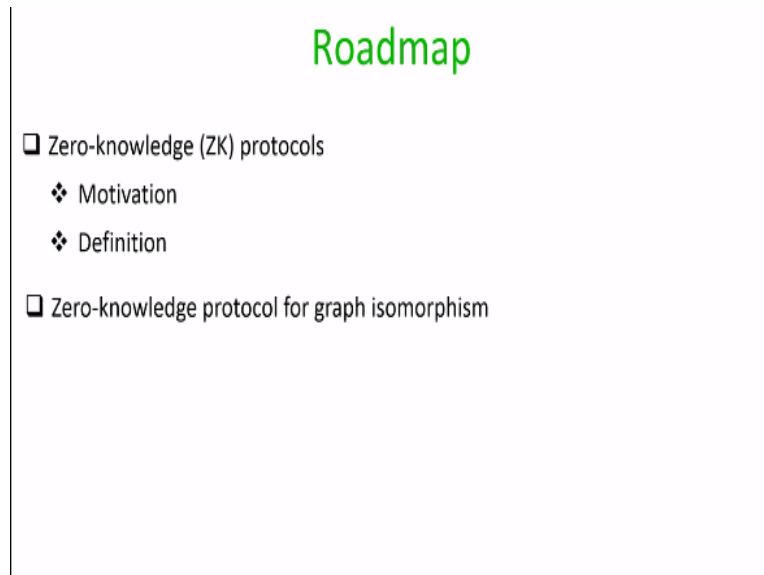


**Foundations of Cryptography**  
**Dr. Ashish Choudhury**  
**Department of Computer Science**  
**International Institute of Information Technology – Bengaluru**

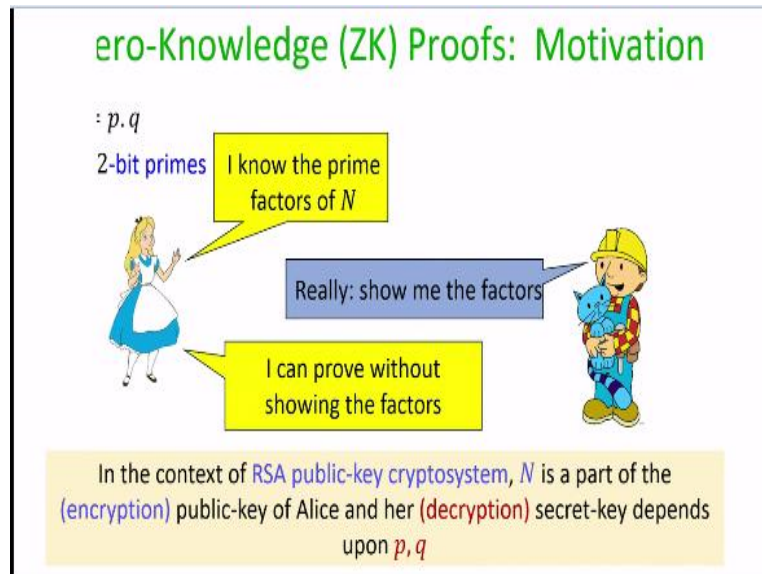
**Lecture – 57**  
**Zero Knowledge Protocols Part I**

**(Refer Slide Time: 00:30)**



Welcome to this lecture. So in this lecture we will continue our discussion on interactive protocols where we wanted to solve a bigger problem rather than the problem of secure communication. So in this lecture we will introduce Zero-knowledge protocols. We will see some of the motivation first in the zero-knowledge protocols and a formal definition. And we will see a zero-knowledge protocol for the graph isomorphism problem.

**(Refer Slide Time: 00:52)**



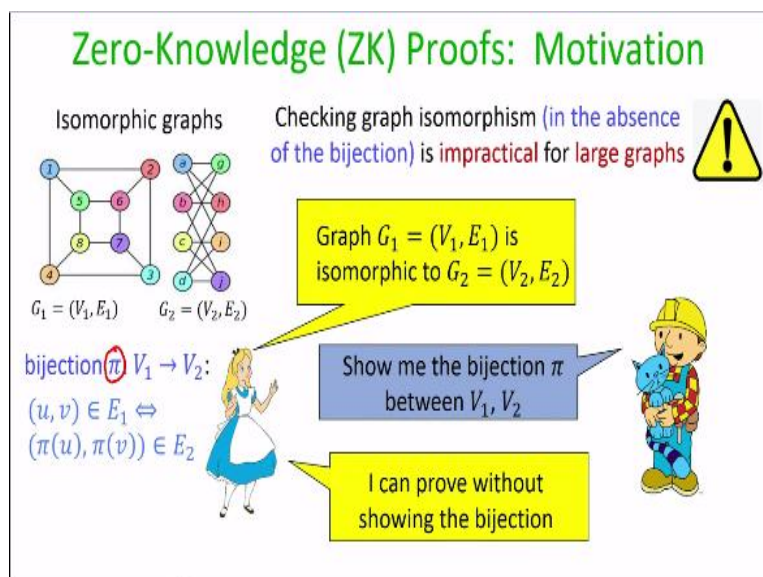
Now let us start by trying to understand a motivation for zero-knowledge proofs. So imagine we have two parties Alice and Bob and say Alice had picked two random prime numbers  $p$  and  $q$  say each of size 512 bits and it has computed the product of  $p$  and  $q$  to obtain the product  $N$  and she goes and claims to Bob that I know the prime factors of  $N$ . Now if indeed she wants to prove to Bob that she knows that prime factors of  $N$  then a simple way to prove that is to show the values of  $p$  and  $q$ .

Because if the values of  $p$  and  $q$  are given to Bob, Bob itself can multiply those two numbers and see whether it matches  $N$  or not. But that is what we call as proof in clear. Because  $p$  and  $q$  are here witness for Alice for the statement that she knows the prime factors of  $N$ . One way of proving her statement is to show the witnesses in clear but what a zero-knowledge protocol is going to do here is it's going to allow Alice to convince Bob that indeed she knows the prime factors of  $N$  without actually showing the witnesses namely  $p, q$ .

Because  $p$  and  $q$  might come secret information for Alice and in this whole process of zero-knowledge proof basically Alice ends up convincing Bob that she knows  $p$  and  $q$  without actually revealing  $p$  and  $q$ . Now you might be wondering where this  $p$  and  $q$  is useful. So if you remember in the context of RSA public-key cryptosystem, the value  $N$  is nothing but a part of the public-key of Alice and  $p, q$  are nothing but part of a decryption key.

So if indeed Alice want to convince to Bob that N is her public key and she knows the corresponding secret key without showing the components related to the secret key namely p and q; she we can convenience Alice by using this zero-knowledge proof.

(Refer Slide Time: 02:45)



Let us see another motivation. Imagine Alice has two graphs here. And pictorially these two graphs are drawn in a different way. The vertex names are different; the edge names are different then so on. But it turns out that information wise the two graphs are isomorphic or structurally the graphs are isomorphic to each other and what I mean by isomorphic graphs here is that, if you consider these two graphs then for these two graphs there exist a bijection from the vertex set of the first graph to the vertex set of the second graph.

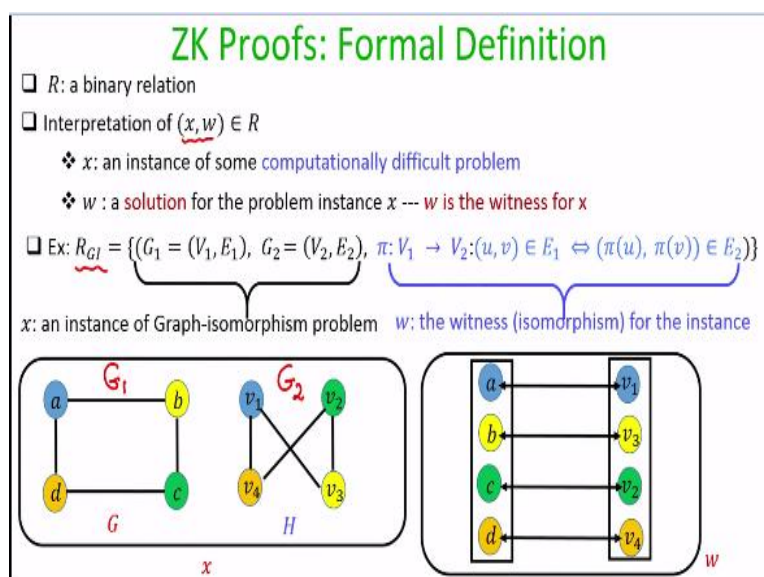
And that bijection has the property that if there is an edge between the nodes u and v in the first graph then in the second graph there exists an edge between the mapped u set and mapped v set. And this is an if and only if statement. That means in the same way if you have an edge between the mapped u vertex and the mapped v vertex in the second graph then an edge exists between the u vertex and the v vertex in the first graph. So in that sense these two graphs are isomorphic even though pictorially they are looking different.

So imagine Alice has two isomorphic graphs, so she makes the description of the two graphs available to Bob and she makes the statement that she claims to Bob that these two graphs are

isomorphic. So that is the statement. Again Bob cannot directly check whether the two graphs are isomorphic or not and verify Alice claim because its impractical to verify whether two graphs are isomorphic or not in the absence of the bijection which is available or which maps the vertex set of the first graph to the vertex set of the second graph.

So one way for Bob to verify the statement of Alice is it can ask Alice that please give me your witness namely the bijection  $\Pi$ . If you give me the bijection  $\Pi$  then I can verify whether indeed for every edge in the graph, first graph the corresponding mapped vertices also constitute an edge in the second graph and so on. But that will be a proof in clear. So what a zero-knowledge proof or a zero-knowledge protocol will allow Alice to do is; it will allow Alice to convince Bob that indeed these two graphs are isomorphic without actually showing the witness namely the bijection  $\Pi$ .

(Refer Slide Time: 05:10)



So that means a zero-knowledge proof is a kind of an interactive protocol between two entities a prover and a verifier which allows a prover to prove a statement to verify without actually showing anything about the underlying witness. So now let us formalize this statement. So imagine  $R$  is a binary relation and the interpretation of this relation is as follows. So if I have a pair  $(x, w)$  present in this relation  $R$  that should be interpreted as if  $x$  is an instance of some computationally difficult problem.

When I say computationally difficult problem informally it means in poly amount of time it is not known how to solve that problem. But again that is a very loose statement but that is an intuitive understanding of computationally difficult problem here and a  $w$  here is a solution for that problem instance  $x$  namely you can consider the  $w$  to be a witness for the problem instance  $x$ . So to understand this relation  $R$  let us consider the graph isomorphism relation here.

So an  $(x, w)$  pair in the graph isomorphic relation  $R_{GI}$  will look like this  $\{(G_1 = (V_1, E_1), (G_2 = (V_2, E_2)))\}$  and the interpretation of elements present in this graph isomorphism relation should be as follows. So the first part here is the description of the two graphs namely it is a problem instance. That means someone wants to prove or disprove these two graphs are isomorphic. And the corresponding witness is nothing but the isomorphic or the bijection from the vertex set of the first graph to the vertex set of the second graph.

So for instance if you consider these two graphs here these two graphs are indeed isomorphic. So first graph is your graph  $G_1$  and the second graph here is your graph  $G_2$ . So that is the  $x$  component of the any  $x, w$  which might be present in this graph isomorphism relation. And a corresponding witness with respect to this specific  $G_1, G_2$  graph is nothing but the bijection from the vertex set of graph  $G_1$  to the vertex set of graph  $G_2$  and so on.

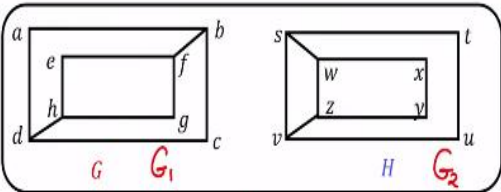
(Refer Slide Time: 07:18)

---

### ZK Proofs: Formal Definition

- $R$ : a binary relation
- Interpretation of  $(x, w) \in R$ 
  - ❖  $x$ : an instance of some computationally difficult problem
  - ❖  $w$ : a solution for the problem instance  $x$  ---  $w$  is the witness for  $x$
- Ex:  $R_{GI} = \{(G_1 = (V_1, E_1), G_2 = (V_2, E_2)), \pi: V_1 \rightarrow V_2: (u, v) \in E_1 \Leftrightarrow (\pi(u), \pi(v)) \in E_2\}$

$x$ : an instance of Graph-isomorphism problem      $w$ : the witness (isomorphism) for the instance



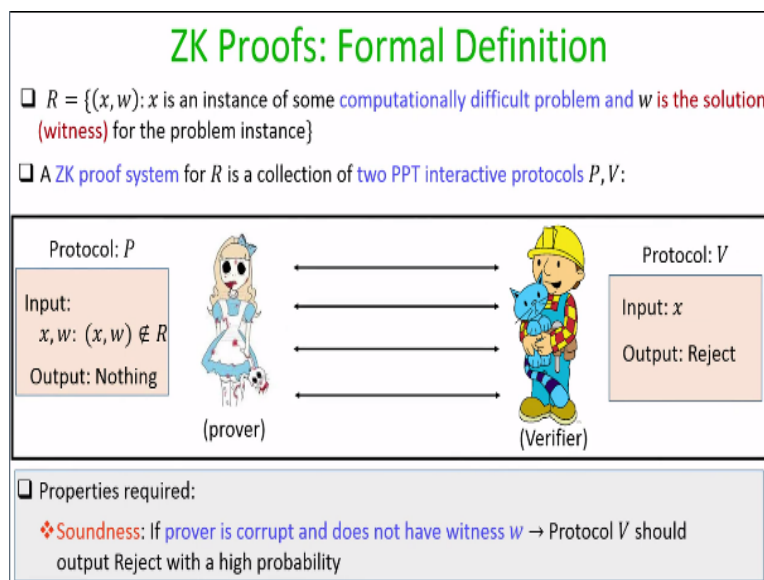
No witness for this instance  $x$  (as the graphs are non-isomorphic) and hence does not belong to  $R_{GI}$

$x$

---

On the other hand, if I consider these two graphs to be my  $G_1$  and  $G_2$ , it turns out that these two graphs are not isomorphic to each other and as a result if this is my candidate  $x$  then for this candidate  $x$  I do not have any candidate  $w$  such that  $x, w$  belongs to this relation  $R_{GI}$ , right. That means only those  $x, w$  will be present in  $r$  where the  $x$  instance has a corresponding witness  $w$ . If for an instance  $x$  there exists no witness  $w$  such that  $x, w$  satisfies that relationship then we say that,  $(x, w)$  is not present in the relation  $R$ .

(Refer Slide Time: 07:57)



So now let us go into the formal definition of zero-knowledge proofs. So we are given some publicly known relation which will have elements of the form  $x, w$  where  $x$  is an instance of some computationally difficult problem and  $w$  is the solution or the witness for that instance. Then a zero-knowledge proof system for the relation  $R$  consists of two poly time algorithms two randomized algorithms one for the prover and one for the verifier.

The input for the prover will be an  $x, w$  pair and the goal of the prover is to prove it knows a  $w$  such that  $x, w$  belongs to  $r$  whereas the input for the verifier algorithm will be the problem instance  $x$ . And the goal of the verifier is to verify whether indeed Alice knows a  $w$  such that  $x, w$  belongs to  $R$  or not. So in the zero-knowledge proof system the prover will send messages or it will interact with the verifier where the messages for the prover will be computed as per the protocol  $P$  and the internal randomness which are used by the prover.

And at the end of the protocol prover outputs nothing whereas the verifier algorithm it will interact with the prover where the messages of the verifier will be computed as per the algorithm  $V$  and the internal randomness chosen by the verifier and the verifier is either going to output accept or reject. Accept means it accepts the fact that Alice knows some witness, reject means it does not believe in Alice statement.

Now what are the properties we require from a zero-knowledge proof system, the first property is the completeness property which demands that if both prover and verifier are honest and if indeed prover knows an  $x$ ,  $w$  such that  $x$ ,  $w$  belongs to  $R$  and both prover and verifier participates, performs all their actions as per the protocol  $P$  and  $V$ . Then with a very high probability the output of the verifier should be accept.

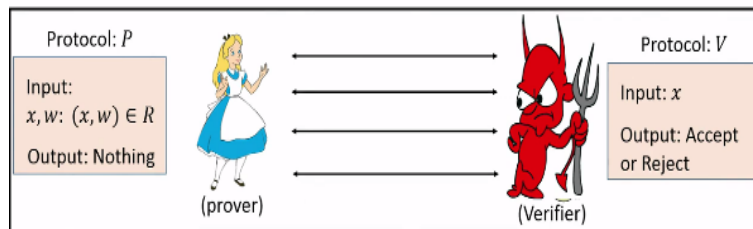
That means verifier should accept the statement of prover. The second property is Soundness property which we consider with respect to a corrupt prover. So what we require here is that if my prover is corrupt and it does not have an  $x$ ,  $w$  such that  $x$ ,  $w$  belongs to the relation  $R$  then irrespective of however she participates in the protocol the output of the verifier should be reject. That means a malicious prover who does not have  $x$ ,  $w$  or who does not have a witness such that  $x$ ,  $w$  belongs to the relation  $R$  with very less probability the prover should be able to convince the verifier that she knows the witness  $w$ . That means with very high probability the verifier should be able to catch a malicious prover. That is the soundness requirement.

**(Refer Slide Time: 10:50)**

## ZK Proofs: Formal Definition

□  $R = \{(x, w) : x \text{ is an instance of some computationally difficult problem and } w \text{ is the solution (witness) for the problem instance}\}$

□ A ZK proof system for  $R$  is a collection of two PPT interactive protocols  $P, V$ :



□ Properties required:

❖ Zero-knowledge: If prover is honest and verifier is corrupted  $\rightarrow$  probability distribution of the values received by the verifier is "independent" of  $w$

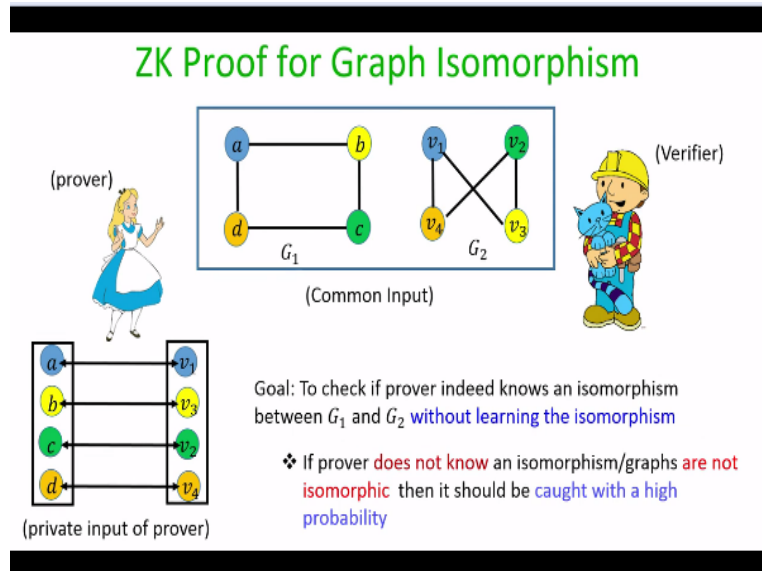
And the third property is a zero-knowledge property which is with respect to a corrupt verifier and a honest prover. So zero-knowledge property demands that if the prover is honest and the verifier is corrupt then irrespective of the way the verifier algorithm or the verifier participates in this protocol the verifier learns absolutely nothing about the witness  $w$  that is available with the prover. So here nothing is in-quote, unquote "nothing". It is not formal.

What exactly we mean by nothing is learned about the witness. If you want to little bit more formal we can say that, we say that the protocol has the zero-knowledge property if the probability distribution of the transcripts seen by the verifier is independent of the actual witness which is available with the prover. Again I am not formally proving what exactly it means to say that the probability distribution of the transcripts seen by the verifier is independent of  $w$ .

The actual formalism is little bit subtle and involved. So let us not go into the actual detail but for your understanding you can imagine that verifier should not learn anything about the witness if the verifier is corrupt by participating in this protocol.

**(Refer Slide Time: 12:06)**



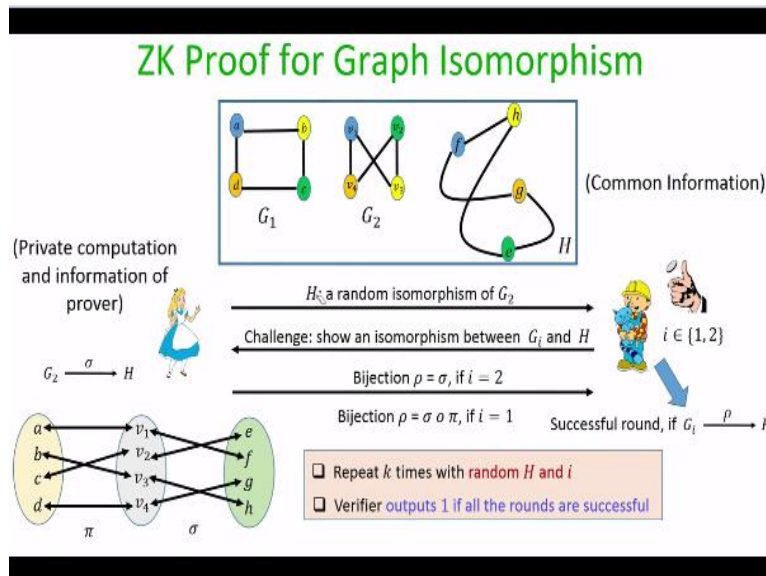


So now let us see a zero-knowledge proof system for a graph isomorphism problem. So the common input that is available to both prover and a verifier is an instance of a graph isomorphism problem namely the description of two graphs. And Alice wants to convince Bob or the verifier here the prover; the Alice is prover and the Bob is verifier. So Alice wants to prove to Bob that indeed these two graphs are isomorphic.

And she claims that she knows the isomorphism between these two graphs, that means she claims that she has a private input mapping the vertex set of the first graph to the vertex set of the second graph with respect to which these two graphs are isomorphic and the goal here is to design a protocol which allows Bob to learn whether indeed these two graphs are isomorphic or not without actually learning the isomorphism.

And it should be sound in the sense that if prover does not know the isomorphism or the graphs are not isomorphic at the first place then she should be caught while giving the proof with a very high probability.

**(Refer Slide Time: 13:07)**



So let us see how exactly the zero-knowledge protocol is designed here. So this rectangle box I have highlighted the public information so the public information available both to Alice and Bob are the description of the two graphs. And the private information available with Alice, the prover is the witness  $\Pi$  or the mapping from the vertex of first graph to the vertex set of the second graph.

So the first round of the zero-knowledge protocol is as follows. So what Alice does is, she creates a random isomorphic copy of the second graph  $G_2$ . And that is very easy to do, what she has to do is she has to basically come up with a random permutation say  $\sigma$  mapping the vertex set of the second graph and the resultant mapped vertices is going to give her another graph  $H$ .

So once she computes a random isomorphic copy of the second graph she sends the description of the random isomorphic copy of the second graph to Bob. So that is kind of a commitment. So I stress  $\sigma$  is randomly chosen and known only to Alice here. Now what Bob does is, Bob picks a random coin and with probably  $1/2$  the random coin could give the output 1 or with probability  $1/2$  it could give the output 2.

And now what Bob's challenge is, Bob challenges Alice to show an isomorphism between the  $i^{\text{th}}$  graph and the new graph  $H$  which Alice has committed. So if  $i = 1$  then basically Bob is challenging Alice to show an isomorphism between graph  $G_1$  and the new graph  $H$  whereas if  $i =$

2 then Bob is challenging Alice to show an isomorphism between the graph  $G_2$  and the new graph  $H$ . I stress here that when Alice was committing the graph  $H$ , she does not know well in advance whether she will be challenged to show an isomorphism between  $G_1$  and  $H$  or between  $G_2$  and  $H$ . She does not know that in advance.

From here viewpoint with probability  $1/2$  she could be challenged to show an isomorphism between  $G_1$  and  $H$  and with probability  $1/2$  she could be challenged to show an isomorphism between  $G_2$  and  $H$ . Now once Bob throws the challenge Alice has to respond. And the response will be different depending upon whether  $i = 1$  or whether  $i = 2$ . Namely if the challenge is  $i = 2$  that means if Alice is challenged to show the isomorphism between  $G_2$  and  $H$  then she can simply supply the bijection  $\sigma$  which she has used to compute  $H$  from the graph  $G_2$  graph.

So that will be her response. So I am denoting the response by  $\rho$  here. So  $\rho$  will be; the response  $\rho$  will be nothing but the mapping  $\sigma$  if  $i = 2$ . On the other hand, if Bob has challenged Alice to show an isomorphism between graph  $G_1$  and  $H$ , then basically Alice can respond back by composing the mapping  $\Pi$  which was her witness with the secret mapping  $\sigma$  that he has chosen to go from the graph  $G_2$  to  $H$ .

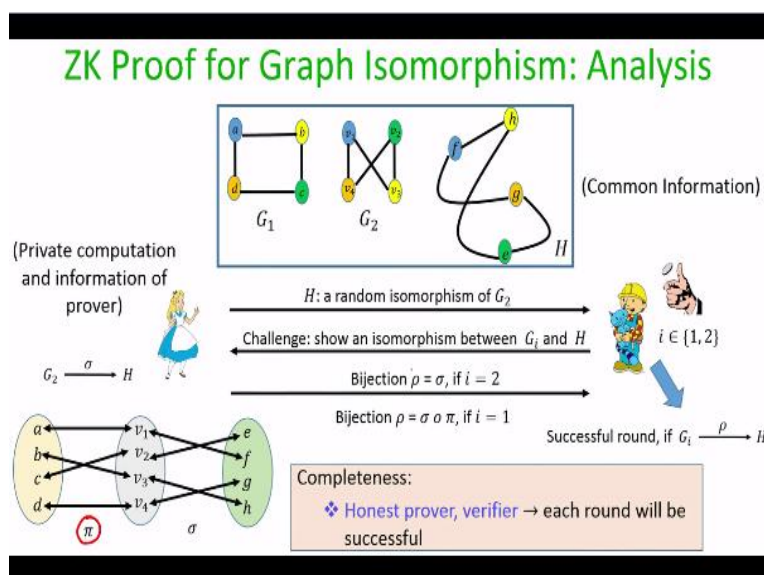
Because if we compose  $\Pi$  and  $\sigma$  then by using  $\Pi$  from  $G_1$  we come to  $G_2$ , Alice comes from  $G_1$  to  $G_2$  and then composing again with the mapping  $\sigma$  from  $G_2$  she can go to the graph  $H$ . So that means the way to go from  $G_1$  to  $H$  is to compose the mapping  $\Pi$  with the mapping  $\sigma$ . And if indeed Alice knows  $\Pi$  she should be able to compose  $\Pi$  and  $\sigma$  and she can respond back with this response  $\rho$ .

Now Bob has to verify whether indeed Alice has properly responded or not. So what Bob checks is it knows that Alice is suppose to show an isomorphism between the  $i^{\text{th}}$  graph and  $H$  and the response from Alice is mapping  $\rho$  and Bob just has to verify whether indeed the graph  $G_i$  takes Bob to the graph  $H$  as per this mapping  $\rho$  or not. If it is then Bob is convinced that Alice has successfully passed this round otherwise Bob says that Alice has failed in this round.

And what Bob is going to do is Bob is going to repeat this whole process namely Alice committing something Bob challenging and Alice again submitting the response  $k$  number of times and for each of these rounds Alice has to pick random commitment  $H$  that means she will be freshly picking the graph  $H$  for every round or every iteration and in the same way Bob will be randomly picking the challenges  $i$  for every round, independent of every previous round. That means it will not be the case that in every round Bob will be picking  $i = 1$  or  $i = 2$ .

And the verifier Bob is going to output 1 namely it will say that indeed Alice knows the secret mapping  $\Pi$ , if all the rounds are successful from the viewpoint of Bob, that means in all the rounds Alice has successfully submitted the responses  $\rho$ . Whereas if any of the rounds Alice fails then Bob outputs reject. That means Bob rejects the claim of Alice. That is the zero-knowledge protocol here for the graph isomorphism problem.

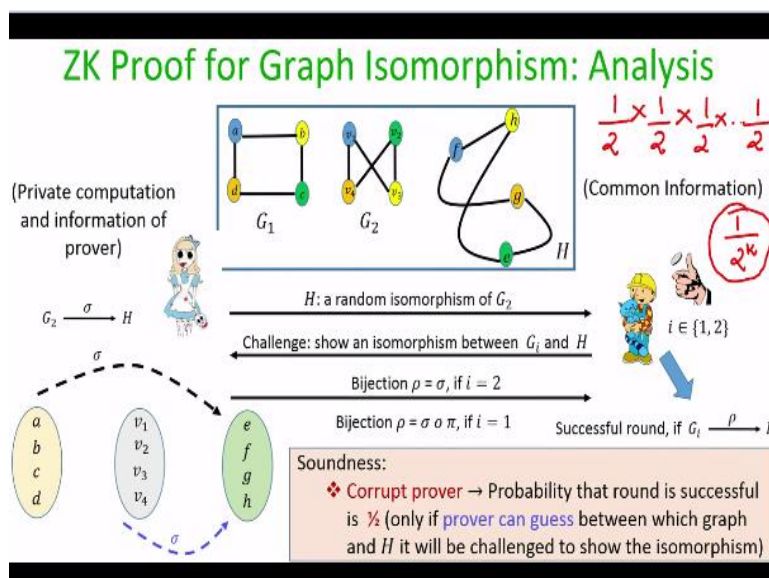
(Refer Slide Time: 18:47)



Now let us do the analysis, let us see whether this graph isomorphism protocol satisfies the requirements of correctness, soundness and zero-knowledge or completeness, soundness or zero-knowledge. So the completeness property or the correctness property here is straightforward, if indeed Alice is honest that means she knows the secret mapping  $\Pi$  between the graph  $G_1$  and  $G_2$  and if she is following the protocol instructions honestly and if verifier is also honest then each of the round will be successful. Because it does not know whether Bob challenges with  $i = 1$  or

with  $i = 2$ . Alice will always be able to successfully respond with the right mapping  $\rho$  and hence all the rounds will be successful.

(Refer Slide Time: 19:30)



Now let us analyze the soundness property here and recall for the soundness property we have to consider the case when Alice is potentially corrupt. And she does not know the mapping between  $G_1$  and  $G_2$  or at the first place there may not be any mapping between  $G_1$  and  $G_2$  showing that they are isomorphic and so since she is corrupted she might not follow the protocol, and remember as per the protocol she is supposed to send an isomorphic copy of  $G_2$  in every round but she not do that as well.

So for soundness we have to analyze that with how much probability she can successfully cheat even though she does not follow the protocol and she does not have the isomorphism between the graph  $G_1$  and  $G_2$  available with her. It turns out that the only way she can cheat in a round is to guess in advance what will be the challenge from the honest Bob. Because if she can guess correctly well in advance whether  $i = 1$  or whether  $i = 2$  then at the first place itself she can create an isomorphic copy of  $G_i$ .

So recall that as per the protocol steps she supposed to create an isomorphic copy of  $G_2$  namely  $H$  should be an isomorphic copy of  $G_2$ . But if Alice is corrupt she may not be able to follow the protocol, she may try to guess in advance that  $i$  could be 1,  $i$  could be 2 and with respect to that

guess she can create an isomorphic of that graph  $G_i$ . And if indeed her guess is correct she will be able to successfully submit the response  $p$ .

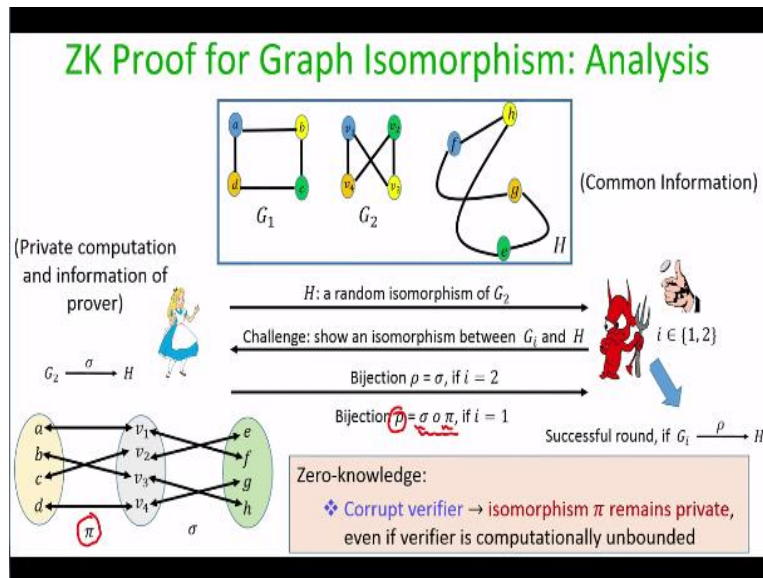
But the probability that she can correctly guess whether  $i$  is going to be 1 or whether  $i$  is going to be 2 is  $1/2$ . That means with only probability  $1/2$  she can cheat in one round and end up successfully passing that round. But since we are repeating this process  $k$  number of times, right because remember Bob is not convinced just by doing this protocol once; depending upon how much confidence he want to be in the claim of Alice, he may repeat the protocol  $k$  number of times.

So what is the probability that in each of the  $k$  iteration a bad Alice who does not know the isomorphism between  $G_1$  and  $G_2$  successfully end up passing all the  $k$  rounds. The probability that she is successful in the first round is  $1/2$ ; the probability that she successful in the second around is also  $1/2$ . And remember, the probability of success; getting successful in the second round is independent of the probability of getting successful in the first round.

Because in the second round also the challenge of Bob will be independent of the challenges that Bob has picked in the first round. So that is why the probability that Alice is successful in the second round is  $1/2$  and like that the probability that Alice is successfully able to cheat Bob in all the rounds by correctly guessing in advance the challenge of Bob in all the rounds is  $1/2 * 1/2 * 1/2 * 1/2 \dots k$  times which is nothing but  $1/2^k$ .

So if  $k$  is significantly large say imagine  $k = 100$  and if a Alice does not know the isomorphism between  $G_1$  and  $G_2$  then definitely there exists at least one of the rounds with very high probability where Alice will be caught and hence Bob will reject the statement of Alice. The only case Alice will be successful in all the rounds is when she is able, when she is lucky enough to guess the challenge of Bob in all the  $k$  rounds in advance which can happen only with probability  $1/2^k$  which is very small if  $k$  become significantly large.

**(Refer Slide Time: 23:11)**



Now let us try to understand the zero-knowledge property here where; remember for zero-knowledge we have to consider the case where Alice is honest and Bob is corrupt, the verifier is corrupt. And the goal of the corrupt verifier is to analyze the protocol transcript and learn the secret permutation  $\Pi$  which maps the graph  $G_1$  to graph  $G_2$  namely the isomorphism between graph  $G_1$  and  $G_2$ .

It turns out that in each round the corrupt verifier does not learn anything about the mapping  $\Pi$ , because if the challenge from Bob is  $i = 2$  then the response that Alice throws is the mapping from the graph  $G_2$  to  $H$  which does not reveal anything about a secret mapping  $\Pi$ . On the other hand, if the verifier challenges Alice with  $i = 1$  then in that case Alice responds with the composition of the secret mapping  $\Pi$  with another randomly chosen mapping  $\sigma$ .

And since  $\sigma$  is randomly chosen; in the iteration you can imagine that  $\sigma$  is acting as some kind of mask here, because since  $\sigma$  is randomly chosen and  $\Pi$  is any how randomly chosen and available only with Alice; this overall composed mapping  $\sigma$  does not reveal anything about the secret mapping  $\Pi$  because the masking here namely the secret mapping  $\sigma$  is randomly chosen by Alice.

And since  $\sigma$  is randomly chosen in each iteration independent of all the iterations even if a malicious prover keeps on challenging Alice with  $i = 1$  it will fail to learn about the secret mapping  $\Pi$ , and this holds even if the verifier is computationally unbounded. So in that sense a

malicious Bob does not learn anything about the secret witness  $\Pi$  available with Alice. So that shows that this zero-knowledge proof system that we have designed for the graph isomorphism problem indeed satisfies all the requirements of a zero-knowledge proof system for a graph isomorphism problem.

So that brings me to the end of this lecture. Just to summarize in this lecture we have introduced the problem of zero-knowledge proof system, we have formally stated their requirements namely the completeness, soundness and zero-knowledge requirement and we have also seen an instance of the zero-knowledge proof system for the graph isomorphism problem. Thank you.

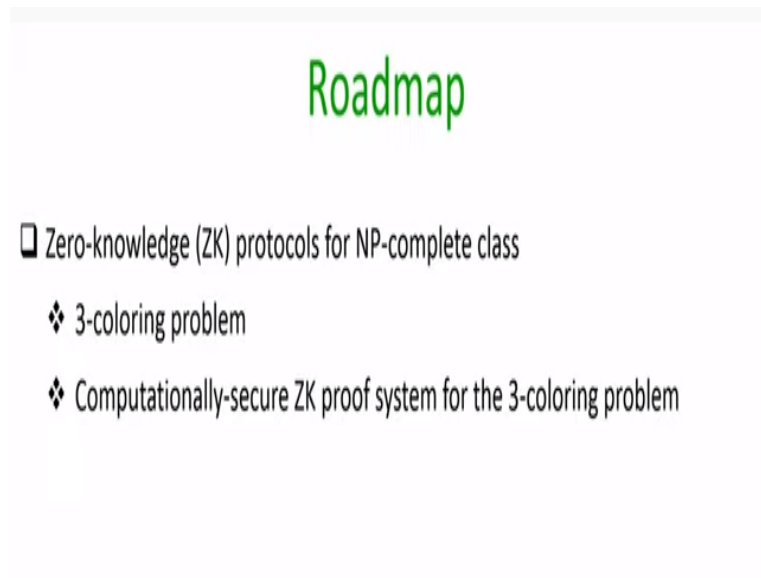


**Foundations of Cryptography**  
**Dr. Ashish Choudhury**  
**Department of Computer Science**  
**Indian Institute of Science – Bangalore**

**Lecture – 58**  
**Zero-knowledge Protocols Part II**

Hello everyone, welcome to this lecture. Just a quick recap. In the last lecture we had started our discussion on zero-knowledge protocols. So in this lecture we will continue our discussion on zero-knowledge protocols specifically we will introduce zero-knowledge protocols for NP-complete class.

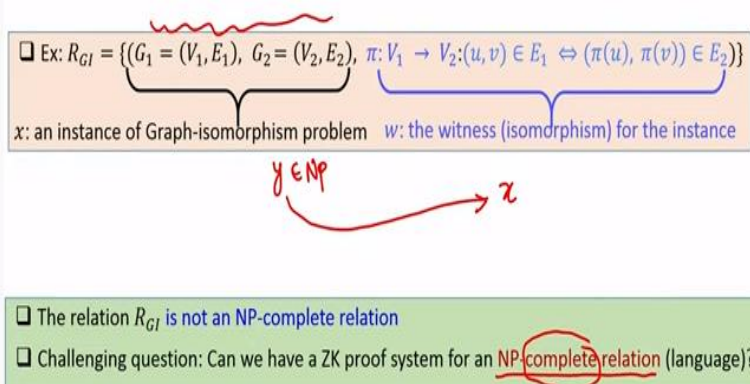
**(Refer Slide Time: 00:44)**



For that we will introduce the 3-coloring problem and we will see a computationally secure zero-knowledge proof system for 3-coloring problem.

**(Refer Slide Time: 00:51)**

## ZK Proof System for NP-Complete Class



So, recall that in the last lecture we have seen that how exactly we can specify a relationship. For instance, if you recall the relationship for graph-isomorphism problem then the relationship will consist of  $(x, w)$  pairs where  $x$  is a problem instance. So in this example if we consider a graph-isomorphism problem then the  $x$  instance is basically the publicly known description of 2 graphs and the witness component corresponding to this  $x$  instance will be the isomorphism mapping between the vertex set of the first graph to the vertex set of the second graph.

And indeed if we have an  $x$  or problem instance where the 2 graphs are isomorphic then we should have a corresponding witness  $w$ . In the last lecture, we have seen that how we can come up with a zero-knowledge proof system which allows the prover to show whether an instance  $x$  has a corresponding witness  $w$  available with the prover and not without revealing anything about the witness  $w$ .

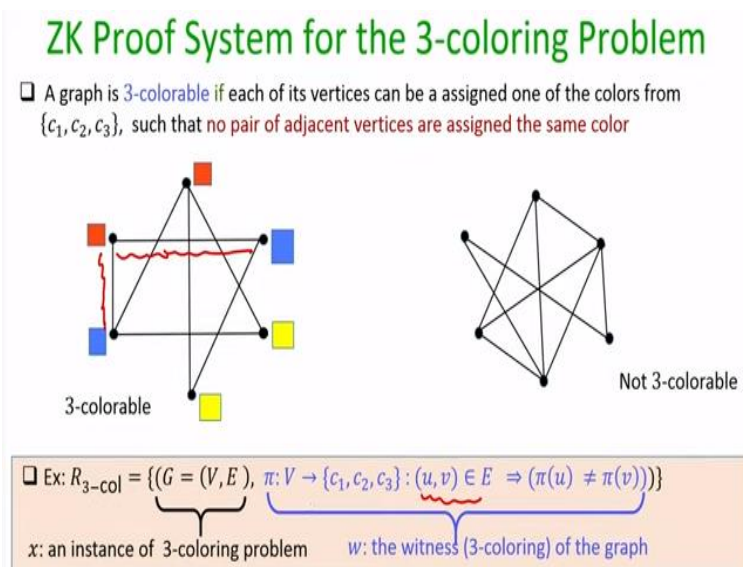
But it turns out that the relation graph-isomorphism is not an NP complete relation and next challenging question is can we have a zero-knowledge proofs system for any NP complete relation? So, for people who might be wondering what exactly is NP-complete problem or NP-complete relation is. A problem  $x$  is called an NP-complete problem, if given a witness  $w$ , we can verify whether indeed the witness  $w$  is a right witness for the problem instance  $x$  and in polynomial amount of time. Specifically by performing non-deterministic computation for polynomial amount of time. That is the first requirement. The reason we call such problems as NP-complete, the

completeness aspect here denotes that if we have any other problem  $y$  belonging to the class NP then that problem instance  $y$  can be reduced to an instance of the problem  $x$  in polynomial amount of time. In that sense this problem  $x$  will be called as an NP complete relation.

That means if you have a solution to solve problem instance  $x$ , that means if you can find out witnesses for problem instance  $x$  in polynomial amount of time, then by just using the reduction of problem instances of  $y$  to the problem instance of  $x$ , you can also get solutions for your problem instances  $y$ . So that is a rough definition of NP complete relation.

So, we are now interested to see whether we can come up with a zero-knowledge proofs system for any relation which is NP complete. So, it turns out that the graph- isomorphism is not an NP complete relation.

(Refer Slide Time: 03:35)



So, what we are going to now do is we are going to see a zero-knowledge proof system for another computational problem which we call as 3-coloring problem which is a well-known NP complete problem. So let us first see what we mean by a 3-colorable graph? We say a given graph with  $n$ -vertices is 3-colorable if each of its vertices can be assigned one of the colors from publicly known colors  $c_1, c_2, c_3$  such that no pair of adjacent vertices are assigned the same color. That means we have to color the vertices in such a way that every pair every the endpoints of every edge should have different colors.

If you consider this graph for instance, then this is a 3-colorable graph. So, for instance if my  $\{c_1, c_2, c_3\}$  are  $\{\text{red, blue, yellow}\}$  then we can color the vertices in this graph using these 3 colors by one of these assignments by assigning these colors to the respective vertices. And now you can see that no 2 adjacent vertices namely no 2 adjacent vertices are assigned the same color here.

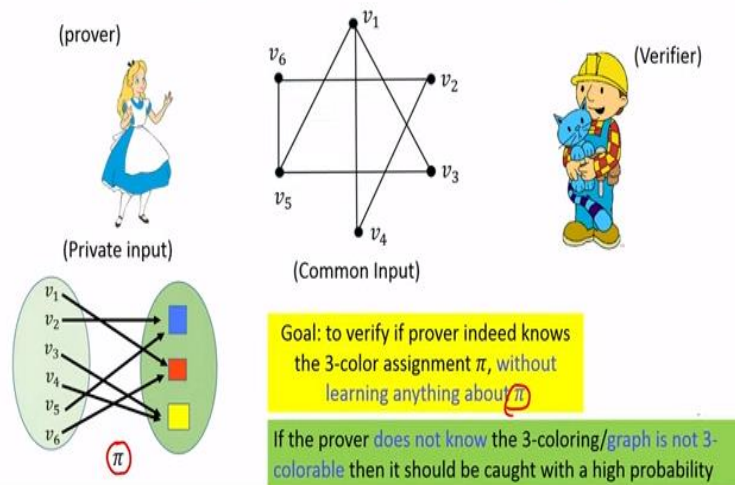
For instance, if I consider these 2 vertices, they are adjacent in the sense that they are the end points of a single edge say same edge and they are having different colors. In the same way if I consider this edge the end points are getting different colors and so on. On the other hand, if I consider the graph on your right-hand side then it is not 3-colorable. That means it is not at all possible to color all the vertices of this graph with just 3 colors satisfying the condition that no pair of adjacent vertices are assigned the same color.

The 3-coloring problem or the 3-coloring relation is a well-known NP complete relation and the  $(x, w)$  entry in the 3-coloring relation will look like this. So, the  $x$  instance will be the public description of a graph namely the number of vertices and the vertex set, and the edge set of the graph will be publicly known.

If indeed this  $x$  instance namely this graph is 3-colorable then the corresponding witness  $w$  will be the mapping of or the assignment of the color  $\{c_1, c_2, c_3\}$  to the vertex set which I call a  $\pi$ . And the coloring the witness  $\pi$  should satisfy the restriction that if the edge  $(u, v)$  belongs to the edge set then the color assigned to the node  $u$  and the color assigned to the node  $v$  should be different. If indeed it is possible to come up with such an assignment of color  $\pi$  for the given problem instance  $x$  then  $(x, w)$  will be considered as a valid entry or satisfying the relationship 3-coloring. If we cannot find a witness  $w$ , then corresponding to a given  $x$  then that  $x$  will not be present in the 3-coloring relation.

**(Refer Slide Time: 06:29)**

## ZK Proof System for the 3-coloring Problem



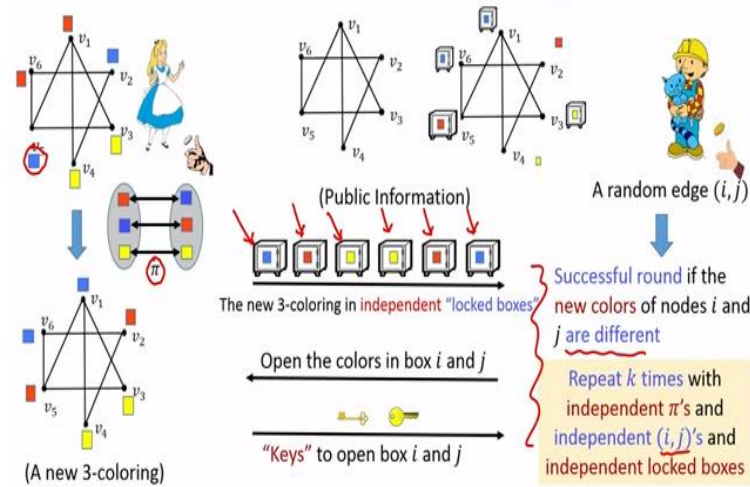
So now let us see as you know knowledge proof system for the 3-coloring problem. So, imagine a Alice is the prover and Bob is the verifier. The common input for both Alice and Bob is the description of a graph. Say, the private input for the prover namely Alice here is a 3-coloring of the publicly known graph namely an assignment  $\pi$  mapping the vertex set to the color set  $\{c_1, c_2, c_3\}$  and what Alice wants to prove to Bob that indeed the given graph is 3-colorable and she has an assignment  $\pi$  available with her.

So, the goal here is to come up with a zero-knowledge proofs system which should convince Bob that indeed Alice knows the corresponding mapping  $\pi$  without revealing anything about the actual mapping  $\pi$ . At the same time the zero-knowledge proofs system should ensure that if prover does not know the 3-coloring of the given graph or if the graph is not 3-colorable at the first place then with very high probability while giving the proof, Alice should be caught by Bob.

Recall, that this is your soundness property, the second requirement. And the first property namely Bob should not learn anything about the 3-coloring is the zero-knowledge property.

**(Refer Slide Time: 07:56)**

## ZK Proof System for the 3-coloring Problem



So now let us see how exactly the zero-knowledge proof system for the 3-coloring problem will look like. Alice has the private 3-coloring available with her. So, what she does is she randomly permutes the color that she has available with her namely see she has the secret mapping  $\pi$  sorry she has the original coloring of the graph. So, what she does is she creates a random permutation of the color set.

For instance, she can say create a permutation we are red gets mapped to blue, blue gets mapped to red and the third color remains as it is as per the permutation. Basically, now what she is doing is she is creating a new 3-coloring of the graph that is available with Bob with respect to the new mapped colors. So wherever in the original graph whichever vertices were colored with the red color those vertices in the new coloring will be assigned blue color because the red color gets mapped to blue color.

In the same way in the old coloring whichever vertices were assigned the blue color those vertices in the new 3-coloring will be assigned a red color and so on. All this Alice is doing at her end. Now once Alice computes the new 3-coloring of the graph what she does is she keeps the new assigned colors of the respective vertices in a locked box. And here locked boxes and quote unquote. We will later see what exactly are the properties we require from this locked boxes and how do we instantiate it using cryptographic primitive.

So, on a very high level what these locked boxes means is that seems in this example we have 6 vertices. So what Alice is doing is whatever is the new color assigned to the vertex one that is kept inside this locked box where the locked is available with Alice. Its locked box in the sense that without having access to the key, Bob cannot open the first box and see what is the new color of the first vertex. In the same way the new color of the second vertex is in the second lockbox the new color of the third vertex is in that third locked box and so on.

Bob right now cannot see the colors that are available in the locked boxes. So from the viewpoint of the Bob if indeed Alice knows the original 3-coloring then from the viewpoint of the Bob, Bob will feel as if he is now seeing a new 3-coloring of the same publicly known existing graph with the exception that he actually do not know what are these exact colors now because all those new colors are in the locked box.

So, this first round of message is like a commitment from Alice to Bob. That means she is saying that “okay! I have now computed a new 3-coloring. I will not show you the new 3-coloring. Those new 3-colorings are actually available in these locked boxes.” That is the commitment from my side as per the zero-knowledge proof system. Now once Bob receives the commitment of Alice, what Bob does is it creates a challenge for Alice. The challenge is basically a random edge  $(i, j)$  from the graph.

Remember Alice will not be knowing what exactly is the random edge that Bob will challenge when Alice is committing the new colors in the locked boxes. So once Bob picks the random edge, he challenges Alice that “you please open the new colors in the box  $i$  and the box  $j$ . I want to check whether indeed they are different colors or not. Because if at all you know the original 3-coloring then as per your mapping the new coloring should also be a 3-coloring.”

And hence since  $i$  and  $j$  are adjacent nodes the new color of the node  $i$  and a new color of the node  $j$  should ideally be different so that is what Bob is challenging Alice to show. And to respond to Bob's challenge what Alice reveals is it reveals is the keys for the box  $i$  and the box  $j$ . Now I need another property from the locked box here. So remember, one of the properties of the locked boxes

is that whatever is kept inside the locked box Bob cannot see its content until and unless he gets access to the keys of the box.

The second property that I require from this locked boxes is that once Alice has kept something inside the locked box and if later, she is asked to reveal the contents of any of the boxes she later cannot change the content that she has already kept inside that particular box. So in this case Bob is challenging Alice to open the box  $i$  and box  $j$  and Alice is forced to give the keys for the box  $i$  and box  $j$ . As per the properties of the lockbox whatever she has committed inside the  $i$ th box and the  $j$ th box that she is not allowed to change.

Now Bob will verify. Indeed Bob will take the keys for  $i$ th box and the  $j$ th box, will open those boxes so they are the new colors with respect to the new 3-coloring for the graph. Bob will consider it to be a successful round if it finds that the new colors of the node  $i$  and the node  $j$  are different which should ideally be the case. If this is not the case, then the round is unsuccessful and Bob stops the protocol here itself. So this is how one round of the zero-knowledge proofs system works.

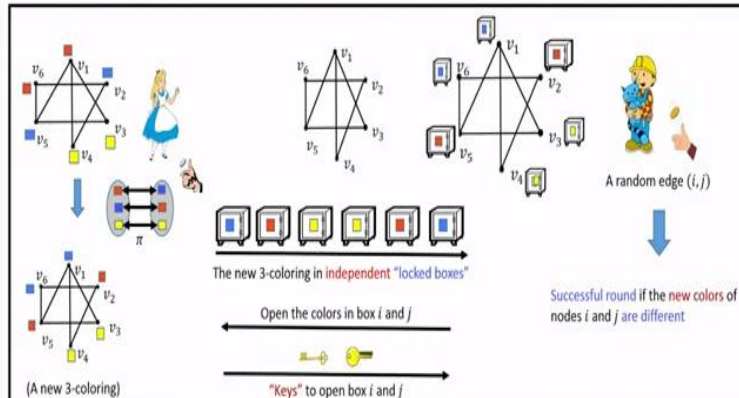
Now to boost the confidence in this proof what Bob can do is Bob can repeat this process  $k$  number of times independently where in each round Alice can use a different 3-coloring with respect to the old 3-coloring techniques. That means every round she will be picking an independent  $\pi$  mapping the existing 3-coloring to a new 3-coloring and independently Bob will be picking fresh challenges  $(i, j)$  in every round.

That means it will not be the case that  $(i, j)$  which is picked as the challenge edge by the Bob will remain the same in all the rounds. They will be picked independently, and Alice will not be knowing anything about the challenges for the subsequent round in advance and if all this  $k$  rounds are successful Bob will consider that indeed Alice knows the actual or the original 3-coloring for the existing graph.

**(Refer Slide Time: 14:18)**



## ZK Proof System for the 3-coloring Problem: Analysis



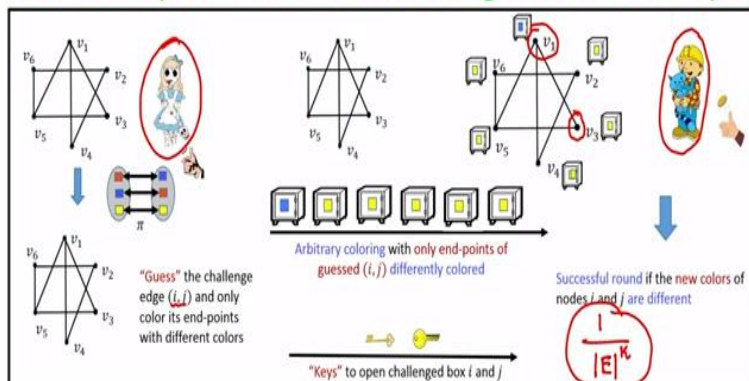
Completeness:

❖ Honest prover, verifier  $\rightarrow$  each round will be successful

So that is a zero-knowledge proof system. Now let us try to do the analysis for the zero-knowledge proofs system whether it satisfies the requirement of correctness, soundness and zero-knowledge. Correctness or completeness, so remember completeness property means that if Alice and Bob are honest and if indeed Alice has a witness for the graph namely she has the original 3-coloring then with high probability the proof should go through and Bob should be convinced. It is easy to see that if indeed Alice knows the original 3-coloring then indeed in all the rounds she will be able to successfully convince Bob. She will not fail and hence each round will be successful, and Bob will accept the proof.

(Refer Slide Time: 15:03)

## ZK Proof System for the 3-coloring Problem: Analysis



Soundness:

❖ If prover is corrupt then it has to correctly guess the challenge edge  $(i, j)$  for the round

❖ Success probability of a round being successful is at most  $\frac{1}{|E|}$

Now let us consider the soundness property. So, remember for soundness property we have to consider the case when the prover is corrupt. Prover is corrupt in the sense that either the graph is not 3-colorable, or it might be the case that the graph is 3-colorable, but Alice does not know what exactly is the 3-coloring of the original graph. For simplicity and without loss of generality assume that the graph is not 3-colorable.

So now let us analyze what is the probability that any round is successful with respect to a potentially corrupt Alice who does not know or who for a graph where the graph is not 3-colorable. It turns out that if Alice is corrupt and Bob is honest then the only way Alice can still successfully pass the round is when she can guess in advanced the edge  $(i, j)$  which is going to be picked as the challenge by Bob.

Because if the graph is not 3-colorable then Alice cannot create any new 3-coloring for the graph because the graph is not 3-colorable at the first place. Then the only way Alice can win is that she can guess. She can pretend in her mind that this might be the edge  $(i, j)$  which Bob can challenge me to show. So, what she can do is she can assign arbitrary coloring to the end points of the graph. In fact, she can assign same colors to all the nodes in the graph except the end points  $i$  and  $j$ .

Because Alice can just guess that this might be the edge which Bob can ask me to open. What Alice can do for instance she can guess that  $i$  might be a say 1 and  $j$  might be equal to anything say semantics number 3? So, for instance she can imagine that she might be challenged to show the new colors for the graph for the edge  $(v_1, v_3)$  or for the end points  $v_1$  and  $v_3$ . What she can do is for the first locked box she can keep the color blue and the third locked box she can keep the color yellow and then in all the other boxes she can just keep the same colors.

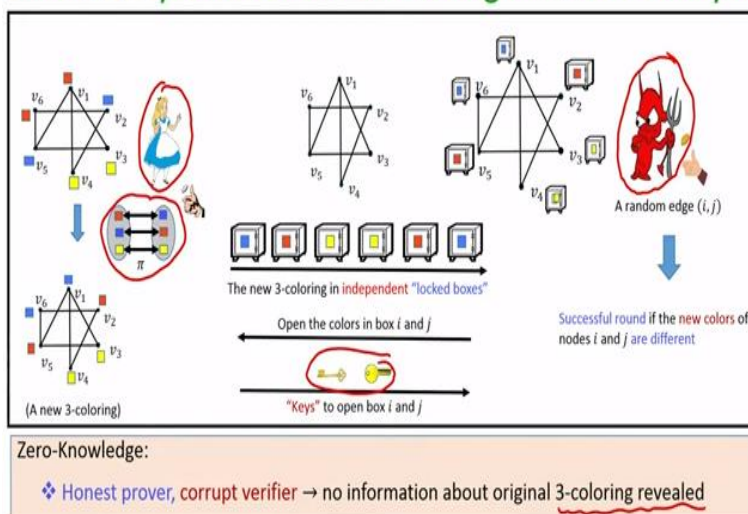
And if indeed she is lucky, then it might be the case that she is indeed asked to open the first locked box and the third locked box. Then, she will be successfully showing Bob that “Hey! I have assigned different colors to the node number  $v_1$  and node number  $v_3$ . But what is the success probability of Alice guessing in advance that what will be the random  $(i, j)$  which Bob will challenge Alice to open.

The probability that Alice is successfully able to guess the random challenge  $(i, j)$  is nothing but  $1$  over the size of the edge set. Approximately the size of edge set in the worst case can be  $n^2$ . That means success probability of the round being successful is bounded by  $1$  over the edge set namely  $1/n^2$ . Remember that there are  $k$  such rounds. That means the only way Alice without even knowing the 3-coloring of the graph can successfully pass all the  $k$  rounds is when for each of the  $k$ -rounds she can guess correctly in advanced that challenge  $(i, j)$  which Bob will be asking in each round.

And the success probability of guessing that in one round is  $1/E$ . The probability that she can do it in all the  $k$  rounds is nothing but  $1/|E|^k$ . By setting  $k$  to be sufficiently large it can be ensured that this quantity  $1/|E|^k$  becomes very small. Hence definitively in one of the  $k$  rounds Alice will get caught. If she gets caught in any of these  $k$  rounds, Bob will suspend the proof system and the claim of Alice will be rejected. So that proves the soundness property.

(Refer Slide Time: 19:07)

### ZK Proof System for the 3-coloring Problem: Analysis



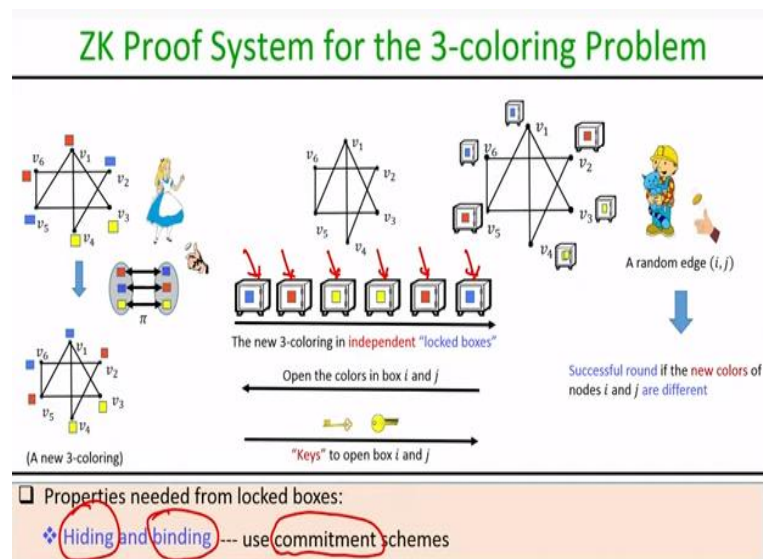
Now let us analyze a zero-knowledge property and remember for the zero-knowledge property we have to consider the case when our prover namely Alice is honest and indeed, she has a witness which she wants to hide from a malicious verifier. So in this case the verifier is the corrupt guy and the goal of the verifier is to try to learn about the original 3-coloring. It turns out that even if the verifier is corrupt, it learns absolutely no information about the original 3-coloring.

This is because what he is seeing in the first round. He is seeing the new 3-coloring but not the exact new 3-coloring, but rather the verifier is seeing the new colors assigned to the vertices in the graph all of which are kept in the locked boxes. It is only a pair of locked boxes which is opened by Alice in response to the challenge thrown by our verifier.

So even if the verifier sees the color of the node  $i$  and the node  $j$ , they are the new 3-coloring. They correspond to the new 3-coloring and it learns only the colors for the  $i$ th node and the  $j$ th node but not the entire new 3-coloring. Remember in each of the rounds, Alice is picking a fresh independently chosen  $\pi$ . That way she is randomly permuting the existing 3-coloring and creating a new 3-coloring.

So that means the new 3-coloring in the first round will be independent of the new 3-coloring in the second round and like this the new 3-coloring in the  $k$ th round will be independent of all the new 3-colorings in the previous round. That means in each of the round, verifier is just going to learn that okay I will be seeing the new colors of the node  $i$  and node  $j$  which I know are going to be distinct. And that is why it does not reveal anything about the original 3-coloring which was available with Alice. So that proves the zero-knowledge property.

**(Refer Slide Time: 21:01)**



Now coming back to the question that what are the properties we need from the locked boxes? As I said we need 2 properties. We need basically the hiding property. Namely if prover is honest, if

Alice is honest and if she has kept some contents inside the box, then until and unless the keys for those boxes are given to Bob, he cannot open and see the contents that are kept inside the box. So that is what I mean by the hiding property here.

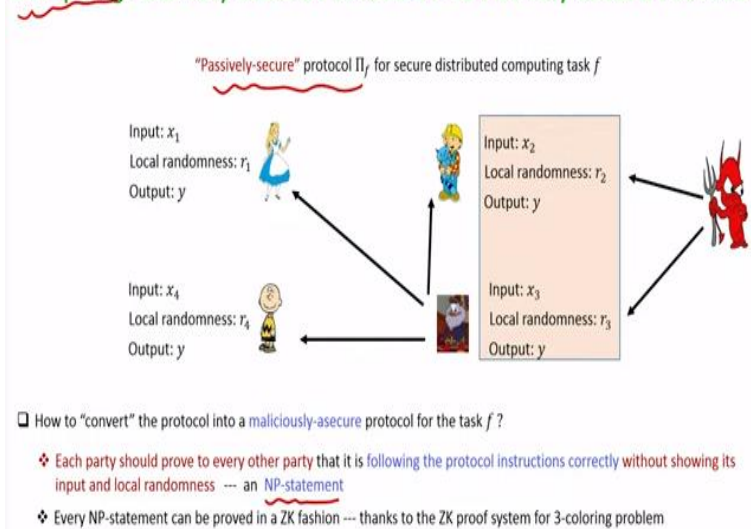
Binding property means if Alice is corrupt then it should not be the case that she put something inside the box but when she is supposed to open the box, she can turn it or she can change it into any new content. So that is a binding property and now we very well know how to instantiate this locked box both with both these 2 properties.

Basically, we can use a commitment scheme so that means what Alice has to do is in each round, she has to compute a new 3-coloring and a new color for the vertices. She has to commit by using any commitment scheme. When Bob challenges to open the new colors of  $i$ th node and the  $j$ th node, Alice has to give the opening information corresponding to the  $i$ th commitment and the  $j$ th commitment.

So, this proves that we now have a 3-coloring. We now have a zero-knowledge proof system for the 3-coloring problem and it's well known that the 3-coloring problem is an NP-complete problem that means now we have a zero-knowledge proof system for any NP relation.

(Refer Slide Time: 22:31)

### Compiling Passively-secure Protocols into Actively-secure Protocols



Now let us see the power of the zero-knowledge proof system. What we can now do is we can see a very nice framework namely a compiler which can compile any passively secure protocol into actively secure protocol. Imagine, you have a distributed computing task say  $f$ , it can be any abstract computational task. It could be say, for example a task involving multiple parties 2 parties, 3 parties or say 4 parties or any  $n$  number of parties where each party have some input say  $x_1, x_2, x_3, x_4$ .

I am taking the case where I have 4 parties. The goal of the parties is basically to compute  $f(x_1, x_2, x_3, x_4)$  and in such a way that even if there are some bad guys in the system, they do not learn anything about the  $x$  inputs of the good guys other than what they can learn from their own input and the function output. This is a very abstract problem. This problem you also call as multiparty computation problem.

The way any multi-party computation protocol will work as follows: the parties will have their own inputs and they will choose to some local randomness say  $r_1, r_2, r_3, r_4$  respectively and then they will interact with each other as per the instructions of this protocol  $\pi_f$ . At the end they will obtain the function output  $y$  where  $y = f(x_1, x_2, x_3, x_4)$  and for the moment imagine that this protocol  $\pi_f$  is passively secure.

It is passively secure in the sense that if even if there is an adversary who can control or who can see the input and the local randomness of some fraction of these  $n$  parties and whatever messages they have exchanged during the protocol, by seeing their inputs the output and the messages that they have received and they have sent during the protocol, the bad guy does not learn anything additional about the inputs of the good guys. So that is what I mean by saying that this protocol  $\pi_f$  is passively secure.

Now imagine I want to compile this protocol. Compiling this protocol in the sense I want to retain the protocol  $\pi_f$  and I want to ensure that the protocol remains secure even if there is an active adversary or a malicious adversary. Active adversary, that means I want to compile this protocol into a maliciously secure protocol sorry for the typo it should be maliciously secure protocol.

So what I mean by malicious security here is that even if the bad guys who are under the control of their adversary try to deviate from the instructions of the protocol, they should not learn anything about the inputs of the good guy. I do not want to design a new protocol or a fresh protocol. I just want to retain the protocol  $\pi_f$  which is guaranteed to be secure against the passive adversary.

So how can we compile the passively secure protocol into a maliciously secure protocol? That is a question that we want to know the answer. Now we want to use a zero-knowledge proof system here. So, it turns out that a generic way to convert the passively secure protocol into a maliciously secure protocol is as follows: if each party proves to every other party that it is indeed following the protocol instructions correctly, then in the presence of a malicious adversary, the protocol  $\pi_f$  will be achieving its task.

Now the question is what we mean by saying that a party proving to other party that indeed it is following the protocol instruction correctly. By that I mean that each party has to prove to every other party that the messages that they are sending are indeed with respect to their randomness and their input as per the instructions given by the protocol  $\pi_f$ . How can the other parties verify whether indeed each party is following its protocol instruction or not?

Well it can check the messages that particular party is sending and the witness whether that party is sending or performing its action properly or not will be the parties input and the local randomness. For instance, in this case if Alice wants to verify whether indeed this third party is following his protocol instruction correctly or not? then one way of verifying that is Alice checks the messages which this third party is sending. Along with that if this third party shows his input  $x_3$  and his randomness  $r_3$  which he has used as part of this protocol  $\pi_f$  to Alice then indeed Alice can perform the action of this third party, because the description of the protocol is known what was not known was  $x_3$  and  $r_3$ .

But now  $x_3$  and  $r_3$  is also given to Alice, so she can herself compute the messages which this third party is supposed to send as per the protocol  $\pi_f$ . If those messages match the messages that indeed this third party has sent or communicated during the real execution of the protocol, that proves that indeed this third party is following its step as per the protocol  $\pi_f$ . Like this every party can

verify every other party's action whether they are performing their actions as per the protocol  $\pi_f$ , if that sending party reveals its input and local randomness.

But it turns out that the input and the local randomness of every party cannot be given to other parties because that is what ensures the security of the protocol  $\pi_f$ . If I learn the input and the randomness of every other party, then there is no way I can guarantee the security of the protocol  $\pi_f$ . So, it turns out that the statement which every party wants to prove to every other party namely I am following the protocol instruction is nothing but an instance of an NP statement.

The problem instances is the set of messages that I have sent, and I want to prove to you that corresponding to these messages I have some randomness and some input such that these messages are indeed computed consistently as per those input and the randomness as per the protocol instruction  $\pi_f$ . So, what each party now has to basically do to convince to other party that it is indeed following the protocol instruction it has to basically prove an NP statement.

And interestingly we now have a zero-knowledge proof system for the 3-coloring problem which is an NP-complete statement. Since it is NP-complete statement that means any instance of NP problem or any NP statement can be reduced to an instance of this 3-coloring problem. That means we can now use the zero-knowledge proof system for the 3-coloring problem. Each party can transform an instance of the NP statement namely that it is following the protocol instruction correctly into an instance of the 3-coloring problem and give us zero-knowledge proof for the existence of 3-coloring and convince to the other parties that indeed it is following the protocol instructions.

If the proof gets satisfied that means that gives the guarantee that every party is following the protocol instructions correctly. If the proof does not go through, we can simply stop the protocol there itself. So, in that sense the zero-knowledge proof system, it gives you a very powerful paradigm of compiling a passively secure protocol into a maliciously secure protocol. So that brings me to the end of this lecture.



Just to summarize in this lecture we have seen zero-knowledge proof system for the 3-coloring problem. And 3-coloring problem is a well-known NP-complete problem and we have seen that how using zero-knowledge proof system we can compile any passively secure protocol for any distributed computing task into a protocol which will be secure even against a malicious adversary.

Thank you!