**Internetwork Security**
**Prof. Sourav Mukhopadhyay**
**Department of Mathematics**
**Indian Institute of Technology, Kharagpur**

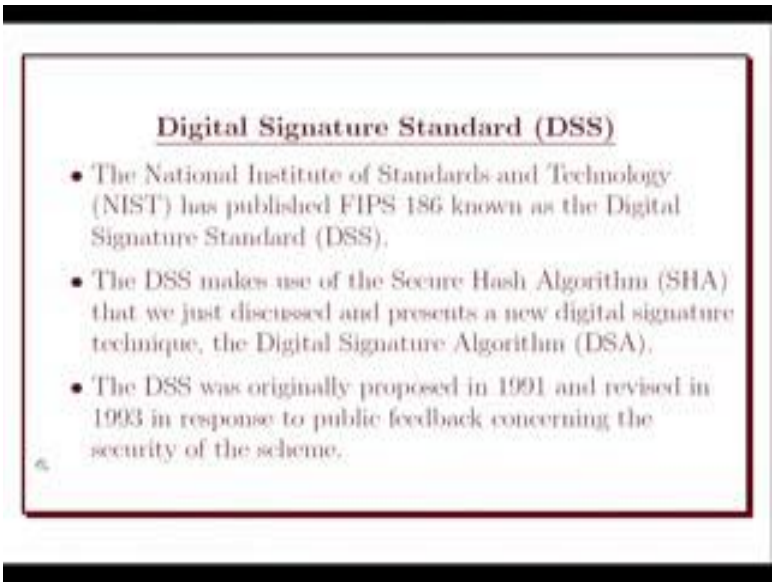
**Lecture - 44**
**Digital Signature Standard (DSS)**

We will talk about Digital Signature Standard which is basically based on we use the SHA 1, I mean SHA, Secured Hash Algorithm to reduce the plaintext in a 160 bit.

(Refer Slide Time: 00:38)


As you know hash function is can take any size plaintext M. So, it can take any size plaintext M to a fix size digest. In general, it is l bit, now for SHA 1, if you use SHA 1 SHA, so this is basically give us 160 bit; 160 bit. So, this is H of M, this we denote by small m. So, it is basically used in digital, so DSS basically used what is called SHA and DSA; digital signature algorithm. So, we will define what is digital signature algorithm.

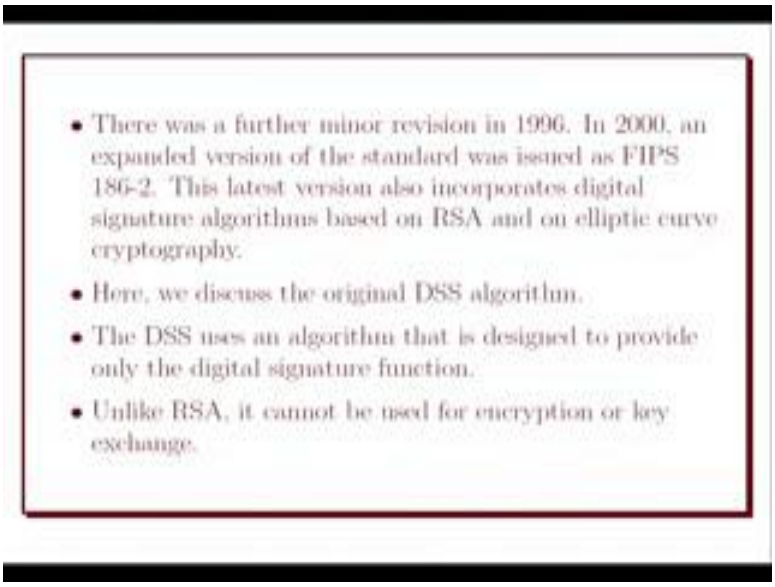
(Refer Slide Time: 01:30)




This is basically NIST publish this in this DSS in 1991. So, this is the first version was proposed and it was revised again in 1993.

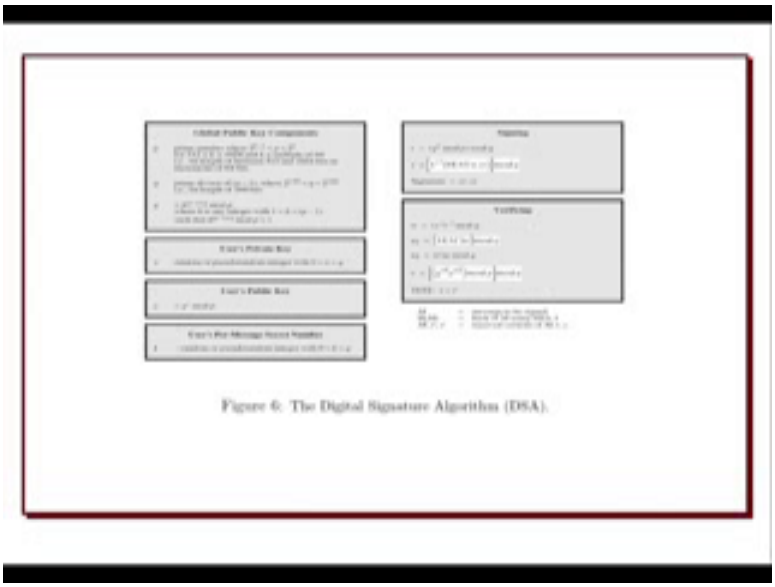
This is the standard made in 1993 for a digital signature and this is used as I said secured hash function; hash algorithm for the digest purpose to for the compression because our message could be long enough. So, we have to make it small and then it is used the digital signature algorithm which we will discuss.

(Refer Slide Time: 02:15)



And there are some modifications on this extended version of this standard like this so, but here we discuss the DSA algorithm which is based on the DSA; DSS which is based on the DSA algorithm.

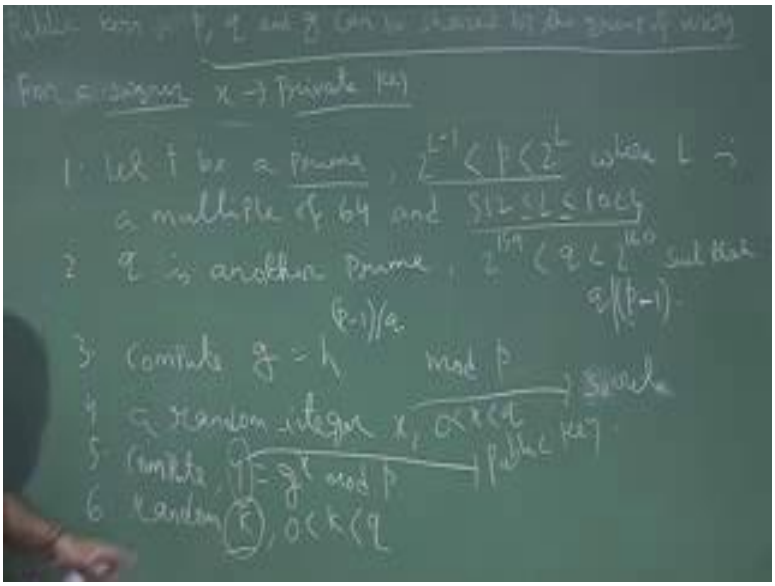
(Refer Slide Time: 02:38)



We will just look at this, this has a few step like key generation step which is called the DSA parameter. So, let us just talk about how we can choose the parameter for DSS.

We are talking about digital signature algorithm, this is basically make use of the Elgamal signature scheme alright, but the message we digest because message is message we are taking to be any message, long messaging and, but it has to be digest to the fixed length 160 bit. So, we had we have to use the SHA 1.

(Refer Slide Time: 03:26)



This is the digital signature algorithm, this is also DSA. So, DSA use the following parameter, DSA uses this is the key generation step, this is this parameter or standard parameter that is why to 4 standard you have to take this.

This parameter or basically we choose a P, let P be a prime, but P is coming from such that 2 to the power L minus 1 less than P less than 2 to the power L where L is basically the L is the number of bit. So, L is basically lies between L where L is a multiple of 64 is the multiple of 64 and L is coming from 512 less than equal to a less than equal to so, P is a large prime P is a L bit prime and L is a multiple of 64 and L is tasking from this. Now we choose another prime q; q is another prime and per key also you have the range coming from 2 to the power 159 less than q less than 2 to the power 160. So, q is 2 to the power 159 bit and such that q should divide P minus 1 such that q should divides P minus 1 q must divides P minus 1. So, we choose another prime such that this q should divides P minus 1.

Now, we choose a g, we compute g; we compute g which is basically, so we compute g which is basically coming from h to the power P minus 1 by q because q divides keep q divides P minus 1. So, that is an integer. So, and where is it is an integer in this range. So, we choose h to be less than P minus 1 to P minus 1 and so, let us write this g is we calculating h to the power P minus 1 by q that is why we need q must divides P minus 1, so this mod P.

Now, we choose a random integer a random integer. So, this is our key generation is all other parameter standard parameter for this standard random integer x such that 0 less than x less than q, x will be the private key for the individual and we compute y, y will be the public key for each individual, we compute y is equal to g to the power x mod P. So, g we know, so you can compute g to the power x mod P and this y will be the public parameter of the individual and x will be the private parameter of the individual. So, now, 6, I will just erase this, so this is 5 and then 6 basically 6 is we randomly choose a key form and random integer k from 0 to q and this k is required for each signature because k is if you recall the Elgamal cryptosystem the for encryption we need to choose a k by the signer or encrypter.

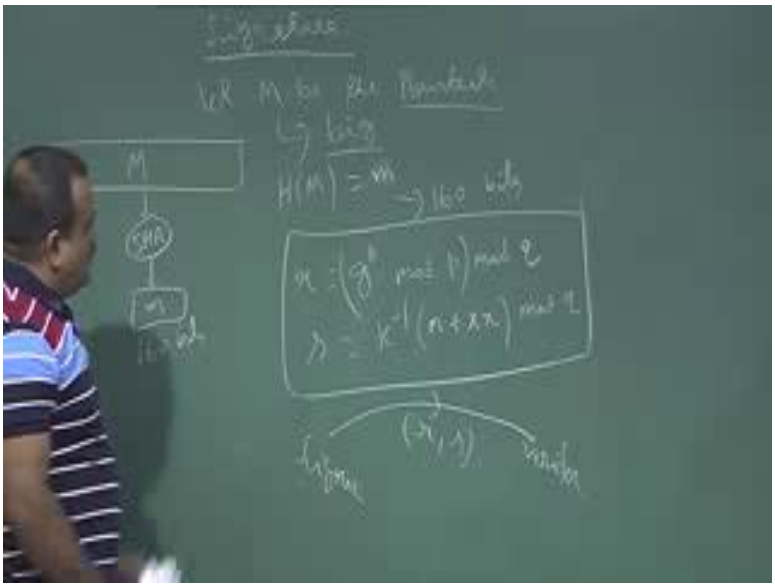
Now, so, this is the parameter specification. So, we have to choose a prime P which is basically L bit and L has to be range from 512 to this any value and, but L must be multiple of 64. So, we have to choose a prime P and you have to choose another prime q which is basically 159 bits, but it must q must divides P minus 1 because we have to calculate this. So, we choose; we compute g for h less than q minus 1 mod P then we choose a random integer x; x is the basically x is the private key of an individual and y is the public key of an individual and the remaining we can make it public. So, public key is basically are public key is basically P q and g. So, these are the key is can be shared by the group of user can be shared by the group of users group of user means because this is a signature standard. So, there are many parties are involved, but one party is in one individually signing, but this checking verification can be done by any party. So, this is there is a group of users Alice, Bob, Rob, Palash, Bimol, so among these so, these are all parameters are publicly known to the all the users. So, these are the part of the public parameter, but for individual; for an individual user we signing an individual for an individual user for signer; for a signer.

We signing basically signing user x is the private key or the secret key this x. So, this x is the private key of or the secret key of the signer and this y is; this y is the public key. So, this k is chosen by the signer at the time of signature because k is another parameter which is sort of norms or time stamp for that particular signature this is basically Elgamal signature. So, this is the parameter set.

Now, let us talk about the signature now using this parameter suppose Alice wants to send a message. So, these are this is the standard parameter will use for this digital

signature standard. So, now, how to sign a message now, using this parameter setup? So, the message could be any length so, but we have to make it come we have to compress it. So, this is the signature part.

(Refer Slide Time: 12:48)




This is doing by the signer and the verification can be done by any other users which belongs to that group let M be the message be the message or plaintext which and M is any arbitrary bit, it is a big message.

Now we will use make use of hash function especially for this standard, we have to use the secure hash algorithm for SHA to so if we use SHA on this, we will get M. So, this is basically 360 bit. So, this is small m. So, this capital M, we use the SHA; SHA has function and we make it 160 bit digest. This, the signer has to apply h on this and get this then signer will compute r and s that is the signature part this is basically the Elgamal signature. So, how we can compute r? Suppose Alice is signing and send it to public channel Bob rob. So, this is the group of users Bimol, Palash. So, Alice is computing what g to the power k because k as is choosing g is public g to the power k mod P mod q and s is basically k inverse m plus x r mod q and Alice is sending this r H 2 to the verifier. So, this is signer and this verifier. So, the signer is sending this r s to the verifier. So, this is the signature part, but here we are using the small m because this is after applying the SHA.

(Refer Slide Time: 15:42)

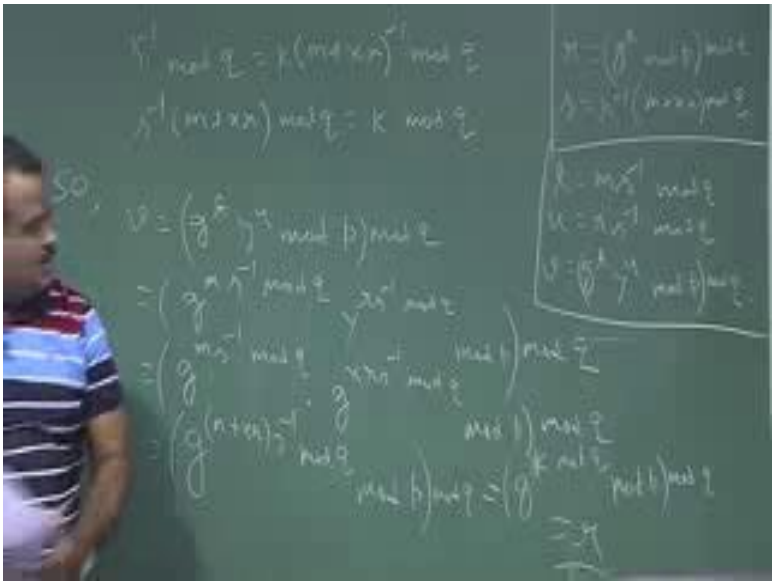
Now, how to verify it? How the verifier can verify it? So, that part we have to talk about. So, let us just write this here, r is basically g to the power k mod P mod q, s is basically k inverse M plus x r mod q, let us have this, in the verification step we need this. So, now, how to verify it? So, verifier is disturbing r and s and verifier is having all the public parameter. So, using that verifier has to verify this. So, this is the verification stage. So, verifier knows this knowing P q g these are all public parameter and the Alice or the signer public key y and verifier also knows M, M is the message. So, message is also has to send. So, this Alice is sending, Alice is the signer, M along with the r and s and Bob is already having this public parameter like P q g y yeah. So, r as in s is name Bob is getting now.

Now Bob has to verify, this is Bob sorry, Bob has to verify this message is signed by the Alice. So, the verification is done by this. So, r and s, so Bob will compute this t which is basically m s inverse mod q. So, Bob can compute because M is basically h of capital M. So, hash function is public. So, Bob can easily get this and s is coming from Alice signature. So, Bob can get the s inverse by the extended Euclidean algorithm and can get this t, Bob can calculate u also r s inverse mod q this is also can be computed and v is basically g to the power t this t is here y to the power u mod P mod q and the verification is like this and verify whether this v is r or not if v is r then the signature is valid if v is r then the signature is valid we have to check this then the signature is valid. So, this we have to check whether v by computing this is giving us r or not. So, r is that. So, let us

write v over here. So, v is basically how can let us just write here t is equal to m s inverse mod q and u is equal to r s inverse mod q and v is basically g to the power t y to the power u mod P mod q.

Now we are checking whether v is equal to r naught. So, y why this will give v is equal to r let us just check this. So, s is this.

(Refer Slide Time: 19:56)

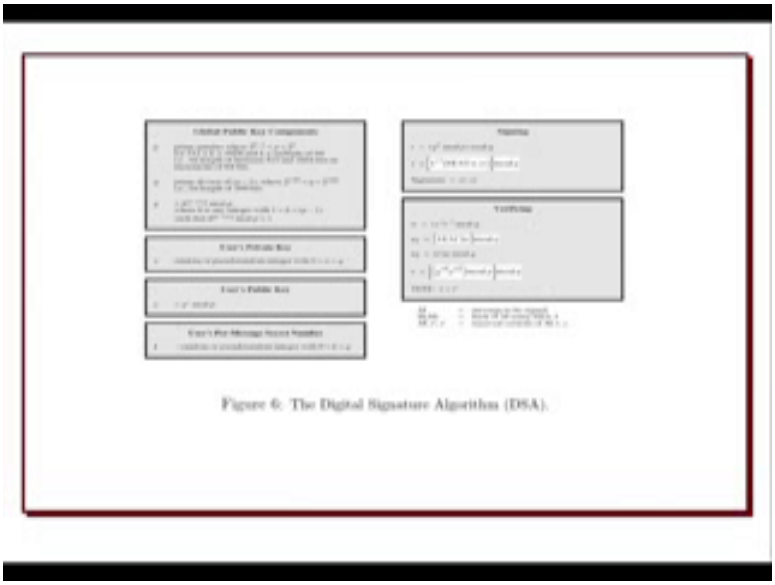


If we just take s inverse so s is this, so s inverse mod q s inverse mod q same as so, this is k inverse these inverse mod q so, k into m plus x r inverse mod q now this implies s inverse m plus x r it will multiply this will give us k mod q this will give us k mod q. So, we are multiplying just m plus x r. So, it will cancel out it will give us k mod q sorry. So, so v, v is basically v is basically g to the power t y to the power u mod P and then mod q. So, this is basically g to the power t, t is basically m is s inverse g to the power m s inverse mod q and because t is m is s inverse mod q y to the power u, u is basically r s inverse mod q then mod P mod q just simplifying this very simple, now y; we know y is basically what? Y is basically g to the power g to the power. So, this is g to the power x basically. So, we can just write g to the power x. So, this will be g to the power m s inverse in q mod q into y is g to the power x, so g to the power x r s inverse mod q mod P mod q.

This will basically give us g to the power m s inverse we take common m plus x r s inverse mod P sorry mod q then mod P and whole mod q and this is equal to what this is

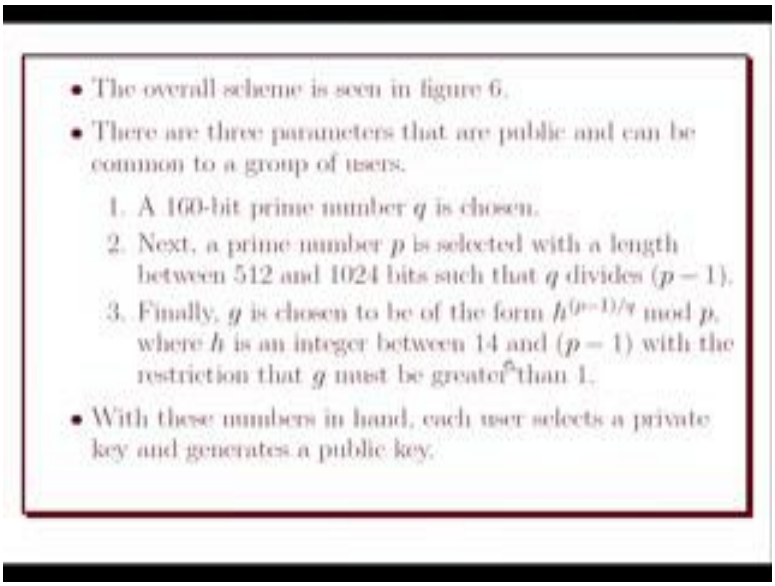
equal to we know this is with s inverse this mod P is equal to k. So, this is basically g to the power k g to the power k mod q then mod P mod q. So, this is basically r g to the power k mod p. So, this is basically r. So, this is valid. So, this is basically r. So, if you get this r then basically we are getting this if you get this r back then that is the verification that is having the verification. So, this is the digital signature standard.

(Refer Slide Time: 23:37)



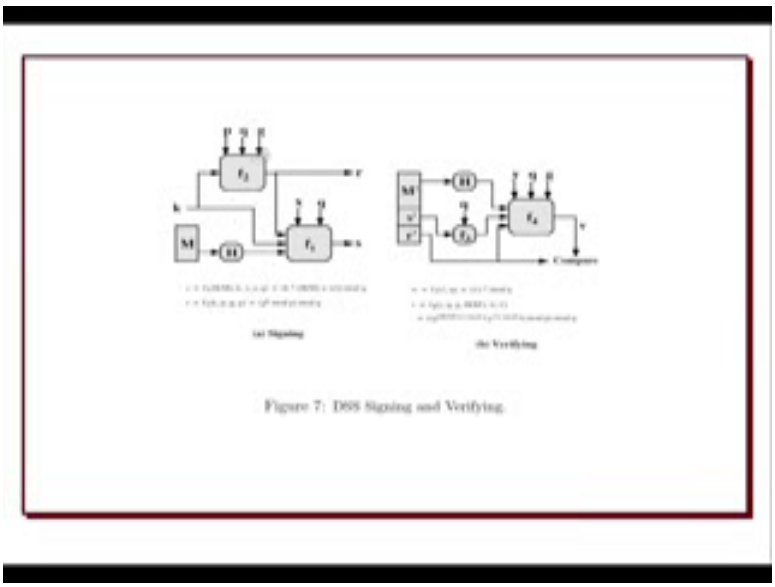

Let us come to the slide please. So, this is the parameter we are choosing, this is the public user individual user public key Alice, this is the user. So, you get can key this is the key which is the random number chosen by the signer and this is the signature here this is our small m h of m and this is the verification if i v is equal to this then we have done. So, this is just what we have discussed.

(Refer Slide Time: 24:05)




Now, let us come to the picture. So, this is the pictorial view of the signing and say verification.

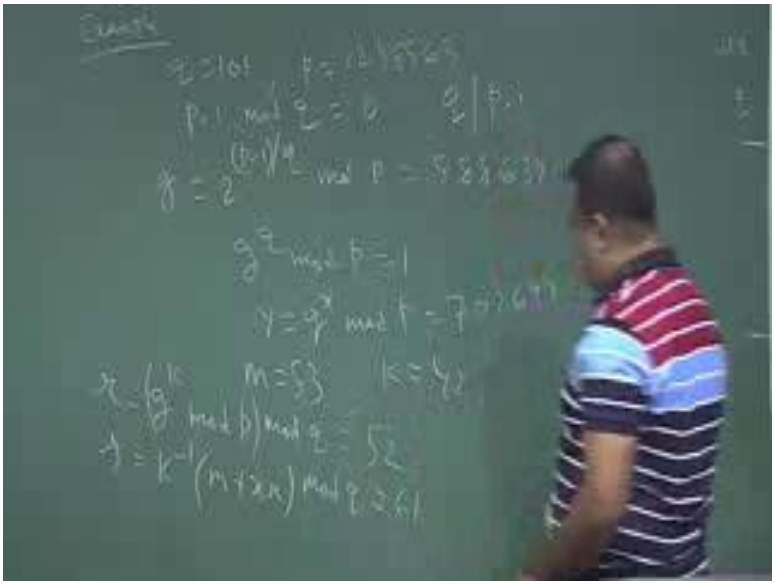
(Refer Slide Time: 24:12)




So, f 1 is basically f of so, we have this inputs are basically P q g and we have k, we are applying SHA here to make it small m then we have applying this f 1; f 1 it is defined like this. So, it is basically the graphical notation of this. So, it is basically giving us r and s. So, once we got r and s and this is sending to Bob after receiving these r and s how Bob is verifying? Bob is having the message also you have to send the message then Bob

is calculating this and getting by calculating this v is equal to r or not checking and verifying. So, let us take an example how it is working.

(Refer Slide Time: 25:17)



A quick example, so suppose we choose q is equal to 101 and P is equal to 1248563 then we can be easily check P minus 1 mod q is basically 0. So, that means, q divides P minus 1 that we can verify, now we choose g, we choose h to the, h is equal to 2. So, P minus 1 by q mod P which is if you calculate 5886434 then you compute so, we can just check this g to the power q mod P is basically 1 now suppose Alice has to sign a message say twenty three. So, now, y is basically so y is basically no, this 23 is not the message, 23 is the Alice's public key, Alice's secret key, this is the private key of the signer and then we compute, we need to compute y, y is basically g to the power x mod P which is basically 702644.

Now suppose we have a big message and we convert it and we got M is equal to 53 small m and we choose k is equal to 42 then we have to compute r and s. So, r is basically what? R is g to the power k mod P then mod q if you compute this will be getting 52 and then s is basically k inverse m plus x r mod q, this will be getting 61, you have to calculate this 61. So, these 2 will be sending to the verifier 52. So, r and s 52 and 61 so, after receiving 52 and 61, the verifier will be verifying this by calculating just t.

(Refer Slide Time: 27:47)

T is equal to m s inverse mod q which is basically 82 and u r s inverse mod q which is basically 29 and check v, then compute v g to the power t, it should be r y to the power u mod P mod q. So, this is 52 which is same as this r. So, the verification is done. So, the verification is done. So, this is the digital signature and for this signature; standard purpose, we need to use those parameter as specified by the designer and as accepted by NIHT as a standard.

Thank you.