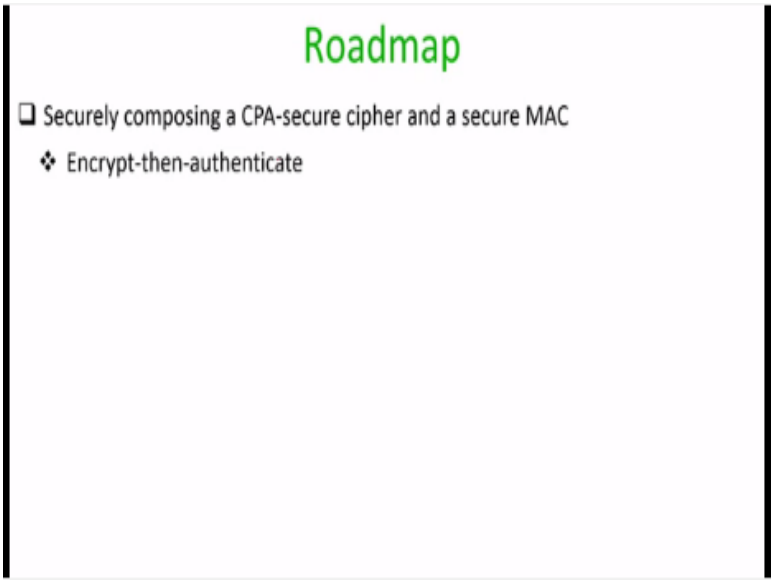
Foundations of Cryptography
Dr. Ashish Choudhury
Department of Computer Science
Indian Institute of Technology - Bangalore

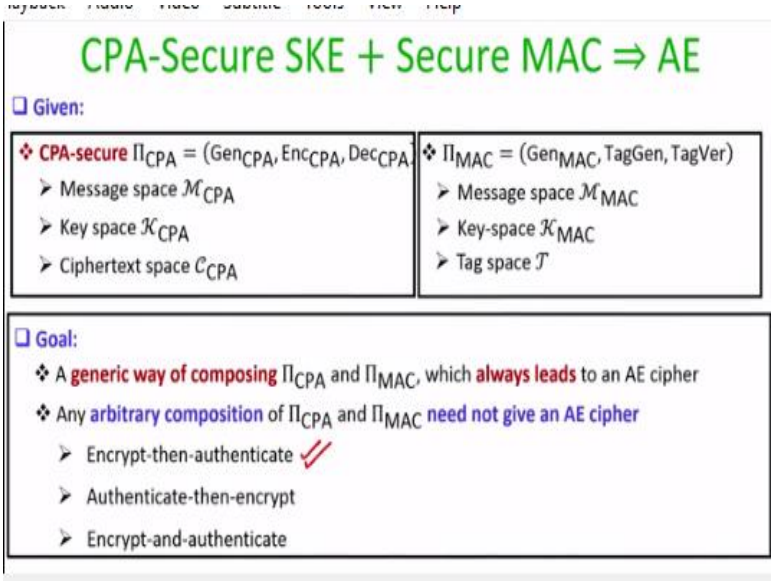
Lecture – 34
Composing CPA Secure Cipher with a Secure MAC Part I

(Refer Slide Time: 00:34)



Hello everyone, welcome to this lecture, just a quick recap. In the last lecture we had seen the definition of authenticated encryption scheme and the plan for this lecture is to construct one such authenticated encryption scheme namely, in this lecture we will see how to compose a CPA secure cipher and a secure MAC and obtain an authenticated encryption scheme. There are several approaches for composing a CPA secure cipher and a secure MAC. In this lecture, we will discuss one of those approaches of composing namely, the encrypt then authenticate approach.

(Refer Slide Time: 01:02)



So, the problem that we are interested to solve here is that we are given a CPA secure symmetric encryption and we are given a message authentication code and we want to combine them or compose them to obtain an authenticated encryption scheme. So, we have a secure Π_{CPA} = (Gen_{CPA} ,Enc_{CPA} , Dec_{CPA}) over its own plain text space, key space and a cipher text space.

And we have a secure Π_{MAC} =(Gen_{MAC} ,TagGen ,TagVer) and it has its own message space, key space and the tag space and the goal here is to come up with a genetic way of composing these 2 primitives such that the way we are generically composing these 2 primitives always leads to an authenticated encryption scheme.

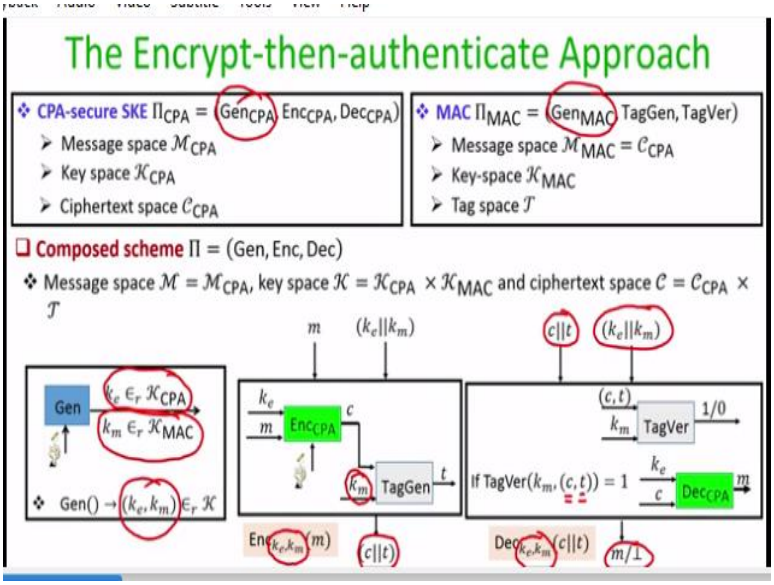
So, what I mean by generic way of composing here is that I would not focus on the underlying instantiation of Π_{CPA} and the underlying Π_{MAC}. We are just going to consider them as a black box, we assume that we are given some instantiation and arbitrary instantiation of a CPA secure scheme and an arbitrary instantiation of secure MAC scheme and our goal is to generically compose them such that we are always guaranteed to obtain an authenticated encryption scheme by that generic composition.

It turns out that there are several ways of composing an arbitrary Π_{CPA} secure symmetric encryption and an arbitrary secure Π_{MAC} but all those ways of composition need not give you necessarily an authenticated encryption cipher. So, what we are going to do in this lecture and

the subsequent lecture is that we are going to discuss 3 of the approaches and in this lecture; we are going to discuss only the first approach.

And we are going to show that this first approach always leads you to an authenticated encryption cipher whereas, a remaining 2 approaches need not always lead you to an authenticated encryption cipher.

(Refer Slide Time: 03:04)



So, let us discuss the first approach namely, the encrypt then authenticate approach. So just to recall what we have given here is; we are given an arbitrary Π_{CPA} secure symmetric encryption and an arbitrary secure Π_{MAC} and we are going to compose these 2 primitives to obtain an authenticated encryption cipher which has its key generation algorithm, encryption algorithm and encryption algorithm. The message space of this composed scheme will be the message space of the underlying CPA secure scheme, the key space of the composed scheme is going to be a pair of keys, were the first key will be picked as; where the first part of the key will belong to the key space of the CPA secure encryption process and the second part of the key will belong to the key space of the underlying message authentication code.

And a cipher text space of the composed scheme will again be a pair, where the first component of the pair will belong to the cipher text space of the CPA secure scheme and the second component of the cipher text will belong to the tag space of the underlying message

authentication code. So that is a message space, key space and a cipher tag space of the composed scheme.

Now, let us see how exactly the composing happens in this encrypt then authenticate approach, so the key generation algorithm of the composed scheme is as follows: it independently runs the key generation algorithm of the CPA secure scheme and output is key as per that key generation algorithm which I denote by k_e and independently it runs instance of the key generation algorithm of the message authentication code and obtains a key as per the key generation algorithm of the MAC which I denote by k_m.

And the overall key or the overall output of the composed key generation of the key generation algorithm for the composed scheme is considered to be the pair (k_e, k_m). I stress here that the key (k_e, k_m), they are independent of each other because they are obtained by running independently the key generation algorithm of the CPA secure scheme and the key generation algorithm of your MAC. We are assuming that these 2 primitives namely the CPA secure scheme and the MAC, they are independent of each other.

Now, let us see the encryption algorithm of the composed scheme, so imagine we are given a plain text *m* which we want to now encrypt as per this composed scheme and as per our notion, the key here basically consist of 2 parts; the first part of the key is for the CPA secure scheme and the second part of the key is for computing a MAC as per the message authentication code. So you can imagine it as a box here, which takes as input as the message and the key, and as the name suggest encrypt then authenticate, so what we are going to do first is we are going to encrypt the message as per the CPA secure scheme using the k_e part of the overall key for the composed scheme and it will produce a cipher text c.

Now the cipher text c is treated as a message for authentication and what we now going to do is; we are going to compute a tag for this c; creating the c as a message and the tag is now computed using the second part of the key namely k_m and whatever comes out of this tag generation algorithm, we denote it as t and now, the overall cipher text for this composed scheme will be the c part that we have obtained by encrypting the message and the t part that we have obtained by

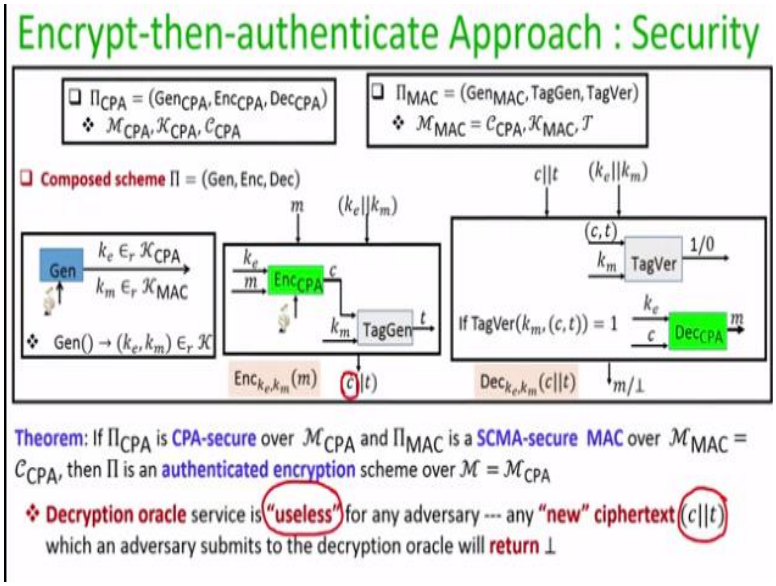
computing the tag on the c part, so that is how overall cipher text that we have obtained by this encrypt then authenticate approach.

Now, let us see how the decryption happens, so again the decryption algorithm will take a key which will further consist of 2 parts and it takes the cipher text which again consist of 2 parts; a c part and the t part. And it takes the key, now to decrypt the cipher text in the composed scheme what we do is; we first perform the tag verification. So we are just performing the reverse operation what we have done for the encryption in the encryption process, so in the encryption process, we have first encrypted and then we have computed the tag, while decryption we will just perform the operation in the reverse order.

So, we will first check if the TagVer(k_m,c) = 1. So if the output is 1, that means, the t part of the cipher text is indeed a valid tag on the c part of the cipher text, then what we do is we decrypt the c part of the cipher text using the k_e part of the overall key and recover the plain text. Whereas, if the output of the tag verification is 0, then we do not perform the decryption as per the CPA secure scheme be simply output box, so depending upon whether the tag verification internally outputs 1 or 0, we either end up outputting the plain text m or we output part indicating that the c concatenated with t, the cipher text that we are receiving in the decryption algorithm is an invalid cipher text.

So that is a way we are going to compose the CPA secure encryption and a secure MAC as per the encrypt then authenticate approach.

(Refer Slide Time: 08:49)



Now, we want to formally prove that if the CPA secure scheme; if the symmetric encryption that we have taken in this composition is CPA secure and if the MAC that we have taken in this composition is secure say, strong CMA secure, then irrespective of what exactly is the underlying instantiation of the Π_{CPA} secure scheme and what exactly is the underlying instantiation of the Π_{MAC} component, the overall composed encryption process that we have obtained is always an authenticated encryption scheme for a message space which is same as the message space of the underlying CPA secure scheme, so that is a theorem which we want to prove.

So, before going into the formal details of the proof, let us first try to understand, what exactly is the intuition, why exactly this theorem statement is true. So, trivially it is easy to see that the composed scheme is CPA secure because the composed scheme basically consists of a composed encryption process, it internally consists of the underlying CPA secure scheme. The only difference in the authenticated encryption scheme is that as per the definition of authenticated encryption, we require it to be having both cipher text integrity property as well as CCA security property.

So, I claim here that any cipher text (c*, t*) which is different from several legitimate cipher text which an adversary has seen in the past is always going to decrypt back to an invalid output and this is because if at all there is a malicious adversary who wants to forge a new cipher text, say

(c*, t*) which is different from previous cipher text that it has seen in the CCA experiment as a response from the encryption Oracle service.

Then that means that internally, adversary has to create or it has to compute a tag t* on the message c* under the unknown key k_m, only if that is the case that is possible for an adversary, then only the adversary could come up or it can forge a cipher text (c*, t*) which is different from all the legitimate cipher text that it has seen as a response from the encryption Oracle service.

But that will be a violation of the strong CMA security of the underlying MAC, so remember we are assuming that the underlying MAC is strong CMA secure and if it is strong CMA secure, then it is very unlikely that a poly time adversary based on several cipher text that it is seen in the past could come up with the new cipher text of the form (c*, t*), where t* is a tag on c*, where c* is treated as the message.

So that is the first observation here that means, if it is difficult for an adversary to come up with the new legitimate cipher text which decrypts to a non-null output, then it means that any CCA attacker from the viewpoint of that CCA attacker, the decryption Oracle service is completely useless because if at all it tries to create a new cipher text, it cannot do that because creating a new legitimate cipher text is as good as forging the underlying message authentication code which it can do only with very less probability.

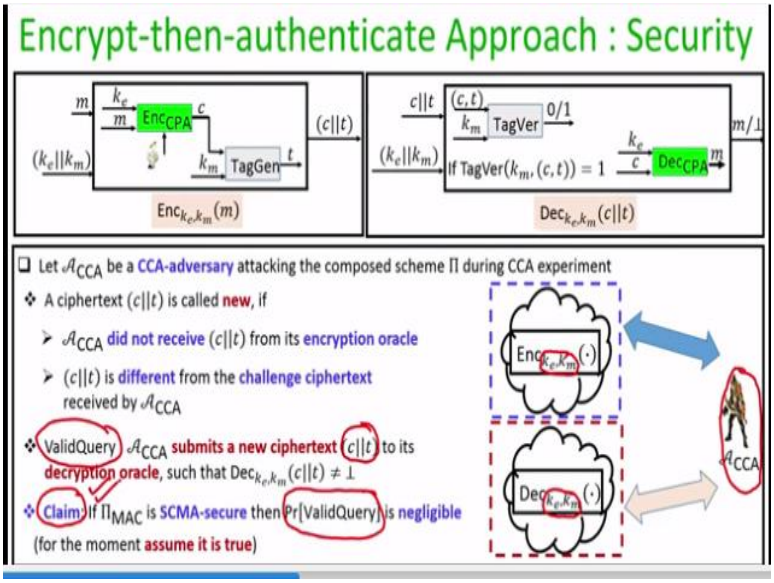
And that means that any new cipher texts or any new bit strings, c concatenated t which it tries to create and submit to the decryption Oracle service will always return back bot to him and that means decryption Oracle service is completely useless for such a CCA attacker and hence such a CCA attacker basically reduces to an instance of a CPA attacker and that automatically means that if our composed scheme Π is CPA secure.

That means, basically what we are now doing is; we are reducing the CCA attacker to a CPA attacker and since our composed scheme Π uses and underlying CPA secure scheme for computing the c part of the cipher text, it will be ensured that even c concatenated t will also be

CPA secure, it will give you CPA secure encryption. So that is how overall idea, the crux of the proof here is to basically show that the way we are composing the MAC and the CPA secure scheme, decryption Oracle service is going to be useless for any CCA attacker and any CCA attacker basically reduces to a CPA attacker and since we are using a CPA secure scheme internally to perform the encryption in the composed scheme, the composed scheme also will be CPA secure.

However, it turns out that even though this intuition is very simple to understand formally establishing this through reductions is the challenging task.

(Refer Slide Time: 13:44)



So, let us go into the details of the reduction proof here, so I am retaining the encryption process and the decryption process of the composed scheme and imagine that you have an arbitrary CCA attacker who is participating in an instance of the CCA experiment against this composed scheme Π, so as per the rules of the CCA experiment is adversary A_{CCA} will have access to both the encryption Oracle as well as the decryption Oracle service.

And what I have done here is I am showing the interaction of this adversary with the encryption Oracle service with the first box and with the second box, I am basically denoting the interaction of this adversary with the decryption Oracle service. When I am saying encryption Oracle service; why it is an encryption Oracle service because the adversary does not know the value of

the random k_e part and the k_m part of the key for the composed scheme which the encryption oracle is going to use to encrypt the plaintext which the adversary is going to submit.

In the same way, when the adversary is asking for the decryption of cipher text, adversary would not be knowing the k_e part and the k_m part of the key of the composed scheme which the decryption Oracle of the composed scheme is going to use to respond to the decryption Oracle queries of this adversary. So, now before going in to the formal reduction, let me introduce some definitions here.

So, in any instance of the CCA game, I will call a cipher text (c | t) to be a new cipher text, if the following conditions hold. The first condition is that A_{CCA} has not obtained the cipher text from its encryption Oracle and or the cipher text (c | t) is different from the challenge cipher text received by this adversary in the CCA game. So, if any of these 2 conditions hold, then I will call that cipher text (c | t) to be a new cipher text.

Basically, the definition of a new cipher text is that it is new in the sense that adversary has produced this string on its own that means, it has not seen this string as the encryption of any message of its choice which it has submitted to the encryption Oracle service or it is different from the encryption of the challenge pair of plaintext which the CCA attacker is going to submit in the CCA game.

If any of these 2 conditions hold, then I call this bit string (c | t) to be a new cipher text, now let me define in event which I called as valid query and this event valid query denotes the event that this CCA attacker A_{CCA} submits a new cipher text to its decryption Oracle such that the decryption of this new cipher text under the unknown key k_e followed by k_m is a legitimate output that means, its output is bot, so that is the event valid query.

So, I claim that if the underlying MAC which I am using in the composed scheme is strong CMA secure, then the probability with which the event valid query occurs is negligible, so we are going to prove this later. For the moment assume this claim is true, I also pause here for a moment and I would like to stress here that what exactly is the event valid query.

If you see closely the way I have defined the event valid query that is almost the same as the even that the CCA attacker could break the cipher text integrity property in the presence of the encryption Oracle service as well as in the presence of the decryption Oracle service because valid query means that this attacker A_{CCA} is able to come up with a new cipher text, which is different from any cipher text which it has seen as a response from the encryption Oracle service as well as the cipher text which is different; as well as it is different from the challenge cipher text such that its decryption is a non bot output.

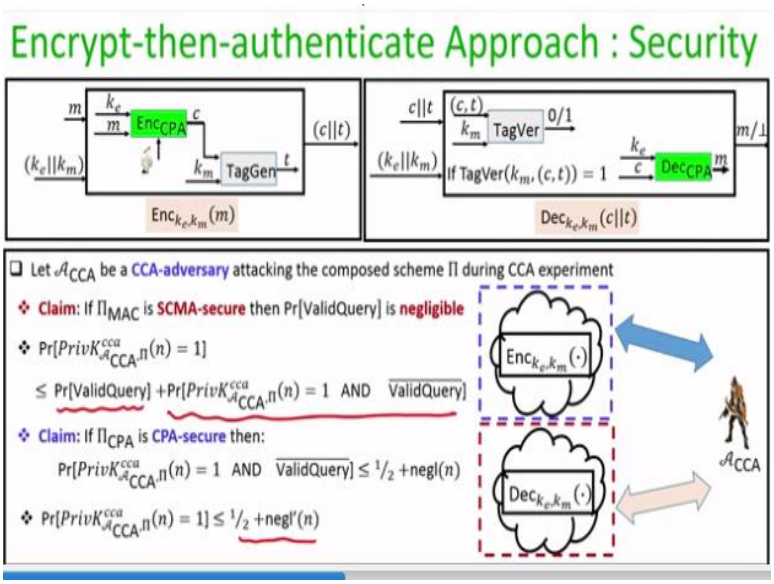
That is precisely what we require when we say that an adversary wants to break the cipher text integrity property but in the cipher text integrity property, the goal of the adversary is to come up with the new cipher text, a new valid cipher text, only based on the encryptions of messages of its choice that means, only in the presence of an encryption Oracle service.

But now, this event valid query is the event that the CCA attacker could come up with a forged cipher text whose decryption is a legitimate output, even the adversary has got access to the encryption Oracle service as well as decryption Oracle service. That means, if at all this claim which I am claiming here or making here is true that means, the probability that the event valid query occurs with the negligible probability, if this claim is true.

That automatically, implies that my encryption process that I have done; I have designed here by composing the CPA secure scheme and the MAC has also the cipher text integrity property that means, the probability with which the cipher text integrity property will be satisfied is always upper bounded by the probability with which the event valid query can occur in the presence of any CCA attacker.

And if I indeed prove that the probability that the event valid query occurs with the negligible probability, it automatically implies that the cipher text integrity property is also satisfied by my scheme. So we will again touch upon this fact later on when we will prove the cipher text integrity property of the composed scheme.

(Refer Slide Time: 19:28)



But for the moment, assume that this claim is true. So now I want to prove that the composed scheme that we have designed here is indeed CCA secure, so let us try to calculate that what is the probability that any CCA attacker attacking this composed scheme is able to win the instance of the CCA game that means, what is the probability that the output of the CCA experiment is 1?

The probability that this adversary can win the CCA game can be further divided into 2 cases namely first case where the adversary can win the CCA game under the presence of the event that event valid query occurs and second case where the adversary can win the CCA game even when the event valid query does not occur. That is an overall probability of the CCA attacker winning the CCA game.

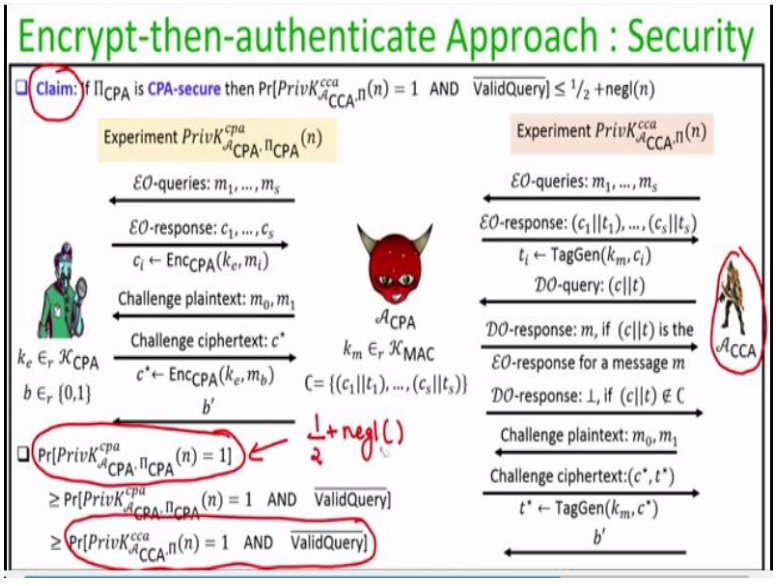
And then what I can do is that this first quantity here I can always upper bounded by the probability that event valid query occurs, so that is how I obtain an upper bound on the overall probability with which the adversary can win the CCA game. So that is an overall probability and what I am going to show next is that if my underlying symmetric encryption process in the composed scheme is CPA secure, then what I am going to prove is that the probability that the CCA attacker can win the game even in the absence of the event valid query is upper bounded by 1/2 + negligible.

Intuitively this is true because if the event valid query does not occur that means, all the new cipher text which the CCA attacker is going to submit to the decryption Oracle is going to give him back an output part, then basically this CCA attacker reduces to an instance of CPA attacker and is my underlying symmetric encryption process is CPA secure, then it automatically means that overall probability of the CCA attacker winning the game is upper bounded by 1/2 + negligible.

So, that is a claim I am going to prove next, so now assuming for the moment that indeed this claim is true, then I am already claiming that the probability with which the event valid query can occur is negligible and if I indeed prove this new claim which I am making here that the probability that this CCA attacker can win the game in the absence of the event valid query is upper bounded by ½+ negligible.

Then, I obtain that overall the probability that the CCA attacker can win the game is upper bounded by ½+ negl(n)+ some other negligible function and a sum of 2 negligible function is also a negligible function, so this will prove that the probability that CCA attacker is able to win the CCA game is ½+ negl(n) and hence the scheme is CCA secure. So, now let us see the proof of this new claim, which I am making here.

(Refer Slide Time: 22:33)



So, I want to prove that if the underlying symmetric encryption which I am using in the composed scheme is CPA secure, then this probability is upper bounded by ½+ negl(n) and this I am going to establish formerly through a reduction.

So, imagine an arbitrary CCA attacker who is playing an instance of the CCA game against the composed scheme Π, using such an attacker I am going to design another attacker which I denote it as A_{CPA}, whose goal is to win an instance of the CPA game against the underlying symmetric encryption process which is used in the composed scheme.

So, basically the goal of this adversary A_{CPA} is to win an instance of the CPA game against the underlying symmetric encryption process which I am using in the composed scheme. So in the left hand side part basically, of the overall reduction you have an instance of the CPA game but against the underlying symmetric encryption scheme which is used in the composed scheme and on the right hand side part of the experiment you have an instance of the CCA game. This adversary A_{CPA} is going to play a dual role; on the left hand side part of the reduction he is acting as an adversary in an instance of the CPA game, whereas in the right hand side part of the experiment he is acting as a verifier for an instance of the CCA game against the composed scheme.

So, now let us see how exactly the interactions happen in this reduction, so when this adversary A_{CPA} invokes the adversary A_{CCA}. The adversary A_{CCA} can demand for encryption Oracle queries, where it submits several plain text and asked for the encryptions of those messages as per the composed scheme, to respond to those queries, what adversary A_{CPA} does is it simply forwards those queries as if it wants to know the encryption of those messages as per the underlying symmetric encryption scheme.

Now, what the verifier for the CPA game is going to do is it is going to respond to this queries and to respond to those queries, it picks the key as per the key generation algorithm of the underlying symmetric encryption scheme which it I denote by k_e and it is going to encrypt those messages and the cipher text here which are returned back to the adversary A_{CPA} are basically the encryption of these messages m_i's under the key k_e.

Now, this adversary A_{CPA}, he has to act as a verifier and he has to respond to the encryption Oracle queries that he has seen in the CCA part of the experiment so, what it is going to do is to respond to the encryption Oracle queries, this adversary A_{CPA} itself picks a uniformly random key from the key space of the underlying MAC as per the key generation of the algorithm of the MAC component and what it is going to do is: it is going to compute tag t_i's on the c_i cipher text which it had seen from the verifier of the CPA part of the experiment of the reduction. Each of those c_i's concatenated with the t_i's are sent back as the response from the encryption Oracle service to the adversary A_{CCA}.

So, if you see closely here what is happening here basically, adversary A_{CCA} it expects the encryption of these messages m_1, m_2,.. m_s as per the composed scheme Π. And how exactly the encryption for these messages would have been computed, it would have been computed by first encrypting these messages as per the underlying symmetric encryption scheme, so that is what is the c_i part here and on top of that a layer of message authentication tag will be computed and that is what this adversary A_{CPA} itself is doing. So, if I just consider the view of this adversary A_{CCA}, it view is exactly the same as it would have expected when it participates in the instance of the CCA game and it sees the response of this encryption Oracle queries from the corresponding experiment.

So, distribution wise the probability distribution of the cipher text at adversary A_{CCA} finally sees in response to his encryption Oracle queries is exactly same as per an instance of the CCA game that it expects. The difference here is that now the adversary's interaction basically is with the CPA attacker who is computing the layer of MAC tag on the cipher text which it is seeing from the verifier or the experiment or the CPA experiment. So, that is how the encryption Oracle queries are handled by this adversary A_{CPA}.

Now suppose this CCA attacker ask for the decryption Oracle queries namely suppose, it submits a cipher text (c | t) and asked for the decryption of such cipher text. Now, what this adversary A_{CPA} has to do: he has to act as a verifier for the CCA experiment and he has to decrypt those cipher text. But it cannot decrypt those cipher text completely because for decrypting those

cipher text, you need to have both the k_m part of the key as well as the k_e part of the key. The k_m part of the key is available with this CPA attacker but the k_e part of the key is available with the experiment; with the CPA experiment which it cannot fetch because if it can fetch the k_e part of the CPA game, then trivially this adversary A_{CPA} can win the CPA game.

The challenge for this adversary A_{CPA} is basically to win the CPA experiment even without knowing the key k_e, so the way this adversary A_{CPA} is going to respond to this decryption Oracle queries is as follows: it sees whether this string (c | t) which has been submitted by the CCA attacker for the decryption Oracle service is already present in the list of the cipher text which this adversary A_{CPA} has sent back to the adversary in response to an encryption Oracle queries.

Basically, it checks whether the cipher text is a result of encryption of any of the previous messages for which the adversary A_{CCA} has asked the encryption Oracle service, if it so happened that indeed this is the case then what this adversary A_{CPA} does is; it just response back with the message m as the output of the decryption Oracle service for the cipher text (c | t).

On the other hand, if this adversary A_{CPA} sees that the cipher text (c | t) is a new string which is not present in the list of the cipher text which it has already sent as a response to the encryption Oracle queries to the adversary A_{CCA}, then basically the response for this decryption Oracle service for such new cipher text is bot, that is what is the strategy of this adversary A_{CPA} to respond to the decryption Oracle service.

Now, as per the CCA game, this adversary A_{CCA} will submit a pair of challenge plaintext (m_0, m_1) and what this adversary A_{CPA} does is it simply forward those plaintexts as the challenge plaintext in the CPA part of this reduction. What this CPA experiment does is it randomly decides one of the message for encryption and encrypt text message as per the key k_e and what this adversary A_{CPA} does is it adds its own layer of message authentication tag on c* treating c* as the tag as per the key k_m and that sent as the challenge cipher text to this adversary A_{CCA}.

Again if you see the view of the adversary that means, the probability distribution of the challenge cipher text and adversary A_{CCA} is seen is exactly the same as this adversary A_{CCA}

would have seen by participating in a real instance of the CCA game. The cipher text (c*, t*) would have been computed by first encrypting the message m_b under a key k_e, which is uniformly random.

And then follow it by adding a layer of tag under a key k_m, which is also uniformly random and that is how exactly the distribution of the challenge cipher text (c*, t*) that this adversary A_{CCA} is seeing in this experiment. Now, again this adversary A_{CCA} could ask for the decryption Oracle queries and to respond to those encryption Oracle queries, my adversary A_{CCA} is going to again follow the same strategy namely, it sees whether the decryption Oracle query which had been submitted is a response to the set of cipher text with adversary A_{CPA} has already sent as a response to the encryption Oracle services. If it so happens then the corresponding game is sent as the output otherwise, bot is sent as the output.

Once the adversary A_{CCA} has participated in all the phases of the CCA game, it outputs a bit b'. Namely, it identifies whether this challenge cipher text is an encryption of message m_0 or message m_1. Now using the response that this adversary A_{CCA} submits, my CPA adversary submits the same response b', namely it says that the challenge cipher text c* that it has seen is an encryption of the message $m_{b'}$.

So, before analysing the success probability of this CPA attacker that we have constructed in this reduction, let us see or let me highlight the way the CPA attacker has responded to the decryption Oracle queries that this adversary A_{CCA} is submitting. If you see closely here, then the way this decryption Oracle queries are handled, its consistent with that event valid query does not occur. So, recall the event valid query means that the adversary A_{CCA} submits a new cipher text which decrypts to a legitimate output. So, assuming that an event valid query does not occur, then any new cipher text (c | t) which has been submitted by the adversary is going to give the output bot.

And that is what exactly this adversary A_{CPA} is doing while responding to any decryption Oracle query which is a new cipher text. So, now I claimed that the probability that the CPA attacker wins the CPA game and the event valid query does not occur is exactly the same that the event

valid query does not occur in the CCA game and a CCA attacker wins the game. So, sorry for the typo here, there is a slight overlapping happening here, so what exactly this expression means is the expression I want to mean here is that I want to state here that the probability that this adversary wins the CPA game and the event valid query does not occur in the CPA game is exactly the same as the event valid query does not occur for this adversary A_{CCA} and this adversary A_{CCA} wins the CCA game.

These 2 probabilities are same and now what I can say is that from the rules of probability I can say that the probability that the adversary A_{CPA} wins the CPA game, I can always lower bounded by the probability that the adversary A_{CPA} wins the CPA game and the event valid query does not occur. Since this expression namely the adversary A_{CPA} wins the CPA game and the event valid query does not occur is exactly the same as the adversary A_{CCA} wins the CCA game and the event valid query does not occur for the CCA adversary.

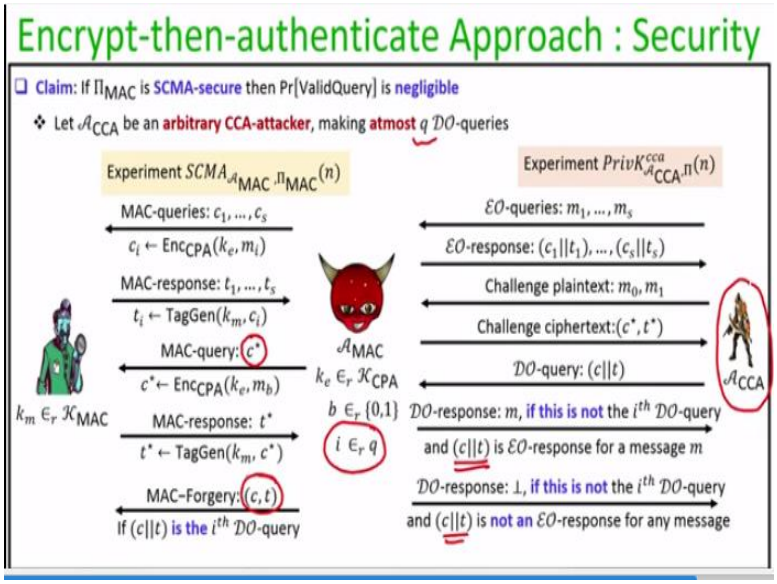
Overall, I can say that the probability that this adversary A_{CPA} can win the game is lower bounded by the probability that adversary A_{CCA} wins the CCA game and the event valid query does not occur.

Now, as per my assumption, the underlying symmetric encryption scheme that I am using in the composed scheme is a CPA secure scheme. That means, any adversary A_{CPA} the probability that it can win the CPA game is always upper bounded by some ½+ negl(n) that follows from the definition of CPA security of the underlying symmetric encryption scheme which I am using in the composed scheme.

That automatically means that the probability that the adversary A_{CCA} wins the CCA game and the event valid query occurs is upper bounded by ½+ negl(n) and that is what exactly the claim I wanted to prove, which I have now formally established through this reduction. That means, if for this adversary A_{CCA}, it cannot submit any new valid cipher text which can decrypt to a legitimate output.

Then, basically this adversary A_{CCA} becomes an instance of an adversary A_{CPA} and with whatever probability, this adversary A_{CPA} can win the CPA game with exactly the same probability this adversary A_{CCA} can win the game and that is what I have formally established through this reduction.

(Refer Slide Time: 36:42)



So that proves one part of the claim. The other part of the overall proof of the CCA security remains there. So our goal is to prove the second claim namely, I want to prove that if the underlying MAC component which I am using in the composed scheme is strong CMA secure, then the probability that the event valid query occurs is negligible.

Again, I establish this formally by a reduction namely, I am going to prove that if the event valid query does not occur with the negligible probability, then it implies that there exist MAC forger who can forge the underlying message authentication code which we are using in the composed scheme with a significant probability. So, let us see how exactly we establish that reduction.

So, again assume we have an arbitrary poly-time CCA attacker attacking the composed scheme and say it makes at most q number of decryption Oracle queries. If q is some polynomial function in your underlying security parameter, using this adversary I design a MAC forger which I denote as A_{MAC} for my underlying MAC component. So now in the overall reduction, this MAC forger is going to play a dual role: in the left-hand side part of the reduction it is acting

as an adversary in instance of the MAC forgery game or the strong CMA game against underlying MAC component whereas, in the right hand side part of the reduction it is acting as a verifier or basically, it is creating an instance of the CCA game against the composed scheme.

So as per the rules of the CCA game, this adversary A_{CCA} could ask for encryptions of several messages of its choice as for the composed scheme to respond to them. What does a MAC forger does is it itself picks a key for the underlying CPA secure scheme which I denote by k_e and it encrypts all the messages which have been submitted by this adversary A_{CCA} and those resultant cipher text are submitted as the messages for which this adversary A_{MAC} wants to know the tag.

So, remember in the CMA game this adversary can request and ask for MAC tag for several messages of its choice, so he is now basically submitting the cipher text which are again the encryptions of this underlying messages m_1 to m_s and it is basically requesting to some; or it is basically asking its experiment to submit the tags on these messages.

To respond to this tag request from the adversary A_{MAC}, what the experiment here does is it takes MAC key on its own or as per the key generation algorithm of the underlying MAC and it adds a layer of the tag on this messages and those tags are sent back as a MAC response from my experiment. Now what this adversary A_{MAC} does is; it takes the cipher text c_i's which it has computed on its own and adds the layer of tags which it has seen from the experiment from the strong CMA experiment and the resultant thing is sent back as the cipher text to the adversary A_{CCA}.

So what exactly is happening here; if you see the view of this adversary A_{CCA}, if it participates in a genuine instance or the real instance of the CCA experiment, the probability distribution for the encryptions of these messages as per the composed scheme will be exactly the same as the probability distribution of the cipher text that he is now seeing from this adversary A_{MAC}. Why? because in a real instance of the CCA experiment, the encryptions of these messages m_1 to m_s would have been computed as follows; first these messages would have been encrypted as per the underlying CPA secure scheme that is what this adversary is doing, by picking the key

himself uniformly randomly, then those encryptions or layer of tag would have been applied by picking the MAC key uniformly randomly.

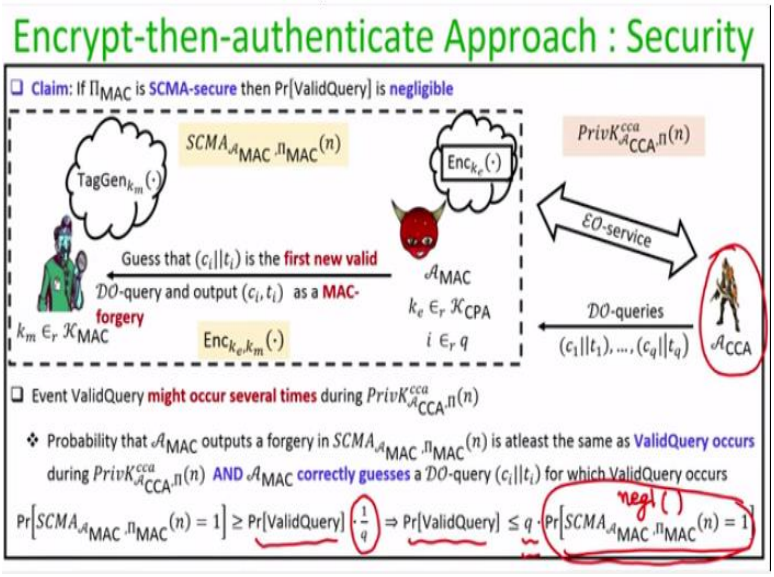
That is what this adversary is obtaining by taking the help of the CMA experiment by asking for the tags on the cipher text which he has computed and forwarding the cipher text concatenated with the resultant tags as the overall cipher text for the composed scheme to the adversary A_{CCA}. So distribution wise, the probability distribution that this adversary A_{CCA} is now seen from its interaction with the encryption Oracle query service is exactly the same as it would have expected in a genuine instance of the CCA game.

Now, this CCA adversary submits a pair of challenge plaintext. What this MAC adversary does is; it randomly picks one of them and it includes that message m_b and ask for the tag on the cipher text c*. The MAC experiment responds by computing a tag on that hence, once a tag is submitted to this MAC adversary, it forwards the (c*, t*) as the challenge cipher text to the CCA attacker.

Now, comes the interesting part; if this adversary A_{CCA} submits a decryption query. To respond to the decryption query what this MAC adversary does is the following; it randomly picks an index i from the set 1 to q, so remember q is the upper bound on the number of queries, DO queries that is adversary A_{CCA} can make. Now, to respond to this DO query, what this adversary A_{MAC} is going to do is the following.

If it seems that if this DO query is the i^{th} DO query and if this DO query is response to the encryption Oracle query for some message m, then just output m. Whereas if this is not the i^{th} DO query and it is a new cipher text, then the response of this adversary A_{MAC} is the bot, whereas if this i^{th} decryption Oracle query (c | t) is the i^{th} DO query, then what this MAC forger does is it simply submits a forgery (c | t).

(Refer Slide Time: 42:58)



So, let us try to understand what exactly is happening. So the way we have constructed the reduction if you see this adversary A_{CCA}, its interaction is basically with 2 individual entities but it does not know that it is actually interacting with 2 individual entities because that is underlying details of the reduction. He is basically interacting with a MAC adversary who has selected the encryption key for the CPA secure scheme and also with the verifier for the CMA experiment who has selected the MAC key.

Whenever he is asking for an encryption Oracle query, the responses are coming by the MAC adversary computing the encryptions and the CMA experiment computing the tags. But when it comes the decryption Oracle service, our MAC adversary cannot simply respond to those decryption Oracle service, because to respond to those decryption Oracle's queries, the MAC adversary need to know both the k_e part as well as the k_m part but it has access only to the k_e part.

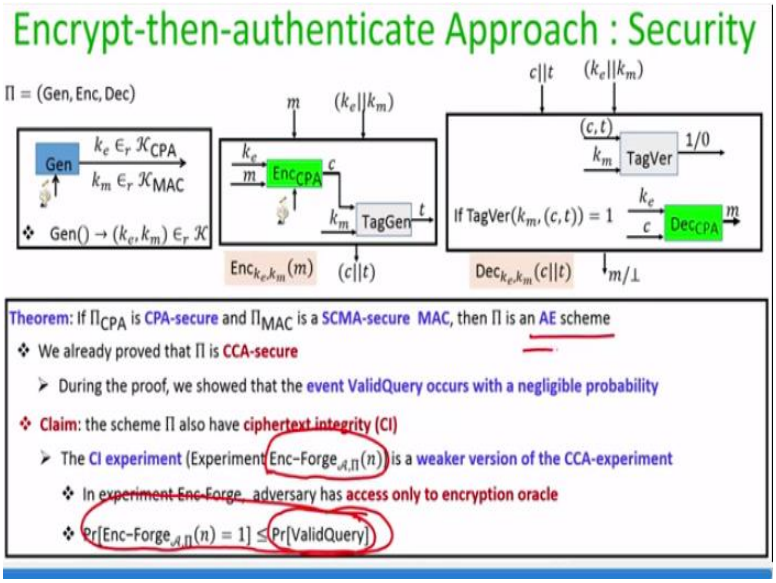
So basically what we have done here is this MAC adversary is just guessing that (c_i, t_i) is the first instance of the DO query where the (c_i, t_i) is a new valid cipher text. That means, it is a new cipher text which this adversary A_{CCA} has come up with and it is just guessing that the new cipher text is going to give him a legitimate output. Just thinking that indeed it is the correct first instance of new valid decryption Oracle query this MAC adversary is submitting that (c_i, t_i) as a MAC forgery in the MAC part of the overall reduction.

So, it turns out that if the event valid query occurs several times during the interaction with the adversary A_{CCA}, then the probability with which the MAC adversary could output a forgery in the CMA part or the CMA experiment of the overall reduction is at least the same with which the event valid query occurs for this adversary A_{CCA} and the guess of the MAC adversary A_{MAC} that indeed the i^{th} query (c_i, t_i) is the first instance when this adversary A_{CCA} is submitting a new valid cipher text is indeed the correct place.

So, that means I can say that the probability with which this MAC forger could win the CMA experiment in this reduction is at least the probability with which the event valid query occurs for this adversary A_{CCA} and the guess of the adversary A_{MAC} is correct which could happen with probability 1/q and that automatically implies that the probability that the event valid query occurs is upper bounded by q times the probability that the MAC adversary can win the CMA game.

But since we are assuming that our underlying MAC is a secure MAC, we know that this quantity namely the adversary A_{MAC} could win the CMA game is some negligible function because that is what is the definition of a CMA secure MAC. Since q is also some polynomial function in the security parameter, polynomial function in the security parameter multiplied by a negligible probability is going to give you a negligible function, which proves that the probability that the event valid query occurs is a negligible function.

(Refer Slide Time: 46:17)

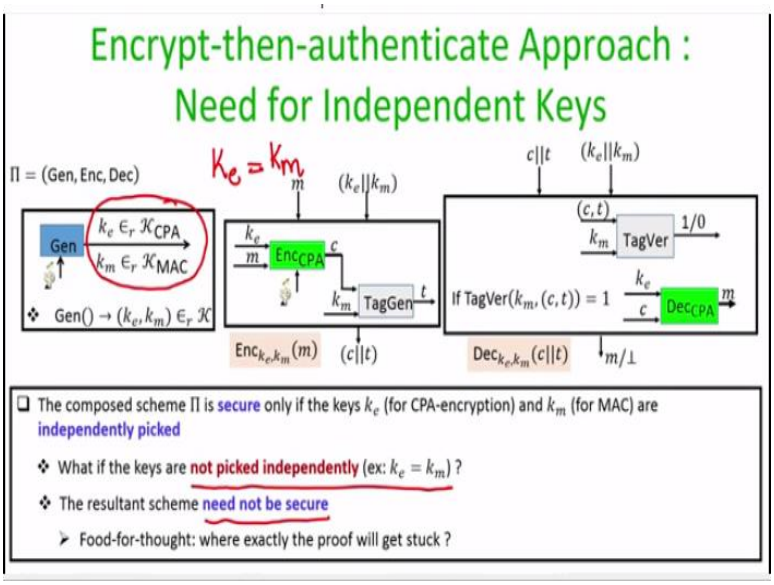


So now, let us come back to the original theorem which we wanted to prove that a composed scheme is indeed an authenticated encryption scheme. We had already formally proved that it is CCA secure and during the proof, we have formally established that the event valid query can occur only with a negligible probability.

What remains is to prove that this composed scheme also have the cipher text integrity property but again, this follows from the fact which I had discussed earlier that the cipher text integrity property in that which we formally established through the experiment Enc-Forge. The experiment Enc-Forge, the goal of the adversary to come up with a new valid cipher text only with the help of access to the encryption Oracle service. Whereas, we have proved that event valid query which can happen with negligible probability even in the presence of both encryption Oracle service as well as the decryption Oracle service.

That means, I can formally state that the probability with which any poly time adversary can win the Enc-Forge experiment or the cipher text integrity experiment against the composed scheme is upper bounded by the probability with which the event valid query can occur which I have already formally established which can occur with negligible function, so that proves the of cipher text integrity property. $Pr[Enc-Forge]_{(\mathcal{A},\Pi)}(n)=1] \leq Pr[ValidQuery]$

(Refer Slide Time: 47:41)



That means that the composed scheme is always an authenticated encryption scheme, so finally let me discuss why exactly we need 2 independent keys to compose the CPA secure component and the MAC component to obtain an authenticated encryption scheme. So if you see the key generation algorithm, I am operating the composed scheme with 2 independent keys, one for instantiating the CPA secure scheme and one for instantiating the MAC component.

It turns out that indeed picking up the independent keys is required to ensure the overall security of the composed scheme. So now you might be wondering what happens if we do not pick independent keys, what if I just operate the CPA secure part and the MAC part with the same key. It turns out as I am saying that the resultant scheme would not be secure, the resultant composed scheme would not be secure.

And it is a food for thought for you as an exercise for you to identify what exactly or what exactly can go wrong or where exactly the proof that we have given rigorously will get struck, if in the composed scheme instead of picking k_e and k_m independently, the key generation algorithm of the composed scheme picks just a single key both for instantiating the CPA secure component and for instantiating the MAC component.

I am leaving the detail as an exercise for you, so that brings me to the end of this lecture. In this lecture we have seen how to compose CPA secure symmetric encryption and a secure MAC and

using the encrypt then authenticate approach such that the composed scheme is always provably an authenticated encryption scheme, thank you!