Hardware Security
Prof. Debdeep Mukhopadhyay
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

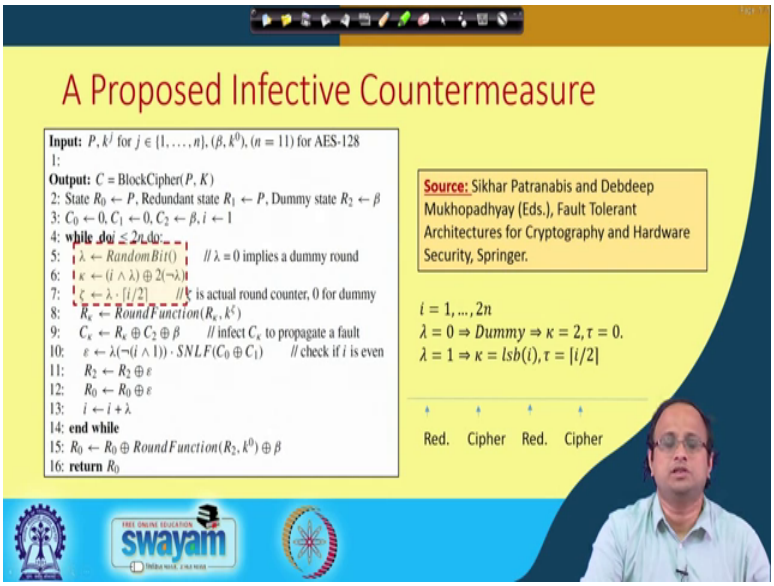
Lecture – 53
Infective Countermeasures for DFA (Contd.)

So, welcome to this class on Hardware Security. So, we shall be continuing our discussions on Infective Countermeasures for Differential Fault Attacks.

(Refer Slide Time: 00:26)

So, in particular today, we shall be discussing about few attacks. Like we already started discussing about an attack against a potential infective countermeasure using a constant fault model. So, we shall be trying to see how we can extend this to you know like even attack the entire state matrix which we did not or could not attack in the last days discussion. We shall be discussing about how to perform an attack using a random fault model which makes the attack more generic. And then finally, we shall be discussing about some modifications to the countermeasure we should be able to protect against these kind of attacks.

(Refer Slide Time: 01:04)

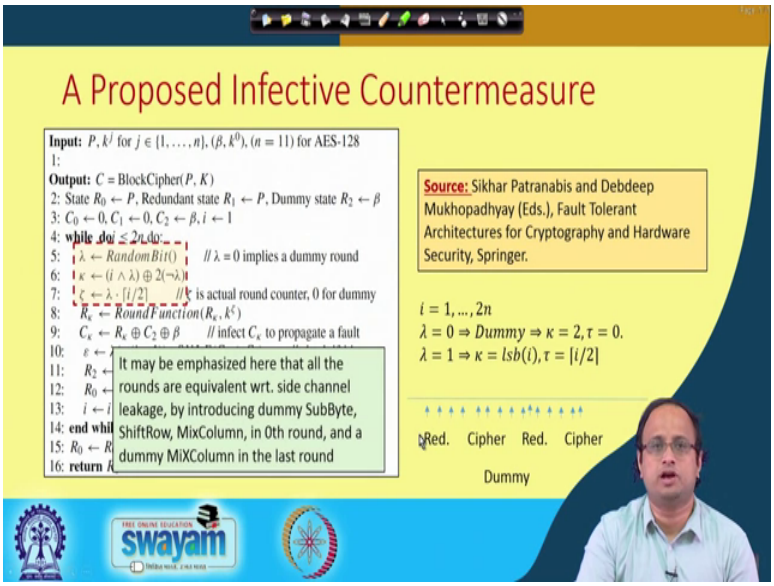


So, to start with this is a quick reminder of the infective countermeasure. The infective countermeasure essentially has got 3 rounds. Apart from the cipher round there is a redundant round and there is also a dummy round which comes in spuriously. So, the idea is that the redundant, it is a kind of repetition of sequences of redundant ciphered redundant cipher and so on, but you know like there are also intermediate dummy rounds which are indistinguishable from the redundant and the cipher down. And therefore, it becomes very difficult for the attacker to know like when to induce the fault because the dummy round kind of confuses the attacker.

At the same time because of the way the algorithm of the countermeasure has been developed right, if there is a fault either in the redundant round or in the cipher round or even in the dummy round, then basically in fix the following operations. And finally, right it results in a cipher text which essentially cannot be exploited or which is supposed to be not exploited by the adversary or at least it makes it more difficult for the adversary to exploit.

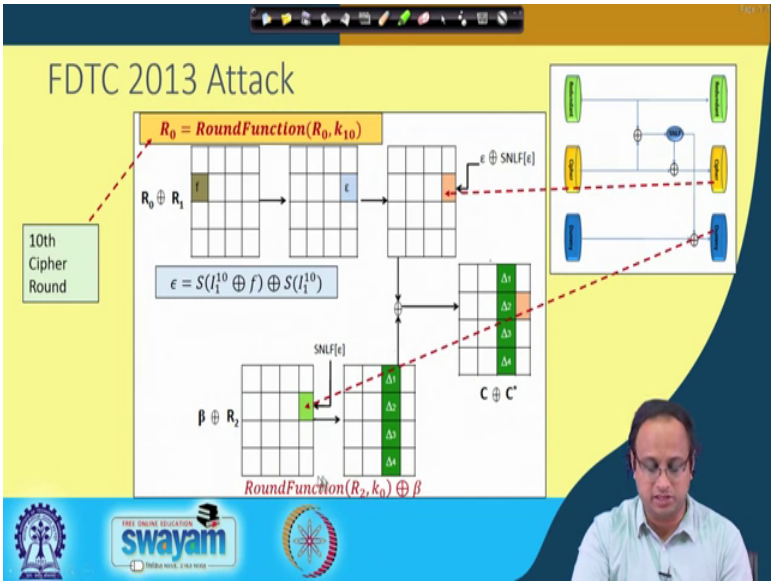
So, the whole idea is that it tries to make the differential of the fault or the differential of the output faulty cipher text as uncorrelated to the secret key as possible to rule out a possibility of a differential fault analysis.

(Refer Slide Time: 02:22)



So, therefore, right I mean of course, we have seen few points in last class like these are the rounds and the ciphers and the redundant cipher, the actual cipher round and also the dummy rounds are constructed in a way so, that they are equivalent. So, that they cannot be distinguished by also other side channel techniques like power or electromagnetics so, that it becomes difficult for the attacker to induce a fault at an opportune location.

(Refer Slide Time: 02:47)



So, here is a quick reminder of the attack that we saw in the last class. And we saw that if there is a fault. For example, here the fault is at this location or at this white of the state

matrix. And this fault is at the input of the tenth cipher round then what happens is because of this fault right, the fault essentially gets affected here. So, if and it basically affects R 0. So, R 0 essentially stands for the actual computation or the cipher round.

Now, because of this fault, the fault essentially means that the fault basically propagates of through the tenth round. And therefore, it gets shifted of course, after getting modified by an S box and therefore, right if you if we assume that one of this the correct state is I 1 10. So, I 1 10 stands for example, the first byte of the tenth round input. For example, the tenth round input is say denoted as you know like this is the zero th byte, is the first byte, this is a second byte and this is a third byte and so on as we have seen in previous classes.

So, if I take one of the states as I 1 10; that means, the correct state as I 1 10 then because of the fault this gets affected as I 1 10 and becomes I 1 10 XOR with f. So now, if I consider both the you know like the correct operation that is you know like the correct execution path and also the faulted execution path. Then the differential at the output of the S box would be basically captured by say epsilon where epsilon stands for the XOR of S of I 1 10 XOR with f XOR with S I 1 10.

But at the same time always remember that there is also a shift row. And because of the shift rows what happens is that this particular byte gets shifted to this location. And now there is an interesting thing which happens. So, this is essentially a usual encryption process for AES. Now when we consider you know like that the countermeasure then there is an additional thing which happens accordingly.

So, what happens here is that because of you know like the presence of this function which is called as SNLF, we what we measured is the XOR of the redundant and the cipher round. And we know that that XOR right now stands as epsilon because of the rest of the bytes are same in both cases of R 0 as well as in case of R 1 that is in both cases of redundant round and cipher round, remember that the tenth round redundant round has already taken place ok.

So, the sequence is redundant cipher, redundant cipher and so on. And so, if you are inducing a fault at the, you know like at the tenth round cipher then; that means, of the tenth round the redundant operation has already taken place without any error. So, therefore, my reference point is correct; that means my R 1 stands for my reference

point. And what essentially we are basically comparing with in R 0 is basically the cipher, but with the fault. And the fault in this case is the epsilon. So now, what happens is because of this XOR I get an input of epsilon and that gets converted into SNLF epsilon as we know that SNLF function if there is a non-zero value, then it also produce a non-zero random output or random looking output. And that essentially pollutes both the cipher as well as the dummy round.

So, if I consider for example, the cipher right then what happens is, this gets further polluted with epsilon it was already there it gets further contaminated with SNLF epsilon. And therefore, this disturbance becomes epsilon XOR with SNLF epsilon. Now, if you know like look at the dummy round infection then this SNLF epsilon also affects the dummy round. And you note that the dummy round was initialized at some beta value suppose you know like you have got beta 0 beta one and so on this is beta.

But now because of this infection which happens in the dummy round as shown by this equation. So, this particular you know like diffusion is basically that what basically do is you apply the round function on R 2 comma k 0 and XOR with these with beta. So, note that if R 2 is beta then this value also computes to beta and therefore, you get beta XOR with beta and that is 0, but now because of the fault all you know like 15 bytes have got the same bytes as that of beta, but this byte is different. And this byte right is kind of contaminated with again this error which stands for SNLF epsilon. And therefore, right we basically have this additional error which we need to kind of accommodate.

Now, remember the round function is nothing about an usual AES operation. And therefore, right what will happen is that this particular byte will get distributed into this column, but again there will be a further shift and we know that the shift which happens right because of AES is a round computation is a shift for like in the second row. There is a shift to the left by one position. And therefore, by this column comes here and then there is a mixed column which comes in the round function and that spreads this disturbance to this entire column. And therefore, this column becomes something like delta 1, delta 2, delta 3 and delta 4.

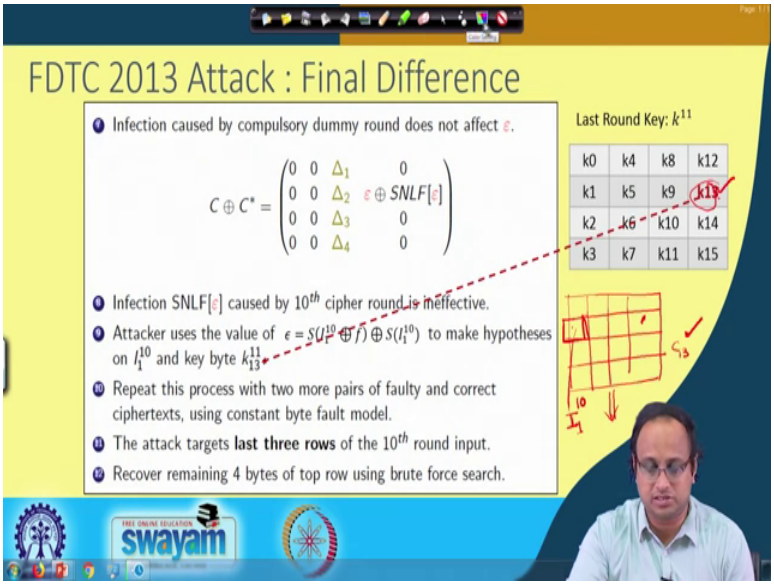
And now this particular disturbance right affects your actual cipher round. So, therefore, if you go back and see the equation, in the infective countermeasure. Then this is the line that I am talking about you see that this particular XOR, is XOR with R 0 which is your

actual cipher, which is essentially returned as your computed cipher. And therefore, write what will happen over here is that because of this like the original cipher will also get further contaminated. And you will have this particular disturbance as a result.

Now, the interesting thing is that because of the shift, this particular pollution affects the third column sk. Whereas, this part this disturbance which is stands which stands for epsilon XOR SNLF epsilon gets exposed to the adversary.

So, if the adversary basically measures the XOR of the faulty output which is essentially nothing but the faulty R 0 value. And XOR it with a correct computation and then we can basically get this differential and therefore, right he can easily know what is the value of epsilon XOR, of SNLF epsilon, where epsilon stands for this value.

(Refer Slide Time: 08:43)



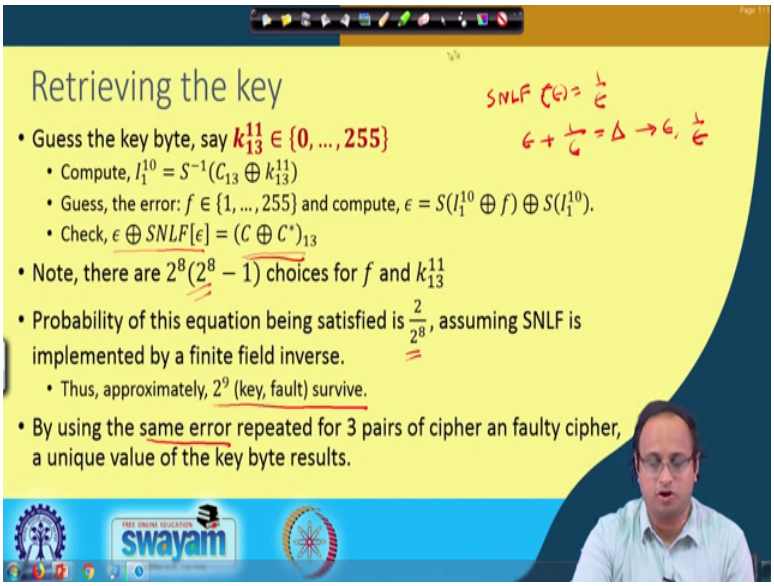
So now, how can you know like so, this basically leads to an attack and because of this right essentially what the attacker will try to do is, the attacker will now. And this attacker would work pretty much on all the last three rows of AES because in all of them right because of the shift this particular disturbance will get exposed to the adversary. I will emphasize later on that if this fault is happening you know like at the first row, then this will not happen and then the because you know like; because there is no shift in the first row of the AES matrix in when we are considering the AES round.

So now, what will happen is essentially here shown here. For example, you know like as I said that epsilon is related with you know like S of I 1 10 XOR with f, XOR with S I I 1 10. What I would try to do is basically, I will try to get the value of this I 1 10. Remember this I 1 10 will basically in turn give me a value of this key byte. Because this key byte will basically get kind of processed with this particular will essentially you know like will get leaked because of this fault.

Of course, right in order to get the actual value of the key you have to probably take more pairs of faulty and correct cipher texts. And again using a constant byte fault model we can essentially retrieve the corresponding key.

Now, why constant fault model or how will they attack exactly work is illustrated subsequent to this.

(Refer Slide Time: 10:05)



So, here we see the process of retrieving the key. So, what we do is that we basically try to guess this key byte. So, remember that this key byte like this particular key byte. For example, k 13 can take 0 to 255; that means 256 possible values. So, if I do that then what I will now do is I will kind of invert and compute S inverse. And XOR C 13 remember that C 13 would stand for a specific location in the cipher which get XOR with this key byte and then compute the inverse of that.

So, this essentially is nothing but I 1 10. So, one of the reasons right why there is a relationship between this key byte and the I 1 10 position is quite straightforward. Because you can observe that you know like that the key byte that is there. Like if you consider the state matrix then I am talking about this position in my essentially is referred as you know like I 1 and 10 essentially stands for the round.

So, one of the reasons why we use this along with this key byte is because of the shift rows operation. And you can observe that if you consider the shift row subsequent to this of course, there is an S box also. Then this particular byte you will get left shifted and therefore, this will come to this location. And again therefore, you know like this will get basically combined with this k 13 ok and with result in your cipher text C 13.

So, if you basically would like to work backward then; that means, that if I would combine a C 13 with k 13 compute the S inverse to get this particular byte which stands for which stands for I 1 10 ok. So, therefore, with this you know like perspective one can easily understand that this is correct that I 1 10 is equal to S inverse of C 13 xor with k 13 11.

So now, what we do is we basically guess this error. Note that the error can be anything except 0 because if it is a 0 then it is not an error. So, there are 255 possibilities. And what we do is we basically, we compute epsilon which stands for S of I 1 10 XOR with f XOR with S of I 1 10. So, epsilon as we know right is equal is equal to S of I 1 10 XOR with f XOR with S of I 1 10.

So now what we do is basically right I mean we already can compute this because we have basically estimated the value of I 1 10. We basically make a guess for this error and let me compute epsilon, but now we also have this value of epsilon XOR SNLF epsilon which has been exposed to the adversary because of the attack that I just mentioned and because of the observation that I just mentioned. Therefore, right if you basically get this value then what you can do is basically, you can check whether indeed you know like the XOR of C and C 13 because you can take the XOR at the position in the cipher text and you can check whether this computed value of epsilon satisfies this equation or not.

So, what is the chance of a particular thing getting you know like satisfying this equation. I will come to that, but first observe that the possible number of guesses that you are making is k 13 there are 256 choices that is 2 power of 8 and there are 2 power

of 8 minus 1 choices for f. So, they have a total there are around 2 to the power of 8 into to the power of 8 possible choices for the topple of f and k 13 11. And what is the probability that a random equation will satisfy this equation?

So, suppose I make a make an assumption which is in there also stated in the paper that SNLF epsilon stands for you know like the finite field inverse. And if that be so then there are approximately you know like two solutions which will probably satisfy because you can observe that the equation is like epsilon plus 1 by epsilon. So; that means, if epsilon satisfies the equation then 1 by epsilon also satisfies the equation ok. So, what I am saying is that if you basically say that and you similar things can be also observed for other non-linear function.

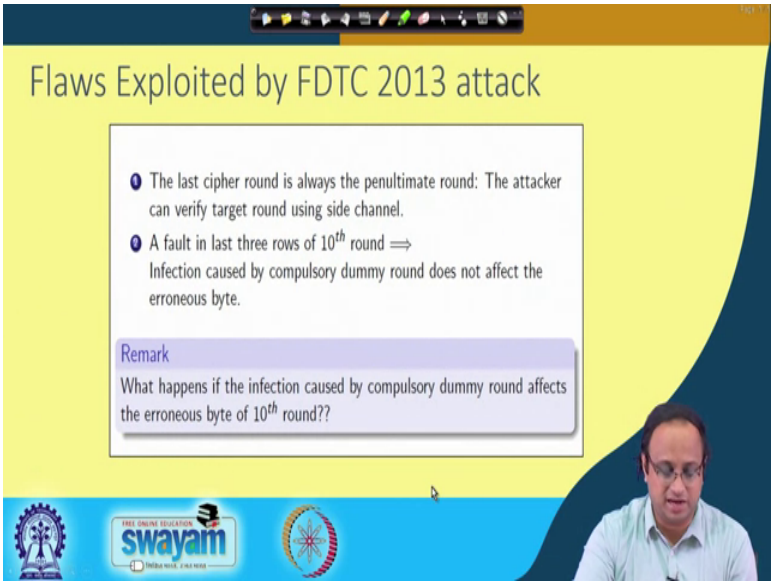
So, if your SNLF epsilon is nothing but 1 by epsilon that is the finite field inverse. Then the equation is epsilon plus 1 by epsilon is equal to sum constant value, because this value is already observed from the cipher and from the faulty cipher. So, here if epsilon is a root then 1 by epsilon is also a root of this equation ok; therefore, there are 2 solutions. And there are 2 power of as a probability of this equation getting satisfied is therefore, 1 by 2 to the power of 8 because you know like this value is already known from the output. And the probability that a random choice will satisfy this equation is 1 by 2 power of 8. Because there are 2 power 8 bytes and out of them I am considering 1 byte actually it is not one value, but there are 2 values which can satisfy depending upon the value of your epsilon.

So, therefore, right I mean roughly speaking you know like you will have if you consider this, then approximately 2 to the power of 9 values will survive and we will basically kind of satisfy this equation. And you can see that this is nothing but if I multiply then it is 2 to the 2; if I multiply this probability with this then 2 to the power of 8 gets canceled with 2 to the power of 8 and you have roughly 2 into 2 to the power of 8 which is 2 to the power of 9 key fault pairs which survives this test.

Now, what I will do is therefore, I will repeat this error ok by using the same error note that if I kind of vary the error then I, this will not this attack will not work. So, I keep the same error and therefore, the error is constant and I repeat it for 3 pairs of cipher and faulty cipher to obtain a unique value of the key byte. So, this attack can be pretty much

kind of extended for other byte positions and therefore, we can essentially perform a fault attack in this particular model.

(Refer Slide Time: 15:42)



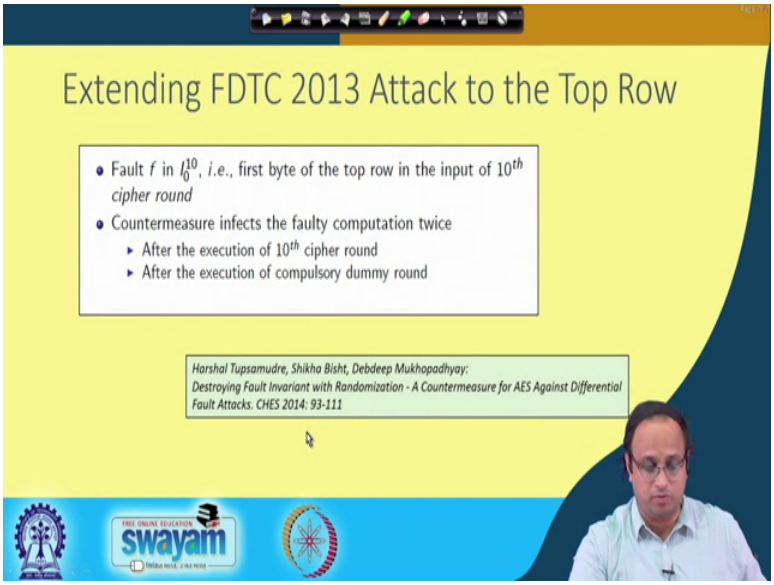
So now, the question is right I mean. So, basically like if I summarize what we have seen is that the last cipher round and you know like the reason, why it works is because the last cipher round is always the penultimate round remember that when the final you know like round comes before the final compulsory dummy round which is the fifteenth line in my algorithm for infective countermeasure. Then you will find that a last round is always a cipher in this case ok. And therefore, the attacker can verify the target round using side channels because it can probably know when is the penultimate round which is happening. And therefore, can for certainties kind say that this is my; you know like tenth round of being in operation.

Likewise, right what we observe is that if there is a fault in the last 3 rounds of tenth round. Then the infection caused by the compulsory dummy round does not affect the erroneous byte. Because of the shift right, because of if you see that if there is a fault then what is happening is that, this particular byte is not getting affected because this diffusion is getting is getting shifted ok and therefore, the attacker has an access to the output differential.

So now the question is what happens if the infection caused by the compulsory dummy down affects the erroneous byte of tenth round ok. And actually you know like more

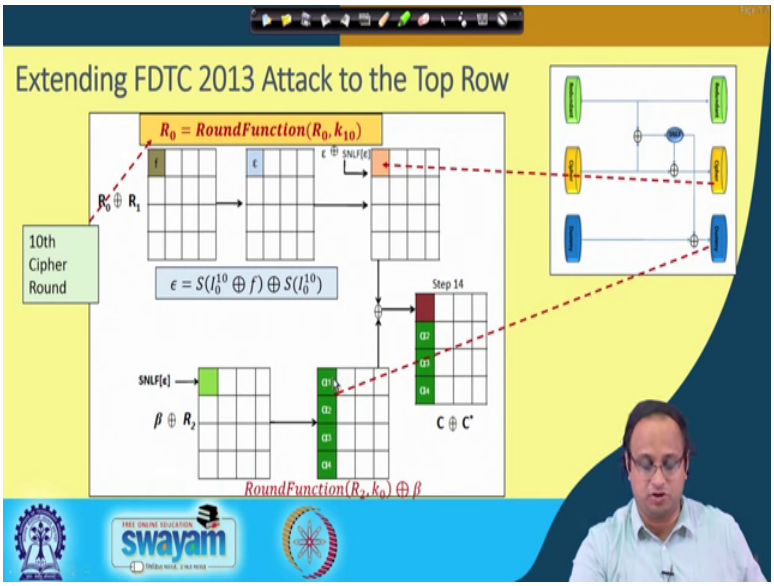
importantly like suppose you know like the, this particular attack happens in the top round top row then what will happen.

(Refer Slide Time: 16:54)



So, the fault f in I 0 10 that is the first byte of the top row in the input of the tenth cipher round is now where the fault is seated. And we can easily understand that you know like that because of this the countermeasure infects the faulty computation twice after the execution of the tenth cipher round like in the previous case and again after the execution of the compulsory dummy round. So, this is something or this attack propagation is similar to what we have already seen. The only thing is that the fault location is now in the first row instead of the second row as we have seen in the previous illustration.

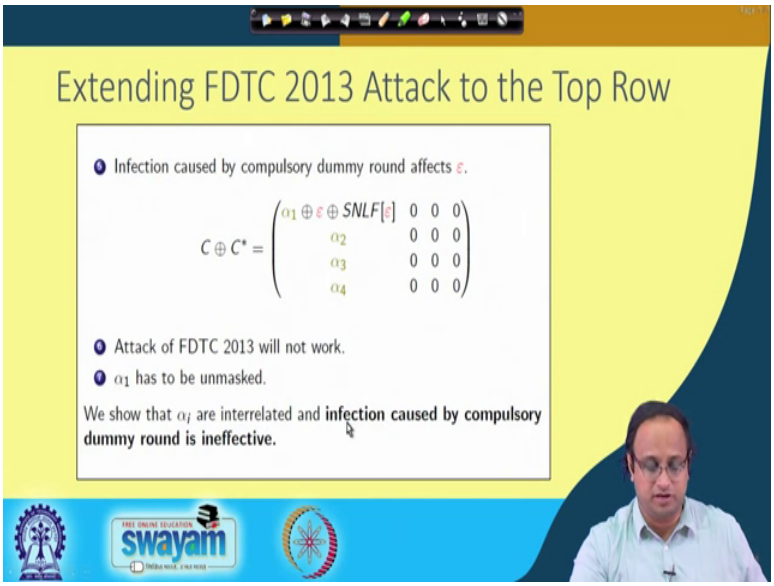
(Refer Slide Time: 17:34)



So, therefore, right if the fault is here. Then again similar things will happen and, but there is a small difference that will also happen which is essentially important. So, here this difference will result in suppose alpha 1, alpha 2, alpha 3 and alpha 4, but remember right this does not get shifted because in the first row there is no shift ok. So, this basically gets in also in the first column this also in the first column.

So, when we XOR it this particular difference that is epsilon XOR SNLF epsilon which we exploited in the previous attack is not known to the adversary because it gets masked by this alpha 1 which is seemingly a random component.

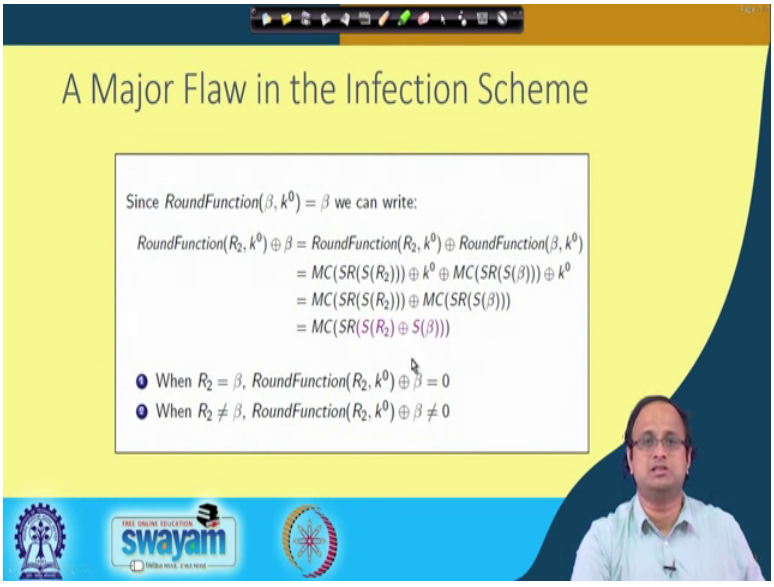
(Refer Slide Time: 18:11)



And therefore, right it is difficult for us to extend the FDTC attack to the top row or you know like then and it was claimed the attack will probably not work. Because alpha 1 has to be masked or you know like alpha 1 has to be unmasked rather to get the value of epsilon XOR of SNLF epsilon.

But now we show that all this alpha ''s like alpha 1 alpha, 2 alpha, 3 or alpha 4 are actually interrelated and therefore, you can remove their effect and obtain the value of epsilon XOR SNLF epsilon.

(Refer Slide Time: 18:40)



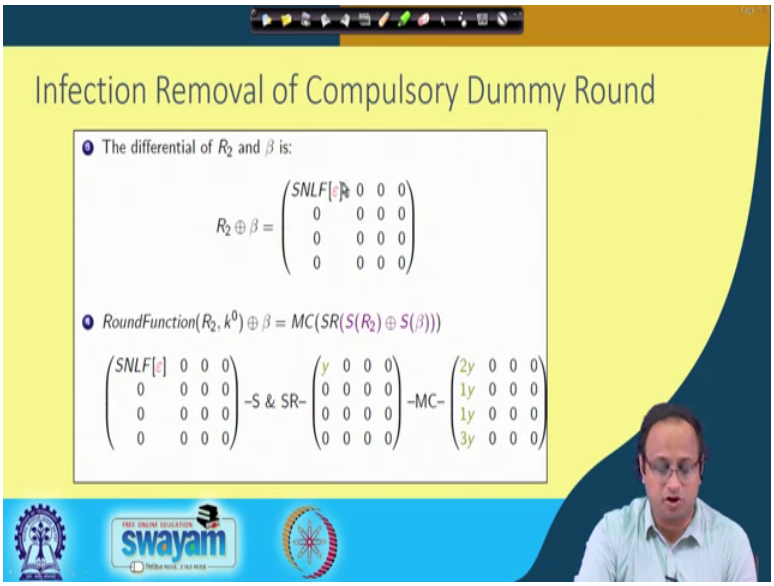
How can you do that? Essentially that is because of a major flow in the infection scheme ok. See if you remember again the fifteenth equation that we had. So, there the important step that we did was we basically corrupted R 0; that means, in the compulsory dummy round; we basically corrupted R 0 with the XOR of beta and the round function applied on R 2 and k 0. Remember that if write the R 2 or the value of R 2 is beta then the round function results in beta. Because this round function or the dummy round was idempotent with respect to k 0 and therefore, a beta input was resulting in a beta output.

But now imagine that R 2 because of some kind of pollution is not equal to beta, then what is the result of this computation. So, you can easily observe that this round function can be broken up into the AES components which stands for the S box followed by the shift row function followed by the mix column operation. And that applies for both these down functions because they are exactly same. So, here right this on operates on R 2 comma k 0 whereas, this operates on beta comma k 0. And you can therefore, write this operation as MC SR S applied an R 2 XOR with a 0 which is a round key, XOR with again MC applied on S R XOR applied on S of beta XOR with k 0.

Now, this case k 0 and this case 0 gets canceled and note that mixed column and shift rows are linear operations. And therefore, I can again take them in common and what I have here is S of R 2 XOR of S of beta which I cannot combine because the S function is a non-linear function. But then we observed that if R 2 is equal to beta then these 2 things will be same and therefore, this result will be 0. So, you will have exactly what you want, which is essentially 0 in this case and therefore, there is no further propagation of or there is no contamination which happens because of this.

But if R 2 is not equal to beta then this value will result in a non-zero value because R 2 and beta are not being same S of R 2 and S of beta will also not be same because the S function is bijective in case of AES

(Refer Slide Time: 20:39)

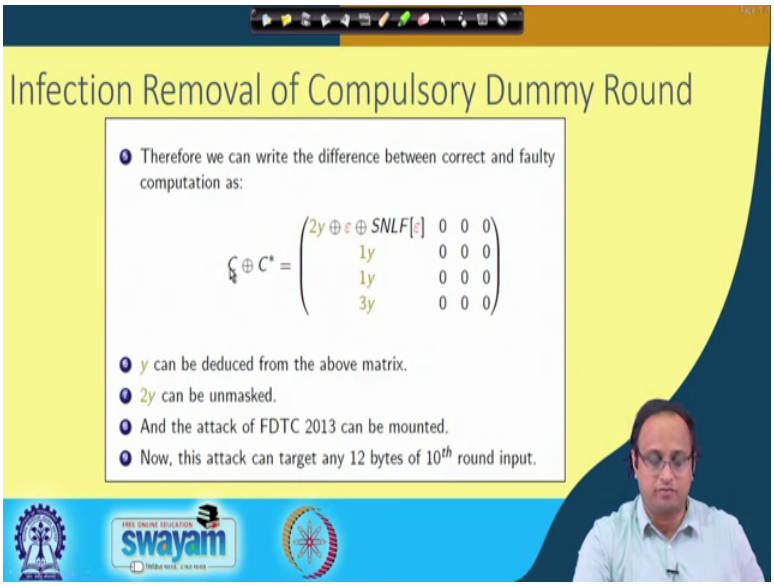


So, therefore, right what will happen is that suppose the fault is in the first round and therefore, right we are now considered in the diffusion in the dummy round. So, remember right the fault is over here, which is essentially standing for SNLF epsilon. So, if we just observe the depiction here the fault is here right. So, this is the fault SNLF epsilon.

So now because of this you know like round computation what will happen over here is that the round function R 2 comma k 0 XOR with beta will be essentially as shown previously in the mixed column applied on shift rows applied on the XOR of S of R 2 and of S or beta. So, this is your, you know like disturbance. So, if we apply the S function then this gets converted into some unknown value. And again because of shift row this does not get shift because in the first row there is no shift.

Now, we apply this mixed columns and because of this mixed columns we again get this you know like interrelationship as we have seen in the context of DFA on AES we get relationship which is like 2 y, y and 3 y.

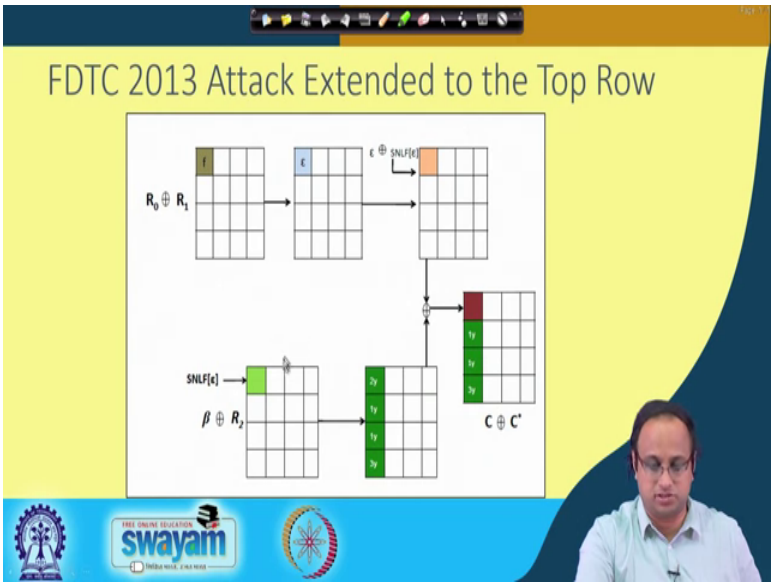
(Refer Slide Time: 21:44)



So, that what does it mean? It means that if right the attacker observes the XOR of the correct and the faulty cipher text, then the alpha 1, alpha 2, alpha 3 and alpha 4 that we were seeing previously are not actually random there is a relationship between them.

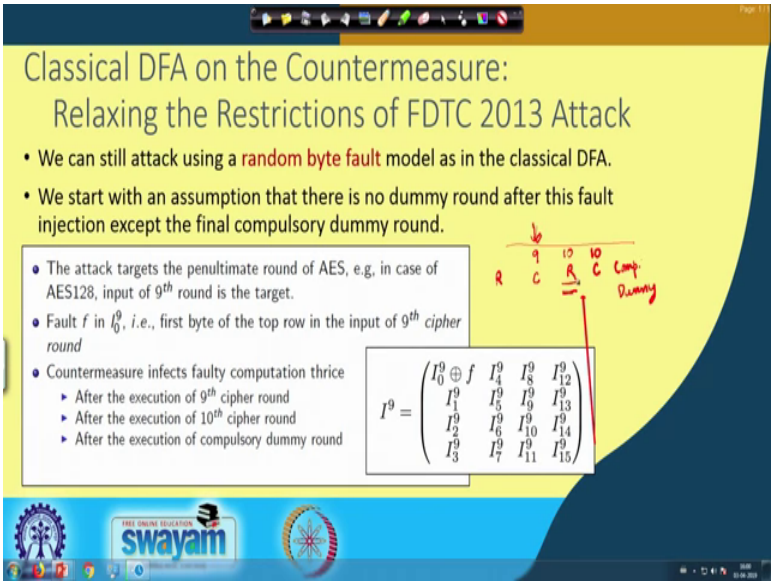
And; that means, that if the attacker observes this disturbance. For example, the attacker knows the value of y. And if the attacker knows the value of y the attacker can also know the value of 2 y and therefore, from there can unmask this contamination and again get the value of epsilon XOR SNLF epsilon. And therefore, can pretty much repeat the attack that we just now saw on the remaining 3 rows of the AES state matrix.

(Refer Slide Time: 22:22)



Therefore, right these attack will work pretty much on any of the byte positions. So, therefore, right this is the illustration of how the attack works and it will be probably work even over here because, you can remove this disturbance and get the value of epsilon XOR of SNLF of epsilon and repeat the attack.

(Refer Slide Time: 22:35)



But again right as we discussed that this particular attack requires a constant fault model. And therefore, right what we expect is a kind of you know like assumption that the fault essentially every time when you are repeating the fault, the fault is a constant fault.

However right what we would probably like to do is basically we would like to apply the classical DFA on this countermeasure where we would like to assume that the fault is a random byte fault. So, if we assume that then you know like again we will basically assume here that the fault is inflicted in the penultimate round of AES that is in this case of 128 AES and the input of the 9th round which we make as a target.

So, the fault f is in I 0 n. So, I 0 n stand for here. And therefore, the faulty state matrix of the input of the 9th round is shown in this state matrix where everything remains the same except these particular byte which is contaminated with a random fault. So now, we make an assumption; the assumption or we start with an assumption which we will relax subsequently is that there is no dummy round after this fault injection except the final compulsory dummy round ok.

So, we basically make an assumption that there is no you know like after the fault induction except the final compulsory dummy round. There is no dummy round that happens ok. And that implies that you know like if you are inducing the fault for example, here and the fault is at the input of the 9th round ok. So, that is where I would like to induce the fault, but what may happen is that you know like since you are doing a cipher operation here. That means, there is a previous redundant round which has already taken place; that means, after this right there will be again a redundant 10th round operation followed by again an 11th I mean the final tenth round cipher which takes place. And here there is a compulsory dummy round ok. So, this is the compulsory dummy round which takes place.

So, therefore, right what will happen is that this countermeasure will infect the so, if you are able to induce the fault indeed at this position. Then; that means, the fault will kind of infect or rather the there will be an infective count; infection in the computation in 3 locations ok. So, the one is of course, like after the execution of the 9th cipher round, one is at the after the execution of the 10th cipher round ok and you will basically get the fault also at the you know like after the execution of the compulsory dummy round.

So, these are you know like the positions where you will essentially have the fault which is essentially created. Remember that you will not have the fault here because or you will not have an infection where you are doing the redundant computation because in our particular example right we basically do the check that whether there is an difference

between the redundant and the cipher round and we basically contaminate only after the cipher round. So, we do not do that check after a redundant round and therefore, right this error will not kind of infect after the redundant round.

So, where will that infect, this will basically get infected only after the cipher round ok. So, this I believe can be easily observed right if you go back and see the algorithm. So, in the previous case right if we just walk back few steps and take a look at the original algorithm right, I mean this becomes quite clear. So, for example, if you observe this algorithm then this infection; that means, the infection of computation of error epsilon which is SNLF of C 0 XOR with C 1 only happens if I is even ok. And I becomes even. So, we start with I equal to 1 and then we have the even dumb.

So; that means, the second term is always the even dumb. So; that means, right this check happens only when you are doing the cipher computation and not when you are doing the redundant computation. So; that means, right if you are inducing the fault at the 9th round cipher then the infection will happen subsequent to this not in the 10th round redundant round, but in the 10th round cipher.

And therefore, right we will have the fault indeed at these 3 positions or 3 instances, which is essentially after the execution of the 9th round cipher, after the execution of the 10th round cipher and also after the execution of the compulsory dummy round ok. So, next class right we will see that how we can use this set up to perform an end to end attack on this particular countermeasure and show that it can be completely broken ok.

Thanks for your attention.