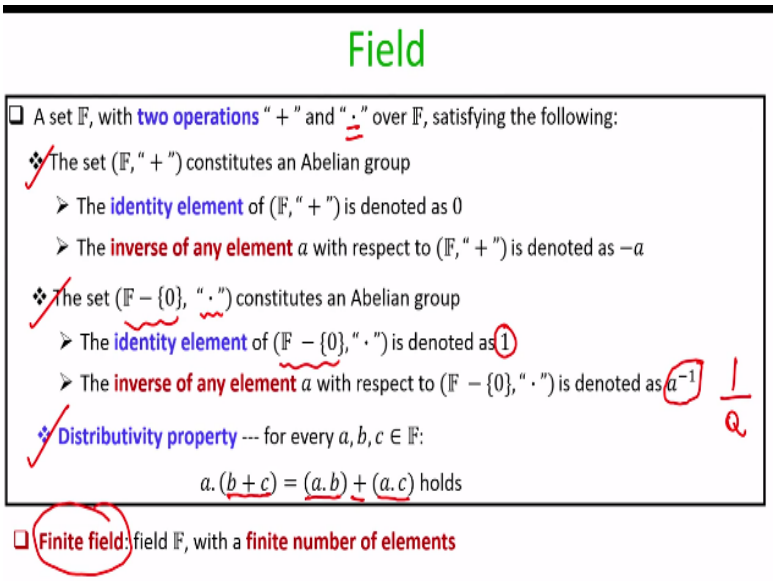
Foundations of Cryptography
Prof. Dr. Ashish Choudhury
(Former) Infosys Foundation Career Development Chair Professor
Indian Institute of Technology – Bangalore

Lecture – 26
Information - Theoretic MACs Continued

Hello everyone, so, this is a continuation of the previous lecture, and in this lecture, we will see the formal constructions of information theoretically secure message authentication codes.

(Refer Slide Time: 00:39)



So now let us define what we call as field. So, you can imagine that field is a special type of group because it will have now two operations which we denote as plus and multiplication. So, this plus and multiplication are notations, it does not necessarily mean the numeric or the integer plus and the integer multiplication. So you have a set F and 2 operations over the elements in the set F, namely (+, .) operations.

For F to be a field, the following properties should be satisfied, the first property that we require is that the set (F, +) should satisfy all the axioms of Abelian group, namely the closure property, associativity property, existence of the identity element, existence of the inverse element, and the commutativity property. What we do is that if indeed the (F, +) satisfies the axioms of the group, then the corresponding identity element we denote by 0, again the 0 is not a numeric 0.

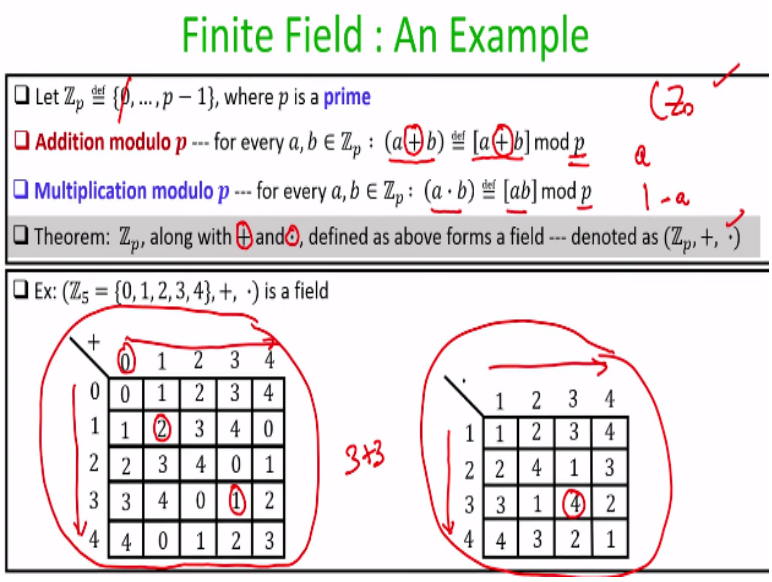
It is just a special label for the identity element, which is present in the set (F, +). In the same way, if the set (F, +) satisfies the group axioms, that means for every element a, there should be an additive inverse and that additive inverse we denote by -a. Again this -a should not be interpreted as the numeric –a, it is just a special label for the inverse of the element a with respect to +.

So, the first property that we require from F is that the set (F, +) should satisfy the axioms of the group and the second property that we require for the set F to be termed as a field is that if we take the set of all elements (F-{0}) or the additive inverse or the identity element with respect to the plus operation, then that set should constitutes a group with respect to "**.**" defined over the set of elements over the F. That means, the closure should be satisfied, the associativity should be satisfied, the existence of identity should be satisfied, every nonzero element in this set F should have a multiplicative inverse, and it should also satisfy the commutativity property.

So, if indeed the (F-{0}, **.**) satisfies the group axioms that means we should have an identity element, which we denote by 1, I stress here this one does not mean the numeric or the integer one. It is just a label for the special nonzero identity element present in the set F, which constitutes the identity element with respect to the dot operation and since we will have the inverse of every nonzero element with respect to the dot operation that inverse, which we term as the multiplicative inverse will be denoted by this notation. Again, this does not necessarily stand for 1/a because as I said that this dot operation is an abstract operation, it cannot be necessarily the numeric multiplication operation.

So, the 2 properties that we require from the F with respect to + and **.** are as follows: we require (F, +) to constitute a group and we require a set of nonzero elements to constitute a group with respect to the dot operation. The third property that we require from a field is the distributivity property, namely we require that the dot operation should be distributive over the plus operation. That means a **.** (b + c) = a **.**b + a **.** c.
If that is the case, then we say that the set F along with operation plus and operation dot constitutes a field and we will be interested in finite field and basically finite fields are those fields where the set F consist of finite number of elements.

(Refer Slide Time: 05:00)



Let us see our candidate example for finite field because our construction of SUF will be based on a finite field. So let me define the set Z_p to consist of all the integers in the range zero to p-1 where p is the prime. Now let me define a special type of plus operation over the set Z_p. The plus operation is defined as follows: a+b := (a+b) mod p.

So whatever remainder you obtained by dividing the number a + b with respect to the modulus p that remainder will be called as the summation of a and b in this set Z_p that is the way I am defining my plus operation. So you can clearly see that this plus operation is defined in a special way. That is not simply the integer plus operation and let me define a corresponding dot operation over the said set Z_p.

So the dot operation is defined as follows: a **.** b := (a **.** b) mod p. It is easy to see, we can in fact prove that there is a well known theorem which states that set Z_p along with this plus operation that we have defined and with respect to the dot operation that we have defined satisfies the axioms of field. Namely, we can prove that the set Z_p is an Abelian group.

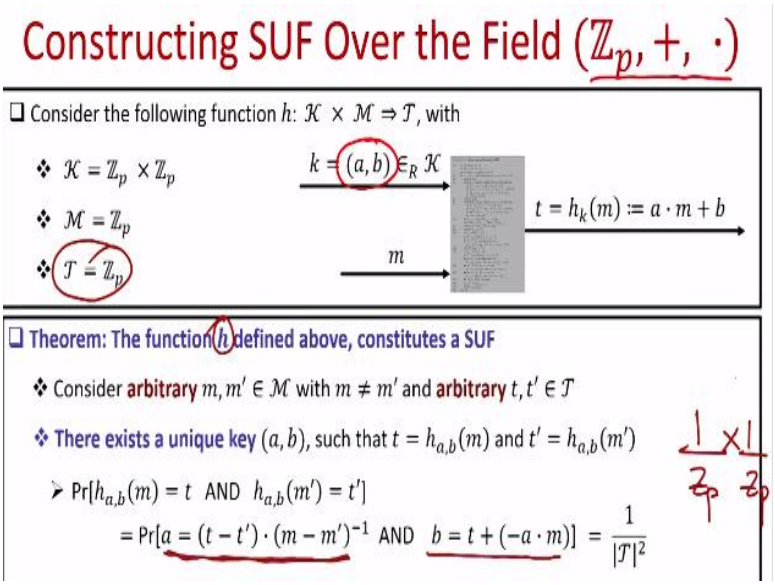
So, you can easily see that it satisfies the closure property because if any 2 numbers from the set Z_p and add it as per this definition, the remainder ϵ {0,..p-1}. So closure is satisfied with respect to plus. The way we have defined plus will satisfy associativity property. The element 0 ϵ Z_p, the numeric 0, and indeed it constitutes the identity element with respect to this plus operation.

For every element 'a' ϵ Z_p, we have a corresponding element p-a also present in Z_p such that the summation of 'p-a' and 'a 'with respect to this plus operation will give you the identity element namely 0 and the plus operation is commutative. That means, the set (Z_p, +) satisfies the group axioms and in the same way we can prove that if we consider the set of nonzero elements, namely Z_p –{0} and focus on {1,…p-1} and consider the dot operation, then again it satisfies that group axioms and we can prove that this plus operation is satisfying the distributivity property with respect to this dot operation. That means, you can prove that this set Z_p along with this plus operation and dot operation constitutes a field and if you want to verify that, let us see an example here.

So, I am taking my prime p= 5. So, that means Z_5 = {0, 1, 2, 3, 4} and what I have done here is I have done the table of the plus operation, the way plus operation will operate on the set {0, 1, 2, 3, 4}. The (i, j)th entry w.r.t "+" denotes the result of i + j where plus is defined as i + j modulo p. So, for instance 1+1 mod 5 = 2**.** In the same way, 3+3 = 6 mod 5 = 1 and now you can verify from this table that you always get an answer which belongs to the set 0, 1, 2, 3, 4.

So closure is satisfied, you have 0 as the identity element, every element has a corresponding additive inverse and so on. The (i, j)th entry w.r.t "**.**" basically denotes i **.** j mod 5, where i and j belongs to the set of nonzero elements of Z_5 and again you can see that all the group axioms are satisfied for the nonzero elements 1, 2, 3, 4.

(Refer Slide Time: 09:43)



So, now we will assume that we are given field namely Z_p and two operations plus and dot and our goal is to basically construct an SUF because once we have an SUF, then we know how to construct or use it to construct information theoretically secure MAC. So, the SUF that we are going to design is as follows: the key space will be the Cartesian product of Z_p and Z_p, namely the key will consist of 2 elements from the set Z_p and the message component of the SUF that we are going to design will be an element of the set Z_p.

So, the message space ϵ Z_p and the output space of the SUF that we are going to construct ϵ Z_p, namely the output will be an element from the set Z_p and the construction is as follows: so, as I said that the key will consist of 2 elements from the set Z_p, let me denote them by (a, b) and the key will be uniformly and randomly picked from the key space. That means a ϵ Z_p is going to be any uniformly random value and b is independent of a and uniformly distributed over the set Z_p. So that is the key.

The message m ϵ Z_p and if you want to compute the value of the SUF with the key k on this input m, a **.** m + b where the dot and the plus operation corresponds to the dot and the plus operation over this finite field Z_p i.e a **.** m = a **.** m mod p and + w.r.t mod p.

So, the dot and plus operation are the dot and plus of the corresponding field, here which is Z_p in this case. So, basically the way to understand this SUF is as follows: The key here you can imagine as a straight line. So, remember a straight line has an equation of the form y = mx + c, where m is the slope of the line and c the intercept of the line. So, (a, b) basically you can imagine as the (m, c) that defines a straight line and the output of this SUF you can imagine as the point on the straight line determined by the value x = m.

So, if I substitute a value of x here, I obtain a corresponding point on the line defined by the slope m and the intercept c. So, that is the way I am computing the value of this SUF. So that is a simple way to interpret the construction of this SUF. Now, we have to prove whether indeed this construction constitutes a SUF or not and we are going to claim here that the function *h* that we have defined like this indeed constitutes an SUF.

So, remember, the definition of SUF is that for every pair of messages (m, m'), m $\neq$m' and for every Tag-Ver (t, t'), pr[h_k(m)=t] = pr[h_k(m')=t'] under an unknown key k, for every key

k. So, to prove that let us consider an arbitrary (m, m'), m ≠m' and an arbitrary pair of candidate tag from the tag-space or output space.

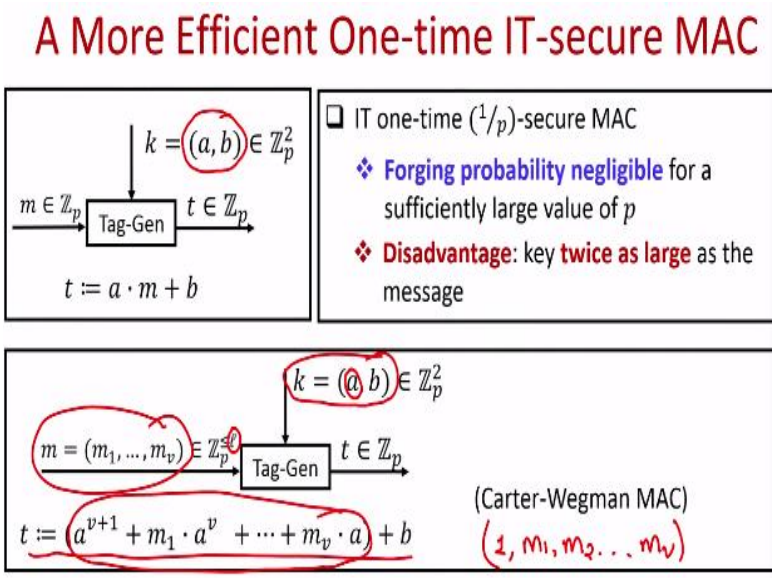
The claim here is that there always exists a unique key which I denote by (a, b) such that $h_{a,b}(m)=t$ and $h_{a,b}(m')=t'$. This is because if indeed t has to be the output of the SUF *h* that we have constructed for the input m with respect to the key (a, b), then this equation should hold and in the same way, if t' is indeed output of the SUF *h* for the input m' under the key (a, b), then this condition has to hold.

So, put in a different way, basically both these equations together imply that if indeed I want that $h_{a,b}(m)=t$ and $h_{a,b}(m')=t'$, then both these conditions should hold simultaneously. That means, a should take this value and b should take this value. So, notice that this $(m - m')^{-1}$ basically that denotes the multiplicative inverse because we are doing all this plus operation, dot operation over the field.

We should consider this $(m - m')^{-1} \neq 1/(m - m')$. No, we are not doing the integer arithmetic here, we are doing the finite field arithmetic here. So, that means (a, b) should take these 2 values and if I put everything altogether in a different context, what I basically require here is that if at all the probability that indeed the output of my SUF for input m is t and the output of SUF on the input m' is t' under the key (a, b) is indeed equal to t' holds.

For that, I require a to take this value and my requirement b to take this value, but remember that the key (a, b) that I have chosen for the SUF are uniformly distributed over the $Z_p \times Z_p$. That means, a is uniformly selected over the set Z_p and so is b. So, what is the probability that my a indeed occupies this value? Well, it is 1/tag-space or $1/ Z_p$, and what is the probability that indeed my b also takes this value that is also $1/ Z_p$. Together the probability that a takes this value and b takes this value will be the product of $1/ Z_p * 1/ Z_p$, which is same as $1/\Gamma^2$ because my tag-space is also nothing but the Z_p and that proves the theorem that indeed the function *h* that we have constructed here constitutes an SUF.

(Refer Slide Time: 16:29)



So even though now we have a candidate construction of one-time information-theoretic secure MAC. The information-theoretic secure MAC that we have constructed, there the success probability of forgery is 1/p because the success probability was of the generic construction that we had, there the success probability of the forgery was 1/Γ, since the tag-space here is nothing but Z_p and $|Z_p| = p$, though the candidate information-theoretic one-time secure MAC that we have constructed, there the success probability is 1/p.

So, if we ensure that p is significantly large, then the forging probability of the adversary becomes negligible, but the disadvantage of this construction is that the key that we are using here is twice as large as the message. Namely the key ϵ Z_p x Z_p, but the message ϵ Z_p.

So, what we will now do is we will see a more efficient construction of one-time information-theoretic secure MAC where we will be authenticating a message of a larger size, namely which will consist of several elements from Z_p using the key (a, b). However, before going into that, let us try to understand that why the MAC that we have constructed using the candidate SUF over the field is only one-time information-theoretic secure. So, the tag generation algorithm here is basically the value of the straight line defined by the slope a and intercept b on the input m.

Now, imagine with respect to the same key k, sender authenticates 2 messages, m and m'? So sender authenticates the message m and the tag will be t= a . m + b and imagine that instead of just using the key (a, b) for authenticating a single message, sender ends up

authenticating another message namely to reuse the key for authenticating another different message m' and the tag for that will be t' = a . m' + b.

Now, imagine that there is an adversary, which observes the message and this corresponding tag and the same adversary observe this message m' and the corresponding tag t'. Now it is very simple for the adversary to recover back the key consisting of (a, b) because from the viewpoint of that adversary, there were 2 unknowns, namely a and b and now he is getting a system of linear equations and 2 unknowns and he has two such equations which are linearly independent and he can solve the system of linear equations over the finite field and end up recovering a, b which is the key.

Once the adversary recovers the key (a, b), it can create a forgery on any new message and forward it to the receiver. So that is why the security of the MAC that we have constructed using the SUF over the finite field gives you the security guarantee for authenticating only a single message. Using a key (a, b) you can authenticate only a single message, you cannot reuse the same key for authenticating more than one message.

So now coming back to our earlier question that is it possible to modify the one-time information-theoretic secure MAC where we can authenticate a larger size message using the same key (a, b) and it turns out that it is possible to do that and that modified MAC is also popularly called as Carter and Wegman MAC and what the MAC does is it allows you to authenticate message consisting of up to l elements from the field Z_p.

So, imagine that you have a message m consisting of say v number of elements from the field, where v could be any value in the range 1 to l and the key for authentication is (a, b). Now, the tag generation algorithm is as follows: what a tag generation algorithm does is it computes this value. So, you can interpret this value t to be as follows. So, what exactly is this? If you treat this particular thing which is there in the bracket, you can interpret it as the value of a polynomial of degree v+1, where the coefficients of the polynomial are 1, m_1, m_2, up to m_v.

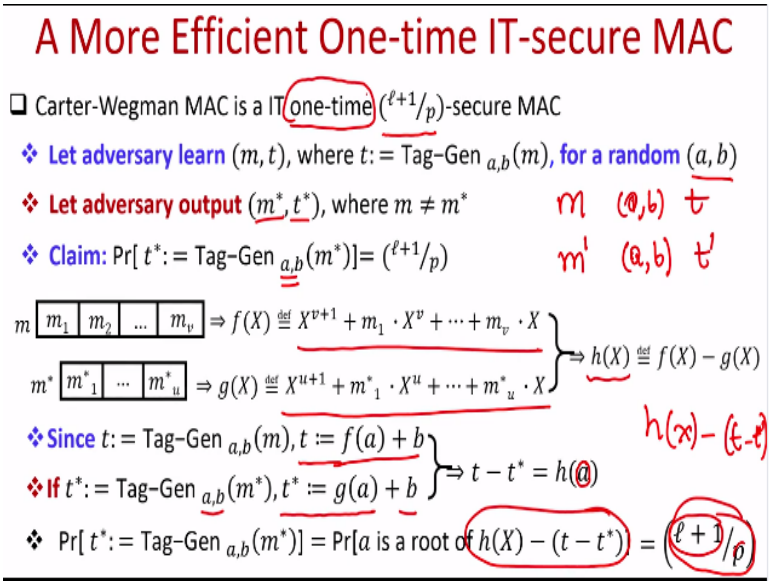
So you have a polynomial in a variable say x of degree v+1 with these coefficients and basically you are evaluating that polynomial at a and adding the value b and that give you the tag on the message that you want to authenticate. So that is a way you can interpret this

Carter-Wegman MAC and in some sense basically, this is a generalization of the one-time IT-secure MAC that we have just constructed. In the previous construction, there was a straight line which you can imagine, a polynomial of degree 1 because you have a message just consisting of a single element.

But now you have a message =(m_1,..m_v), v elements for authenticating that you are actually defining a polynomial of degree v+1 and t:= a^{v+1}+m_1 a^v+m_2 a^{v-1}+..+b to obtain the overall tag. Now, you might be wondering that even though you have v number of elements in the message, why you are defining a polynomial of degree v+1. I leave it as an exercise for you to identify what happens if I define a polynomial of degree v instead of v+1 and end up evaluating it on the key (a, b) and obtain the tag.

It turns out that if we modify this construction and instead of defining a polynomial of degree v+1, we take a polynomial of degree v, then the resultant construction is not going to give you a secure MAC.

(Refer Slide Time: 22:37)



So, now, I claim that the Carter-Wegman MAC that we have constructed gives you one-time authenticity, where the success probability of forgery is (l+1)/p. That means, imagine there is an adversary which learns the value of tag as per the MAC that we have constructed over some message and where the message m is also known, but the key (a, b) is not known to the adversary and now suppose the adversary wants to come up with a forgery on a message m* which is different from m and for producing the forgery, it also outputs the corresponding tag, which I denote by t*.

Now, our goal is to analyze what is the success probability that (m*, t*) indeed constitutes a valid forgery. That means what is pr[t*= Tag-gen$_{a,b}$(m*)]? I claim that the probability of this happening is a (l+1)/p. For this, let us first try to understand that if I have a message m for which the tag is t, then you can say the message m consist of v number of elements from the field, where v is anything in the range 1 to l.

Then the corresponding polynomial defined by the elements of m of degree v+1 will be: $f(X)= X^{v+1}+m_1 . X^v+m_2 . X^{v-1}+..+m_v . X$ this and in the same way imagine the message m* $g(X)= X^{u+1}+m^*_1 . X^u+m^*_2 . X^{u-1}+..+m^*_u . X$. Now, based on these 2 polynomials, let me define the difference polynomial which I call us h(X), which is the difference of the polynomial f(X) and g(X). Now, since that t value is the tag of the message m under the key a, it holds that the t is nothing but the value of the f polynomial on the input a + b because that is how the Carter-Wegman would have been computed on the message m, and if you want that t* should also be a MAC on the message m*, Carter-Wegman MAC on the message m* under the same unknown key (a, b).

Then it should hold that t* should be the value of the g polynomial on the input a + b. That means together both this condition if they hold that means that t-t* should be the value of the h polynomial or the difference polynomial on the input a. Now, that means the probability that t* is indeed the Carter-Wegman MAC for the message m* under the key (a, b) for that to hold, it should hold at the value a or the a part of the key should constitute a root of the polynomial h(x) – t- t*. Because if you see a should constitute a root of the polynomial h(x) - t – t*.

If I take this polynomial and if indeed a turns out to be a root of this, then indeed we get that t is equal to the MAC of message m and t* is a MAC of message m*, but what exactly is the degree of the polynomial h(x) - t – t*? Well, it is a polynomial of degree up to l+1 because remember my v, the number of elements in the message could be anything in the range 1 to l, so in the worst case it could be l. In the same way, the number of elements in the message m* could be up to l.

That means, the degree of f(x) polynomial could be anything in the range 1 to l+1 or 0 to l+1 and then same way, the degree of g(x) could be maximum l+1. That is why the difference

polynomial h(x) could have a degree of $2l+1$ and if I have a difference polynomial whose degree is $l+1$, then it can have up to $l+1$ roots. So basically, for t* to be the tag of the message m* under the key (a, b), it should be the case that a should be one of the $l+1$ possible roots of the polynomial h(x) whose degree is $l+1$.

For the probability that indeed a is one of those roots is $(l+1)/p$ because there are $l+1$ candidate roots and the root could be any value from set Z_p. So that is how we get the success probability of forgery for this Carter-Wegman one-time MAC. Again, it is interesting to see that why exactly this MAC is one-time secure, why cannot we authenticate 2 messages m and m' under the same unknown key (a, b).

It turns out that if we try to authenticate m with the key (a, b) and obtain a tag t as per the Carter-Wegman scheme, and if I have another message m' and if I authenticate the same message by reusing the same key as per the Carter-Wegman process and obtain the tag t', then basically adversary learns a lot of information about the key (a, b). Basically it can recover the full key (a, b) by solving a system of 2 linearly independent and once it learns the key (a, b), it can come up with a forgery on any message m* and forwarded to the receiver which will be accepted.

So that is why using this Carter-Wegman MAC, we can authenticate only one message, we cannot reuse the same key for authenticating multiple messages and it turns out that formally we can prove this. Forget about Carter-Wegman MAC, if you have any kind of information-theoretic secure MAC, then in order to authenticate multiple messages, you have to proportionately increase the size of the key, you cannot have a key of some fixed set size and expect that your scheme is information theoretically secure and at the same time gives you the guarantee of authenticating arbitrary number of messages.

Depending upon the size of the key, you get the restriction on the maximum number of messages that you can authenticate in the presence of a computationally unbounded adversary. This is in contrast to computationally secure MACs where we can use the same key to basically authenticate arbitrary large number of messages, polynomial number of messages because there the adversary is computationally bounded, but as soon as we go to the computationally unbounded world, we get the restrictions on the number of messages which you can authenticate using the key.

So, that brings me to the end of this lecture. Just let me summarize what we have discussed in this lecture. In this lecture, we have seen the constructions of information-theoretic secure MACs, which gives you the security guarantee even against a computationally unbounded adversary. However, if the adversary is computationally unbounded, we cannot get the zero forgery guarantee that means there is always a nonzero error probability which will be associated in the scheme because adversary can always guess the value of the tag on our message which was never authenticated by the sender in the previous sessions.

We also saw a generic construction of information-theoretic secure MAC given strongly universal functions and we have seen the construction of a strongly universal function based on finite field algebra and we had also seen a very efficient information-theoretic secure MAC, namely the Carter-Wegman MAC. Thank you!