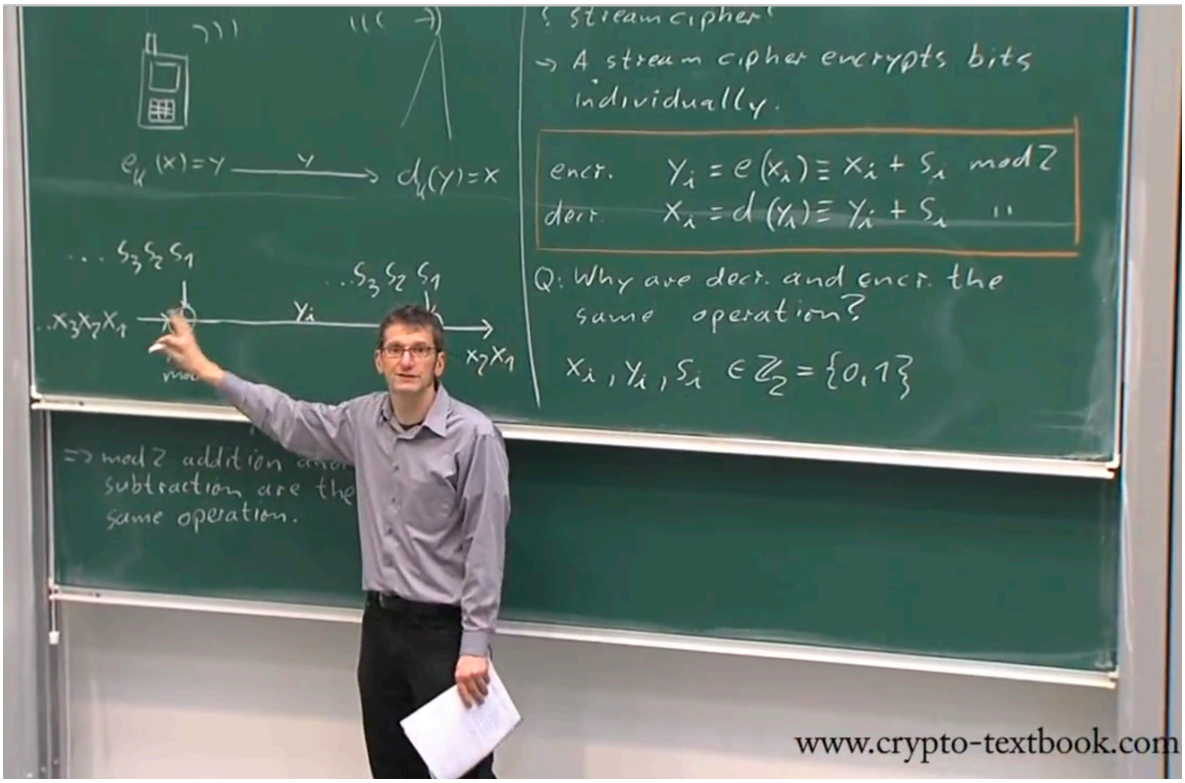
Lecture 3 Stream Cipher :
Mobile Communication(Voice encryption) is using stream ciphers.
Stream Cipher :- Encrypts bits by bits.
Encryptions => yi = enc(xi) = xi + si mod 2 in stream cipher we call key as s.
decr(yi) = yi + si mod 2

Question : why are we adding the bits in both encryptions and decryption ?
Answer : xi = yi + s1 mod 2, xi = xi + s1 + s1 mod2 => xi + 2si mod2.
As 2 mod 2 is 0 so above holds.

Mod 2 addition and subtraction are same.



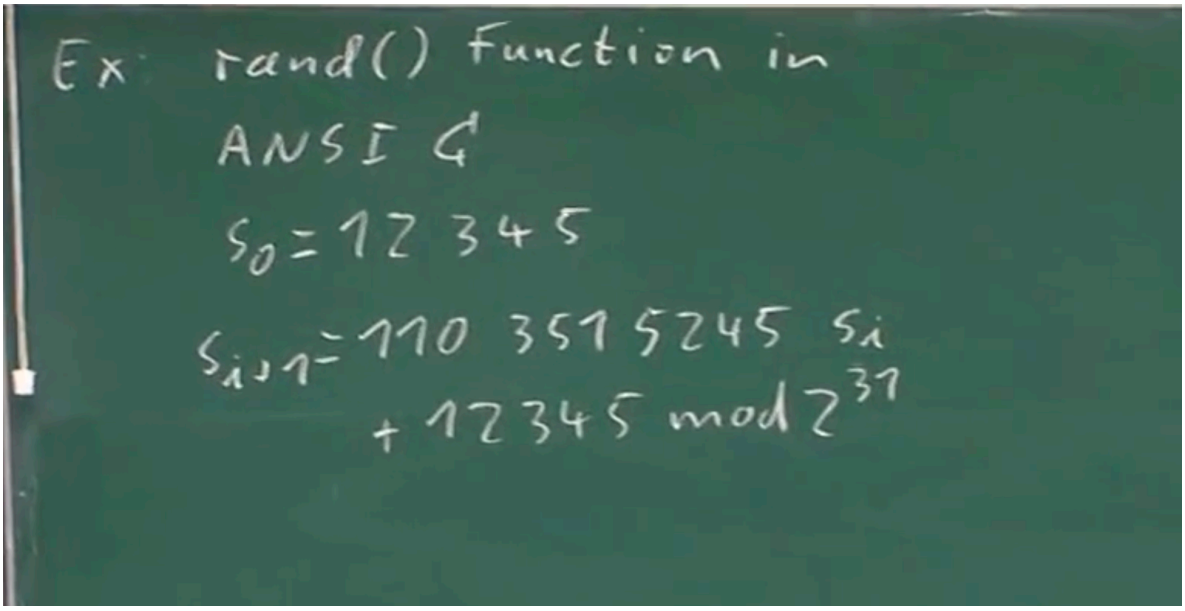
Modulo 2 addition is same as XOR gate.

Question :- Why we use XOR Gate not any other gate ?
Question :- How to generate Si (Key bits) ? Related to Randomness.

Random Number Generator(RNG) :-
Types of RNG's :-
— **True Random Number Generator** (eg coins, dices, noise, mouse movement, key strokes)
In PGP encryption etc, sometimes they ask for random key , we give mouse movements also time between key presses.
Very old tool which uses disk movement.
We cannot regenerate/recreated them.

So they might be useful in creating unique private keys but sometimes we need to recreated.

— Pseudo random number generator (PRNG) :-
PRN's are deterministic. They require seed and they run recursively ie Si+1 = func(si)



you cannot use the above rand function.

— Cryptographic secure PRNG (CPRNG) :-
are prng's with a property ie they are unpredictable.

Meaning given si to si+n-1 , it is computationally infeasible to compute si+n in the given time.
Little about Randomness :-Internal implementation of secure random generator function in java etc.

1. /dev/random (Blocking) is used for getting random bytes
2. /dev/urandom (NonBlocking) is also used.
3. Blocking, blocks until sufficient entropy is not there.
4. Java internally uses /dev/urandom and random.
5. Random function uses various factors like Network, Key Strokes, mouse , disk, cpu utilisation matrix etc.
6. All the above factors are passed to SHA1 and then written to random/ urandom devices.

One Time Pad :- "Perfect" cipher.
Unconditionally secure and which cannot be broken with infinite computing resources.
2^300 is something about the atoms in universe.

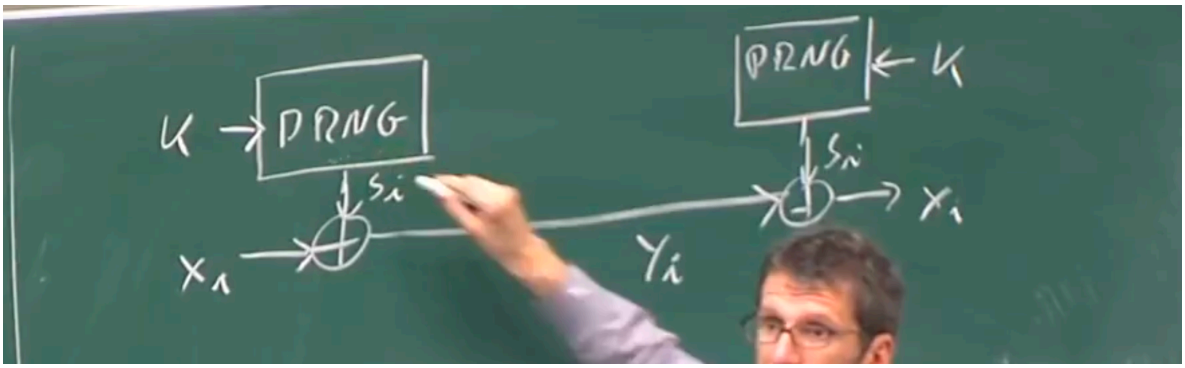
The above is extremely strong statement ie say 1000 bit key then brute force requires 2^1000, but say I have 2^1000 computers and each computer is checking one key then in one second I can broke. So infinite resources is a very huge term. So as per above requirement, 1000 key cipher is also not secure.

There are many ciphers which are practical secure but cannot be called perfect ciphers.

One Time Pad:-
Cipher where key stream bits are coming from truly random function and each key stream is used only once.

Big Draw Back :- Key is as long as the message. In practice it is very difficult to use one time pads. All commercial products don't use one time pad. In military products they might use one time pad. Instead of using one time pad(TRNG) they use PRNG.

Question is what is PRNG ?
Linear Congruential Generator (LCG) :-
Idea is to use Key stream Si generated from PRNG.
Practical Stream cipher works like :-



C random function is LCG.

1. S0 ==> Seed
2. Si+1 = A.Si + B mod m
3. Key comprise of A and B

This is a good random function if A,B and m are chosen wisely (like C function). C function has nice statistical properties but it is really bad cryptographically.

Attack :-
Oscar knows X1,X2,X3 ==> as these are file headers etc and are constants. Eg. say Alice is sending Excel file then first some bits will represent headers. For 31 bit "m" which is the case with C LCG function, we need first 93 bits. As Oscar is looking into the stream so he has Y1,Y2,Y3 so Now he can compute Si ? How as Si is just XOR so by XOR property he can get the Si.

A,B.Xi are Ceil (log2(m))

Now he knows S1,S2,S3 he can find A and B as 2 equations and 2 variables ie
S2 = A.S1 +B mod 31
S3 = A.S2 + B mod 31

It means if we use LCG's and if we observe 3 outputs we can completely compute the LCG. From A and B can get the entire key.