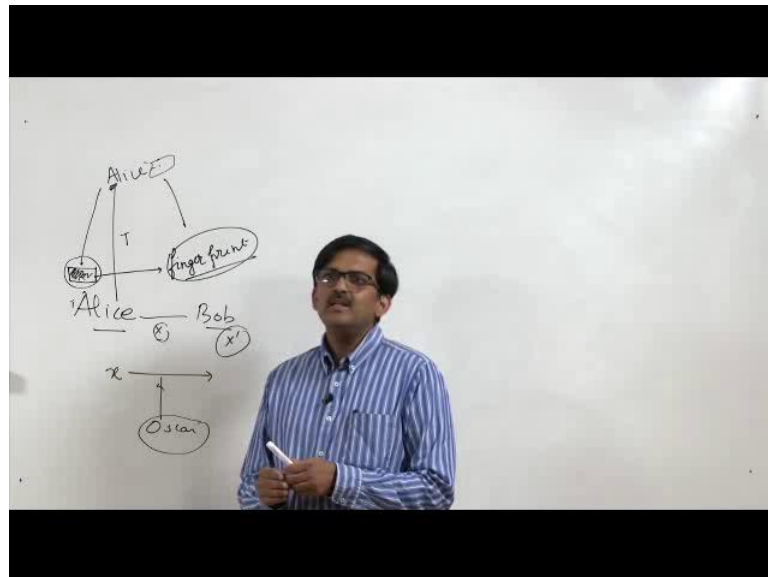


Introduction to Cryptology
Dr. Sugata Gangopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Roorkee

Lecture - 16
Cryptographic hash functions: Introduction

Hello, welcome to the 4th week lecture, the first lecture of 4th week now. So, far if we recall what we have done, we have been basically securing communication between a sender and receiver, Alice Bob.

(Refer Slide Time: 00:53)



In several ways, our name concerned was that, there is a message that Alice wants to send to Bob and in such a way that an attacker Oscar is unable to know the meaning of the message even if he has access to the message. We have seen classical cryptography, we have quickly also seen the weaknesses of classical cryptography, we have also seen block ciphers, mainly private key cryptography and then we have seen RSA, which is the public key cryptography.

So, we have another problem that is somewhat different from communication, at least communication across space that is in some way communication across time. Now,

suppose that Alice has a hard disk which she keeps and goes away and suppose that this hard disk is accessible to more than one people. Now, after sometime Alice comes back, let us suppose, after length of time Alice comes back and checks the same hard disk, what is she ask is that whether anybody has tempered with hard disk or not in her absence? Now, our question is that is it possible to secure this hard disk for Alice. So, that at least she is sure whether everything is all right.

Now, there is another question, here I am like or someone might say that of course, Alice can do this, Alice can well check it, keep a copy of the hard disk, the whole hard disk, take it with her and then when she comes back she can match bit by bit the hard disk she left with the hard disk she has been carrying, but then it beats the purpose, it does not serve the purpose because then Alice is carrying the same hard disk of same size. So, she would have easily carried the whole hard disk from here and removed, I mean whole data from here and removed whatever is over here.

So, ideally what Alice would like to have is a small finger print of this hard disk and carry it with her and when she comes back after sometime, calculate the finger print again from this hard disk that she left behind and if this these finger prints match she should be reasonably sure that no one has tempered her hard disk. So, that is something that we would like to see whether it is possibly to do. There is another thing that is important when Alice is communicating to Bob, suppose Alice sends a message x .

Now, she might be encrypting that message, but still when Bob gets the message he get something else in principle other than x . This x point that Bob gets might be x itself or the attacker Oscar might be interfering with this message and might have changed the message. So, Bob definitely would like to know whether the message that he has obtained from Alice has been tempered with or not and in some way he would like to know whether the message that is allegedly send by Alice is really send by Alice. So, we start the topic of this week lecture, which addresses these questions. So, we introduce cryptographic hash functions.

(Refer Slide Time: 06:19)

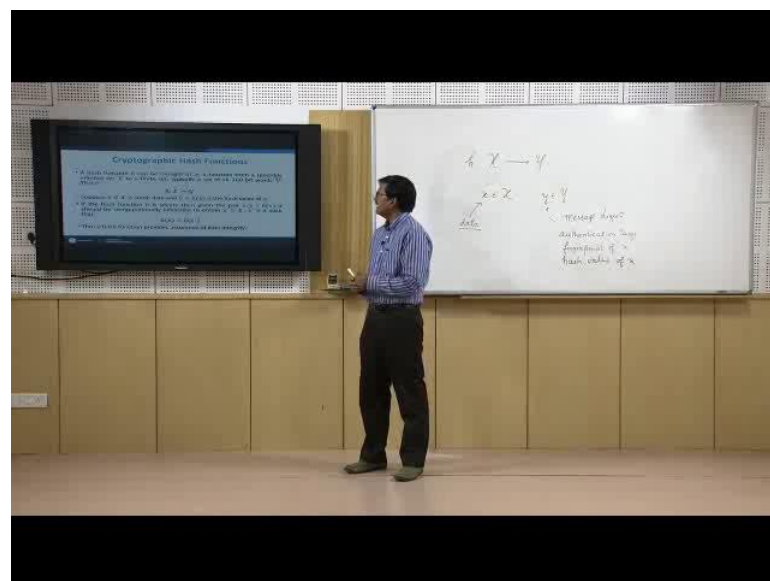
Cryptographic Hash Functions

- A hash function h can be thought of as a function from a (possibly infinite) set \mathcal{X} to a finite set, typically a set of all 160-bit words \mathcal{Y} . That is
$$h: \mathcal{X} \rightarrow \mathcal{Y}.$$
- Suppose $x \in \mathcal{X}$ is some data and $y = h(x)$ is the hash value of x .
- If the hash function h is secure then given the pair $x, y = h(x)$ it should be computationally infeasible to obtain $x' \in \mathcal{X}$, $x' \neq x$ such that
$$h(x) = h(x').$$
- Thus a hash function provides assurance of data integrity.

IIIT ROORKEE NPTEL ONLINE CERTIFICATION COURSE

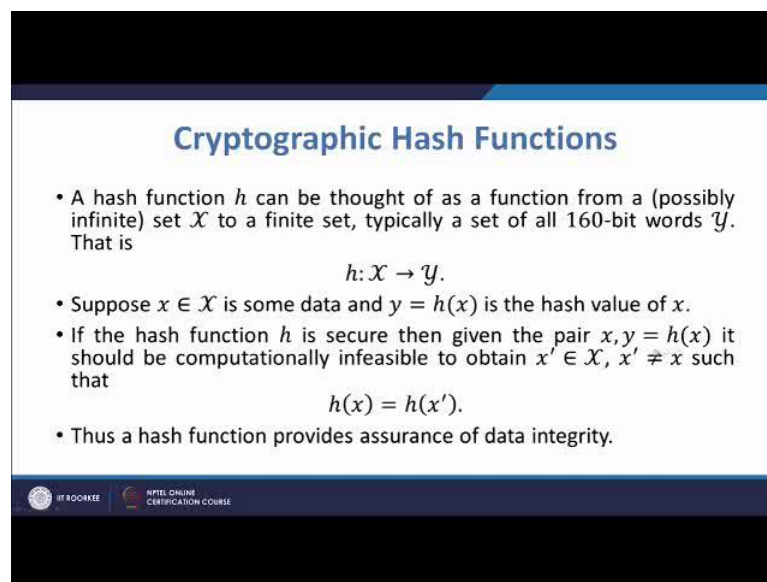
A cryptographic hash function technically is a function from its space, let us say x to another space, let us say y such that x is very large possibly infinite and y is relatively small.

(Refer Slide Time: 06:34)



We would like to see how small possible keeping the security intact is. So, we will be denoting hash functions by h . So, this is hash function x to y and we have got certain names over here. Very often instead of message, we will call elements of x as data because in reality very often there will be data like, as I have been saying that it might be a whole hard disk of Alice which she is leaving behind. So, that is data and the elements of y will be called message digest or authentication tags. Sometimes we may be referring to y as finger print of x or hash value of x .

(Refer Slide Time: 08:53)



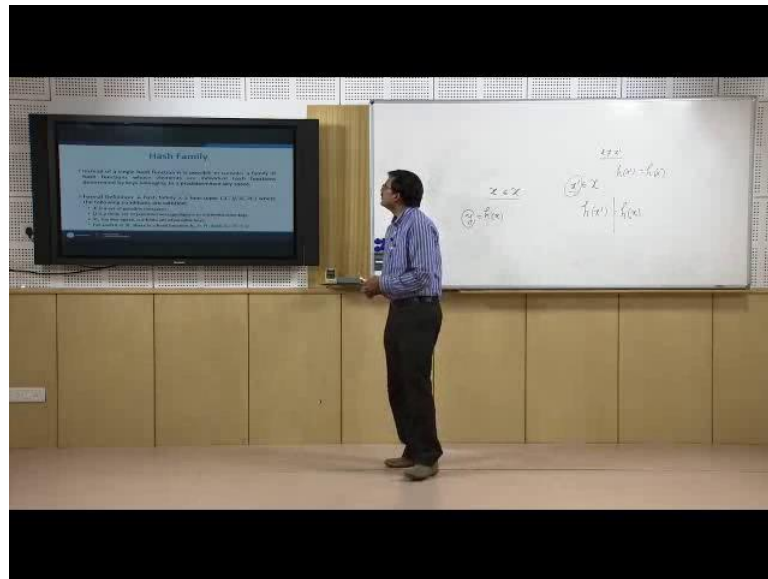
Cryptographic Hash Functions

- A hash function h can be thought of as a function from a (possibly infinite) set \mathcal{X} to a finite set, typically a set of all 160-bit words \mathcal{Y} . That is
$$h: \mathcal{X} \rightarrow \mathcal{Y}.$$
- Suppose $x \in \mathcal{X}$ is some data and $y = h(x)$ is the hash value of x .
- If the hash function h is secure then given the pair $x, y = h(x)$ it should be computationally infeasible to obtain $x' \in \mathcal{X}$, $x' \neq x$ such that
$$h(x) = h(x').$$
- Thus a hash function provides assurance of data integrity.

IT BOORKE NPTEL ONLINE CERTIFICATION COURSE

Now, some basic level of security; what we want in a hash function? So, let us look at this, if the hash function h is secure when given the pair x and y equal to h of x , it should be computationally infeasible to obtain another data x prime belonging to x , such that x prime is not equal to x and $h(x)$ is equal to $h(x)$ prime. So, this is how we will get some kind of data integrative. So, what will happen for Alice, if Alice has access to a hash function?

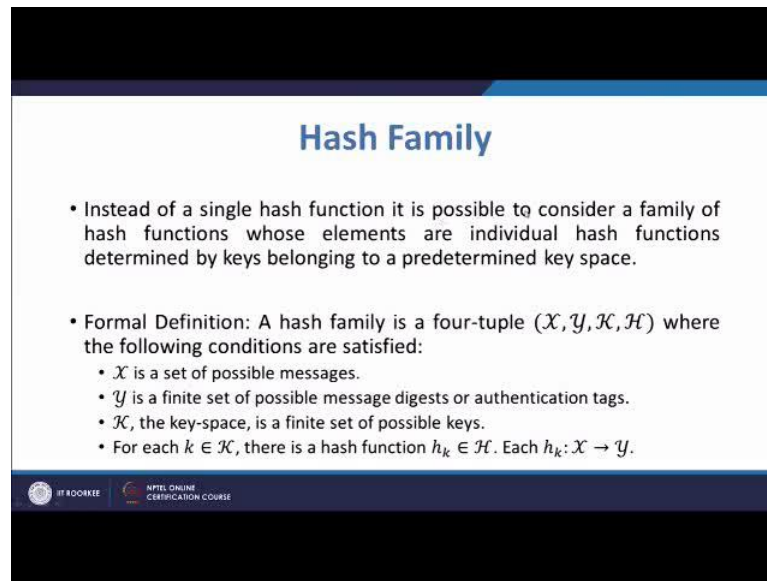
(Refer Slide Time: 10:00)



The data in her hard disk x , which she wants to protect she will apply the hash function $h(x)$ and get y . This y is small, much smaller than x and she will carry the y and when she comes back, she will look at the look at a hard disk that she has left with the data. So, data is x prime and then she will compute $h(x)$ prime and she will match $h(x)$ and $h(x)$ prime. If she finds that both are same then with high reliability she will infer that no one has tampered with her data, but if they are not same then she will definitely be able to say that someone has tampered the data and so therefore, it must be very, very difficult to find out an x prime such that x is not equal to x prime, but $h(x)$ prime is equal to x .

Now, we can move on to something more general from hash functions to hash families. What we can do is that we can create a class of functions which are dependent on a key. So, that whole class will be called a hash family and the key is like our usual cryptographic keys, which will be exchanged between Alice and Bob and the particular hash function from the family will be obtained by them.

(Refer Slide Time: 12:10)



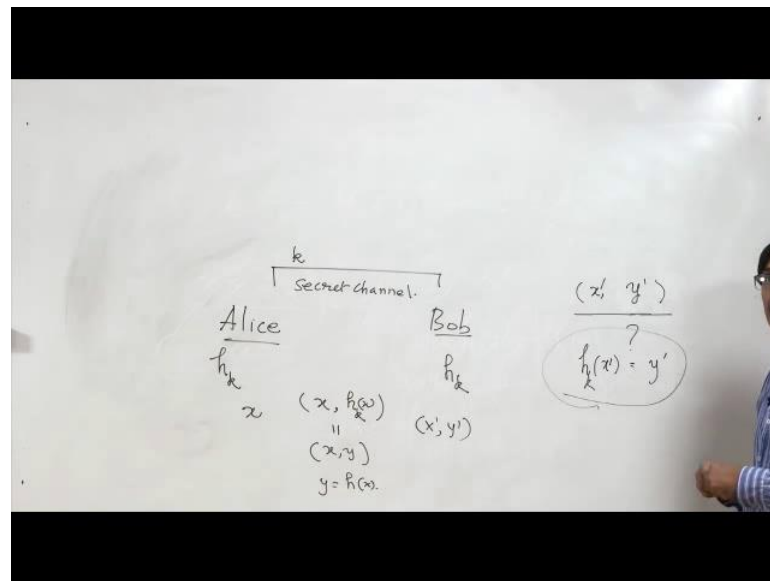
Hash Family

- Instead of a single hash function it is possible to consider a family of hash functions whose elements are individual hash functions determined by keys belonging to a predetermined key space.
- Formal Definition: A hash family is a four-tuple $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$ where the following conditions are satisfied:
 - \mathcal{X} is a set of possible messages.
 - \mathcal{Y} is a finite set of possible message digests or authentication tags.
 - \mathcal{K} , the key-space, is a finite set of possible keys.
 - For each $k \in \mathcal{K}$, there is a hash function $h_k \in \mathcal{H}$. Each $h_k: \mathcal{X} \rightarrow \mathcal{Y}$.

IT ROOFTEE NPTEL ONLINE CERTIFICATION COURSE

So, let us see instead of a single hash function it is possible to consider a family of hash functions, whose elements individual hash functions determined by keys are belonging to a predetermined key space. Now, formally we will define a hash family as a four-tuple. So, we will have a space of data that is x , then we will have a space of message digest or authentication tags which is y and we have a space of key and the family of functions. So, given key k belonging to capital k , Alice and Bob will be able to obtain a function in h , which they will denote as h_k and which will work for their requirements. Now, it might be like this.

(Refer Slide Time: 13:21)



Suppose, Alice and Bob are communicating. Now, what Bob and Alice want to ensure is that, when they are sending the message nobody is tampering with it. So, what Alice does is that Alice takes the data first, what they do is that they agree upon a key through a secret channel. Now, this has to be done by a through a secret channel and since that agreed upon a key both of them will have the hash function h_k and now suppose Alice wants to send x to Bob, she sends a pair she sends x and $h_k(x)$ this pair which we will denote by x, y where $y = h_k(x)$.

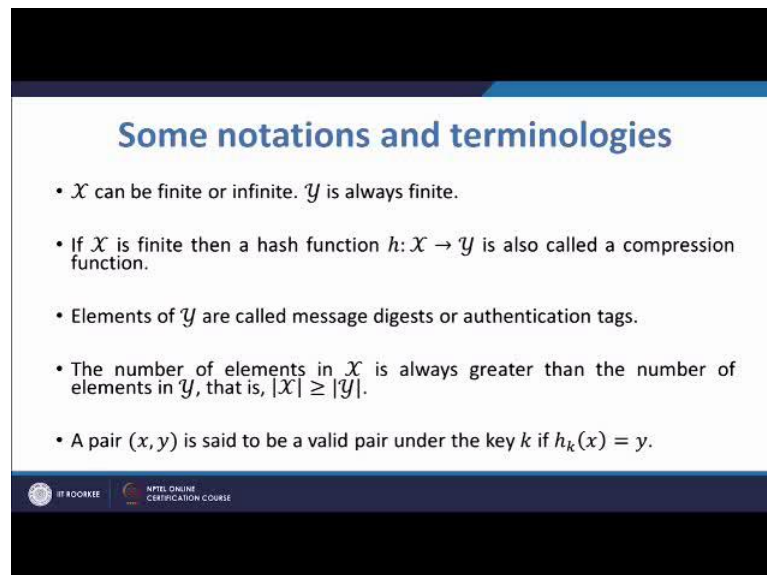
Now, Bob obtains this pair and now in reality Bob is not sure whether he has obtained this pair or a modification of this. So, that is why we are writing that Bob has obtained this and this is not $h_k(x)$ this is y , Bob has obtained this. So, what Bob does is that Bob computes $h_k(x)$ and checks whether it is equal to $h_k(x)$, whether it checks that whether it is equal to this? What this hash function must guarantee me that if I do not know the key; it will be computationally infeasible to find out a pair which is valid for this particular function.

So, an attacker cannot produce a pair like this; x' and y' and send to Bob take it out. So, that Bob when checks with this equation will find that this equality holds. So, if such is the case then Alice and Bob have obtained some authentication. So, if whatever

Bob gets, Bob applies hash function that particular key hash function to the message and checks the message digest if they agree then Bob should be quiet certain that the message is send by Alice and no one else and Alice is message has not been tempered with. So, that is what we would like to achieve.

What we see is that unkeyed hash functions are also in a sense fall within this same frame work. We can always say that in unkeyed hash function comes from a hash family where the key space is singled out. Now, let us come to some notations and terminologies.

(Refer Slide Time: 17:43)



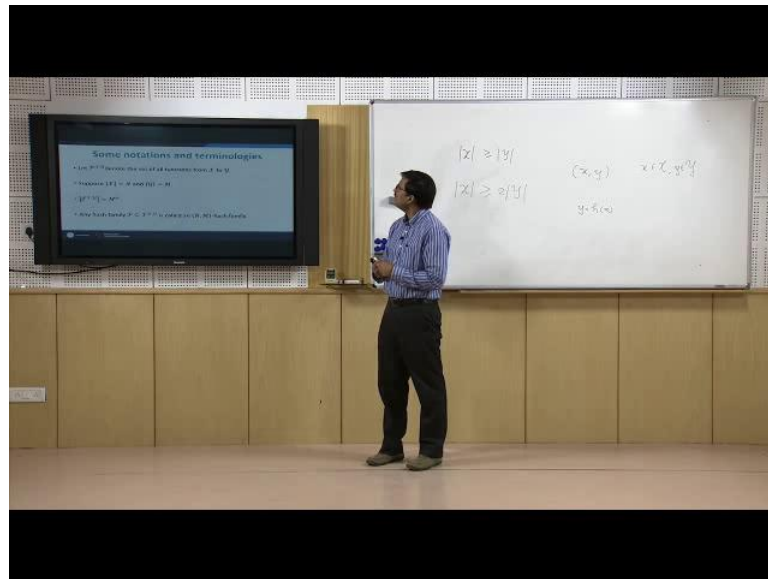
Some notations and terminologies

- \mathcal{X} can be finite or infinite. \mathcal{Y} is always finite.
- If \mathcal{X} is finite then a hash function $h: \mathcal{X} \rightarrow \mathcal{Y}$ is also called a compression function.
- Elements of \mathcal{Y} are called message digests or authentication tags.
- The number of elements in \mathcal{X} is always greater than the number of elements in \mathcal{Y} , that is, $|\mathcal{X}| \geq |\mathcal{Y}|$.
- A pair (x, y) is said to be a valid pair under the key k if $h_k(x) = y$.

IT 4001XEE NPTEL ONLINE CERTIFICATION COURSE

So, as we have already seen whether set of data x can be finite or infinite, advice always finite and relatively small then if x is finite then the function h from x to y is also called a compression function. We will be using this terminology very often later and we have already said that why is a is the elements of y are called message digest or authentication tags and the number of even if x is finite and the number of elements in x is much larger the number of elements in y .

(Refer Slide Time: 18:35).



So, of course, we will have this relationship always. In fact, we will be having something else, something more stringent as this, that is number of elements in x will be twice more than twice the number of elements of y and there is a concept of valid pair. If I give you a pair x, y where x is an element of capital x and y is an element of capital y when we will say that x, y is a valid pair if and only if y is equal to $h(x)$. So, this is also a terminology that is use very often.

(Refer Slide Time: 19:42)

Some notations and terminologies

- Let $\mathcal{F}^{X,Y}$ denote the set of all functions from X to Y .
- Suppose $|X| = N$ and $|Y| = M$.
- $|\mathcal{F}^{X,Y}| = M^N$.
- Any hash family $\mathcal{F} \subseteq \mathcal{F}^{X,Y}$ is called an (N, M) -hash family.

IT ROORKEE NPTEL ONLINE CERTIFICATION COURSE

Now, from more terminologies by f superscript x, y ; we will denote the set of all functions from x to y and very often the number of elements in x . If x is finite will be denoted by capital n and number of elements in y will be denoted by capital m and in that case the number of functions in f superscript x comma y will be m raise to the power n . Let us quickly have a look at this counting.

(Refer Slide Time: 20:23)

The lecturer is standing in a classroom. To his left is a screen displaying a slide titled "Security of Hash Functions". To his right is a whiteboard with the following content:

$f: X \rightarrow Y$

$X = \{x_1, x_2, \dots, x_n\}$

$Y = \{y_1, y_2, \dots, y_m\}$

$|X| = N, |Y| = M$

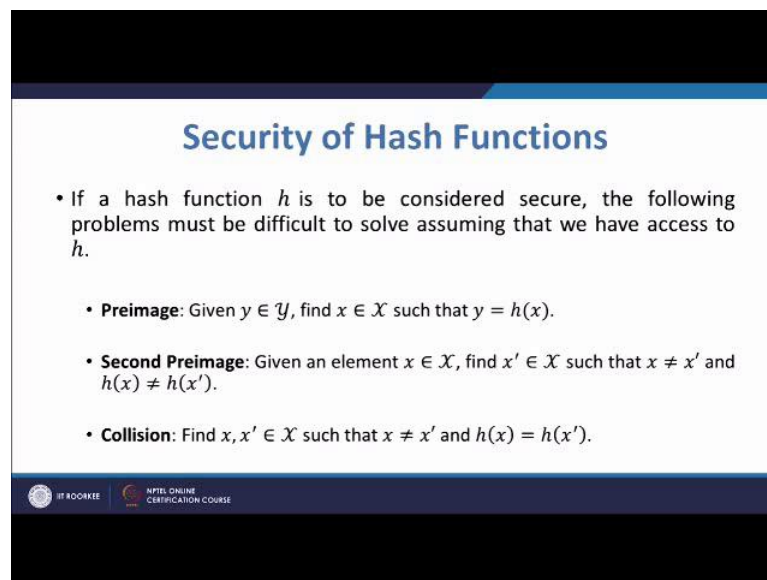
N^M

Diagram showing a set X with elements x_1, x_2, \dots, x_n mapping to a set Y with elements y_1, y_2, \dots, y_m . The mapping is labeled $f: X \rightarrow Y$.

We are considering all the functions from X to Y and we are denoting that by f superscript X comma Y and we have been told that the number of elements in capital X is N and the number of elements in capital Y is M . So, if we list all the elements in script capital x that is a domain. So, we can list like this we come up to x sub capital n and what we see is that we can choose the image of x size that is x_1, x_2 and so on in N ways, each of the images can be chosen in M ways and therefore, the total number of choice is M into M into M capital N times this gives me M raise to the power N . So, we have a huge space of m raise to the power N and within that space our hash families lie. So, any hash family you just script f is a subset of $f(X, Y)$ and these are called NM hash families.

The last topic that we discuss in this lecture is security of hash functions. So, what has been found is that hash functions must have some properties to be secure and these properties are formulated as problems.

(Refer Slide Time: 23:03)



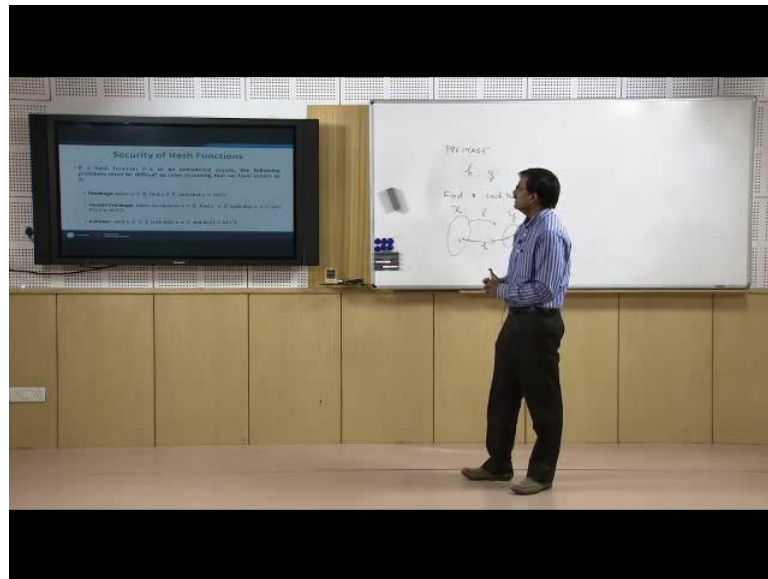
Security of Hash Functions

- If a hash function h is to be considered secure, the following problems must be difficult to solve assuming that we have access to h .
 - **Preimage:** Given $y \in \mathcal{Y}$, find $x \in \mathcal{X}$ such that $y = h(x)$.
 - **Second Preimage:** Given an element $x \in \mathcal{X}$, find $x' \in \mathcal{X}$ such that $x \neq x'$ and $h(x) = h(x')$.
 - **Collision:** Find $x, x' \in \mathcal{X}$ such that $x \neq x'$ and $h(x) = h(x')$.

IT ROOKIEE NPTEL ONLINE CERTIFICATION COURSE

So, these problems are preimage, second preimage and collision. So, let us think in this way, we have a problem called preimage.

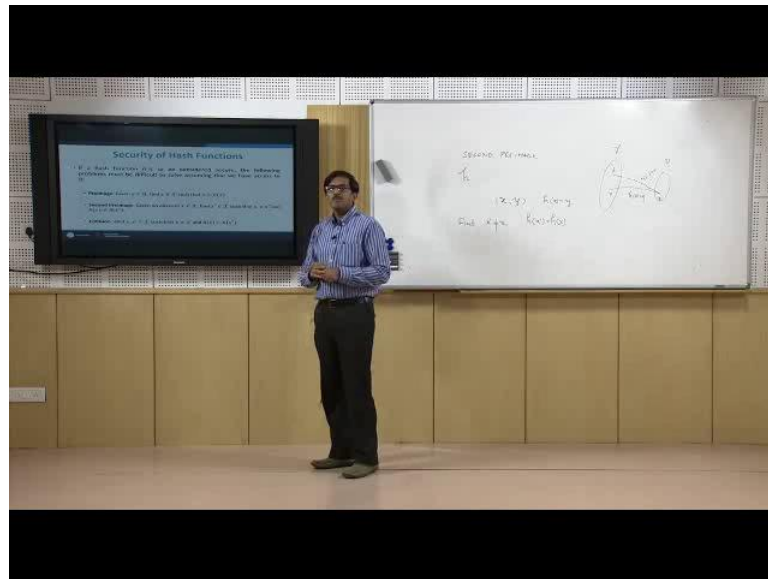
(Refer Slide Time: 23:26)



What is that problem? The problem states that I am given something, I have got the access to the hash function h and I have been given a y . So, these are the things that are given to me and I have been told that find x such that $h(x)$ equal to y . So, this means that in a way I have a huge space x , I have relatively smaller set y and well I am given the function and I am given a y and I have been asked to find out x such that x is taken to y . This is preimage. Now, if a hash function has to be secure then this preimage problem must be very difficult to solve that is to say it should not be possible to solve it in feasible time.

Now, second preimage is a problem which looks like preimage, but it is slightly different from that.

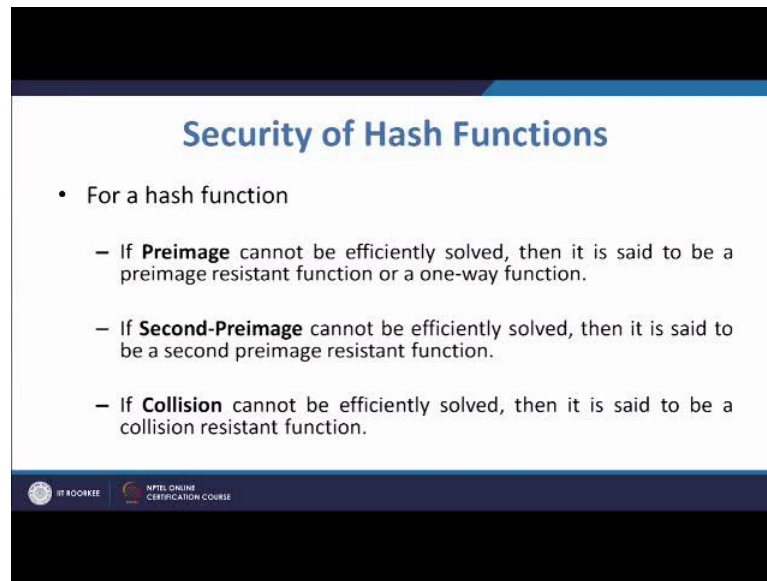
(Refer Slide Time: 25:29)



Here again we have a rare access to h , the hash function and here I have been given the pair which is so to say valid pair. So, I have been given a pair such that $h(x)$ equal to y and I am asked to find out another x that is a second preimage, we denote it by x prime. Find x prime which is not equal to x , such that $h(x)$ prime is equal to $h(x)$. So, if you draw a diagram, it will be like this that again I have this space x and I have this space y . I have been given a y and a preimage of that. So, I know that $h(x)$ is equal to y . I am asked to find something else, some other element in x such that this also maps to y $h(x)$ dash should be equal to y , so that is the second preimage.

Now, lastly the third problem which should be difficult to solve or so to say practically impossible to solve that is called collision.

(Refer Slide Time: 27:27)



The slide is titled "Security of Hash Functions" in a blue font. Below the title is a bulleted list with three items, each starting with a hyphen. The first item discusses "Preimage", the second "Second-Preimage", and the third "Collision". At the bottom of the slide, there are two logos: one for IIT ROORKEE and another for NPTEL ONLINE CERTIFICATION COURSE.

- For a hash function
 - If **Preimage** cannot be efficiently solved, then it is said to be a preimage resistant function or a one-way function.
 - If **Second-Preimage** cannot be efficiently solved, then it is said to be a second preimage resistant function.
 - If **Collision** cannot be efficiently solved, then it is said to be a collision resistant function.

It says that I have something or access to something much less, I have just access to h . I just know the hash function and I should not be able to compute two distinct data sets, x and y , sorry, x and x prime. I should not be able to find x and x prime which are distinct whose hash values are same. If I am able to do that; that means, that I have got a collision, but I should not be able to do this. So, that is collision.

Now, at the end if for a hash function, it is not possible to find or if it is not possible to solve preimage problem, it is called preimage resistant hash function. If for a hash function it is not possible to find second preimage then it is called a second preimage resistant hash function, and lastly if for a hash function it is not possible to find a collision, it is called a collision resistant hash function. This ends today's lecture here, we will consider these properties in details in the forthcoming lectures.

Thank you.