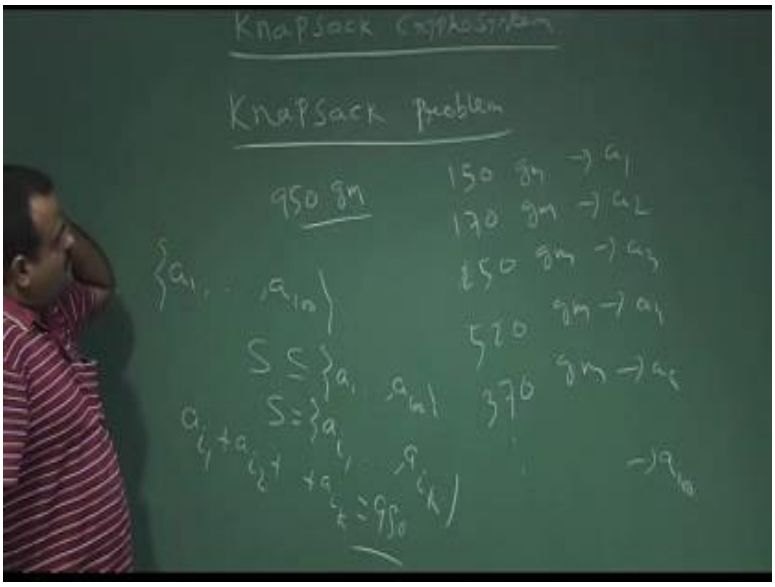
Internetwork Security
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture - 24
Knapsack Cryptosystem

I will talk about knapsack cryptosystem; Merkle Hellman knapsack cryptosystem. So, this is a public key cryptosystem.

(Refer Slide Time: 00:35)



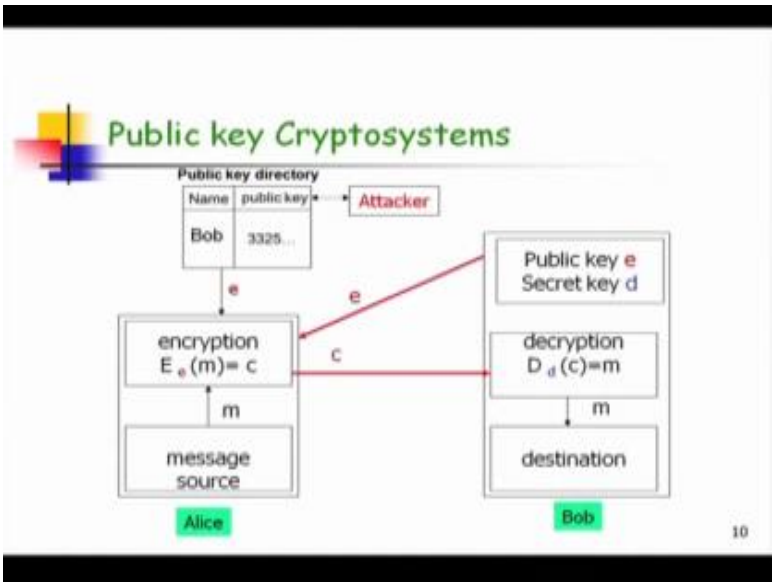
Basically it is a coming from the; what is called knapsack problem. So, problem is suppose we have; suppose you are going to market to buy an apple; buy some apple. So, your friend asks to buy say 100 kg apple. So, say 100 grams apple, say or 950 grams apples and you see in the markets so, there are apples whose weight are say 1150 grams 170 grams, like this, 250 grams. So, say 520 grams and say 370 grams like these. So, these are the apples. So, these should be denoted by a 1, a 2, a 3, a 4, a 5 and so on.

Your target is to buy 950 gram, it totally exactly so; that means, you have to choose. So, this is a collection of apples, a 1, a 2, say there are say 100 apples, a 100. So, this is the collection of apples a 1, a 2, a 100, among these collection, you need to find a subset, this is also called a subset sum problem, you need to find a subset such that whose sum is this 950. So, this is the all collection, among these we need to find a subset which is basically

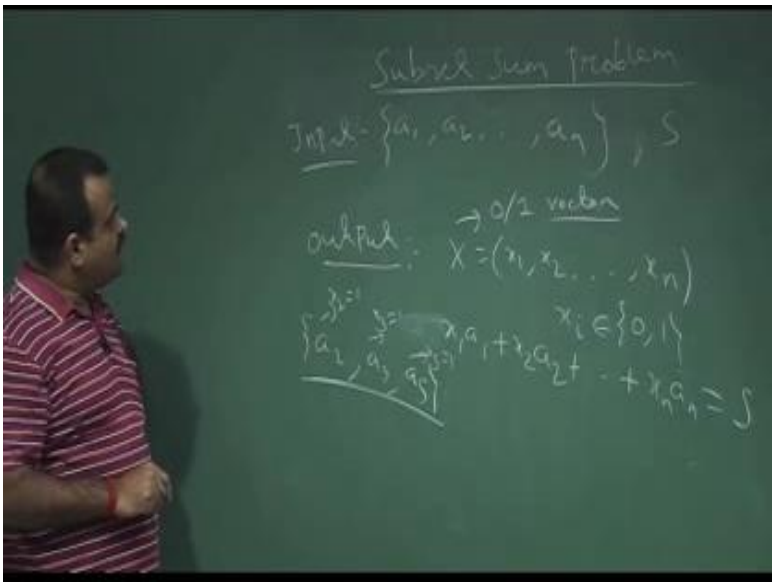
say S could be say a i 1. So, there are K apples such that summation of a i 1 plus a i 2 a i K is 950.

That is the problem, so, problem is we have given some integer. So, this is also called subset sum problem we and this set is called a knapsack, this collection; this collection, this set is this subset is called knapsack. So, we keep this apple into a bag. So, that bag is called knapsack.

(Refer Slide Time: 03:24)



(Refer Slide Time: 03:44)



We can formally write this problem, this is called a subset sum problem, no computer please. So, we call this subset sum problem. So, problem is we have given some integer a 1, a 2, a n. So, these are input and we have another integer which is S which is the tar, which is called target, which is called target, which is also another integer and now have to find out a subset the output will be so, we have to find out a 0 on vector x say x 1, x 2, x n, such that all x i's are from coming all x i's are 0 1 such that summation of a 1 x 1. So, x 1 a 1 plus x 2 a 2, x n a n, this is S.

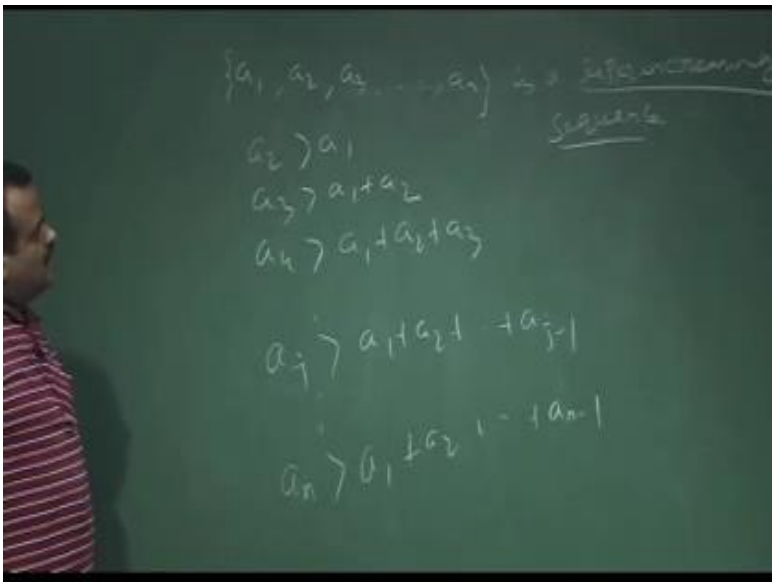
That means, if a if my subset is a, a 2, a 3, a 5, if this is our subset which is sum is giving us S then the corresponding x 2 is 1, x 3 is 1 and x 5 is 1 and remaining are 0. So, that is that is the collection. So, this is called knapsack this is knapsack. So, you have to find out this 0 1 vector, this is the 0 1 vector basically this will give us a subset.

In general how many subsets are there? There are 2 to the; if this is the power set number of subsets 2 to the power n. So, in general we have to the exhaustive search is we have to try for all subsets and we check whether this is matching or not.

In general we can try for all K i x i's. So, each x is 0 or 1. So, that is 2 to the exponential time algorithm which is hard. So, in general so, this is the problem, this is called subset, this is called subset sum problem. So, this is; this original version is a, this is sum problem is there exists a is there exists that yes or no that is called decision problem. So, decision problem in itself is hard not only it is a n p had problem and so, this problem is computationally infeasible problems, it is hard problem.

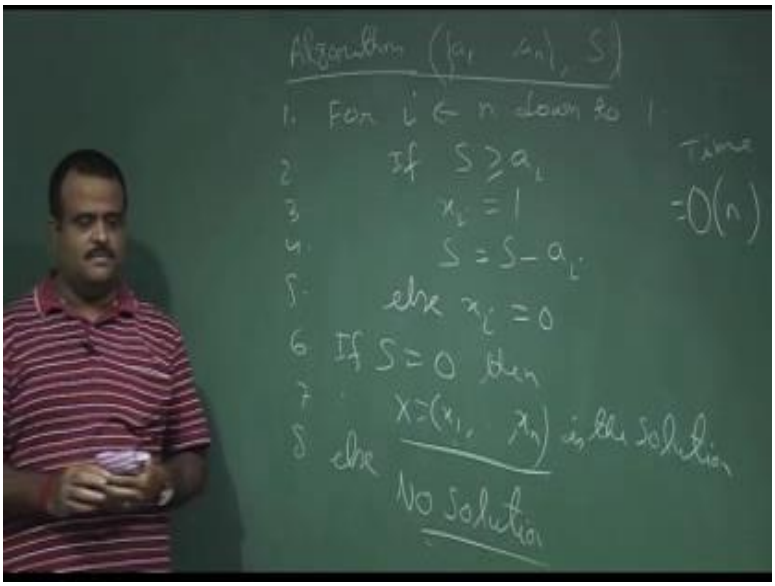
Now, how we can make it a feasible computational feasible problem if we take this a input in a super increasing way we have our a i's are super increasing sequence name this problem this may be feasible so, that we will talk about.

(Refer Slide Time: 07:00)



If this a i's are super increasing sequence, f a is super increasing, what do we mean by super increasing? Super increasing sequence; super increasing means a 2 must greater than a 1, a 3 must be greater than a 1 plus a 2, a 4 must greater than a 1 plus a 2 plus a 3, this way. So, a j must be greater than a 1 plus a 2 dot, dot, dot, a j minus 1. So, a 1 must be greater than a 1, a 2 a n minus 1. So, if you take any point any integer this must be bigger than sum of 2 previous integers so that that is called super increasing property. So, if our a i's are super increasing sequence then this subset sum problem having a polynomial algorithm to get the so, that vector. So, what is that algorithm? So, let us write their algorithm we are taking a super increasing sequence un habit target is and we need to get the subset who sum is basically S.

(Refer Slide Time: 08:39)

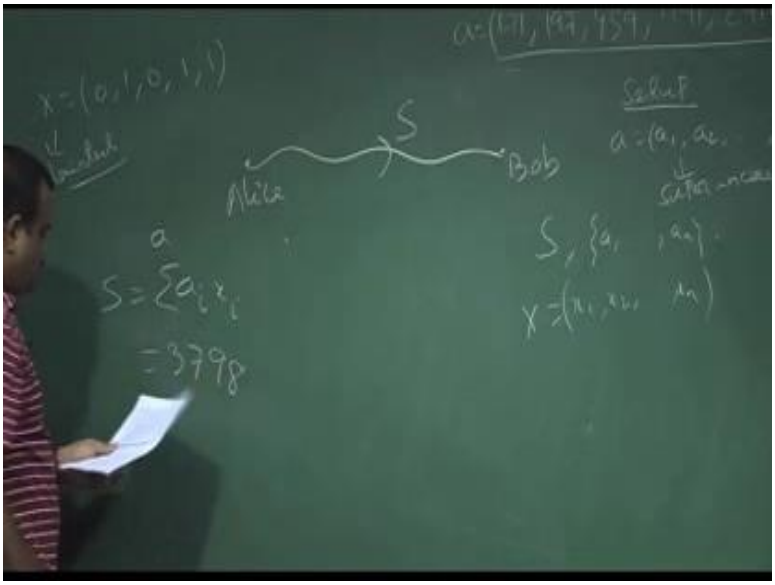


This is the algorithm to finding the sorry, algorithm. So, input is this super increasing sequence a 1, a 2, a n and a target sum S, this is the input. So, we have a for loop for i n down to 1. So, what we are doing? for this, for loop, we are starting from a n. So, if S is greater than a i. So, for first point if S is greater than if S is greater than a n then we have to choose a n otherwise because all the other if you take the sum of all the other elements from one to n minus one n is greater than that. So, if our target is greater than n there is no other way other than choosing n into the knapsack.

We have to choose a n. So, we take x i equal to 1; x i equal to 1 means xn equal to 1; that means, we are putting n into the knapsack and we decrease by S, S minus a i else. So, this is 3, 4, 5, else x i is 0. So, this is the loop and then finally, if S is 0 then we return this x 1, x 2, x n, as a solution then we return x 1, x 2, x l is the solution else no solutions else there is no solution; no solution exists solution. So, this is the algorithm to finding the knapsack when the a n's are in a n's are super increasing. So, this is a, what is the time complicity for this? This is basically a loop of size n. So, this is a order of n algorithm. So, this is a polynomial time algorithm on the size of the input. So, this is good, but i general it is hard here, we have a tab dot, tab dot is this; this sequence is super increasing sequence, if this sequence is super increasing sequence they then we can this problem is a easy problem otherwise i general this is a hard problem if we have not this information about this e i's whether super increasing or not.

This is the knapsack; knapsack problem, it is in general hard, but if our sequence of super increasing sequence then is it is a easy problem. So, we want to use this knapsack or subset sum problem to find out a to have a cryptosystem, what is called knapsack cryptosystem and which is developed by Merkle Hellman. So, let us talk about the cryptosystem.

(Refer Slide Time: 12:35)



This is a public key cryptosystem. So, in public key, so, Alice and Bob; suppose Alice wants to send a message to Bob, so, what Alice, so, now, this set up Bob has to do because Alice wants to send a Bob; message to Bob. So, Alice has to get the public key of Bob so; that means, Bob has to generate his public key private key pair and Bob has to make his public key public so, this setup as to be done by Bob because Bob has to receive the message.

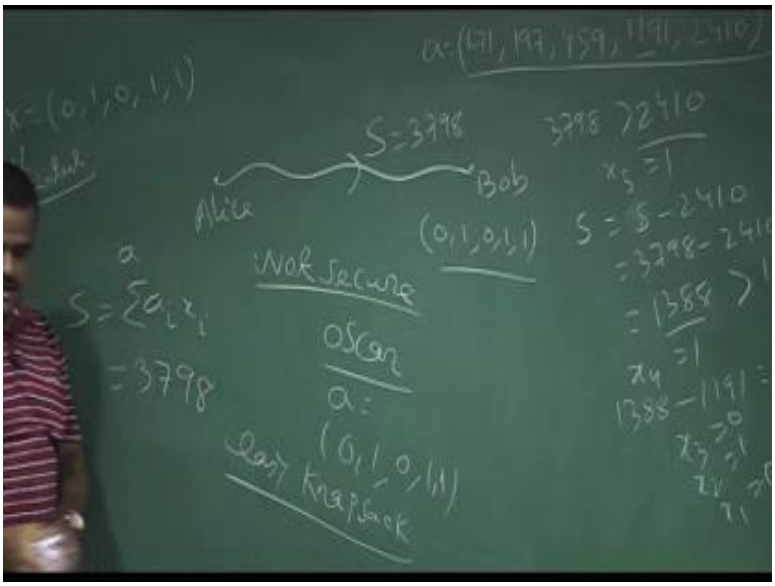
What is this cryptosystem? So, what we are doing? We are choosing a super increasing sequence a which is a 1, a 2, a n, this is a super increasing, increasing sequence and this is this Alice sending to Bob while Bob is sending to Alice. So, now, plaintext is a n bit number. So, plaintext is so, this is a x plaintext is x 1 x 2 x m, it is a n bit number. So, what Alice is doing as a encryption. So, Alice is generating this S summation of x i a i 1 to n this sum is in the ciphertext and Alice is sending this S to Bob.

Now, what Bob is doing? Bob is getting a S and Bob is having this super increasing sequence. So, Bob will apply the algorithm which we have just now discussed the

polynomial time algorithm and Bob can get the x 1 x 2 x n the message. So, this is the decryption of Bob we can take an example suppose the super increasing sequence is say 171, 197, 459, 1191, 2410, suppose this is the super increasing sequence, we are using here. So, now so, this is n is equal to 5.

Now, suppose Alice is choosing secret a, Alice is choosing this vector x say 0, 1, 0, 1, 0, 1 and 1. So, this is the plaintext, Alice is choosing this plaintext. So, what Alice is doing? Alice is encrypting using this with the super increasing sequence a and Alice is getting the ciphertext S summation of a i x I, if we are doing this just take this sum a, Alice is getting 3798; 3798 Alice is sending to Bob.

(Refer Slide Time: 17:10)



Now, what Bob will do? Bob has to apply the algorithm which we have discussed that super increasing algorithm. So, now so, that n that loops is there, i is from n down to 1. So, 3798 is greater than 2410, we are applying that algorithm which we have discussed. So, then we have to add these otherwise, this is super increasing, otherwise by summing this all, we will cannot able to cover this. So, we have to add this so; that means, if x 5 is so, x 5 must be 1 and a S is reduced to now S minus this 2 for 1 0. So, this is 3798 minus 2410. So, this is basically 1388.

Now again we compare this with this, now this is greater than 1388 is greater than 1191 so; that means, x 4 has to be 1 and then we deduct this x from 1191 and then we got say

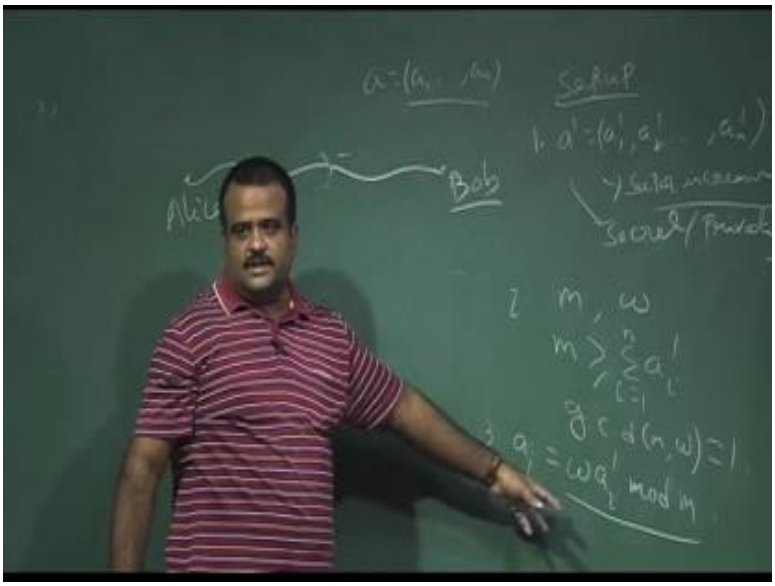
we got this number and slowly we will just continue this passion and he we can see that x 3 is 0 x 2 is 1 and x 1 is 0. So finally, Alice got this x, plaintext is a, Bob got 01011.

Now, this can be achieved by Oscar also because very small because this is the super increasing sequence and this is Alice, Bob is betting public. So, Oscar is also knowing this super increasing sequence a. So, the same thing same algorithm, Oscar can also mound and Oscar can get this x. So, this cryptosystem is not secured. So, this is called easy knapsack easy knapsack cryptosystem and it is not secured because anybody can any third party can mound this.

Now, the idea is how we can make it strong, how we can make it hard because this super increasing properties is a if we just make it is public then everybody if everybody knows this then it is everybody can apply that algorithm and get the message. So, now, the challenge is how we can make it hard. So, the idea is to do the modular operation. So, let us go for this knapsack cryptosystem this is called easy knapsack which is not secured now we will go to the original knapsack cryptosystem.

We have to somehow destroy this super increasing property and then we made that sequence has public, but again Bob as to get back that super increasing property in some way so, that Bob can apply that algorithm to get the message. So, that is the idea. So, idea is somehow we need to destroy this super increasing property before making this sequence public. So, that is the idea. So, what we are doing. So, Alice wants to send a message to Bob. So, Bob as to generate his public key private key pair.

(Refer Slide Time: 21:05)

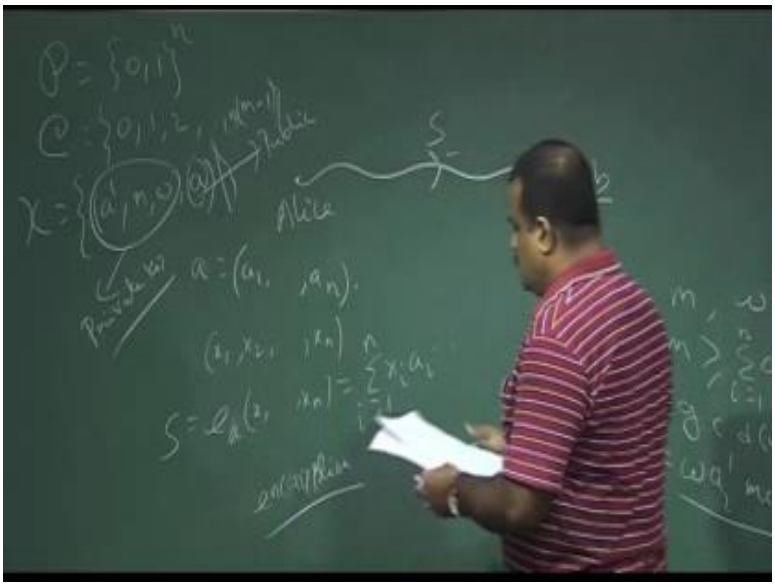


This set of Bob is doing. So, what Bob is doing? Bob was choosing a super increasing sequence this is a super increasing sequence and this is not public this is secret this is a secret key of Bob or private this Bob is not making public . So, now, what Bob is doing? Bob is choosing m and w such that m is greater than summation of a i prime i is equal to one to n and w is basically chosen such that gcd of m and w is 1. So, that we will have a, we will have these w inverse would w inverse I mean exists under model. So, that is the idea.

What we are doing here? So, we are just choosing this m and w such a way this and then we denote this we just multiply this we create a i's which is basically w a i prime mod m. So, by taking gcd of m m w is 1, we will grantee that w n will exist under this.

Now this will destroy the super increasing properties. So, this is the tack ties, these way we destroy the super increasing property and now these sequence we can make it public this a i's. So, this a i is now public. So, this a is basically a 1, a 2, a n. So, this is no more super increasing we have destroyed the super increasing nature of it by multiply w mod m. So, this is the trick we are doing. So, now, we can make this public.

(Refer Slide Time: 23:30)



This is the public key of Bob and this is Alice is getting this, now here the plaintext phase is basically n bit and the ciphertext phase is we have to choose mod m. So, it is number between 0 to m minus 1 n into m minus 1.

And the key space is this a prime which is the super increasing sequence m w n a this is the key space and among this among this, this is the public this is the public. So, this is doing by Bob this setup is doing by Bob and these all are private key of Bob this Bob is not revealing these are all secret to the Bob.

Now, what is the encryption? Now a Alice is choosing a message m. So, Alice is choosing a message m which is a n bit number and then encryption is basically Alice is just doing this. So, this is the k, k is the public key of this is we can say a a is the public key of Bob using the public key of Bob Alice is just doing the summation of a i k i i is equal to 1 to n.

This is the encryption and this is this is our S and this is the ciphertext Alice is sending to Bob now how Bob will. So, this is encryption this is this is encryption, now at the decryption what Bob will do?

(Refer Slide Time: 26:02)

At the decryption what Bob will do? So, Bob will just Bob is getting S which is nothing but summation of x i a i. So, now, Bob will apply w inverse on both side and w inverse is exists because we have taken gcd of m and w is one. So, that will guarantee that w inverse will exists under mod m. So, Bob will compute this mod m and this is say this is say S prime.

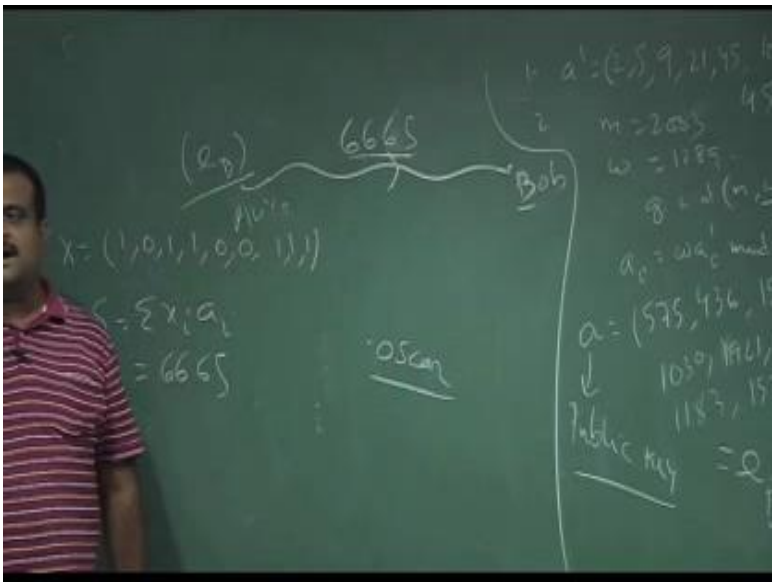
Now, this is basically, what is S prime? So, S prime is basically so, w inverse what was S? S is basically so, summation of a 1 x 1 plus a 2 x 2 plus a n x n. So, now, this is w inverse a 1 x 1 plus w inverse a 2 x 2 dot, dot, dot, plus w inverse a n x n, now this is nothing but so, a i are basically a i's, we get from that super increasing sequence by multiply w a i prime mod m. So, if multiply by the w inverse with this it will give us just the a i mod m. So, this is basically a i prime. So, a one prime x 1 plus a 2 prime x 2 dot, dot, dot, a n prime x n. So, this is a now we can now Bob can apply that algorithm polynomial time algorithm and can get because this now this sequence of super increasing sequence.

Now Bob will apply that polynomial time algorithm and we will get back this x i's x 1 x 2 x n. So, this is the plaintext. So, this is the decryption process. So, Bob as to apply that polynomial time algorithm, but now this is super increasing sequence. So, this is the this is the n a, this is the knapsack cryptosystem where we are we are choosing Bob is choosing originally super increasing sequence, but Bob is destroying the super increasing

property and again Bob is at the decryption phase or Bob is getting back that property and doing this.

Now let us take some example quick example. So, suppose so, this is Bob has to do because Alice as to send a message to Bob. So, this setup as to done by Bob.

(Refer Slide Time: 29:00)

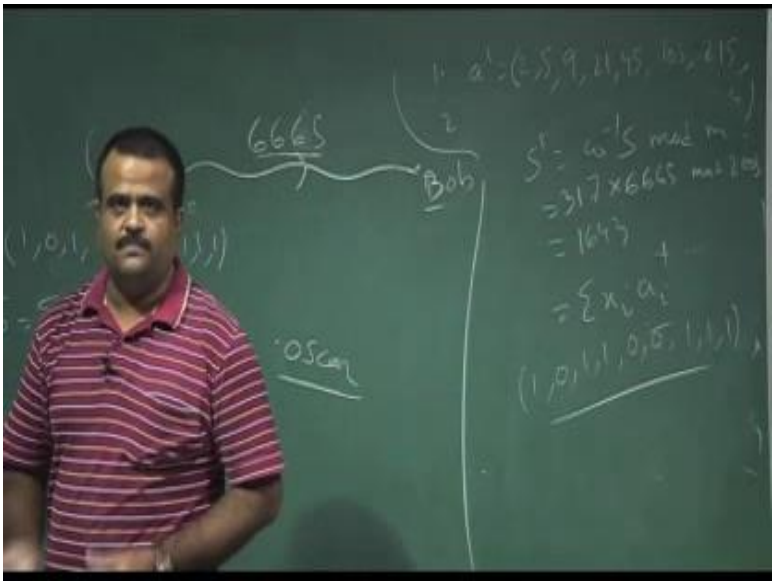


Bob is choosing a super increasing sequence a prime say 2, 5, 9, 21, 47 by 45, this 215, 450, 946. So, one can easily verify this is a super increasing sequence. So, this is Bob is doing because this setup because Bob as to get the message. So, this setup is done by Bob. So, this is 1 of the Bob's public key. So, this Bob is choosing a super increasing sequence sorry this is a private key of Bob and Bob is keeping this as a secret key of Bob.

Then Bob is choosing m and w and one can verify that gcd of m and w is 1 so, that will guarantee that w inverse will exist on that mod m then we can multiply this, these are the a i's. So, a i prime we can just compute w a i prime mod m. So, if you do that then we get the a vector as 575, 436, 1586, 1030, 1921, 569, 721 comma 1183, 5740. So, this is the public key of Bob and this Bob makes this public, this is no more super increasing we destroy the super increasing property and this is basically e b public key of Bob and this is sending to Alice or Alice as to get it from a public key directory now suppose Alice choose some message x say 1, 0, 1, 1, 0, 0, 1, 1, 1.

Now Alice will Alice is having this public key of Bob which is no more super increasing. So, Alice will compute summation of x i a i and this is basically 6665, now this is the ciphertext Alice is sending to Bob and now for Oscar; Oscar is knowing this which is not super increasing Oscar is knowing this sum. So, for Oscar it is a hard problem because subset sum problem is hard in general.

(Refer Slide Time: 31:55)



Now, decryption is just we have seen. So, Bob can just apply this S w inverse S mod n. So, this is 317 into 6665 mod m. So, this will give us 1643. So, now, we know this is basically summation of x i a i prime which is this sequence is super increasing, now Bob can apply that algorithm which we have seen, the polynomial time algorithm and Bob can get the plaintext 1, 0, 1, 1, 0, 0, 1, 1, 1. So, this is the decryption. So, this is the knapsack cryptosystem.

Thank you.