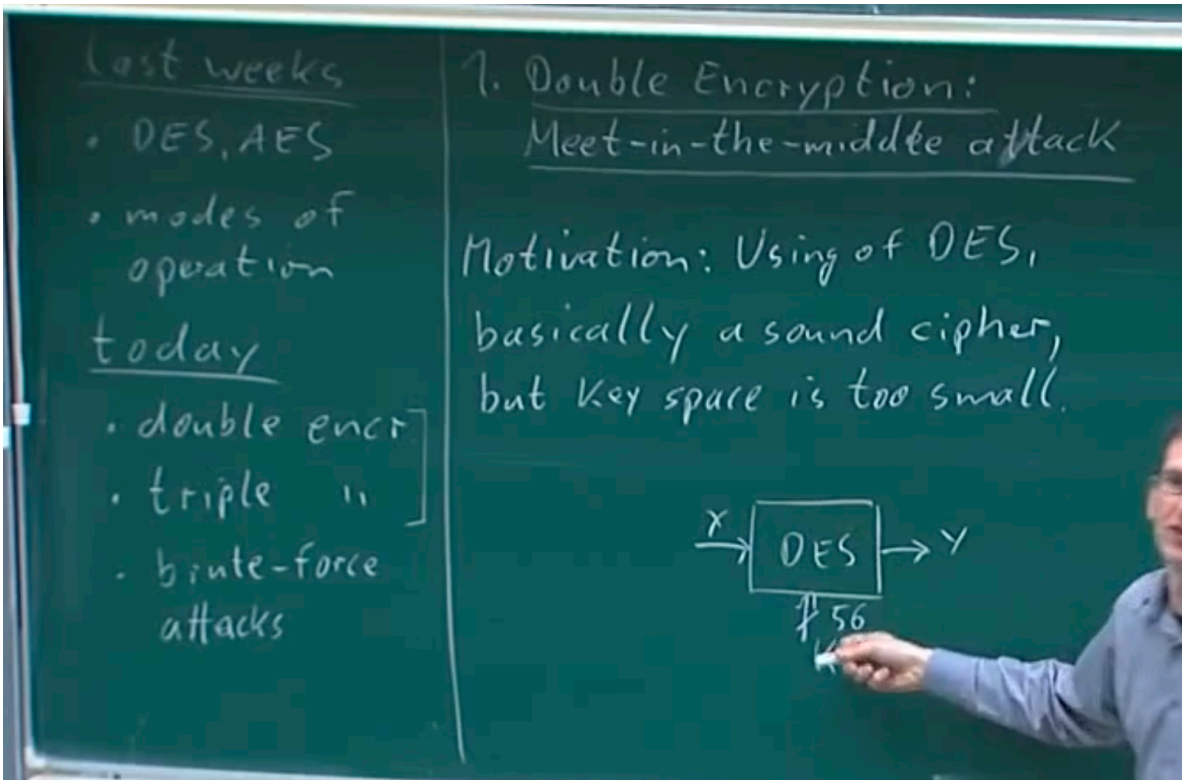
Lecture 10 Multiple encryption and brute force attack

Symmetric cryptography.

Two important subjects which we are discussing in this lectures :-
Multiple Encryption => Double and Triple Encryption
Brute Force => How to do bruteforce.

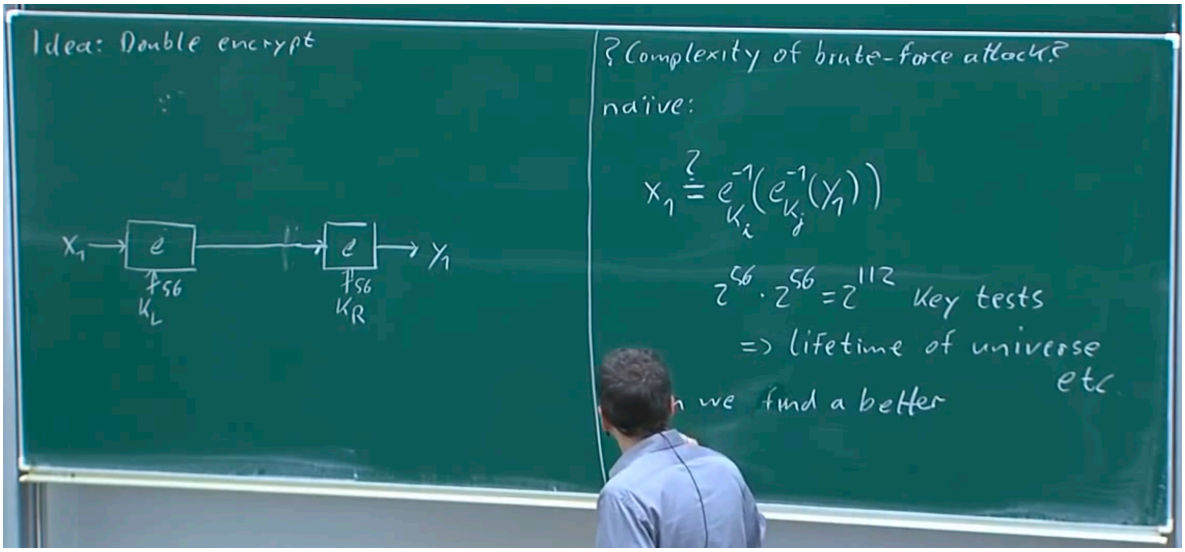
Double Encryption: is a bad idea and why ? due to the attack called **Meet in the Middle**.

Motivation: Using DES, basically a sound cipher, it is not broken but it has problems due to Key Space ie Key Space is too small.



DES is very nice and many hardware products use it like E-Passports but they are not using standard DES.

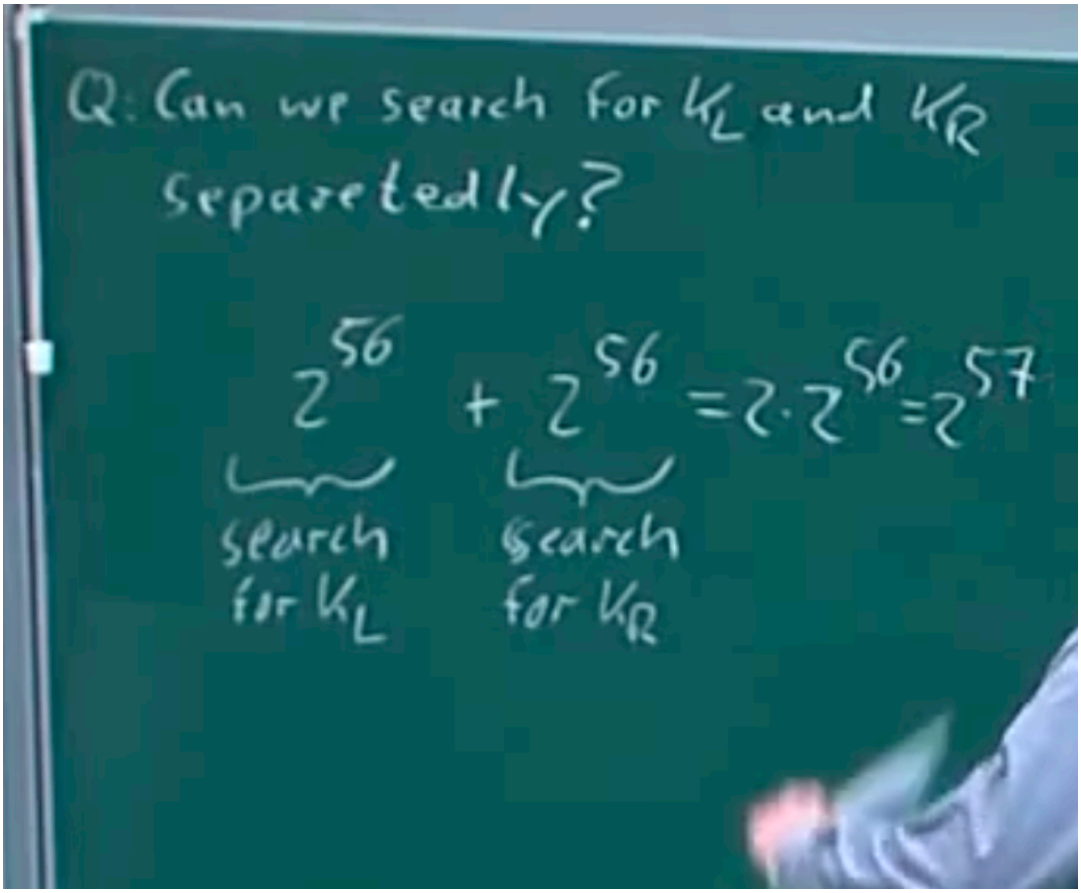
so it has nice properties then how can we make DES stronger ?
So solution is we will not encrypt with DES once but we encrypt it twice.



Here in above picture “e” is DES encryption so what is done is we encrypted X1 we get Y1′ and then we encrypted Y1` and got Y1. **Note** both the encryptions are on different key so we need 2 keys of size 56 bit.
Now finding complexity:
so say we fixed left key then we need to go though all the Key space of right key and we need to fix the left key for 2^56 times so total complexity is 2^112

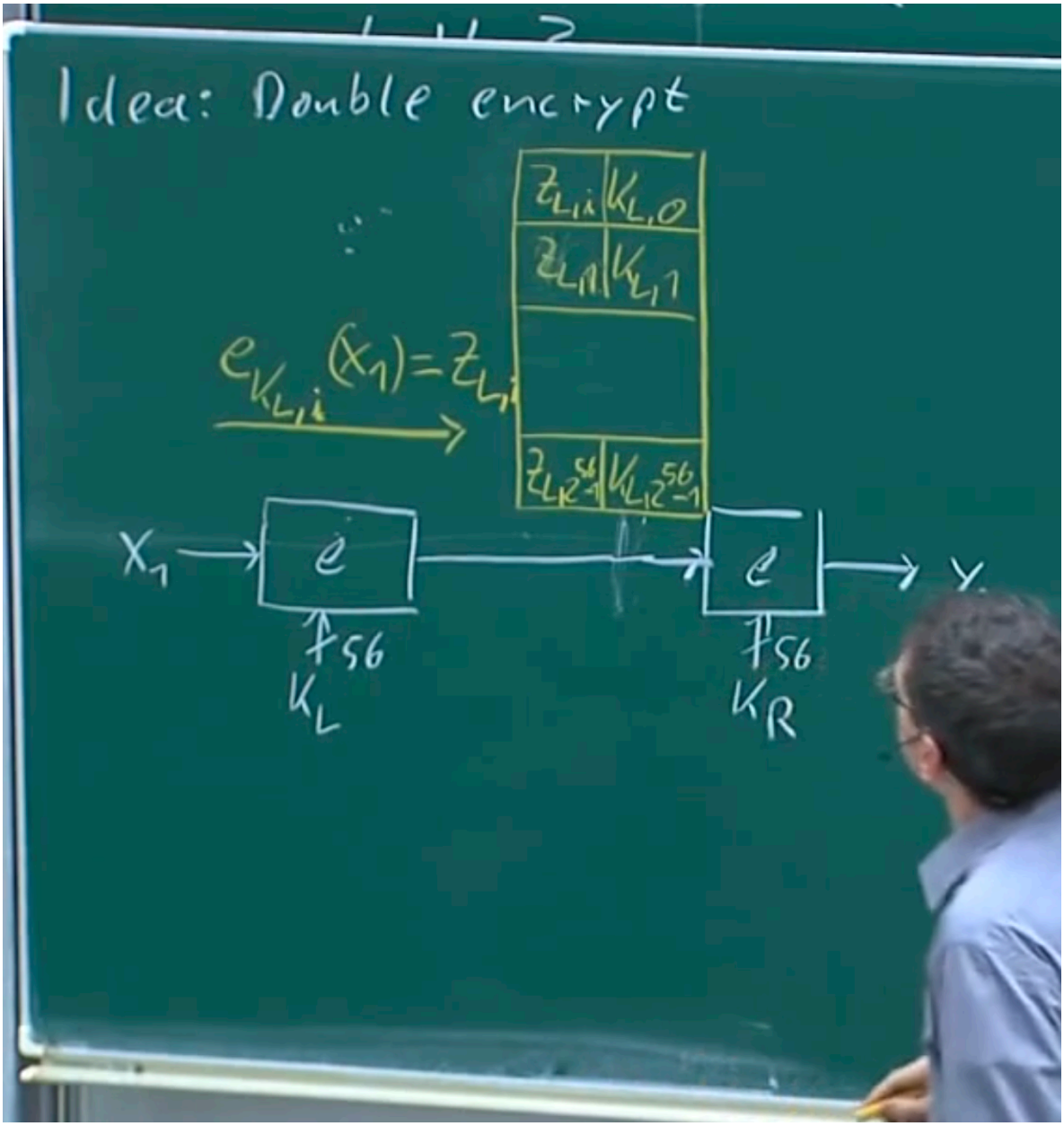
From above keyspace we might conclude that it is very strong and very tough to break but there is an attack called “Meet-In-The-Middle” which can be used to break it without bruteforcing entire key space of 2^112.

Meet-In-The-Middle:
Can we find a better attack?
Actual thought is Can we search for KL and KR separately ie we are encrypting twice and KL and KR are the 2 keys needed for 2 encryptions so is there a way to compute KL and KR separately ?



this attack is 2 phase attack ie 1st phase is Bruteforce on Left side encryption.

1st phase



So what we are doing here is we are having X1 and we are bruteforcing X1 with each Key possible and then storing in a big table(Kind of DP)

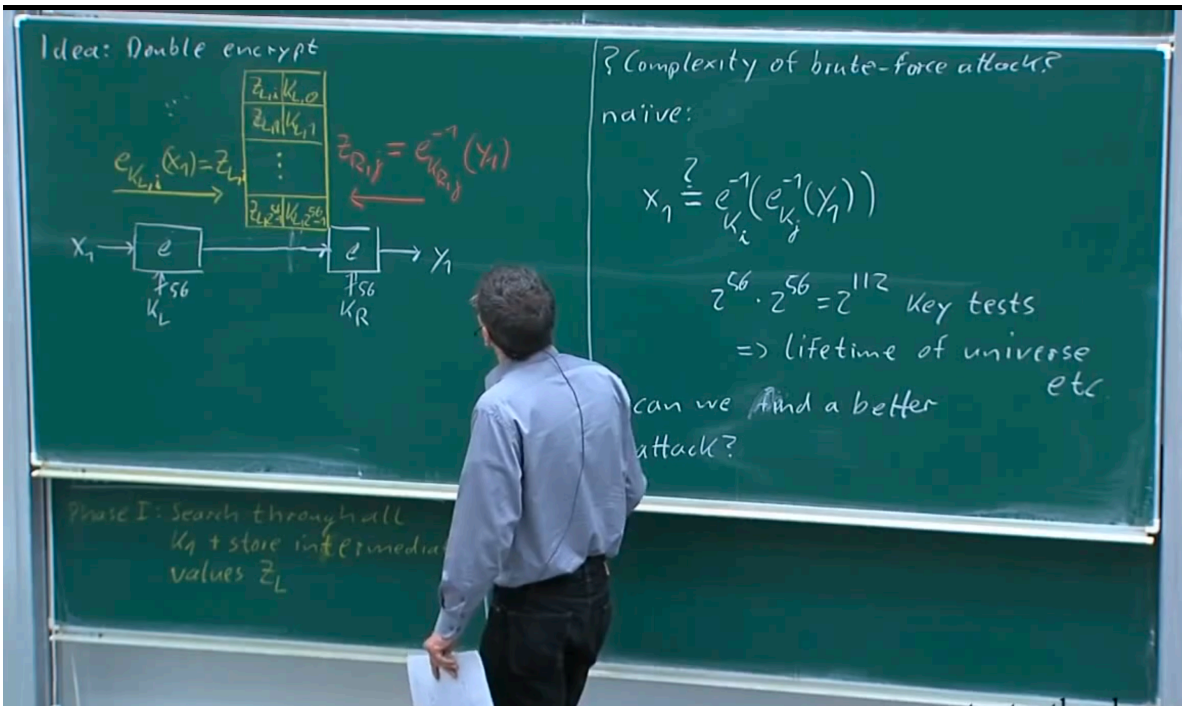
so Z in above table signifies the encrypted value of X and index i is the key (Starting from Key 0 to Key 2^56-1)

So phase one is iterate through all the keys and store values in table.

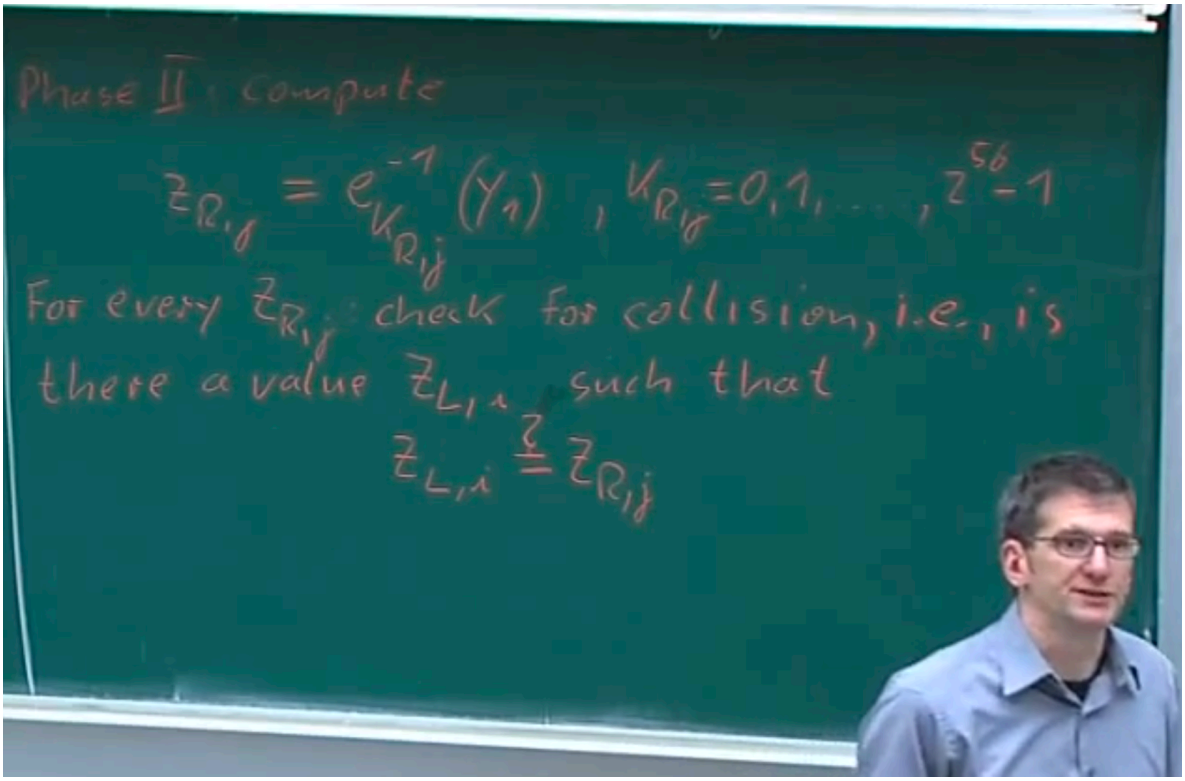
Complexity: As we are going through all the keys so complexity is 2 ^ 56 plus space complexity will be 2 ^ 56 which is quite large.

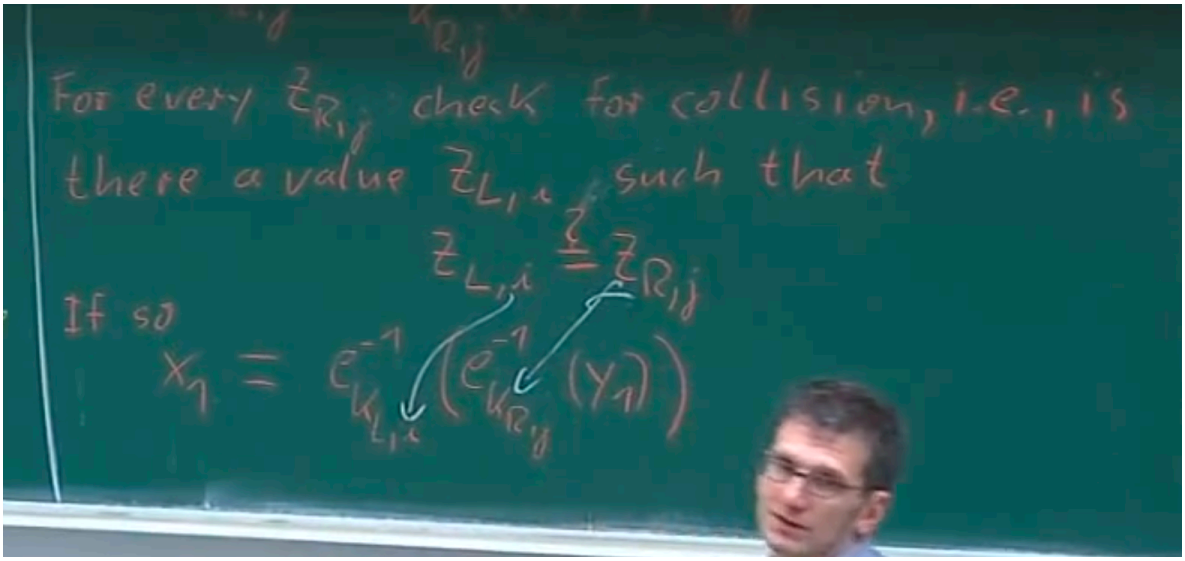
Phase 2:
As we see in phase 1 we are moving from left to right so in phase 2 we will be moving from right to left and that is the reason for this attack is called Meet-in-the-Middle
Also in phase one we are doing encryption and in phase two we will be doing

decryption due to the movement differences between the two phases.



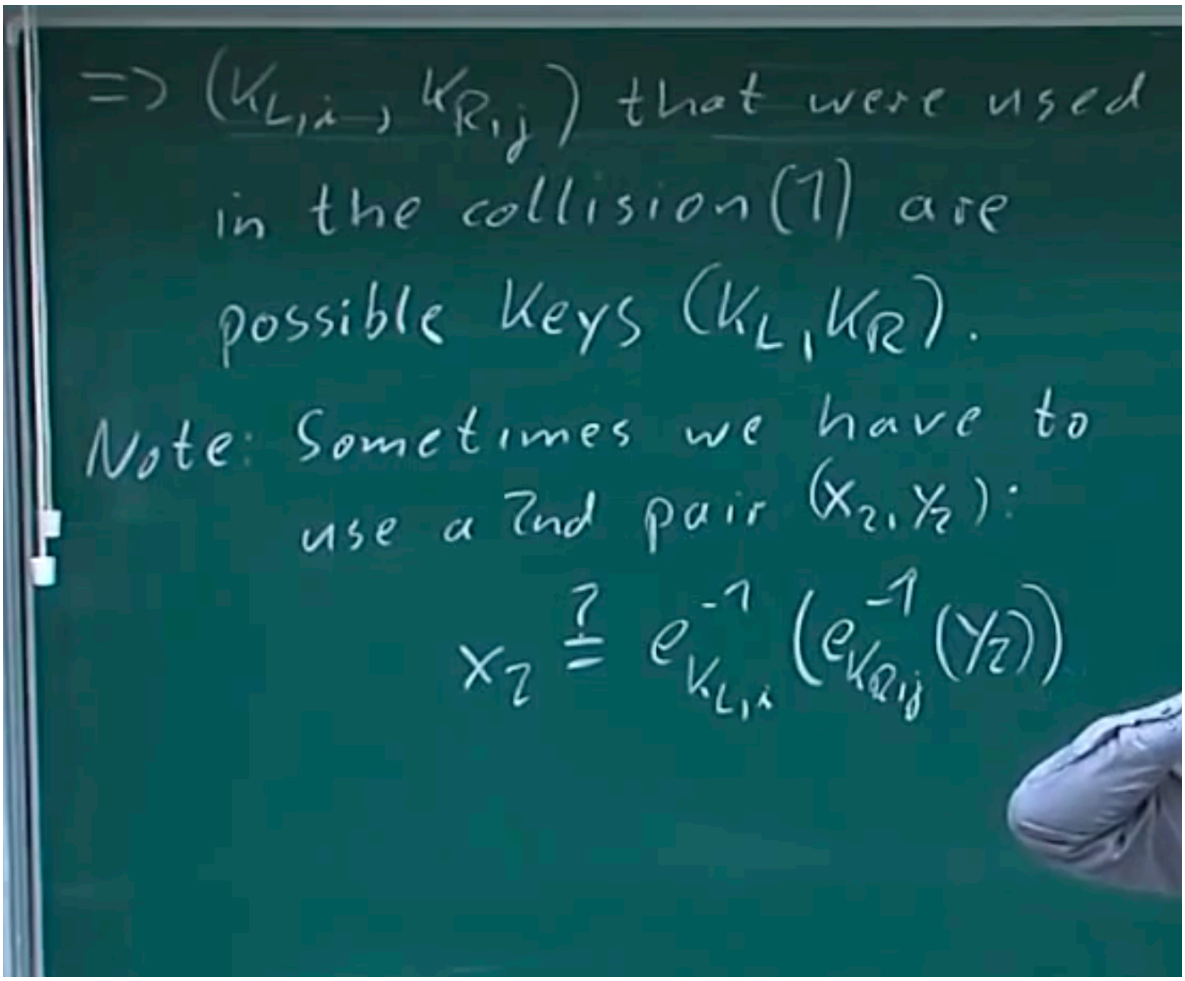
in phase 2 we decrypt using key from 0 to 2^56-1 and each decrypted value we will search in table and if we are able to find then our problem is solved.





the keys we get may or may not be the correct keys ie different keys can give same results in some of the cases and Paar is going to discuss about it later in the lecture.

In case of DES it is highly likely that found key is the actual key.
so for such cases we need to do a second block decryption.

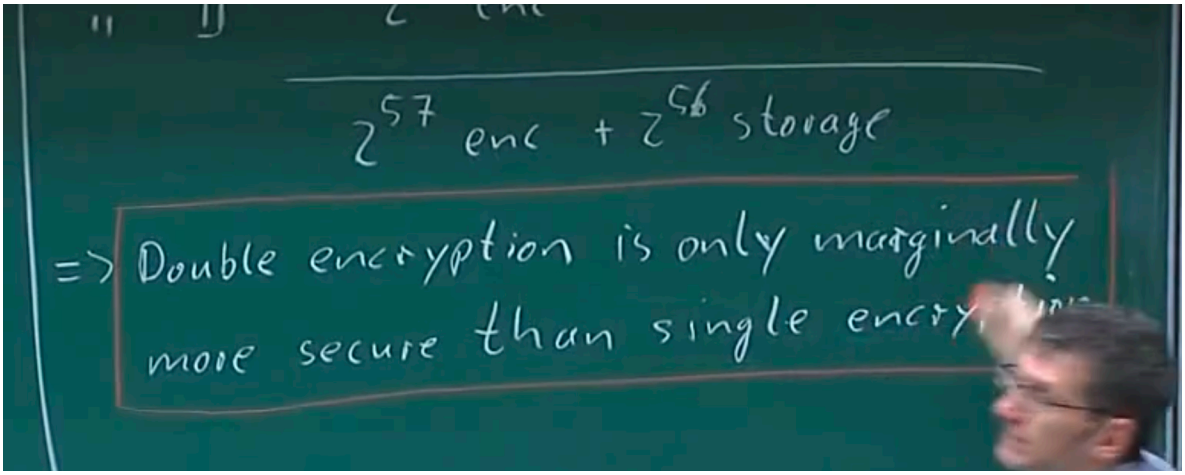


so incase second case fails then we continue phase 2 and do the same check

for other keys.

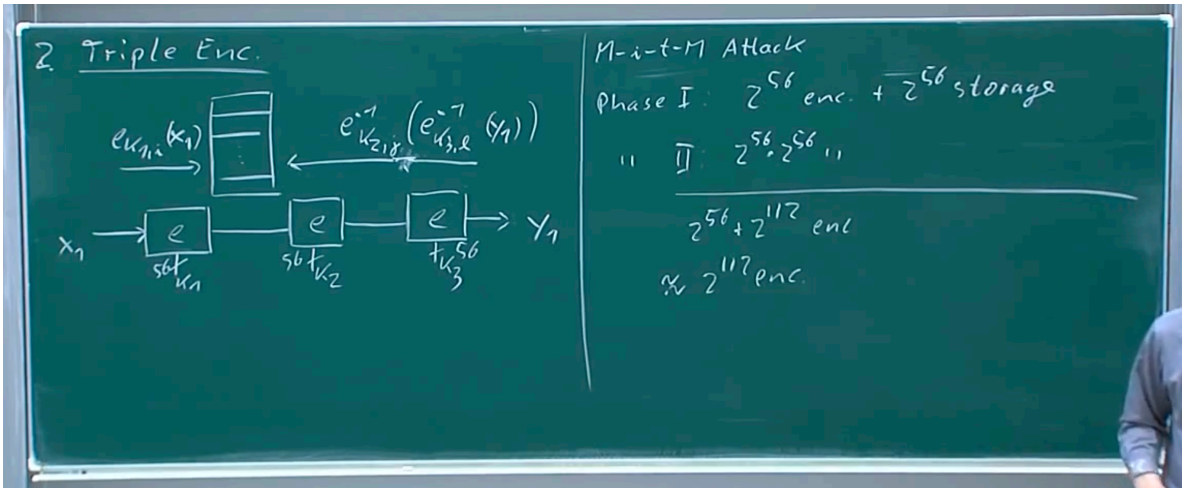
Total Complexity:
Phase 1. 2^56 enc + 2^56 storage
Phase 2: 2^56 dec
Total: 2^57 + 2^56 storage
Question why searching in table is not there ? is it constant ? Paar told that it is sorted so it will be there but marginal so approximately the above complexity. Also as 2^56 is more than log 2^56.

Conclusion: Double encryption is only marginally more secure than Single encryption.



Because of Meet-in-the-Middle attack, even if key size of Double encryption is double but we are not using double encryption and instead go with triple encryption.

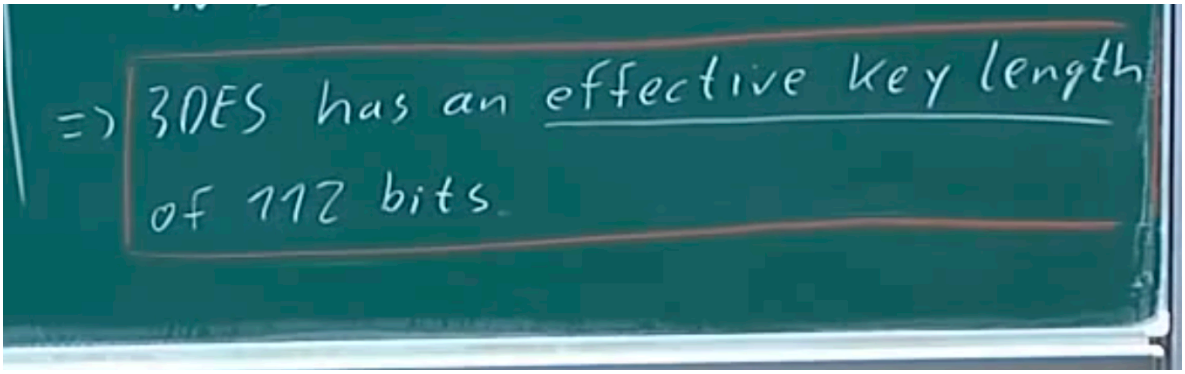
Triple Encryption:
we are encrypting it thrice with three different keys but there are some variants to it.



three times the workload but the security is roughly what we are expecting

from double encryption.
Naive attack should be 2^168 but triple encryption is max security as 2^112

3DES has an effective key length of 112 bits.

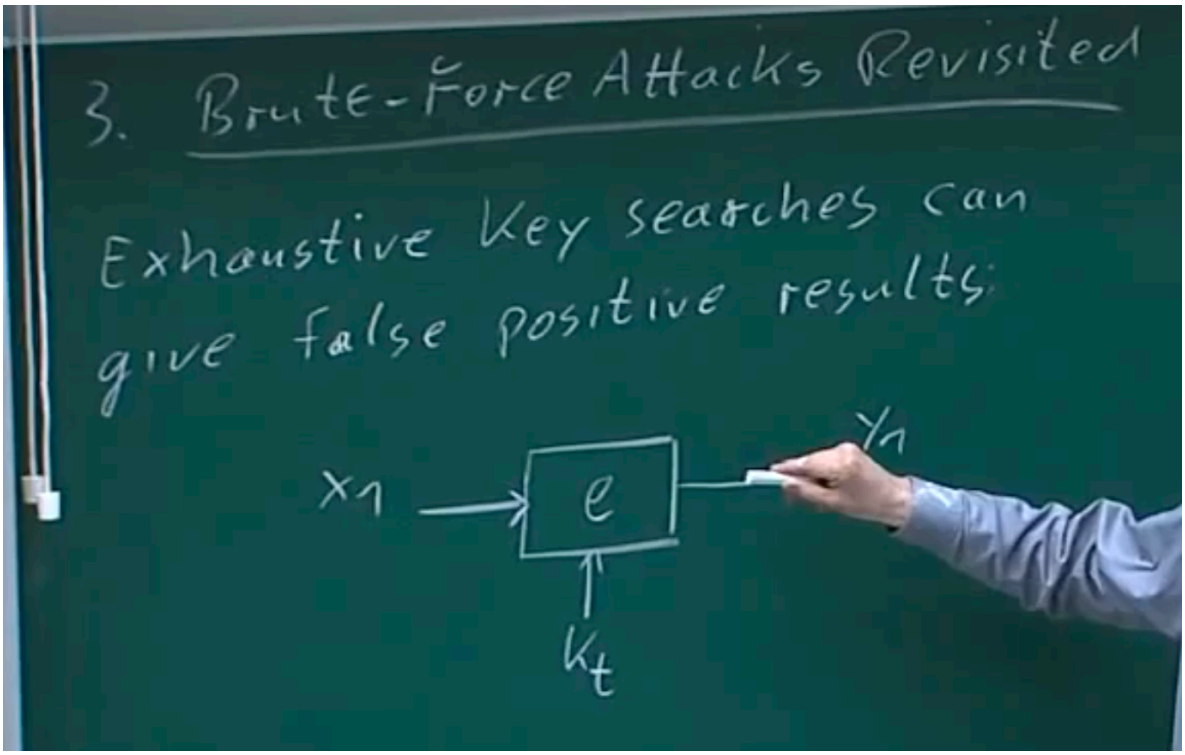


112 is perfectly secure as it is very large number.

Question how about encrypting 4 times ?
With 4 times we will jump 2^112 bits in phase 1 and same in phase 2 so not much of complexity but in case of 5 times encryption we can have phase 2 complexity of 2^168 so that will be very beneficial.

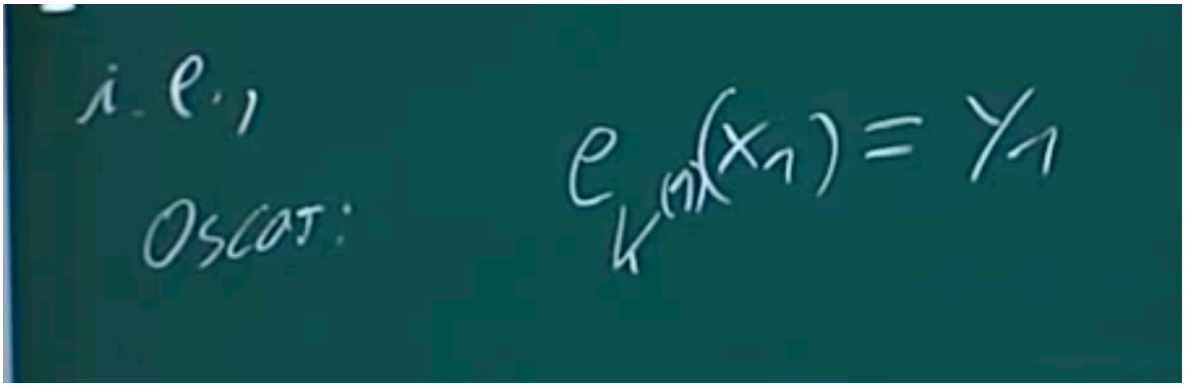
Why EDE ? Need to think more on this. 54

Brute Force attack revisited:
Exhaustive key searches can give false positive results.

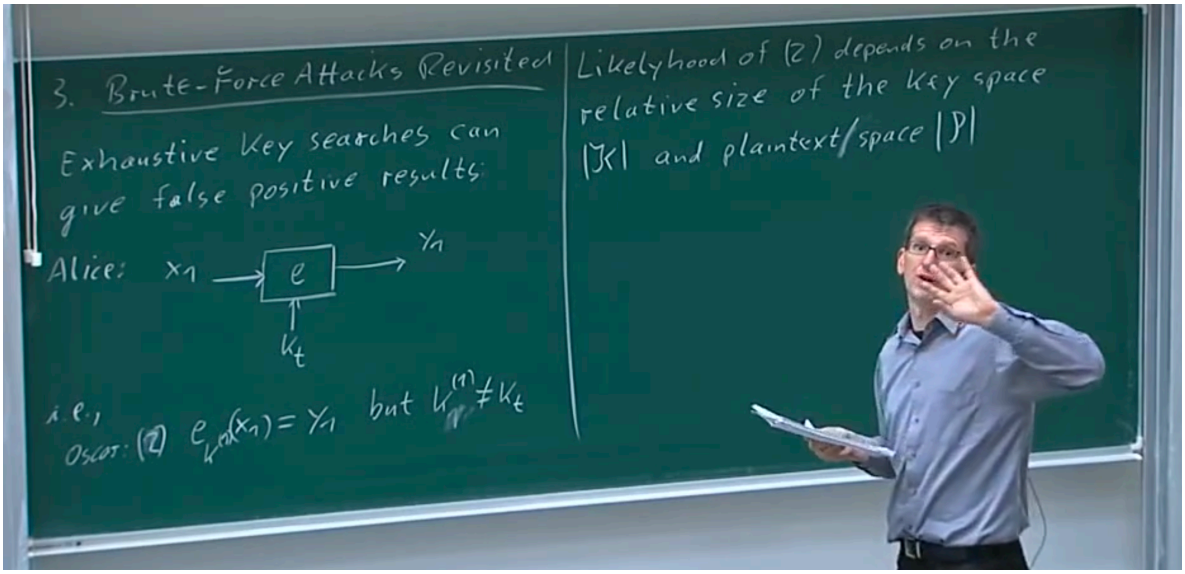


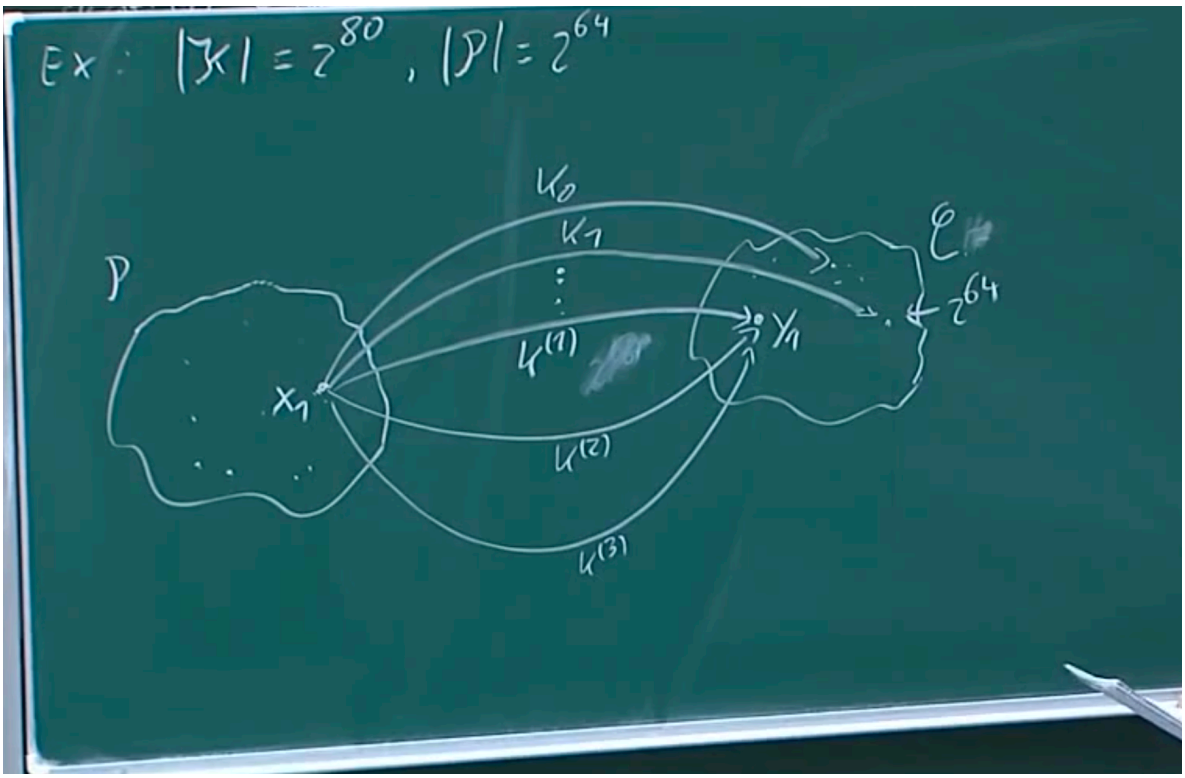
In above picture kt is the target key used by Alice and Bob to encrypt and decrypt

So what is the Weird thing ? what is the false positive ?

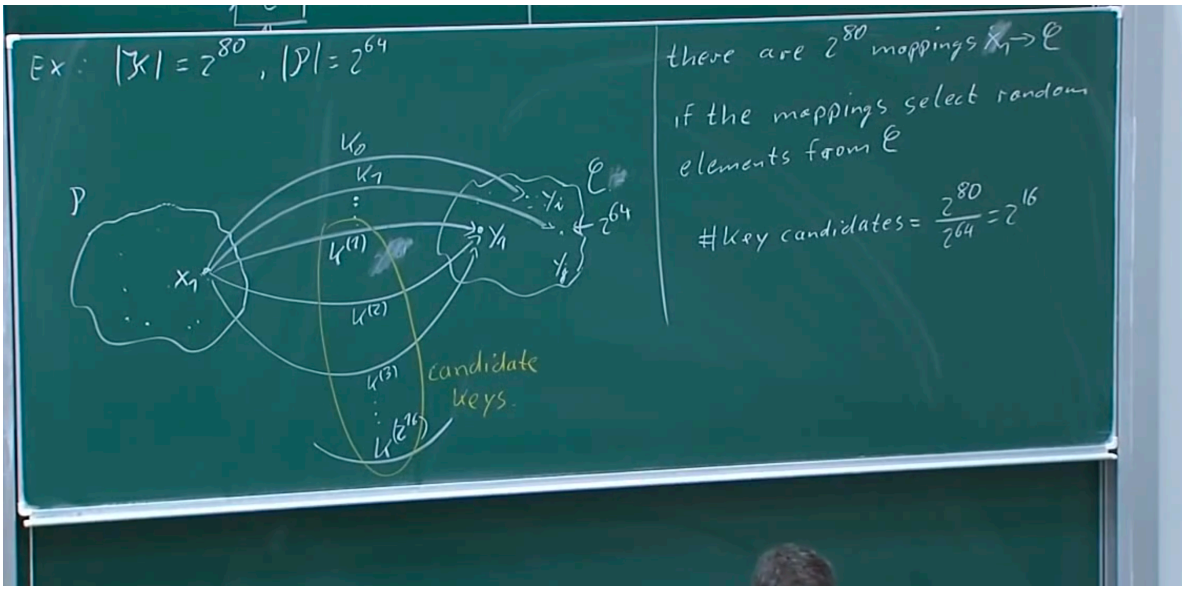


Oscar finds key k1 but that is not the target key ie it is not kt.





Little Background:
say Key space is 2^80 and Plain text space/Cipher text space is 2^64, now if we are trying to encrypt a plaintext will all the keys present in keyspace and we know key space is much much larger than the Cipher Text space/Plain Text space so each time we encrypt a plain text it will match with some cipher text space value and at some point we will exhaust the cipher text space and still we might have the keys left as keyspace is larger so multiple keys will point to a cipher text.

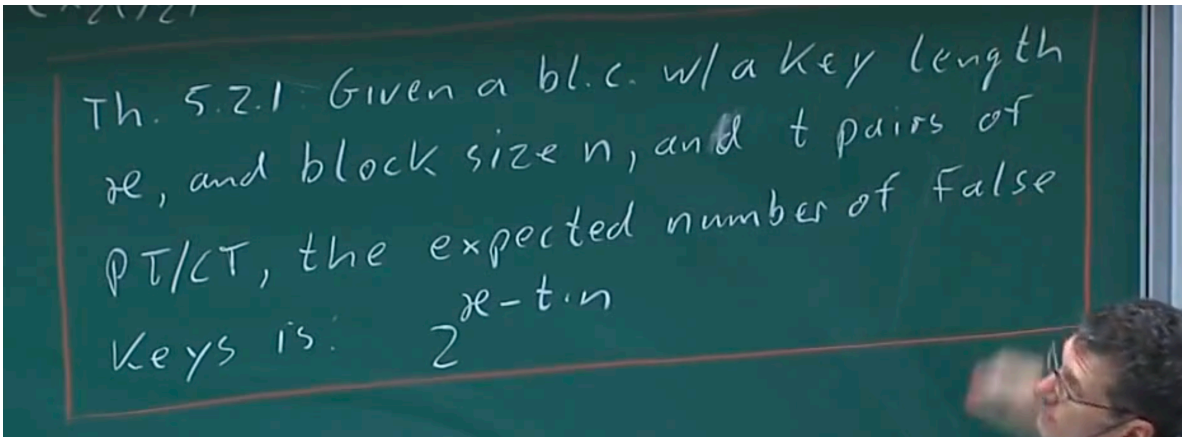


Finding Average keys mapping to the Y1 ie candidate keys:
there are total cipher texts generated are 2^80 which is same as the key space

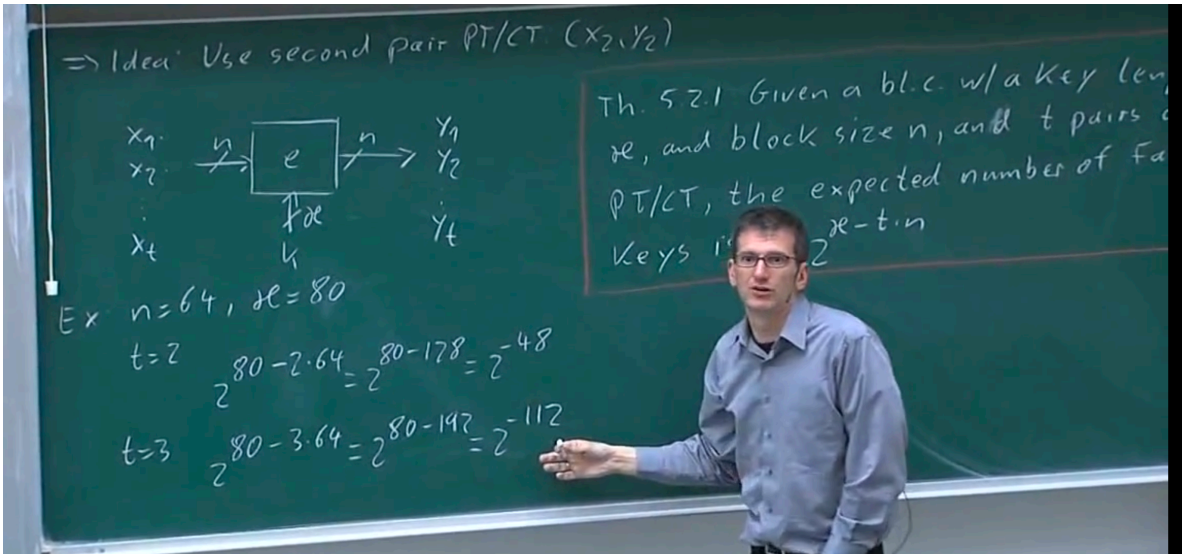
and total cipher text space is 2^64 so there will be cipher texts which are hit more than once and total hits more than once is 2^80 - 2^64 so if we take an average, so on an average each point in cipher text space is hit (2^80 - 2^64)/ 2^64 which is ~ 2^16

so candidate keys on an average is 2^16 which is quite huge so we need to find another chosen plaintext to reach at the target key.

Probability of finding Kt by chosen plain texts:



2^16 - 2^64/2^64 ~ 2^-48 for 2nd choosen plain text and with formula 2^(80 - 2*64) = 2^-48.



So from above image we can know that with even a very less number of chosen plain text we get a very high probability of finding right key or we have a very low probability of not finding target key.