Data encryption Standard

So far we the Feister cipher structure and with the Feister cipher structure, we were processing only one half of the data and the other half becomes, the decryption and for decryption we have to do only one thing, that was we have to use subkeys in reverse order and one of the Feister subkey algorithm is DES , it was the mostly widely used block cipher in the world and it was the standard for long, long time for security applications in the world. It was adopted in 1977 and it was called NBS and now its called NIST. And you can find the detail description of the algorithm in this publication. FIBS PUB 46 if you google it you will the standard publication of the DES algorithm. This algorithm encrypts 64 bit data ,that means 64 bit block size using 56 bit key. Actually its 64 bit ,but I will explain why this guy is using 56 bit. It has a widespread use , and it has been considerable controversy regarding its security. That means it has some secret design, within the algorithm and we are going to see the secret of that design.

DES history

If you look at the history of DES, it was developed in IBM which developed Lucifer cipher. This was the team that was lead by Feistel ,who developed the Feistel cipher. In this algorithm we use 64 bit data but with 128 bit keys. Today is 64 bit , but today's standard is 128 bits and this guy, when they developed this algorithm they developed with 128 bit key size. In that 128 bit key was very pretty secure time. Remember brute force attack where 128 bits take enormous amount of time. 64 or 128 bit does not take too much time ,its just going to take 10 minutes, in 1907's these guys came up with 128 bit key size, and and lot agencies did not like this idea as It was the recent ones, what they did is they redeveloped this algorithm and commercialized with inputs from NSA. What's the full form of NSA??.. national security design, in other words, government got involved in this algorithm .And they reduced this 128 to 56 bit keys. And in 1973 NBS issued requests for proposal for a national cipher standard and IBM submitted their revised Lucifer which eventually became DES

DES design controversy.

Although DES standard is public, there is lot of design about the inputs , NSA and other agencies basically. The basic thing is 56 bit choice instead of 128 bit saying why you are using 56 bit and why you are using 128 bit. In 1970' ,otherwise they agreed on 56 bit actually. And other criteria was design of the algorithm were classified. Design criteria was design of the s boxes were classified that we are using in the DES algorithm but S box design was classified. Algorithm is not standard but the algorithm is still classified. After some controversies, people try to find some back door in the algorithm, people tried to find, some bugs design or something, but so far, no body has found anything and that's why is called controversial, as nothing was proven, so that there is some kind of back door. And some kind of stuffs that will help NSA type of guys, to break the algorithm, ok. Atlast came up with some thing, alright. And then this has become in widely used in financial applications.

DES encryption

This is the actual algorithm, DES encryption .Remember DES, based on this we can understand the whole DES algorithm .first of all the block size which is 64 bits plaintext, which is going to go into the initial

permutation table is defined in DES standards and the table is this, which says 1st bit is going to be replaced with 58 bits. 2nd bit is going to be replaced with 50 bits, and so on. All those bits are going to be replaced with the other ones, that means, permutation, when I say replace then it means you are rearranging the data. If you look at the table do you see anything peculiar?..you guys don't see it??..good god you guys are having problems with your eyes I guess. Well I agree its difficult to see, but you can see something very clearly actually. The first 4 bits are even numbers and the second 4 bit numbers are all odd numbers, its like ,rearranging the even bits on the one side and rearranging the odd number bits plaintext on the other side. Its will be like odd number group will be on the lets say left half of the data and even number group on the right half of the data. In other words I am grouping the even number grouping of the plain text and odd number group of the plaintext. In other word we are using simple permutation cipher, as we saw this table is simple cipher, as we saw in the transposition table and the result of the permutation table will be going to round one. And output of round one will be going to round 2 and like this I will be having 16 rounds. So I have total 16 rounds. {question from a student}. With initial permutation we are grouping the even number bits from the permutation ,and making it left half of the data and odd number bits on the right half of the data. If you look at the, if you see the first 4 rows you are going to see the replacement of even numbers with the actual plaintext. For example first bit is replaced wit 58 bits, second bit with 48 bits and third bit with 42 or 46 something like that and last 4 rows are odd numbers, if you look at the table , the total entries are 64 entries ,that means you are actually grouping the odd number bits somewhere and even number bits somewhere, I am going to go 16 rounds and in each round I am going to use different subkeys, and I need 16 different subkeys and those subkeys are generated using 64 bit keys. Pay attention to that as we said 56 bits, but actually, if that bit is in the character form then it will be 64 bits and that 64 bits, and that will go to another table called permuted choice, lets have a look at it.

Permutation table for DES.

Permuted choice 1 is this one, and if you look at that one I am actually removing the 8 bit ,16 bit and the multiples of 8 bits from the plaintext. So those things are removed from the input key, I have 64 bits as input key, right?.. so how many 8 bits I have?..8 right?..so 8 of the 8bits are going to be removed, and why am I removing those guys??.. and why do I need 56??.. and we are going to have such kind of attitude for such kind of algorithms are concerned , but the thing is there is a reason why, for example as I said if your key is character based key, what is the representation of the characters, if you are using the ascii , how many bits you are using?..8 bits and how many bits are actually representing the character?.. 7 right?..and what is the 8 bit, it is the parity bit and we are using it for the error control. So what we are doing is we are going to remove parity bit from the algorithm that is the idea basically. If your key is character based. But if you character is binary input, the actual key size would be 56 bits ,so you are going to enter 56 , 0's and 1's , and if you are using some kind of character and maximum character that you can use is 8 characters , and each characters is 8 bits and you are going to use 64 bits with the 8 characters. If that is the case then we are going to remove the parity bits and resolve the binary keys ,that is my 56 bit key. All right?.. the algorithm is 56 bit key but you can enter until 64 bit key. And that key is going to go to left table and that will be going to permuted choice table .that I am going to explain to you guys later and the result is going to be first round sub key or first round key, first

key ,round key are all the same thing. So what about the left circular shift, we don't know right now , I am going to explain that how many bits we are going to shift. The thing here is key will be shifted and it will go to permuted choice 2 and the result is subkey. And the output of the first table will be going to another circular table, same table to get the subkey. And we are going to generate like this until we generate 16 subkeys. {question from the student}. That is the design, they designed it like that. They didn't design or divide by 4 bits, that is simplified structure, remember like in feister structure where plaintext was split in to two halves. If the plaintext is 64 bits then its going to be 32 bits left and 32 bits right, that I will explain when we go in to the round one. {question from a student}. The number of rounds as you might remember in feister structure one of the criteria was number of rounds. If you have more rounds you are increasing the security of the algorithm, but you are decreasing the, speed or performance of the algorithm , but in this algorithm, they found the balance, we will see another algorithm and it is called, avalanche effect. Every algorithm will be built under certain criteria and one of the criteria is called avalanche effect. What we doing in this is we are going to change the 1 bit from the plaintext block and we are going to see the changes of that one bit in the ciphertext block. If the ciphertext block number of bits per half is changing based on 1 bit ,that means that the algorithm is pretty secure. And 64 bits DES gives us more than 32 bits changes in the cipher text against one bit change in the plaintext and again I am going to explain those things. If you change one bit in the round key ,you are going the see the changes later, both of them has to create more than half of the different bits, in other words ,they must change more than half of the bits to make it change, to this 16 rounds, we do get that one { question from a student}. We are not look at decryption now, what we are seeing is what will be the effect of one bit change in the ciphertext block, in plain text and also in key, if I am able to new cipher text when compared to others in the new, the number of bits is more than the previous one , that means that this algorithm is pretty, secure algorithm and we will see, how many bits the algorithm is generating, that means DES is generating. So I am going to do this 16 rounds and eventually I am going to swap the left half and the right half. And then I am going to go to the inverse initial permutation and in inverse initial permutation I am going to have 64 bit cipher text. So this is the structure of DES encryption, right now we don't know the details of left rounds and right rounds. So we will take the next step

Single round of DES algorithm

So lets have a look at the actual one round structure. Remember in feister cipher, plaintext are split into two blocks, 32 bit left half and 32 bit right half ,if this is the first round then this is going to be the first left half and the first right half. If this is the third round then this is going to be the output of the second round . That's why it says Li-1 and Ri-1 ,the output of the previous block. So I have two halves as we know left half as we know directly will be XORed with the output of F function ,remember this dashed light area is the F function .Right half directly goes to the table of the F function, if we see expansion table, it will expand, 32 bit into 48 bits and that's why its going to improve two columns into 32 bits. And these columns are the repetition of 1st column and the 4th column and you are going to shift the column like this, one decimal column to up. So the one is going to come and be like this and rest is going to be like this, in the last column and the 4th column Is the repetition of here and with one number shift , so total how many bits you have??..48, this shows the bit places. 22nd bit, 13th bit and so on. That is the

expansion table and I need 48 bit from the expansion table and why do I need 48 bit?..subkey sizes in DES algorithm is 48 bits. And is not 56 ,56 is your actual key and what I am going to do with that actual key is , I am going to shift that into two parts, 28 bit left and 28 bit right. And this 28 bit left is going to shift in the round and the right half is going to be shift it again left, might be circular shift again , and the result is going to go to another table, which is called permuted choice two, and with this permuted choice 2 table I am going to get 48 bit and this way, its output of the expansion table can be XORed with the subkey ,so the subkey becomes the function right now and affect the output of the parameter, and then we are going to go to the substitution, in DES we are going to have 8 substitution tables, and those tables are these DES S boxes, as you can see there are 8 tables, and design of these tables are, we don't know why it is number 11 here and we don't know why this number here is 13 over here, the design criteria is still classified and that's why lot of people have little bit problem with DES, because you don't whats going on in here, you don't know how these guys design these things, you don't know whether it has any weak points or back points, back doors or something like that so that they can use that stuffs and break your algorithm ,so far no body has found any back door in the X boxes . and the uses of these DES S boxes are little bit different ,we are going to come to that point in a minute, ok . I do XOR the expansion table with the subkey, and the result is going to go to the S boxes and its going to pass through 8 S boxes, this is the substitution part of the algorithm ,remember we said , diffusion and confusion, it will come with this substitution and transposition and all these tables are transposition and this part is substitution and the result is going to be 48 bit input, and 32 bit out put , so S box outputs are 32 bits, and that will go to another permutation table, 32 bit output will be the result and that's going to be your right half of the data, for the next round, this is going to be the right half of the data, previous right half is going to become the current left half, and that is the sloppy part and we are going to do how many times, 16 times, and in each round we don't change the structure of S boxes or we don't change the structure of these tables, whatever the tables they design we are using the same tables all over in each round.