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

Lecture - 09
Stream Cipher

So we talked about another symmetric key encryption primitives; which is stream cipher.

(Refer Slide Time: 00:28)

So, Alice and Bob is having the common key k from the symmetric key scenario they have a common key k. Now we have seen in block cipher, we have a function which is take the l bit input and if the l bit output that is the block. Now in stream cipher, this is also another primitives of symmetric key cryptosystem. In stream cipher usually the plaintext are in binary bit say this is 0, 1 bits. So binary bits and we have a key stream generator which can generate the key stream bit which is also a binary bit 0; 1 bits 1, 0, 1, 1, 0, 1 like this. So, this is key stream bits and we do the beta XOR with this 0, 1, 1, 0, 0, 9, like this is the ciphertext.

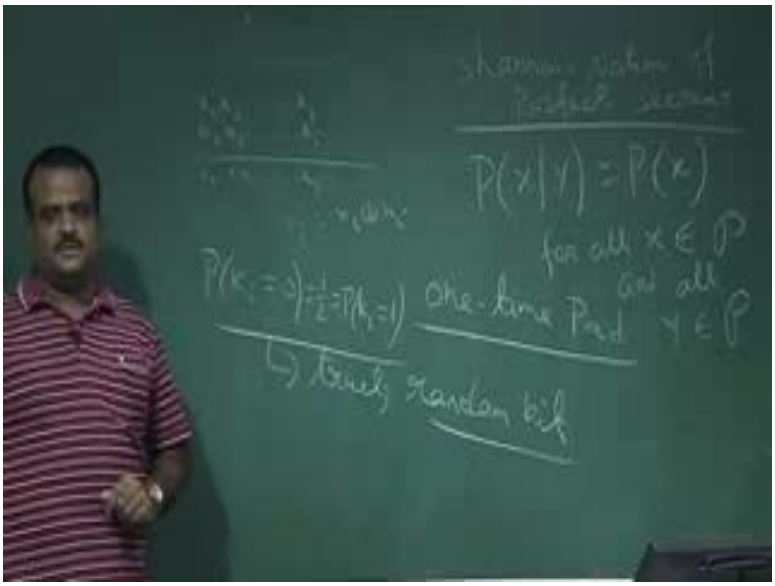
So, this is a typical stream cipher, so they have a common key k. So, from this k one has to get this key stream, so will talk about some key stream generator. So, it will give us the key stream, so this is basically like this; we have this plaintext x i and so on and we have the key stream k 1, k 2, k i is at the key stream bit, now we do the be drive exuding and we get c 1, c 2, c i and so on. So, c i is basically x i; x of it k i; ith bit of the splintex

XOR with the ith bit of the key stream. So, there should be a what is called key stream generator which takes this secret key and suitable to generate this k 1, k 2, k i and so on and this we are going to XOR with this plaintext bits and gives up the ciphertext bits c 1, c 2, c i, and so on and these values is sending to box c 1, c 2, c i, and so on; so this is the encryption. Now how Bob will decrypt it, Bob is receiving this c 1, c 2, c i and so on, so Bob has to get back the message x 1, x 2; so this is a message x 1, x 2, x i and so on.

So, what Bob will do? So, Bob is having this c 1, c 2, c i; so Bob also has to run the same key stream generator and get this key stream bits k i and so on. So, Bob will simply XOR and get x 1, x 2, x i and so on because this c i; x r with key i is basically c i is nothing but x i XOR with key i and this is XORing with key I, so key i; is key i is getting cancelled this is x i.

So, we are getting the x i basically the ith bit of the plaintext, so this is the decryption. So, now the question is how we can get this key stream bit, so we will take some example of a key stream bit generator; one is LFSR; Linear Feedback Shift Register, but before that let us analyze this nature of this key stream bit. So, this is called stream cipher now so in stream cipher basically we need to generate the key stream.

(Refer Slide Time: 05:09)



So, we have this plaintext bits, so one has to generate the key stream and then we will do the XORing to get the c i; s i mean c 1, c 2, c i. So, c i is basically now if you have a coin

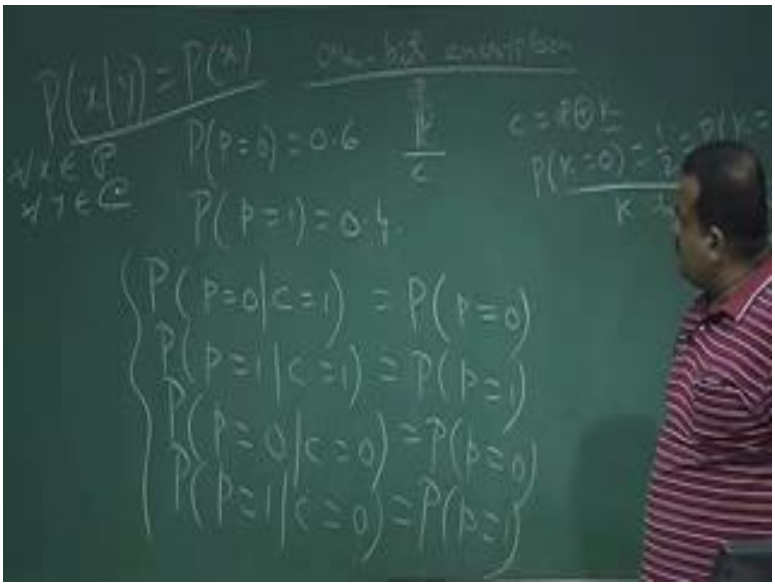
then you can generate the key stream bit by tossing the coin. So if we get ahead, we will put it 0; if we get a tail, we put it 1; just keep on tossing a coin we will get a key stream.

So, now we will define the notion of what is called Shannon notion of perfect secrecy. So, Shannon notion is telling if the probability of plaintext given the ciphertext is same as probability of plaintext and this true for all plaintext and all ciphertext. So that means; so ciphertext is public, ciphertext is seen by anybody. So, this is transmitted over the public channel, so it is seen by the Oscar. So by seeing the ciphertext, if we are not getting any information about the plaintext; if this conditional probability is the basically the unconditional probability of P so; that means, they are not getting any advantage of knowing the ciphertext.

So, then we call our cipher is perfectly secure cipher and then our cipher is called one time pad; if we have a stream cipher which is having this property like if probability of a plaintext beat given some ciphertext bit is same as probability of plaintext bit; that means, we are not getting any advantage of knowing the cipher text and this can be achieved if this key i's are truly random bits, truly random bits means so k i's are by a binary bit, so it could be 0 and 1; if k i is 0, is half is same as probability of k i is 1 then it is called truly random bit.

And then we will see, if our k is satisfying this truly random bit sequence, if we have a truly random bit sequence then we can achieve this channel notion of perfect secrecy and then that stream cipher is called a onetime pad, but the question is whether we can achieve that, is it possible to have a truly random sequence or not; because see we cannot, we are tossing a coin and head tail, so we cannot make a coin where the it is a totally unbiased coin. So, that means probability of head cannot be equal to probability of tail. So, anyway we will be using some algorithm to generate the key stream, so that never gives us a truly random sequence of the key stream. So, that is why it is called sewed random sequence will come to that, but before that let us prove this, if our key stream as a key stream bead is truly random bit then we achieve the channel notion of perfect secrecy.

(Refer Slide Time: 09:56)



So, let us try to prove this and then that is called one time pad. Suppose we are encrypting a single bit, so our x is suppose we are doing one bit encryption. So, we have a message X; which is a single bit and we generate a key bit k and we XOR this and we get a c bit. So, c is basically x plus k or we can say P over here plaintext; plaintext is a single bit, so P x service k. So, this is the single bit encryption and P is the one bit plaintext; k is one bit key, one bit key generated from this secret k and c is the one bit cipher text.

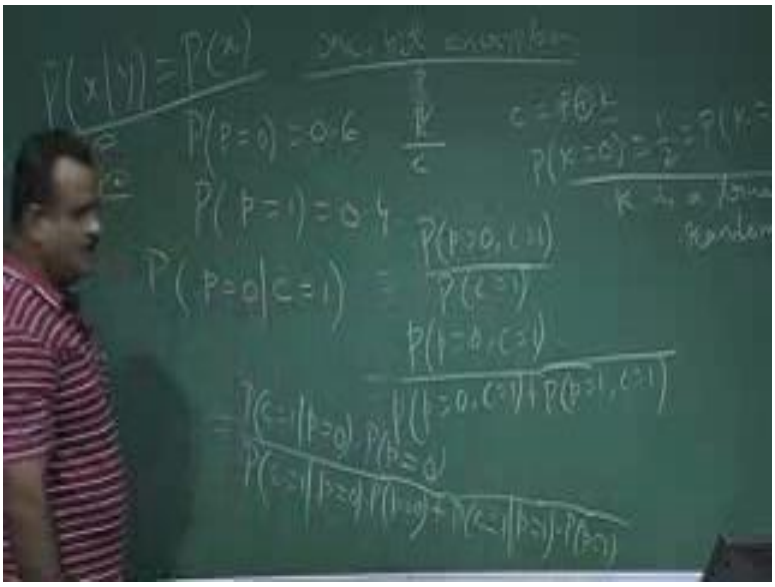
Now, suppose this is a truly random bit k so; that means, suppose probability of this is same as probability of k is equal to 0. So; that means, k is a truly random bit, then we want to prove that this is achieving the channel notion of perfect secrecy; that means, it is a onetime pad basically. So, to show that let us have some prior distribution of P, suppose probability of P is equal to 0 is 0.6 probability that P is equal to 1 is 0.4 is the prior distribution of the P. So, now we want to find out this probability; probability of say P is equal to 0 given, c is equal to 1.

So, that this is probability of P is equal to 0 and we are not getting any advantage of knowing the cipher text. Similarly you have to show the all possibilities P is equal to 1, given c is equal to 1 is equal to P is equal to 1, this we have to show and probability that P is equal to 0 given c is equal to 0. So, is basically probability that P is equal to 0 and last one is probability that P is equal to 1, given c is equal to 0 is basically probability of

P is equal to 1. So, if we can prove all this statement then we can say that probability of x given y is basically probability of x and this is true for all x from the plaintext and for all y from the ciphertext.

And then we say we are achieving the channel notion of perfect secrecy. So, we will try to prove only one then the remaining, you can try at home. So, let us try to prove this probability of this is basically probability of P is equal to 0.

(Refer Slide Time: 13:28)

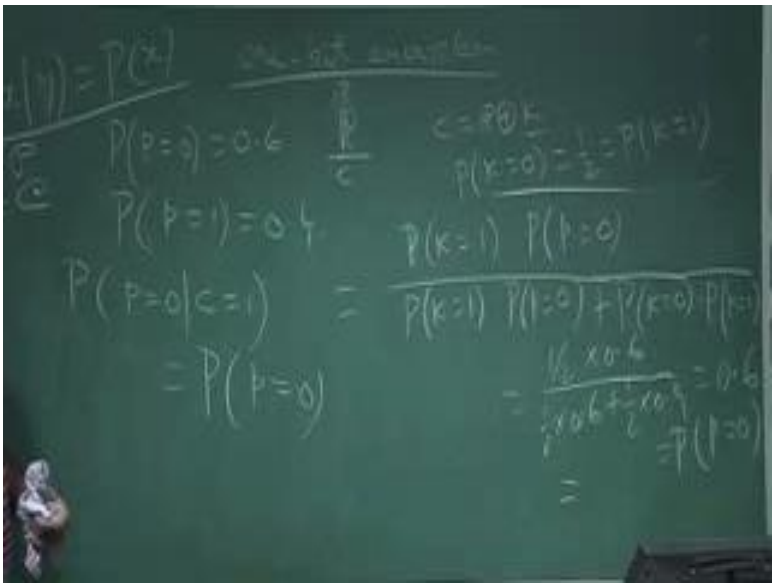


So, this is a conditional probability, so this is basically probability of a given b is basically probability of a intersection b divided by probability of b. So, this is basically probability of c is equal to 1 and this is basically probability of P is equal to 0 and c is equal to 1. So, this can be written as, so by base theorem I mean the total probability rule. So, probability that P is equal to 0 comma c is equal to 1 and probability that P is equal to 0 comma c is equal to 1 plus probability that P is equal to 1 comma c is equal to 1 because P can be 0 or 1, so this is the total space; so this way we can write.

So, these can be written as probability of; so this can be c is equal to 1 given probability of P is equal to 0 into probability of P is equal to 0 divided by; so again this can be written as probability of c is equal to 1 given P is equal to 0 into probability of P is equal to 0 plus probability of c is equal to 1 given P is equal to 1 into probability of P is equal to 1. So, just the total probability rule, so these we can just calculate. So, probability of c is equal to 1, P is equal to 0 this is same as; so we are XORing. So, this is same as k must

be 1 because if we are XORing, so if this is 0 means this is 0. So, this is 1 means k has to be 1. So, this is same as probability k is equal to 1, so this is equal to; maybe I can write here.

(Refer Slide Time: 16:04)



So, I can just right here, so this is equal to probability of k must be equal to 1 into probability of P is equal to 0. Now this one probability that P is equal to 0 and c is equal to 1 same as probability of k is equal to 1, this is probability that k is equal to 1 into probability that P is equal to 0 plus probability of k is equal to 0 into probability of P is equal to 1.

So, if we just put the value this is basically half into 0.6 divided by half into 0.6 plus half into 0.4. So, this is basically 0.6 which is same as probability that P is equal to 0, so basically this we achieve probability of P is equal to 0. So, similarly other probability calculation we need to do and then we can convince that probability of a plaintext given the ciphertext is basically same as probability of plaintext. So, we are not getting any information about the plaintext by knowing the ciphertext.

So, provide if our key stream bit is truly random bit then this will give us a onetime pad and this is perfectly secure cipher under the notion of Shannon, but this is not possible in practice because in practice will use some algorithm to generate this key stream bit. So, there we cannot achieve these truly random bits for k i; so that is why it is called as pseudo random bit generator.

So, this is the result where we can say if our key streams are truly random bit sequence then we can achieve the Shannon notion of perfect secrecy, which is not possible in practice which is impractical. So, truly random sequence; random number generator is not there, so you have a then we say that sequence we are getting from a key stream generator is called pseudo random bit generator.

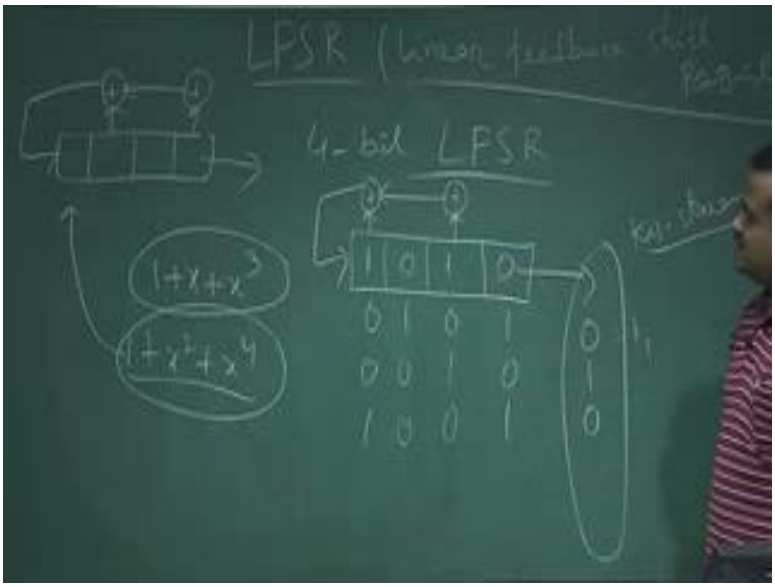
(Refer Slide Time: 18:58)

So, pseudo random sequence generator because we cannot have a random sequence generator that k 1, k 2, k key stream are a sequence. So, we cannot have a; so this is the key stream generator, so this is taking the key and it is giving us the key stream k 1, k 2, k i and so on; depending on the size of the plaintext. Now if this all the key i's are truly random bit, then this generator is called a truly random sequence generator this is sequence generator, but which is not and then it will achieve the; it is basically one time pack and it will achieve the Shannon notion of perfect secrecy, but this is not possible in practice.

So, we call this generator; any generator of key stream is called pseudo random sequence generator and we will see how good this sequence is. If you have a key stream generator, we will see how good means for how much randomness is there. So those we have to taste, some randomness taste we have to do, so those are basically some necessary tests for a truly random sequence, so those properties; a truly random sequence must have. So, if we are getting a sequence; if we take a period of that sequence, if we run some tests

and if we pass that test then those tests are basically necessary condition for a truly random sequence. If we have a random sequence truly random sequence those properties it must have, so anyway we will discuss those.

(Refer Slide Time: 21:37)



So, let us just have a key stream generator, so one key stream generator is LFSR; it is called Linear Feedback Shift Register or in short LFSR. So, it is basically a register; register means basically a collection of flip flops. So we can take an example; if 4 bit LFSR is basically we have a four people of which can store a single bit of information.

So, these are all can store a 0, 1 bit; 1 bit of information it can store and it has some connection like this. So, these are all flip flops, so it has some connection like this, so a connection could be somewhere else also, so these are called feedback bits. So, this position and this position is contributing in the feedback. So, what we are doing in each clocking will output one bit and then, so this bit will come here, this bit will come here, this bit will go out and that is our key stream and this feedback bit will be calculated by this connection; wherever you have connection.

This suppose we have initially we have say 0, 1, 0 suppose this is the initial state of the LFSR and this may be the key, the key shared between Alice and Bob; the secret key shared between Alice and Bob. So, we fill this by the initialization of this LFSR and then what we do; we just clock it each time, so for the first time in the first clock 1 will come here, 0 will come here, 1 will come here, 0 will go here. So, this is the k 1 key stream bit

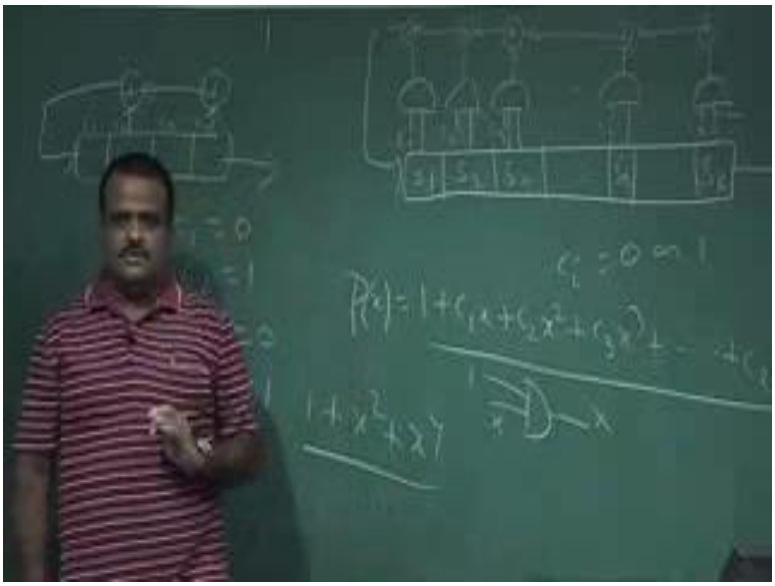
and this will be field; this is linear feedback shift register, we are shifting and this will be filled by these two position, so this is 1, this is 0; so if we XOR it will be 0. So, this is at first clocking, in the second clocking this will come here, this will come here, this will come here 1 will be going out.

And again, so now this is the state; now again these and these will be XOR and this will be putted here. Now again this will be like this, so this will come here, this will come here, this will be here and this 0 will be going here as a key stream and again we have these two. So these and this, so it is 1 basically, this and this will be XOR; so like this. So, this is the key stream build, so this is an example this is therefore, just four bit LFSR, we have a general l b d; LFSR, we will talk about that. So, this connection could be here then we have a different LFSR and it has nice relationship with the polynomial. So, this we denoted by 1 plus, so here we have a connection x and this is x square; no connection. So, x square will not come; this is x cube, so this is the polynomial corresponding to this LFSR.

Now, if we have a polynomial like this 1 plus x square plus x to the power 4 then what is the LFSR corresponding to this; then we have; so this is basically x position x x square. So, we have a feedback over here and then x to the power 4; we have a feed back over here, so this is the LFSR corresponding to this polynomial.

So if we have a LFSR; we can have a polynomial four degree polynomial or if we have a polynomial then we can construct a LFSR and if this polynomial is happened to be a primitive polynomial; we will talk about primitive polynomial then this will give us a full sequence, this period of this will be full; if this polynomial is a primitive polynomial. So, now let us talk about general case l b d; LFSR. So, this is just an example of four b d LFSR this is an example of key stream generator, so it is generating the key stream.

(Refer Slide Time: 26:55)



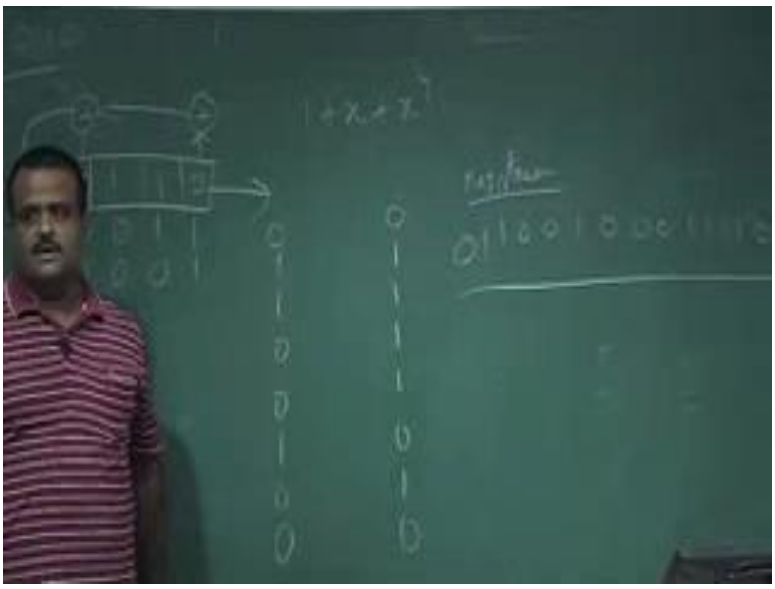
In l b d LFSR, we have basically instead of four bit we have l bits, so we denote this by state is below S 1 or maybe we can start with 1; S 1, S 2, S 3, s n, so these are all fifth lops.

So we will have a connection like this, we will put c 1 then I will explain sorry you will have c 2. So, we will have c 3 like this and then for the S I; this is the ith, so we have c i and these are for all of this; we have like this c L dot, dot, dot, dot, dot, dot. So, these are all this feedback and this is going here, so this c 1, c 2, c l are either c i's are either 0 or 1. If c i's is 1 then that corresponding state or corresponding field will be contributing in that feedback. So for example, this case this is c 1, this is c 2, this is c 3, this is c 4, so for this case c 1 is 0 because this is not contributing, if c 1 is 0 this is the n gate so; that means, whatever value it is having, it will be 0 only because c 1 is 0. So, for this case c 1 is 0, c 2 is basically 1; c 3 is 0 and c 4 is 1 and here we have this P strain.

So, this is one example, so this c 1; if c 1 is 1; then that corresponding field is contributing in the feedback, otherwise it is not and the LFSR corresponding to the; and the polynomial corresponding to this is basically 1 plus c 1 x plus c 2 x s square plus c 3, x cube like this c l x to the power l. So, this is the polynomial corresponding to this LFSR, now for this LFSR; so it is just basically 1 plus c 1 is 0, so c 1; x 1 come c 2 is 1, so x square, c 3 is 0; so it will not comes c 4, so this is the polynomial corresponding to this LFSR.

We can just draw this using this and gate because if the c 1 is 0 then this is always 0, no matter what is the value is having over here and if c 1 is 1 then this will pass, whatever the value over is here because in the and gate, if this is 1 then if we have x; this is basically x. So, whatever value if it is 0, it will pass 0, if it is 1 it will pass 1. So, whatever value over here is that will be going there, so that means this corresponding field will be contributing in the feedback. So, we will take a quick example where we have this going to generate the key stream.

(Refer Slide Time: 31:07)



So, suppose we take the polynomial like this 1 plus x plus x to the power 4; so that means, our LFSR is this, so 1 is there and x to the power 4, so this is there; this is our LFSR.

So, only these two state by these two state are contributing in the feedback, so this is we want to see; so this is the primitive polynomial. So, you want to see whether we are getting a full sequence or not; if you start with any initial state, so that is basically the secret key shared between Alice and Bob. So, suppose we start with say some 0, 1, 1, 0, so suppose this is the secret key k shared between Alice and Bob and they are using this LFSR as a key stream generator in order to encrypt their plaintext. So, now if we run this, so in the first sort; this will come here, this will come here, this will come here; 0 will be out and what about this? This will be basically XOR between these two.

So, this is the first step and next will have, so this is basically these and these we have 1, 0, 0. So, this will come here; this will 1; 1 like this. So, like this if you continue we will be getting like this 1, 1, 0, 0, 1, 0, 0 then we have again this 0, 1, 1, 1, 1, 0, 1, 0. So, this is basically 16 clocking. So this sequence key stream is basically, we are getting 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, so this is a full sequence 16 period 15.

So, this is one way of generating a key stream, so this is the key stream generator LFSR basically and if we choose the polynomial is primitive polynomial, it is giving us the full sequence and it will satisfy some nice randomness property. So, it will satisfy some necessary condition of the random sequence. So those are called randomness taste, we will talk about those in detail.

Thank you.