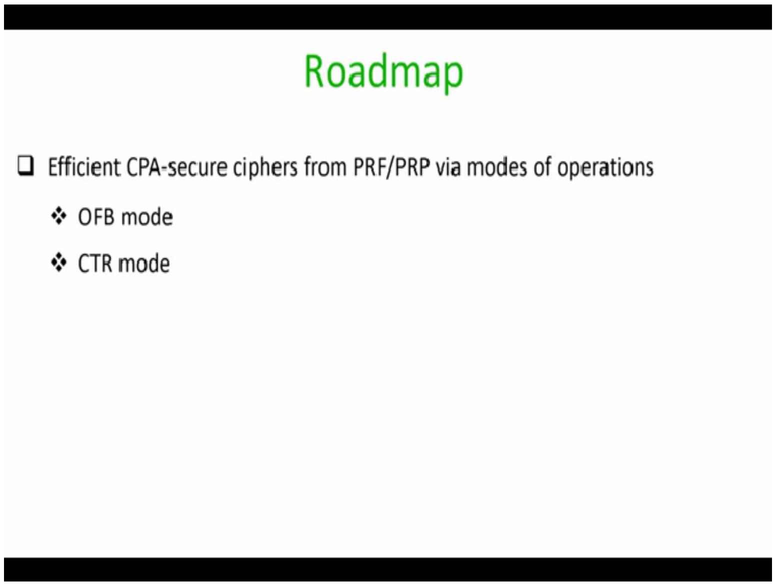
Foundations of Cryptography

Prof. Dr. Ashish Choudhury
(Former) Infosys Foundation Career Development Chair Professor
International Institute of Information Technology-Bangalore

Lecture-17
Modes of Operations of Block Ciphers Part II

Hello everyone, welcome to lecture 16. In this lecture we will continue our discussion regarding the modes of operations of block ciphers.

(Refer Slide Time: 00:36)

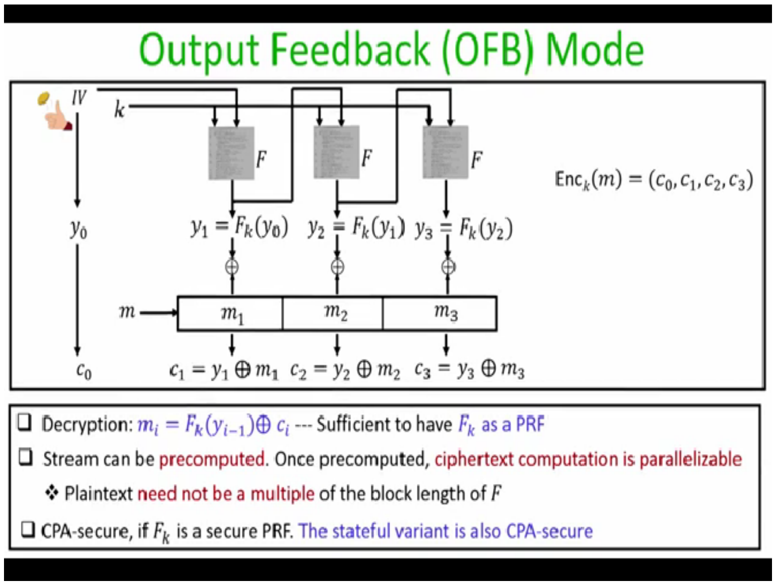


Roadmap

- Efficient CPA-secure ciphers from PRF/PRP via modes of operations
  - OFB mode
  - CTR mode

Specifically the roadmap for this lecture is as follows, we will discuss how to design efficient CPA secure ciphers using 2 of the modes of operations, namely the output feedback mode and counter mode of operation.

(Refer Slide Time: 00:50)



Output Feedback (OFB) Mode

- Decryption: $m_i = F_k(y_{i-1}) \oplus c_i$ --- Sufficient to have F_k as a PRF
- Stream can be precomputed. Once precomputed, ciphertext computation is parallelizable
  - Plaintext need not be a multiple of the block length of F
- CPA-secure, if F_k is a secure PRF. The stateful variant is also CPA-secure

So, let us start with the output feedback mode or OFB mode in short. So, the idea here is basically sender and receiver generate a pseudo random stream of pad, which is actually independent of the plaintext, which sender wants to encrypt. And once the pseudo random stream of pad is generated and the message is available, the pseudo random stream of pad is used for XORing or masking the message. So let us see how the pseudo random stream of pad is generated. So, the sender randomly choose an IV, which we do not as y_0.

And the size of the IV has to be the same as the block length of your underlying function F. And then imagine we want to generate 3 blocks of pseudo random stream of pads here is how we go. So we invoke 3 underlying invocations of the function F, all instantiated with the same key k and we feed the IV as block input for the first invocation, obtain the output y_1, and then we do the chaining.

But now this chaining is different from the CBC mode of encryption that we had seen in the last lecture. Here the chaining is independent of the message block because as of now, there is no actual plaintext which is getting involved here. So, the output y_1 is fed as the block input for the second invocation of the function to obtain the second block of pseudo random stream of pad which we could denote as y_2 and then this is fed as the block input for the third invocation.

And we can continue like this right. So, this is how the pseudo random stream of pad is generated. And now if the message is available for encryption, whatever pseudo random stream of pad that has been generated till now it can be used to master message depending upon the length of the message right. So, you can imagine that encryption here is now a 2 stage process and offline process and an online process where in the offline phase or a message independent phase a pseudo random stream of pad is generated.

And during the online phase when the actual message is available, we do the encryption by using whatever pad that has been generated in the offline phase as the mask right. So, if the IV is also communicated to the receiver and the receiver also can generate a pseudo random stream of pad offline and once the message is encrypted from the sender side and communicated back to the receiver it can be decrypted properly.

So, in this specific example, the encryption of the message will consist of the IV and the actual encryption namely c_1 c_2 c_3, right. And to decrypt, we do the corresponding operation namely to recover back the i^{th} plaintext block, what we do is receiver needs the i^{th} block of pseudo random stream of pad and which can be computed by just doing this operation and once the i^{th} block of pseudo random stream of pad is available it can be XORed from the i^{th} ciphertext block to recover back the plaintext.

That means, it is sufficient in this mode of operation, for the F being just a pseudo random function, we do not need the function f to be an invertible function. That is of, it suffice for F(k) to be just be PRF. And the advantage of this mode of operation is that that the stream can be pre computed and once precomputed the ciphertext computation is parallelizable. That means we cannot generate the stream of pads in parallel because the pad generation happens sequentially.

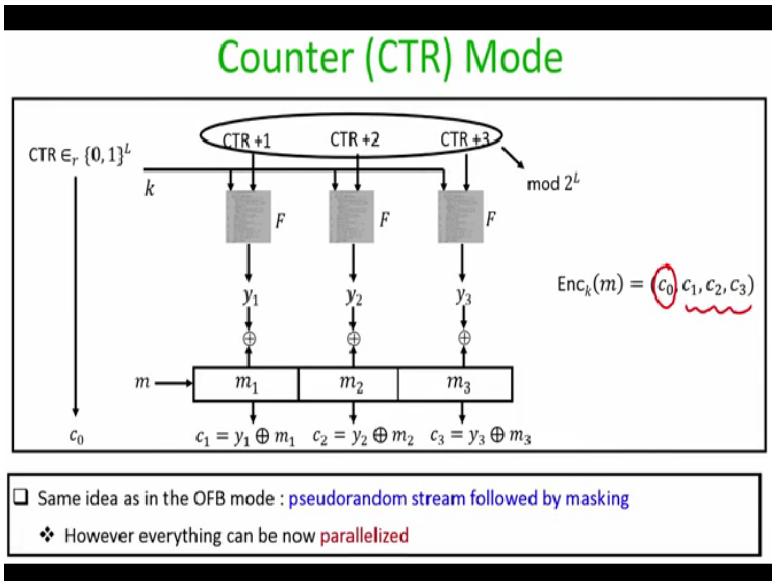
But once the offline phase is over and the pad is available, whatever message block we want to encrypted encryption can happen in parallel right, moreover, the another advantage here is that a plaintext may not be a multiple of the block length of your underlying function F, that means here we do not require any sophisticated padding to make the length of the message to beto be a multiple of the block length of your underlying function before encrypting the message, right.

So this is unlike your CBC mode of encryption. And it can be proved formally that if your underlying keyed function F(k) is a secure PRF, then this mode of operation namely the OFB mode is CPA secure, also we can prove that a stateful variant is also CPA secure that means, imagine a scenario where sender and receiver are synchronized and they have generated a pseudo random stream of pad y_1 y_2 y_3 and which is used to encrypt a message m right.

And now, there is a new message m' for encryption which is available for the sender. So, if the sender now wants to encrypt a message m' using the OFB mode, there is no need to pick a fresh IV the y_3 which has been used which was the last block of the pseudo random pad for the previous message can be used as the IV for the message m' and using y_3 we can kick start the whole process.

And generate a fresh pseudo random stream of pad which can be then used for masking the new message m' and it can be proved that a the stateful variant of this OFB mode is also CPA secure right. So, more or less we actually get all the advantages that we want or we expect from the best possible mode of operation. Namely, now we have CPA security, we have the minimal assumption required from the underlying function namely PRF assumption. And if we assume that the pad has been generated in the offline phase and ciphertext computation is parallelizable and so on. But it turns out that we can do better, right.

(Refer Slide Time: 06:26)



Counter (CTR) Mode

- ❑ Same idea as in the OFB mode : pseudorandom stream followed by masking
  - ❖ However everything can be now parallelized

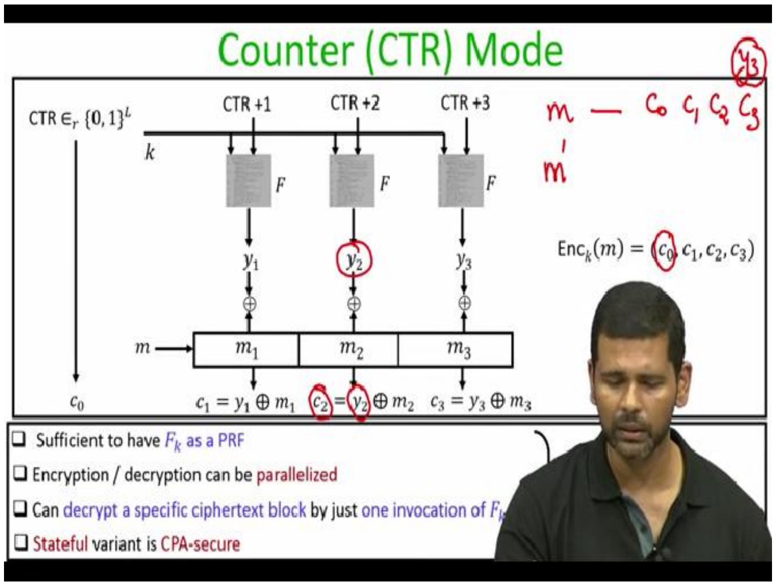
And further better mode of operation is the counter mode of operation of the CTR mode. And the idea here is more or less the same. Namely, we divide the encryption process into 2 phases. The first phase is the message independent phase, where a pseudo random stream of pad is generated, and which is later used to do the actual masking of the message. But now, the way the pseudo random stream of pad will be generated, everything can be parallelized.

That means the pad generation as well as the actual encryption can be made parallelized. So here is how does counter mode of operation is executed. So as in the OFB mode, we start with a random IV, which we call the counter, and the counter is selected randomly from the domain of the block length of your underlying function. So it is a uniformly random bit string of length L and which will be also a part of the cipher text.

And then we generate the pseudo random stream of pad, but unlike the OFB mode, we do not do any chaining here, the way we generate the pseudo random stream of pad is as follows. So, the first invocation of your function F will be with the input counter plus one, the second invocation of the function will be with the input counter plus 2 and so on. That means all the invoke subsequent invocations of the function will be with an input which is one more than the previous value of the counter used in the previous or the previous invocation of the function.

And once the functions Fs are evaluated on this input, we get the pseudo random stream of blocks of pads which can be led to used to actually mask your message for the encryption right. So that is the only difference here, the way we are generating the pseudo random stream of pad, remaining all the operations remain same as in the OFB mode here. So here again, the actual encryption of the message will consist of your counter value, which will be the c_0 block, and actually encryptions blocks are the encrypted blocks will be c_1 c_2 c_3 okay.

(Refer Slide Time: 08:26)



Counter (CTR) Mode

$\text{Enc}_k(m) = (c_0, c_1, c_2, c_3)$

- Sufficient to have F_k as a PRF
- Encryption / decryption can be parallelized
- Can decrypt a specific ciphertext block by just one invocation of F_k
- Stateful variant is CPA-secure

So, as in the OFB mode, we can actually formally prove that it suffice for the underlying keyed function to just be a PRF over to have the counter mode of operation to be CPA secure. Because for the decryption process we do not require any inversion to be computed. So that is why it suffice for the underlying keyed function to be just a PRF. More importantly, the encryption as well as decryption can happen in parallel. This is because when we if you see closely the way we are generating the pseudo random stream of pad.

If I want to generate y_2 I do not need to depend on y_1 which was the case in the OFB mode, in the OFB mode and until and unless you generate, y_1 you cannot compute y_2 and hence until and unless y_2 is computed, you cannot generate y_3 and so on. So that is why the pseudo random stream of pad generation that was not parallelizable. But the way we are generating the pseudo random stream of pad and the counter mode of operation, y_2 can be generated independently of y_1 and so on.

In that sense, both the encryption and the decryption operation can be parallelized right. More importantly in the counter mode of operation, if I want to just decrypt a specific ciphertext block, and I am not interested in decrypting the other blocks that is possible, and that to by just invoking one invocation of the underlying keyed function, for instance, if I just want to decrypt c_2, for decrypting c_2 what I need is y_2.

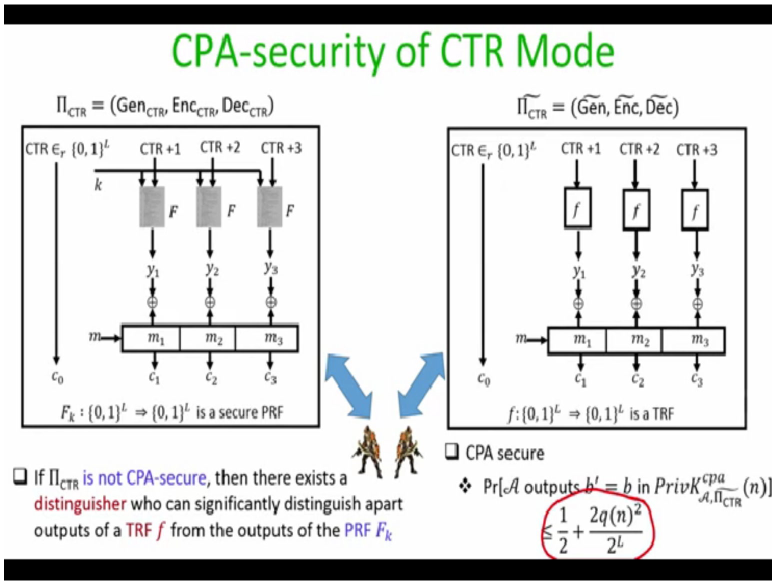
And y_2 just requires me to invoke the underlying function on the value counter plus 2. So if I have the initial counter value, which is there in c_0, and if I want to decrypt c_2 what I have to do is I have to create c_0 as the counter value, compute counter plus 2 invoke my underline keyed function on the input counter plus 2 generate the pseudo random pad y_2 and XOR it from c_2 which will give me back m_2.

So, depending upon my requirement, I can decrypt any specific ciphertext block by just invoking one invocation of the underlying key function, which was not possible in the OFB mode. And more importantly, we can prove formally that the stateful variant of the counter mode of operation is also CPA secure. That means, if I have a message m, and for that my resulting ciphertext is c_0, c_1 c_2 c_3, where y_3 was the last stream of pseudo random pad.

And now I have a new message m**'**, right, so, I do not need to pick a fresh counter value to generate a pseudo random stream of pad for the message m**'**, I can use y_3 as the counter value for generating the next sequence of pseudo random pad for the message m**'**, and the same can be done at the receiver end, and it can be proved formerly that the stateful variant is also CPA secure. So we have now almost exactly all the features that we required from it, good mode of operation of the block cipher.

And that is why this counter mode of operation is very popular, which is used in several instantiations of the real world protocols okay.

(Refer Slide Time: 11:25)



CPA-security of CTR Mode

$\Pi_{CTR} = (Gen_{CTR}, Enc_{CTR}, Dec_{CTR})$

$\widetilde{\Pi_{CTR}} = (\widetilde{Gen}, \widetilde{Enc}, \widetilde{Dec})$

- If Π_{CTR} is not CPA-secure, then there exists a distinguisher who can significantly distinguish apart outputs of a TRF f from the outputs of the PRF F_k

- CPA secure
  - $\Pr[\mathcal{A}$ outputs $b' = b$ in $PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi_{CTR}}}(n)]$ $\leq \frac{1}{2} + \frac{2q(n)^2}{2^L}$

So now we want to do rigorous security analysis of the CPA security for this counter mode of operation right. So, I am going to use this example. So imagine a scenario where a sender has encrypted say a message consisting of 3 blocks right and this is how the counter mode of operation would have operated. And we want to prove that this counter mode of operation is CPA secure. So before going into the actual security proof of the counter mode of operation.

Let us consider a variant of this counter mode of operation where everything remains as it is in the actual counter mode of operation except that all the invocations of the key PRF are replaced by invocations of a truly random function which is an unkeyed function. So, imagine both sender and receiver have agreed upon a truly random function which is an unkeyed function, mapping L bit strings to L bit strings.

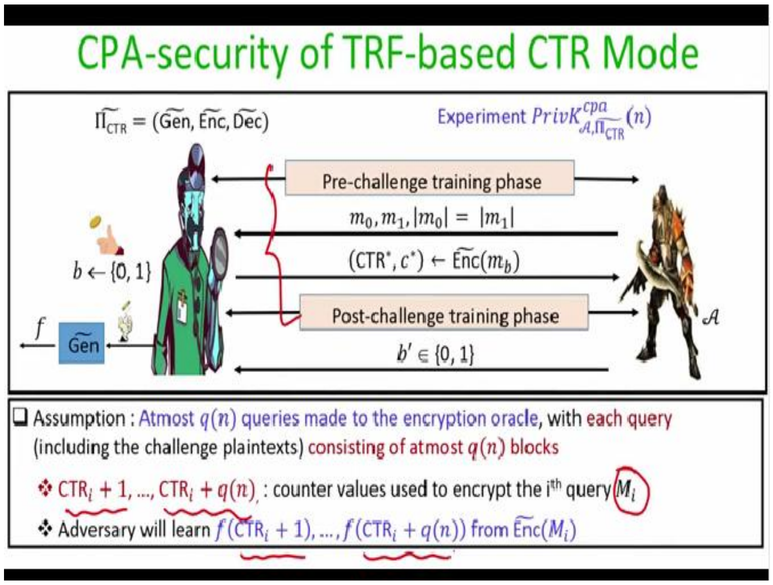
So, the variant of the counter mode of operation which I denote as $\widetilde{\Pi_{CTR}}$i is almost the same as the actual counter mode of operation except that all the invocations of the keyed function F_k are replaced by the truly invocations of the truly random function f. Now, what we are going to prove next is that, if we consider this variant of the counter mode of operation based on truly random function.

And if we consider any arbitrary poly time adversary participating in an instance of CPA indistinguishability game and makes total of q(n) number of encryption oracle queries then the probability with which that adversary can win the CPA game is upper bounded by ½ + (2 q(n)^2 / 2^L), right. So if L is actually if we assume that L is some polynomial function of the security parameter, and anyhow q(n) is a polynomial function of the security parameter.

Then overall we end up showing that the probability of any poly time adversary winning the CPA game against this truly random function based counter mode of operation is half plus negligible, which satisfies the definition of CPA security. So let us first do that. Once we do this, later on we will actually show that actual counter mode of operation which we want to prove formally to be CPA secure, is not CPA secure, Then it means that there exist a distinguisher who can distinguish apart the behavior of the keyed pseudo random function from the behavior of a truly random function and which will be a contradiction to the assumption that keyed function is a pseudo random function. So that is the overall idea of the proof, we will first consider a variant of the counter mode of operation prove it to be CPA secure formally.

And then we will come back and argue that since the only difference between the actual counter mode of operation and a variant of the counter mode of operation is the underlying invocations of the function. So, the actual counter mode of operation should also be CPA secure. Otherwise, there is a distinguisher who can distinguish apart from the keyed function from the outcome of a truly random function right.

(Refer Slide Time: 14:37)



CPA-security of TRF-based CTR Mode

$\widetilde{\Pi}_{CTR} = (\widetilde{Gen}, \widetilde{Enc}, \widetilde{Dec})$
Experiment $PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{CTR}}(n)$
Pre-challenge training phase
$m_0, m_1, |m_0| = |m_1|$
$(CTR^*, c^*) \leftarrow \widetilde{Enc}(m_b)$
Post-challenge training phase
$b' \in \{0, 1\}$
$b \leftarrow \{0, 1\}$
f
Gen
$\mathcal{A}$

- Assumption : Atmost $q(n)$ queries made to the encryption oracle, with each query (including the challenge plaintexts) consisting of atmost $q(n)$ blocks
  - $\underbrace{CTR_i + 1}, \ldots, \underbrace{CTR_i + q(n)}$: counter values used to encrypt the ith query M_i
  - Adversary will learn $f(\underbrace{CTR_i + 1}), \ldots, f(\underbrace{CTR_i + q(n)})$ from $\widetilde{Enc}(M_i)$

So, let us first proceed to prove the CPA security of the truly random function based counter mode of operation. So, consider an arbitrary computationally bounded adversary who participates in an instance of a CPA game which will consist of 4 phases, we will have pre-challenge training phase and we will have a challenge phase, right. And to answer the queries of the pre-challenge training phase basically, the challenger will select a truly random function which would not be known to that adversary right.

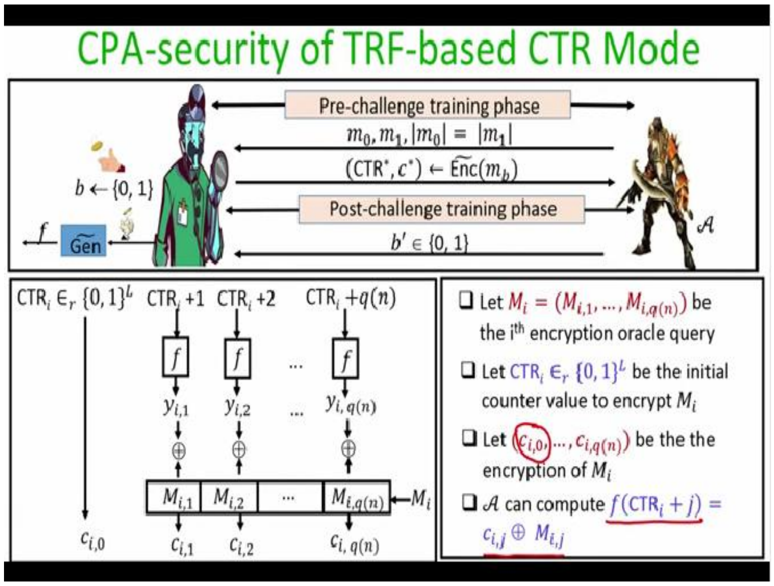
And all those queries will be encrypted by running the encryption algorithm of the truly random function based encryption process. And then once the pair of challenge plaintexts are submitted by the adversary, the challenger will pick one of those messages randomly for encryption. And it will encrypt it by picking a random counter value, which I denote as CTR* and then c* will be basically the value of the CTR* on the truly random function XORed with the message m_b.

And then there will be a post challenge training phase, again, where the adversary will adaptively submit encryption oracle queries and which will be encrypted as per this modified encryption process. And finally, the adversary will output whether this challenge ciphertext is an encryption of message m_0 or message m_1, right. So let us assume that the total number of queries which the adversary is asking throughout this instance of this experiment is q(n).

The q(n) is some polynomial function of the security parameter *l*. Moreover, we also assume that each query also consist of at most q(n) number of blocks. So, actually we will assume the worst case scenario where there are total of q(n) number of queries and each query is exactly consisting of q(n) number of blocks right. So, I denote the i^{th} query for which the adversary has asked the encryption oracle service to be M_i.

And since M_i consist of q(n) number of blocks adverse to answer the encryption oracle service to answer to encrypt this message M_i, the challenger need to take q(n) number of counter values. So, I denote those counter values as counter i + 1, counter i + q(n) and so on. So, the first observation is that if adversary submits the message M_i for encryption and c is the encryption of that message as per this variant of the counter mode of operation, then by seeing the ciphertext adversary will learn the value of the unknown truly random function on this counter values, namely counter i + 1, counter i + 2 and so on.

(Refer Slide Time: 17:20)



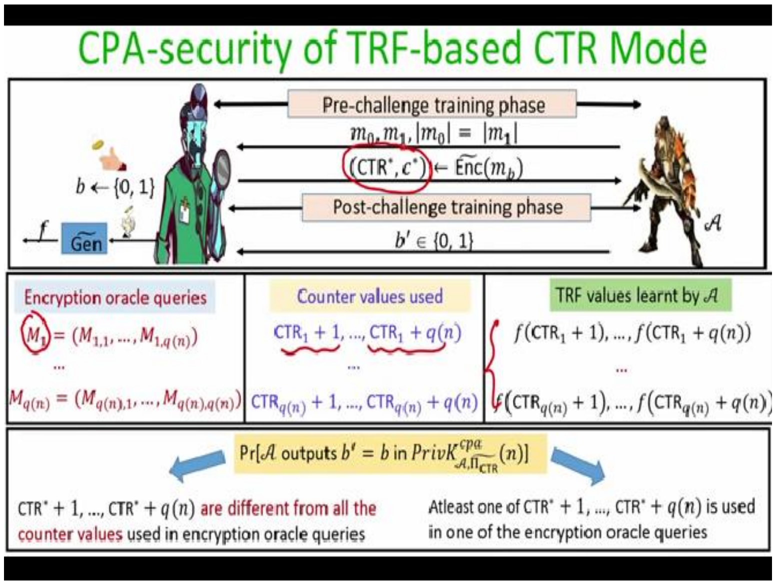
CPA-security of TRF-based CTR Mode

- Let $M_i = (M_{i,1}, \ldots, M_{i,q(n)})$ be the i^th encryption oracle query
- Let $\mathsf{CTR}_i \in_r \{0,1\}^L$ be the initial counter value to encrypt M_i
- Let $(c_{i,0}, \ldots, c_{i,q(n)})$ be the the encryption of M_i
- $\mathcal{A}$ can compute $\underline{f(\mathsf{CTR}_i + j) = c_{i,j} \oplus M_{i,j}}$

Why, so let us consider the i_{th} query M_i which basically consist of q(n) number of blocks, right and for encrypting that message, the challenger will pick a random counter value, which I denote as CTR_i right. And once the counter value is selected, since there are q(n) number of blocks in my message m_i basically the challenger needs to evaluate the truly random function on CTR_i + 1, CTR_i + 2 and CTR_i + q(n), which will generate the sequence of pad blocks which will be XORed with the actual message blocks that are present in the message m_i to generate the ciphertext blocks.

And the whole cipher text will be communicated back to the adversary as the response from the encryption oracle query, since the adversary will know the value of the counter i and counter i + 1 and so on. The adversary just can simply perform this operation namely, it can XORed the j_{th} ciphertext block and the j_{th} message block present in the message m_i and that will give him the value of the unknown truly random function on this counter value namely counter i + 2 right.

(Refer Slide Time: 18:32)


CPA-security of TRF-based CTR Mode
Pre-challenge training phase
$m_0, m_1, |m_0| = |m_1|$
$(\mathsf{CTR}^*, c^*) \leftarrow \widetilde{\mathsf{Enc}}(m_b)$
$b \leftarrow \{0,1\}$
Post-challenge training phase
$b' \in \{0,1\}$
f
$\widetilde{\mathsf{Gen}}$
$\mathcal{A}$
Encryption oracle queries
$M_1 = (M_{1,1}, \ldots, M_{1,q(n)})$
$M_{q(n)} = (M_{q(n),1}, \ldots, M_{q(n),q(n)})$
Counter values used
$\mathsf{CTR}_1 + 1, \ldots, \mathsf{CTR}_1 + q(n)$
$\mathsf{CTR}_{q(n)} + 1, \ldots, \mathsf{CTR}_{q(n)} + q(n)$
TRF values learnt by $\mathcal{A}$
$f(\mathsf{CTR}_1 + 1), \ldots, f(\mathsf{CTR}_1 + q(n))$
$f(\mathsf{CTR}_{q(n)} + 1), \ldots, f(\mathsf{CTR}_{q(n)} + q(n))$
$\Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{\mathsf{CTR}}}(n)]$
$\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)$ are different from all the counter values used in encryption oracle queries
Atleast one of $\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)$ is used in one of the encryption oracle queries

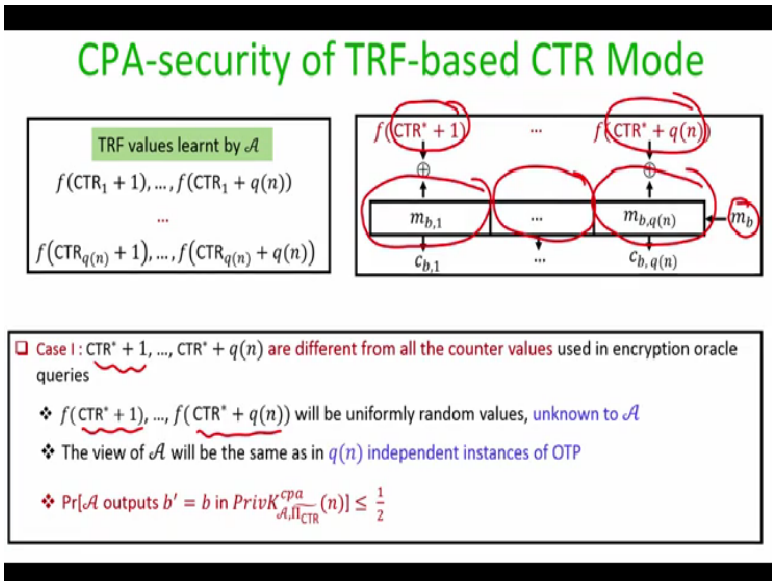
So, since I am assuming that my adversary is going to make q(n) number of queries those queries I am representing as message m_1 m_2 and $m_{q(n)}$ and I am assuming that each of those messages further consist of q(n) number of blocks. And I assume that the our challenger in this instance of the experiment uses this counter values for encrypting these message blocks namely the first message is encrypted using the counter values $\mathrm{CTR}_1 + 1$, $\mathrm{CTR}_1 + 2$, $\mathrm{CTR}_1 + q(n)$ so on.

So, these are the counter values which are used by the challenger. And as I have seen that based on the encryption of his message the adversary learns the corresponding output of the truly random function on the corresponding counter values. So that means adversary basically have access to the value of the unknown truly random function f and all the counter values which have been used by the challenger to respond to the encryption oracle queries.

Now our goal is to analyze what is the success probability of this adversary in identifying whether this challenge ciphertext is an encryption of the message m_0 or whether it is an encryption of m_1 and we can split this overall probability into the summation of 2 probabilities. So consider the first case when the counter values which are used to prepare the challenge ciphertext c* are different from all the counter value which have been used to respond to the encryption oracle queries raised by the adversary.

That is the first case. And the second event is when the counter values which are used to prepare the challenge ciphertext overlaps or gets repeated in at least one of the instances of the encryption oracle queries namely the counter values which are used to prepare the challenge ciphertext overlaps with the counter values which have been used to respond to the adversary's encryption oracle queries. So these are the 2 separate cases.

(Refer Slide Time: 20:33)



CPA-security of TRF-based CTR Mode

TRF values learnt by $\mathcal{A}$

$f(\mathsf{CTR}_1 + 1), \ldots, f(\mathsf{CTR}_1 + q(n))$

...

$f(\mathsf{CTR}_{q(n)} + 1), \ldots, f(\mathsf{CTR}_{q(n)} + q(n))$

$f(\mathsf{CTR}^* + 1)$
...
$f(\mathsf{CTR}^* + q(n))$
$m_{b,1}$
...
$m_{b,q(n)}$
m_b
$c_{b,1}$
...
$c_{b,q(n)}$

- Case I : $\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)$ are different from all the counter values used in encryption oracle queries
  - $f(\mathsf{CTR}^* + 1), \ldots, f(\mathsf{CTR}^* + q(n))$ will be uniformly random values, unknown to $\mathcal{A}$
  - The view of $\mathcal{A}$ will be the same as in $q(n)$ independent instances of OTP
  - $\Pr[\mathcal{A}$ outputs $b' = b$ in $PrivK^{cpa}_{\mathcal{A},\Pi_{\mathsf{CTR}}}(n)] \leq \frac{1}{2}$

And let us take the first case and try to analyze what is the probability that in that case, adversary is able to win the CPA game right. So we are in the case when the counter values namely CTR* + 1, CTR* + 2 and so on, which are actually used for encrypting the challenge plaintext m_b are different from all the counter values which has been used to respond to adversary's encryption oracle queries right, so we are in this case.

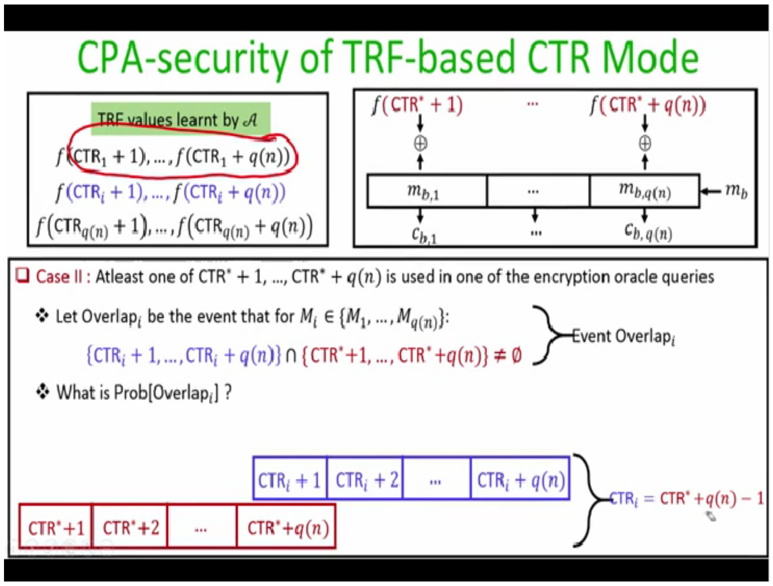
So this is the challenge plaintext m_b encrypted by evaluating first the truly random function on this counter values which have not been used so far in any of the encryption oracle queries to generate the pads. And those pads are XORed with the message blocks of the message m_b to generate the ciphertext blocks. So since this counter values CTR* + 1, CTR* + 2 and so on, is never repeated. That means from the viewpoint of any adversary the value of the truly random function on these new inputs which have never been used truly random.

Because that is what is the property of a truly random function. That means adversary does not know what are the pads which have been used to mask the blocks of the message m_b. That means what we can say is that from the viewpoint of the adversary, his view is as good as if he has seen q(n) number of independent instances OTP namely, the first block is encrypted by running an instance of OTP where the pad is completely unknown to the attacker.

The second block of m_b is encrypted again using a truly random pad, which is unknown to the attacker. And similarly the last block of the message m_b is encrypted by using a truly random pad unknown to the attacker. And as we have seen in the case of OTP, if the pads are truly random and unknown to the attacker, then it is perfectly indistinguishable or perfectly secure from the viewpoint of the attacker.

That means, if the counter values which have been used for computing the challenge ciphertext is not repeated, then we can safely conclude that the probability that adversary wins the CPA game is upper bounded by 1/2.

(Refer Slide Time: 22:47)



CPA-security of TRF-based CTR Mode

TRF values learnt by $\mathcal{A}$

$f(\text{CTR}_1 + 1), \ldots, f(\text{CTR}_1 + q(n))$

$f(\text{CTR}_i + 1), \ldots, f(\text{CTR}_i + q(n))$

$f(\text{CTR}_{q(n)} + 1), \ldots, f(\text{CTR}_{q(n)} + q(n))$

- Case II : Atleast one of $\text{CTR}^* + 1, \ldots, \text{CTR}^* + q(n)$ is used in one of the encryption oracle queries
  - Let Overlap_i be the event that for $M_i \in \{M_1, \ldots, M_{q(n)}\}$: $\{\text{CTR}_i + 1, \ldots, \text{CTR}_i + q(n)\} \cap \{\text{CTR}^*+1, \ldots, \text{CTR}^*+q(n)\} \neq \emptyset$ — Event Overlap_i
  - What is $\text{Prob}[\text{Overlap}_i]$?

Now let us take the second case when at least one of the counter values which has been used to encrypt any of the blocks of the message m_b has been repeated or has been used it to answer any of the adversary's encryption oracle queries, right. That is the second case. And we will now analyze what is the probability that the adversary can win the CPA game in this case. So for that, let me consider an event Overlap_i. So that is an event I am defining.

And the event Overlap_i denotes the event that for when the challenger is encrypting the i^{th} encryption oracle query from the adversary then the counter values which are used for encrypting the blocks of the message m_i gets overlapped with the counter values, which have been used by the challenger to prepare the challenge ciphertext.

So that is what Overlap_i means, Overlap_1 means the counter values which have been used for encrypting the blocks of the first encryption oracle query gets overlapped with counter values used for encrypting the message block m_b and so on. So there could be several ways in which the event Overlap_i could occur, right. So, $[\mathrm{CTR}_i + 1, \ldots, \mathrm{CTR}_i + q(n)] \cap [\mathrm{CTR}' + 1, \ldots, \mathrm{CTR}' + q(n)]$ is one possible way in which there could be an overlap between the counter values used for encrypting the blocks of the message m_i.

And the counter values which have been used to encrypt the blocks of the message m_b. Namely, it could be possible that the last counter value, which is used for encrypting the last block of the message m_i is the same as the first counter value, which is used to encrypt the first block of the challenge plaintext that could be one way in which overlap could occur right or it could be possible that the first 2 counter values which have been used to encrypt the first 2 blocks of the challenge plaintext overlaps are same as the last 2 counter values which have been used to encrypt the last 2 blocks of the message m_i that could be one way in which overlap can occur or it could be possible continuing like this.

If I analyze, it could be possible that another way in which Overlap_i could occur is that the first counter value, which is used to encrypt the first block of the message of m_i overlaps, or the same as the counter value, which is used to encrypt the last block of my challenge plaintext m_b. So these are the various ways in which the Overlap_i event can occur. So, let us understand the consequence of each of these cases right.

So, before going into the consequence, it should be clear to you that if at all the event Overlap_i occurs, that means there exists some indices j and j*, such that the counter value which is used to encrypt the j^{th} block in the message m_i and the j* block in the message m_b are same, and if this event occurs, then in this case, the adversary can clearly win the CPA game with probability 1 because in this case, the adversary will know the counter value, which has been used to encrypt the j^{th} block of the message m_i.

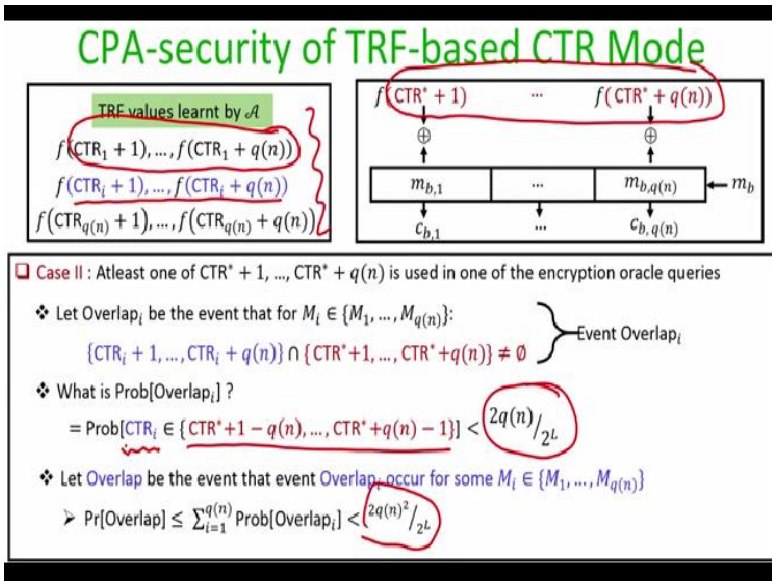
And adversary will see all the counter values which are actually used for encrypting the challenge plaintext m_b because those counter values are part of the challenge ciphertext. So, as soon as the adversary sees that there is an overlap, then based on the value of the truly random function, which it has already seen, by seeing the response of the encryption oracle query, it can easily identify what has been encrypted in the challenge ciphertext, right.

So, now we have to understand what exactly is the probability of the event Overlap_i happening. So as I said earlier, the event Overlap_i can occur in several ways. So if event Overlap_i occurs because of this case, namely, the last counter value of which is used for encrypting the last block of the message m_i is the same as the first counter value which has been used for encrypting the first block of the challenge plaintext.

If that is the way overlap is occurring, that means that the value of the i^{th} counter value should satisfy this relationship. Only if this happens, then the event Overlap_i could occur in this way right. In the same way if the overlap it occurs because of this condition, namely the last 2 counter values for the message m_i and the first 2 counter values for the challenge plaintext m_b overlaps. If that is that way Overlap_i is occuring that implies that the value of the counter i should satisfy this relationship.

And in the same way if I consider the scenario where the event Overlap_i occurs because of the fact that the last counter value used for encrypting the last block of the challenge plaintext overlaps with the first counter value used for encrypting the first block of the message m_i then for that case the CTR_i should satisfy this relationship. So, these are the various ways in which the event Overlap_i could occur.

(Refer Slide Time: 37:57)



CPA-security of TRF-based CTR Mode

TRF values learnt by $\mathcal{A}$

$f(\mathsf{CTR}_1 + 1), \ldots, f(\mathsf{CTR}_1 + q(n))$

$f(\mathsf{CTR}_i + 1), \ldots, f(\mathsf{CTR}_i + q(n))$

$f(\mathsf{CTR}_{q(n)} + 1), \ldots, f(\mathsf{CTR}_{q(n)} + q(n))$

$f(\mathsf{CTR}^* + 1)$
$f(\mathsf{CTR}^* + q(n))$
$m_{b,1}$
$m_{b,q(n)}$
m_b
$c_{b,1}$
$c_{b,q(n)}$

- Case II : Atleast one of $\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)$ is used in one of the encryption oracle queries
  - Let Overlap$_i$ be the event that for $M_i \in \{M_1, \ldots, M_{q(n)}\}$: $\{\mathsf{CTR}_i + 1, \ldots, \mathsf{CTR}_i + q(n)\} \cap \{\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)\} \neq \emptyset$ — Event Overlap$_i$
  - What is Prob[Overlap$_i$] ?

    $= \text{Prob}[\mathsf{CTR}_i \in \{\mathsf{CTR}^* + 1 - q(n), \ldots, \mathsf{CTR}^* + q(n) - 1\}] < 2q(n)/2^L$
  - Let Overlap be the event that event Overlap$_i$ occur for some $M_i \in \{M_1, \ldots, M_{q(n)}\}$
    - $\Pr[\text{Overlap}] \leq \sum_{i=1}^{q(n)} \text{Prob}[\text{Overlap}_i] < 2q(n)^2/2^L$

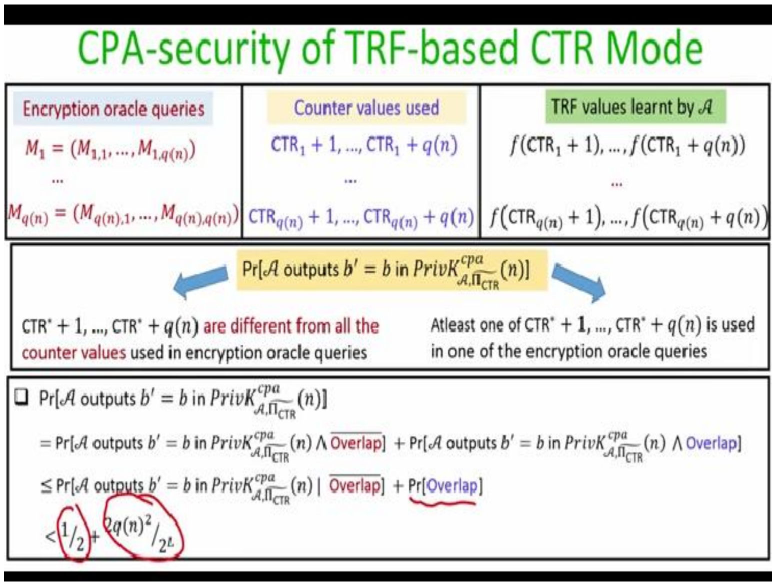
That means I can say that the probability of the event Overlap$_i$ to happen is the same as the counter value i, which is used to encrypt the ith encryption oracle query follows in this range, it should, it should lie in this range. If that if it lies in any of this range, then the event Overlap$_i$ could occur. Now, it is easy to see that what's the size of this range, I can always upper bound the size of this range to be 2 * q(n).

And since each of the counter values which are used in the modified counter mode of operation is selected uniformly randomly from the set of L bit strings, that is why I can say that the probability of event Overlap$_i$ to occur is always upper bounded by this quantity right. So that is the way event Overlap$_i$ could occur. Now our goal is to analyze that what is the probability that the counter values which are used for encrypting the challenge plaintext gets repeated in any of the q(n) number of queries.

So remember, adversary has made q(n) number of queries where each query actually consist of q(n) number of blocks. And whatever we have analyzed till now is was with respect to only one specific query, namely the encryption oracle query for ith message. So it is easy to apply the union rule here because each of this queries encryption oracle queries are independently answered by the challenger.

So if I apply the sum rule, I get at the probability that the event overlap occurs. So now the event overlap is defined to be the event that are count any of the counter values used for encrypting any of the blocks of the challenge plaintext gets repeated while answering any of the q(n) number of encryption oracle queries. So I can apply sum rule right. And by sum rule, I get to the probability that the event overlap occurs can be upper bounded by quantity : $(2\ q(n)^2 / 2^L)$.

(Refer Slide Time: 29:56)



CPA-security of TRF-based CTR Mode

Encryption oracle queries	Counter values used	TRF values learnt by $\mathcal{A}$
$M_1 = (M_{1,1}, \ldots, M_{1,q(n)})$	$\mathsf{CTR}_1 + 1, \ldots, \mathsf{CTR}_1 + q(n)$	$f(\mathsf{CTR}_1 + 1), \ldots, f(\mathsf{CTR}_1 + q(n))$
…	…	…
$M_{q(n)} = (M_{q(n),1}, \ldots, M_{q(n),q(n)})$	$\mathsf{CTR}_{q(n)} + 1, \ldots, \mathsf{CTR}_{q(n)} + q(n)$	$f(\mathsf{CTR}_{q(n)} + 1), \ldots, f(\mathsf{CTR}_{q(n)} + q(n))$

$\Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{\mathsf{CTR}}}(n)]$

$\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)$ are different from all the counter values used in encryption oracle queries

Atleast one of $\mathsf{CTR}^* + 1, \ldots, \mathsf{CTR}^* + q(n)$ is used in one of the encryption oracle queries

- $\Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{\mathsf{CTR}}}(n)]$

  $= \Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{\mathsf{CTR}}}(n) \wedge \overline{\mathsf{Overlap}}] + \Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{\mathsf{CTR}}}(n) \wedge \mathsf{Overlap}]$

  $\leq \Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi}_{\mathsf{CTR}}}(n) \mid \overline{\mathsf{Overlap}}] + \underline{\Pr[\mathsf{Overlap}]}$

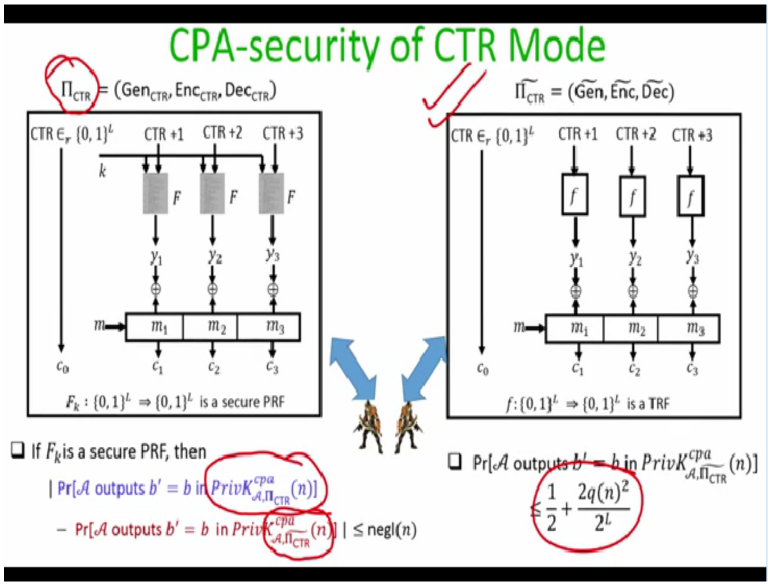
  $< \frac{1}{2} + \frac{2q(n)^2}{2^L}$

So now coming back to the answer that what is a probability of adversary winning the CPA game against a truly random function based encryption process, as I said earlier, it could be divided into 2 cases whether the counter value used for encrypting the challenge plaintext gets repeated or not. So overall, I can say that the probability that adversary wins the CPA game against a truly random function based encryption process can be written as the sum of 2 individual probabilities.

So $\Pr[\mathcal{A} \text{ outputs } b' = b \text{ in } PrivK^{cpa}_{A,\widetilde{\Pi}_{\mathrm{CTR}}}(n) \wedge \overline{\text{Overlap}}]$ is for the case when the event overlap does not occur. That means the event that the counter values that is used for encrypting the challenge plaintext never gets repeated, but still the adversary wins the CPA game. And the second probability is the probability of adversary winning the CPA game under the condition that Overlap has occurred right. That means one of the counter value has been repeated.

Now It is easy to ~~upper bound the first probability here as the probability of~~ sorry, it is always possible to upper bound the second probability here by the probability of the event Overlap occurring, right. So I am simply applying simple rules of probability here, the first probability term is retained as it is. And now we know both of terms here, the value of both the terms here, the first probability is basically 1/2,because if the event overlap does not occur, then we are is then the situation is the same as in the case of q(n) number of independent instances of OTP in which case we know that the probability of adversary winning the CPA game is 1/2. And the probability of overlap we have already computed to be ($2\ q(n)^2\ /\ 2^L$) quantity. So that is the maximum probability with which our adversary can win the CPA game against a truly random function based CTR mode of operation.

(Refer Slide Time: 32:02)



CPA-security of CTR Mode

$\Pi_{CTR} = (Gen_{CTR}, Enc_{CTR}, Dec_{CTR})$

$\widetilde{\Pi_{CTR}} = (\widetilde{Gen}, \widetilde{Enc}, \widetilde{Dec})$

$F_k : \{0,1\}^L \Rightarrow \{0,1\}^L$ is a secure PRF

$f: \{0,1\}^L \Rightarrow \{0,1\}^L$ is a TRF

- If F_k is a secure PRF, then

  $| \Pr[\mathcal{A}$ outputs $b' = b$ in $PrivK^{cpa}_{\mathcal{A},\Pi_{CTR}}(n)]$

  $- \Pr[\mathcal{A}$ outputs $b' = b$ in $PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi_{CTR}}}(n)] | \leq negl(n)$

- $\Pr[\mathcal{A}$ outputs $b' = b$ in $PrivK^{cpa}_{\mathcal{A},\widetilde{\Pi_{CTR}}}(n)]$

  $\leq \frac{1}{2} + \frac{2q(n)^2}{2^L}$

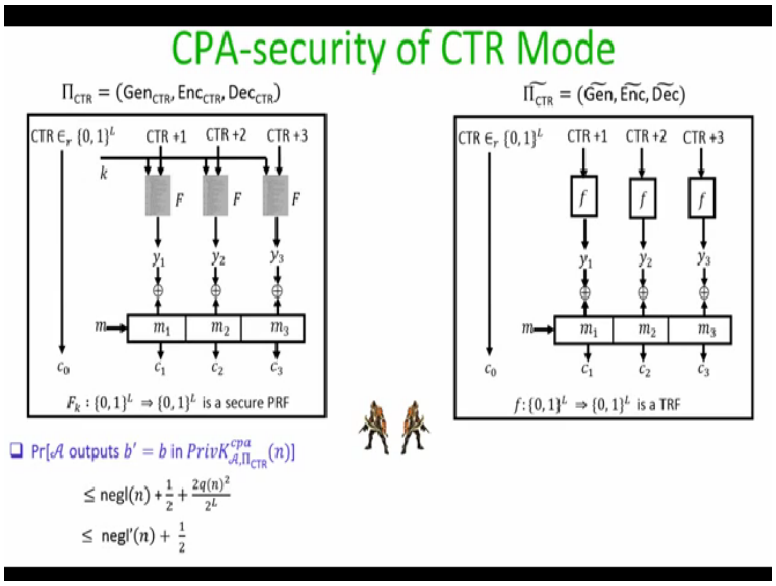
So, this part is done, we have analyzed this, but our actual goal is to show that the pseudo random function based counter mode of operation is CPA secure. So, what we are going to do now is, we are going to prove that if my underlying keyed function F_k is a secure PRF, then the absolute difference in the winning probability of the adversary absolute difference of the adversary winning the CPA game against the PRF mode against a PRF function based counter mode of operation.

And a truly random function based counter mode of operation can be upper bounded by a negligible quantity. That is what our next goal is. So this will follow more or less using the same argument that we had used to prove the CPA security of our first candidate, CPA secure scheme based on the pseudo random function, right. So remember, in that proof, we had first considered a hypothetical truly random function based construction, proved its CPA security.

And then we gave a reduction based argument where we showed that the difference in the winning advantage of the adversary between the 2 instances of the CPA game is upper bounded by a negligible probability if my underlying function is a pseudo random function. So we can use the same or similar argument and end up showing that the difference in the adversary winning the CPA game against the actual counter mode of operation.

And the adversary winning the CPA game against a truly random function based counter mode of operation can be upper bounded by a negligible quantity. And since we know that adversary winning the CPA game against counter mode of operation is 1/2 + (2 q(n)^2 / 2^L) by reshuffling the term we end up getting that the counter mode of operation using the pseudo random function is also CPA security right.

(Refer Slide Time: 34:05)



CPA-security of CTR Mode

$\Pi_{CTR} = (Gen_{CTR}, Enc_{CTR}, Dec_{CTR})$

$\widetilde{\Pi_{CTR}} = (\widetilde{Gen}, \widetilde{Enc}, \widetilde{Dec})$

$F_k : \{0,1\}^L \Rightarrow \{0,1\}^L$ is a secure PRF

$f: \{0,1\}^L \Rightarrow \{0,1\}^L$ is a TRF

❑ $\Pr[\mathcal{A}$ outputs $b' = b$ in $PrivK^{cpa}_{\mathcal{A},\Pi_{CTR}}(n)]$

$$\leq \text{negl}(n) + \frac{1}{2} + \frac{2q(n)^2}{2^L}$$

$$\leq \text{negl}'(n) + \frac{1}{2}$$

So that is what we end up showing, and which concludes that the counter mode of operation is actually CPA secure. So, that brings me to the end of this lecture. To summarize, in this lecture, we had seen 2 of the modes of operation, namely OFB mode and counter mode of operation.

And we had also seen the security proof of the counter mode of operation, the counter mode of operation has several attractive features.

Namely, the encryption and the decryption can happen in parallel. And we do not require any sophisticated assumption from the underlying keyed function, it is enough or suffice for the underlying keyed function to be just a pseudo random function. And the stateful variant is also CPA secure. And these attractive features make the counter mode of operation to be used rigorously in the real world protocol. Thank you.