

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

Lecture-04
Perfect Security

Hello everyone, welcome to lecture 4.

(Refer Slide Time: 00:28)

Roadmap

- ❑ Perfect-secrecy
 - ❖ The first formal notion of security
- ❑ Various equivalent definitions of Perfect-secrecy



So plan for this lecture is as follows, we will discuss perfect secrecy which is the first formal notion of security and which is also the strongest notion of secrecy. And we will also discuss various equivalent definitions of perfect secrecy.

(Refer Slide Time: 00:45)

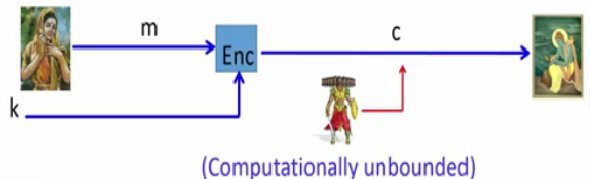
Perfect Security

Shannon C. E, A Mathematical Theory of Communication, Bell system technical journal, 1948



☐ Also called Unconditional security, Information-theoretic security

☐ Attack model : ciphertext-only attack



☐ Informal definition : "Irrespective of any *prior info.* the attacker has about m , the cipher-text c should leak *no additional information* about the plaintext"

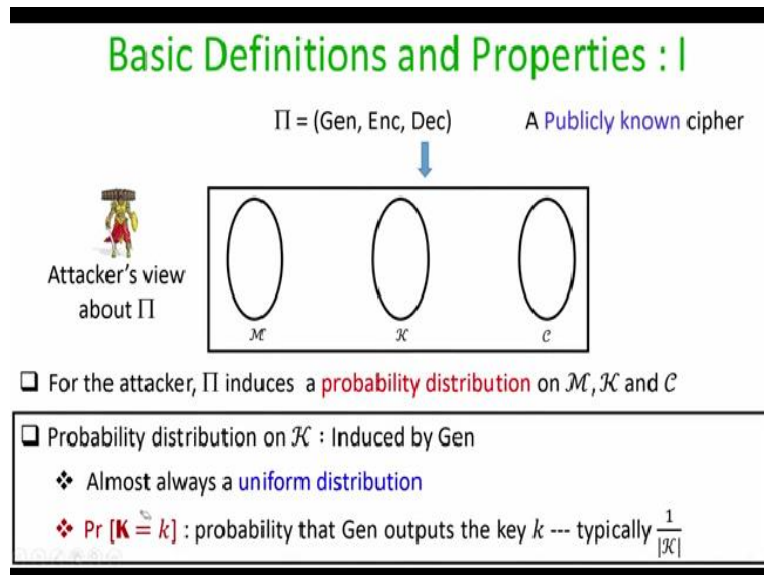
So, the definition of perfect security was given by Claude Shannon in his classical work in 1948 and Shannon is often considered as the father of information theory and this notion of security is also called as unconditional security and information theoretic security. So the attack model that is considered in the definition of perfect security is the ciphertext only attack model, where we assume that we have a sender and the receiver who have agreed upon a random key value k .

And we assume that sender has encrypted a single message m using a publicly known encryption process under that key k , and the ciphertext has been intercepted by the adversary. The interesting part of this attack model is that we assume here that the adversary is computationally unbounded. That means we make no assumption whatsoever about his computing power. We assume that he can do any kind of computation with brute force or any other computation.

So that is the interesting aspect of this security model. Informally, perfect security says that irrespective of any prior information that the attacker has about down the line plain text, seeing the cipher text provides no additional information to him about the underlying plain text. That means seeing the ciphertext is absolutely useless for the attacker, whatever it could infer about the message from the ciphertext, same it could have already inferred before any ciphertext would have been communicated.

So here we have 2 entities, we have the prior information and we have the term no additional information. So we have to now learn a little bit of maths to understand how to formalize these 2 notions namely that of prior information and that have no additional information right.

(Refer Slide Time: 02:36)



So, imagine we are given a publicly known cipher namely a triplet of algorithms key generation algorithm, encryption algorithm and a decryption algorithm. Then any attacker has the following information about the encryption process, right. Namely, he knows it knows 3 spaces, namely the message space, the key space and the ciphertext space, where the message space is the set of all legitimate messages which could be encrypted by the encryption process.

Keys space denotes all possible keys which could be output by the key generation algorithm. And the ciphertext denotes all possible ciphertext, which could be generated by the encryption algorithm right. So now from the viewpoint of the attacker, any public any encryption scheme induces 3 probability distribution, 1 probability distribution over the message space, 1 probability distribution over the key space.

And 1 probability distribution over the ciphertext space. So now let us go over this probability distributions one by one. The first probability distribution is over the key space and this is induced by the key generation algorithm. So remember, as per the Kerckhoffs' principle, we

assume that steps of the key generation algorithm encryption algorithm and decryption algorithm are publicly known to the attacker.

And we also discussed that key generation algorithm has to be a randomized algorithm. That means, every time we run the key generation algorithm it is going to output a possible candidate key from the key space with certain probability. So that is what we mean by the probability distribution over the key space induced by the key generation algorithm. Also, we discussed in one of our lectures that in most of the cases, the key generation algorithm is going to output a uniformly random key from the key space.

Namely, if the key space size is this cardinality of key space, then any candidate key could be generated with probability one over the key space. So that is the uniform distribution over the key space, which adversary will already have if it knows such steps, of key generation algorithm. But it is not necessary that your key generation algorithm always outputs uniformly random keys from the key space.

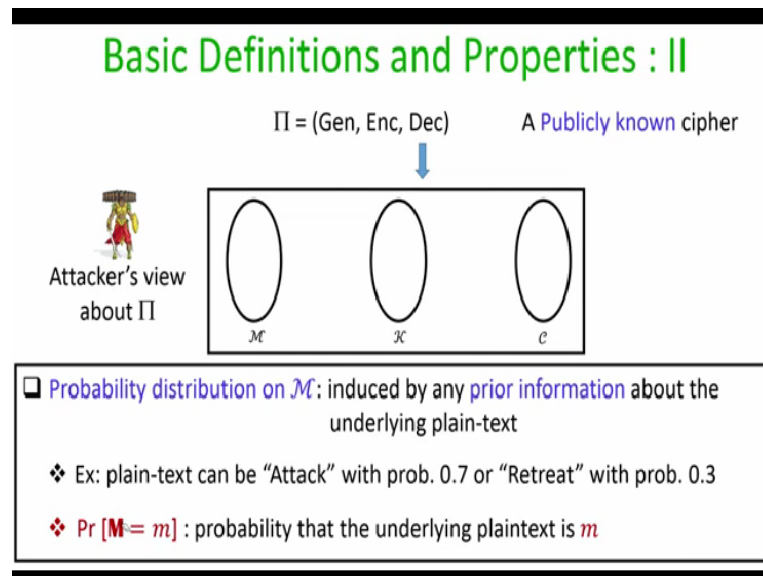
So that would be inducing another kind of probability distribution. Whatever is the case since the steps of the key generation algorithms are publicly known. We know that from the viewpoint of the attacker, there is a probability distribution over different values of keys, which could occur with different probabilities. And to capture that, we introduced this notation bold phase K , which is a random variable.

And which denotes the value of candidate key, which could be obtained by running the key generation algorithm. And since the key generation algorithm is going to be a randomized algorithm, it is not the case that every time we run the key generation algorithm, we obtain the same value of little k . And different values of little k could occur with different probabilities to denote what value of little k has been obtained by running the key generation algorithm.

We introduced this random variable bold phase K . So the notation probability, random variable k is equal to little k denotes the probability with which the key generation algorithm would output

the candidate value of key to be little k . That is the probability distribution that adversary has about the key space.

(Refer Slide Time: 05:58)



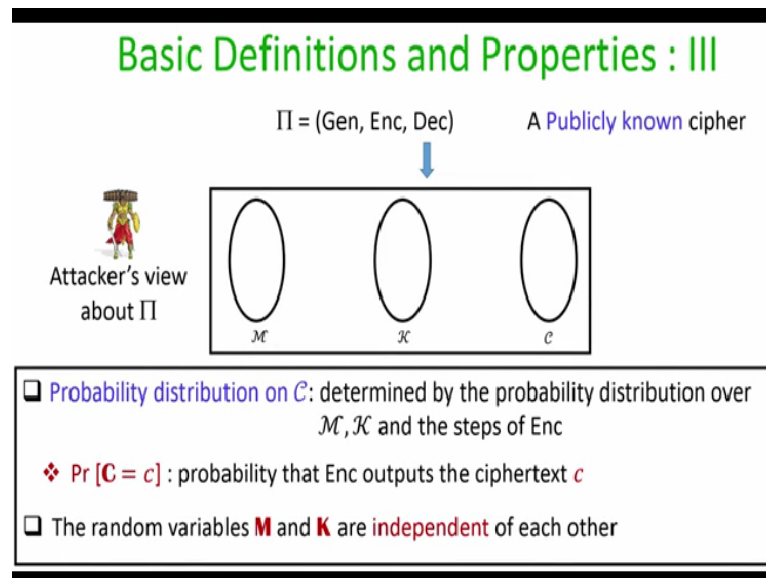
The next important probability distribution is that over the plane tech space, and this models any kind of prior information that the attacker might have about the underlying plaintext. For example, if the attacker already knows or feels that depending upon the underlying context sender would either encrypt a message attack 70% times or the message retreat 30% time. Then that is a probability distribution over the message space that is induced and known to the attacker.

But it might be the case that attacker is completely clueless what exactly is going to be the message it could be any uniform the random message from the plain text space in that case, the probability distribution is a uniform probability distribution. So we introduced this random variable bold phase m , which is a random variable to denote the candidate plain text, which sender might encrypt using the encryption algorithm.

And since attacker does not exactly know what exactly is the message going to be encrypted. We have problems for different message candidate messages to occur with different probabilities. So the notation probability random variable M equal to little m denotes a probability with which

attacker fields, that sender would encrypt a plain text m using the encryption algorithm. So that is a probability distribution over the plain text space.

(Refer Slide Time: 07:20)



And the third probability distribution is the probability distribution over the ciphertext space. And this is determined by the probability distribution over the message space and a key space and the steps of the encryption process. Because once the attacker knows, what is the probability that a certain plane text would be encrypted by the sender, and what is the probability that a candidate key would be obtained by the key generation algorithm.

Then by running the steps of the encryption algorithm over that candidate plain text and the candidate key adversary can find out what is the probability of occurrence of a certain ciphertext. So we introduced this third random variable bold phase $\mathbf{C}$ to denote value of the candidate ciphertext and the notation probability the random variable $\mathbf{C}$ equal to little c denotes a probability with which the encryption algorithm would output a cipher text c right.

So notice that there are random variables, m and k are independent of each other because the key is obtained by running the key generation algorithm, which is run independent of what message is going to be encrypted by the k algorithm. Whereas the random variable c , it depends upon the random value of the random variable m and a random variable k .

(Refer Slide Time: 08:37)

A Numerical Example : I

□ $\mathcal{M} = \{a, b, c, d\}$, $\mathcal{K} = \{k_1, k_2, k_3\}$

□ Attacker's view about the cipher

m	$\Pr[\mathbf{M} = m]$
a	$1/4$
b	$3/10$
c	$3/20$
d	$3/10$

Probability distribution
over $\mathcal{M}$

k	$\Pr[\mathbf{K} = k]$
k_1	$1/4$
k_2	$1/2$
k_3	$1/4$

Probability distribution
over $\mathcal{K}$

	a	b	c	d
k_1	3	4	2	1
k_2	3	1	4	2
k_3	4	3	1	2

Encryption matrix



So now let us see a numerical example to understand this concepts in a better fashion. So imagine I have a candidate encryption process where the candidate messages could be either the character a or b, or c or d. In the same way, imagine I have a key generation algorithm which could output 3 keys with different probabilities and say the attacker has the following probability distribution over the plain text space.

That means it knows depending upon the underlying context, it feels or it knows somehow that the message could be either a probability 1 by 4, or the message could b with probability 3 by 10 and so on. In the same way, assume that the key generation algorithm outputs the candidate key to be k 1 k 2 k 3 with probability 1 by 4, 1 by 2 and 1 by 4 respectively. So that is a probability distribution over the key space that attacker has.

And imagine the adversary knows the steps of the encryption algorithm. So it could compute an encryption metrics where along the columns you have the candidate plain text namely a b c d, right. So these are the candidate plain text. And here you have the candidate keys which could be obtained by running the key generation algorithm. So this value 3 denotes that the encryption algorithm is going to output side text 3 if the plain text would have been a and the key would have been k 1.

In the same way, for example, this last entry denotes that the ciphertext will be 2, if the plain text would have been d, and the key would have been a 3, and so on, and adversary could compute this matrix because it knows the steps of the algorithm and it knows the different candidate values of plain text and candidate values of key, right. So that is the information that is available with the attacker.

(Refer Slide Time: 10:31)

A Numerical Example : II

m	$\Pr[\mathbf{M} = m]$
a	1/4
b	3/10
c	3/20
d	3/10

k	$\Pr[\mathbf{K} = k]$
k_1	1/4
k_2	1/2
k_3	1/4

	a	b	c	d
k_1	3	4	2	1
k_2	3	1	4	2
k_3	4	3	1	2

□ What will be the probability distribution over $\mathcal{C}$?

$$\Pr[\mathbf{C} = 1] : \Pr[\mathbf{M} = b] \Pr[\mathbf{K} = k_2] + \Pr[\mathbf{M} = c] \Pr[\mathbf{K} = k_3] + \Pr[\mathbf{M} = d] \Pr[\mathbf{K} = k_1] = 0.2625$$

$$\Pr[\mathbf{C} = 2] : \Pr[\mathbf{M} = c] \Pr[\mathbf{K} = k_1] + \Pr[\mathbf{M} = d] \Pr[\mathbf{K} = k_2] + \Pr[\mathbf{M} = d] \Pr[\mathbf{K} = k_3] = 0.2625$$

$$\Pr[\mathbf{C} = 3] : \Pr[\mathbf{M} = a] \Pr[\mathbf{K} = k_1] + \Pr[\mathbf{M} = a] \Pr[\mathbf{K} = k_2] + \Pr[\mathbf{M} = b] \Pr[\mathbf{K} = k_3] = 0.2625$$

$$\Pr[\mathbf{C} = 4] : \Pr[\mathbf{M} = a] \Pr[\mathbf{K} = k_3] + \Pr[\mathbf{M} = b] \Pr[\mathbf{K} = k_1] + \Pr[\mathbf{M} = c] \Pr[\mathbf{K} = k_2] = 0.2125$$

So now let us try to compute the probability distribution over the ciphertext space right. So as you can see from the encryption metrics, the ciphertext could be either 1 or 2, or 3 or 4. So let us try to compute the probability with which the ciphertext could be 1 or 2, or 3 or 4. So let us first compute the probability that what is the probability that encryption algorithm produces ciphertext to be 1.

So now if you see the encryption matrix, ciphertext could be 1, if the plain text is b and the keys k_2 , that is one option or one condition under which the cipher text will be 1. The other condition in which the cipher text could be 1 is when the plain text is c, and the key would have been k_3 . And the third condition under which the ciphertext could be 1 is when the plain text is d, and the key would have been k_1 .

And all these 3 conditions or events are independent of each other. So we apply the same rule here. So the first term here denotes the probability that the key would have been k_2 . That means

by running the key generation algorithm sender would have obtained the key to be k_2 , and the probability that sender would have used the message b for encryption. Both these even suffer together, then the ciphertext would have been gone.

The second term denotes the other possible condition in which the ciphertext could have been 1 namely, plaintext at sender wants to encrypt the c . And the key that is obtained by running the key generation algorithm is k_3 . And the third term here denotes the probability of occurrence of the third condition, namely, the plain text is d and the key generation algorithm outputs the k to be k_1 .

And since we know the value of each of these individual probabilities, which is available to the attacker, attacker can compute this and hence it knows that the probability that encryption algorithm could produce ciphertext to be 1 is 0.2625. So that is the value of probability of occurrence of 1 as the ciphertext. In the same way, let us try to compute what is the probability that the ciphertext could be 2 right.

Again, if you see the encryption metrics, there are 3 possible conditions in which the ciphertext could be 2, the first condition is if sender would have encrypted the plain text c , and the key that is used by the sender namely the key generation algorithm outputs the key to be k_1 . So that is underlying them. The second condition in which the ciphertext character would have been 2 is if the plain text would have been d .

And the key generation algorithm for sender would have output the k to be k_2 . So that is the second term. And the third condition under which the ciphertext character could have been 2 is if the plaintext would have been d , and the key would have been k_3 . Again, we know all these individual probabilities and we can sum them together to obtain the probability of ciphertext being 2 being produced by the encryption algorithm.

In the same way you can compute the remaining probabilities, namely, the probability of ciphertext being 3 and the probability that ciphertext character is 4. And that gives you the probability distribution over the ciphertext space.

(Refer Slide Time: 13:50)

A Numerical Example : II

m	$\Pr[\mathbf{M} = m]$
a	$1/4$
b	$3/10$
c	$3/20$
d	$3/10$

k	$\Pr[\mathbf{K} = k]$
k_1	$1/4$
k_2	$1/2$
k_3	$1/4$

	a	b	c	d
k_1	3	4	2	1
k_2	3	1	4	2
k_3	4	3	1	2

□ What will be the conditional probability distribution $\Pr[\mathbf{C} = c \mid \mathbf{M} = m]$?

$$\Pr[\mathbf{C} = 1 \mid \mathbf{M} = a] = 0$$

$$\Pr[\mathbf{C} = 2 \mid \mathbf{M} = a] = 0$$

$$\Pr[\mathbf{C} = 3 \mid \mathbf{M} = a] = \Pr[\mathbf{K} = k_1] + \Pr[\mathbf{K} = k_2] = 0.75$$

$$\Pr[\mathbf{C} = 4 \mid \mathbf{M} = a] = \Pr[\mathbf{K} = k_3] = 0.25$$

Not only we can compute the probability distribution over the cipher text space, we can also compute some additional values, namely some kind of some conditional probabilities. For example, let us try to compute the conditional probability that given the plain text is m , what is the probability that ciphertext would have been c , right. So for instance, suppose let us try to compute the value of probability that ciphertext is 1 given that plaintext is equal to a .

So again, if we see the encryption metrics, if you see the column under the plaintext a , then you can see that if the plain text would have been a no way, it is possible that the ciphertext could be 1 because the encryption of a could give either the ciphertext being 3 or ciphertext being 3 or ciphertext being 4. For no value of k , no value of k the encryption of a would have given you the ciphertext to be 1.

That is why we can say that the conditional probability that ciphertext is 1 given the plain text is a is 0, right. In the same way, the condition probability that ciphertext is going to be 2 given the plaintext is 1 is again 0 because again, you can see the encryption metrics or the encryption algorithm, which states that for no value of k encryption of a under that key would give you the ciphertext value 2.

Now let us compute the conditional probability that given the message a , what is the probability ciphertext is equal to 3. Now, if you see the encryption metrics, under the plain text a , there are 2 values of k s namely k_1 and k_2 , both of which could produce an encryption of $a \oplus b = 3$. And since the key generation algorithm is a randomized algorithm, the probability that the key is equal to k_1 is given to you the probability that key is equal to k_2 is given to you.

And if any of these 2 events occur, then the encryption of a under the key k_1 or under the key k_2 would have given you the ciphertext equal to 3. So the conditional probability that given the plain text is a you obtain the cipher text 3 is 0.75. And in the same way, if you want to compute the conditional probability that given the plain text is a what is the probability ciphertext equal to 4. Then again, if you look into the encryption metrics, you can see that if the plain text is a then the only condition in which you can obtain the ciphertext to be 4 is if the key generation algorithm would have produced the key to be k_3 .

And the probability that the key generation algorithm would produce the key to be k_3 is 0.25 right. In the same way, you could compute other conditional probabilities as well.

(Refer Slide Time: 16:37)

Perfect Security : Original Definition

❑ Informal definition : "Irrespective of any *prior info*, the attacker has about m , the cipher-text c should leak *no additional information* about the plaintext"

❑ Formal definition : An encryption scheme (Gen, Enc, Dec) over a plain-text space $\mathcal{M}$ is perfectly-secure if for every probability distribution over $\mathcal{M}$ and $\mathcal{K}$, every plain-text $m \in \mathcal{M}$ and every cipher-text $c \in \mathcal{C}$, the following holds:

$$\Pr [M = m \mid C = c] = \Pr [M = m]$$

Posteriori probability that m is encrypted in c

a-priori probability that m might be communicated

≈

Observing the cipher-text c **does not change** the attacker's knowledge about the distribution of plaintext

So now let us see the original definition of perfect security as given by Shannon. So informal requirement, recall the informal requirement from the perfect secrecy is that we will say an encryption process to be perfectly secure. If irrespective of any prior information the attacker has

about the underlying plaintext. The cipher text that it intercepts leak, no additional information about the plaintext.

And now we know how to model this prior information. That is precisely the probability distribution over the plane text space that attacker has. And now we can also mathematically capture what exactly we mean by no additional information. So formerly an encryption process namely a collection of algorithms key generation in conduct over a plain text space m is called perfectly secure.

If for every probability distribution over the plain text space and key space, and for every plain text m belonging to the plain text space, according to that probability distribution, and for every ciphertext c belonging to the ciphertext space, the following equality holds, right. So let us try to understand what exactly the lhs and rhs of this equality stands for. So if you consider the rhs part of this equality.

Namely the probability that the plain text is equal to small m little m that is precisely the a priori probability that message m could be communicated by the sender. That is a prior information about the underlying plain text before any communication has happened from the sender to receiver, before any key generation algorithm has been used, and the message has been encrypted that is the a priori information about the underlying plaintext.

Whereas the lhs part of this equality namely the conditional probability that the plain text is equal to m given the ciphertext equal to c that supposed to you are the probability that the message m is encrypted in c . That means, given that adversary has seen or intercepted a cipher text c , what is the probability that the plain text m little m has **is** encrypted in this cipher text c . So, intuitively, when we say that these 2 probabilities are equal, it means that whatever the adversary knew about the underlying plain text before any cipher text was communicated with same probability adversary knows that a plain text could be m and in the given cipher text c .

That means observing the ciphertext c does not change attackers knowledge about the distribution of the plaintext. Whatever the attackers knowledge was before seeing the ciphertext,

the same knowledge it has, even after seeing the ciphertext. And moreover, it holds even if adversary is computationally unbounded. That is the importance of this definition. This definition does not put any kind of restriction on the computational power of the adversary.

Even if adversary knows the steps of the algorithm, even if it knows what could be the candidate keys even if it is allowed to do brute force, its view or its knowledge about the underlying plain text or the distribution of the plain text should not change before and after seeing the ciphertext.

If that holds, then we say that our underlying encryption process is perfectly secure. Notice that in this definition, I have highlighted few things namely, I have said that the condition should hold for every probability distribution over the plane text space.

That means it does not matter whether the distribution of a plain text space is a uniform distribution or a bias distribution, the condition should hold for any possible probability distribution of other message space. In the same way, the condition or the quality should hold for any kind of probability distribution over the key space whether it is a uniformly generated key, whether the key generation algorithm output uniformly generate random keys or bias keys still the condition should hold.

Moreover, once we fixed up probability distribution of the plain text space and the key space, the condition should hold for every plaintext belonging to the messy space and every candidate ciphertext belonging to the ciphertext space right.

(Refer Slide Time: 20:45)

Perfect Security : First Equivalent Definition

❑ **Original definition** : An encryption scheme (Gen, Enc, Dec) over a plain-text space $\mathcal{M}$ is perfectly-secure if for every probability distribution over $\mathcal{M}$ and $\mathcal{K}$, every plain-text $m \in \mathcal{M}$ and every cipher-text $c \in \mathcal{C}$, the following holds:

$$\Pr [\mathbf{M} = m \mid \mathbf{C} = c] = \Pr [\mathbf{M} = m]$$

❖ Interpretation : probability of knowing a plain-text remains the same before and after seeing the cipher-text

❑ **Alternate definition**: for every probability distribution over $\mathcal{M}$ and $\mathcal{K}$, every plain-text $m_0, m_1 \in \mathcal{M}$ and every cipher-text $c \in \mathcal{C}$, the following holds

$$\Pr [\mathbf{C} = c \mid \mathbf{M} = m_0] = \Pr [\mathbf{C} = c \mid \mathbf{M} = m_1]$$

❖ Interpretation : probability distribution of cipher-text is independent of plain-text

So now, what we will do is we will see some alternate equivalent definitions for perfect security. So this is the original definition of perfect security as given by Shannon and the interpretation of the quality of these 2 probabilities is that the probability of knowing a plaintext remains the same before and after seeing the cipher text. That is the interpretation of the original definition of perfect security as given by Shannon.

Now let us see the first alternate definition. The first alternative definition says that we will say an encryption process encryption scheme to be perfectly secure for every probability distribution over the plaintext space and key space. And for every pair of message m_0, m_1 which occurs with non-zero probability with respect to that probability distribution and every cipher text c the following equality should hold.

The equality says that the conditional probability that ciphertext c given the plaintext is m_0 is same as the conditional probability that ciphertext is c given the plaintext is m_1 . The interpretation of this equality is that the probability distribution of the ciphertext is independent of what exactly is your underlying plain text. That means if the adversary sees the ciphertext c over the channel, then it does not matter whether the plain text is m_0 or whether the plain text is m_1 with equal probability from the viewpoint of the attacker, the ciphertext c should be in candidate encryption of m_0 as well as a candidate encryption of m_1 .

There should not be any bias in the probability. But whether it is an encryption of m_0 or whether it is an encryption of m_1 , that means the ciphertext distribution should be independent of the underlying plaintext.

(Refer Slide Time: 22:32)

Perfect Security : First Equivalent Definition

□ If $\Pr[C = c | M = m_0] = \Pr[C = c | M = m_1] = \delta, \forall m_0, m_1 \in \mathcal{M}, c \in \mathcal{C}$

Then $\Pr[M = m | C = c] = \Pr[M = m], \forall m \in \mathcal{M}, c \in \mathcal{C}$

Proof: Let $m \in \mathcal{M}, c \in \mathcal{C}$ be arbitrary plain-text and cipher-text

$$\Pr[M = m | C = c] = \frac{\Pr[C = c | M = m] \cdot \Pr[M = m]}{\Pr[C = c]} \quad (\text{Bayes theorem})$$

$$\Pr[C = c] = \sum_{m' \in \mathcal{M}} \Pr[M = m'] \Pr[C = c | M = m'] = \sum_{m' \in \mathcal{M}} \Pr[M = m'] \cdot \delta = \delta$$

Similarly, $\Pr[C = c | M = m] = \delta$

$$\text{Hence, } \Pr[M = m | C = c] = \frac{\delta \cdot \Pr[M = m]}{\delta} = \Pr[M = m]$$

So now what we are going to do next is we are going to prove that both these 2 definitions are equivalent. Namely, we will show that if there is an encryption process which satisfies the original condition of Shannon's perfect security, then the same encryption process has to satisfy the alternate definition and vice versa. So let us first prove the equivalence of the definition assuming that we have an encryption process which satisfies the condition of alternate definition.

Namely, we assume that we have an encryption process where for every probability distribution, the conditional probability that m_0 is encrypted in c and m_1 is encrypted in c is same. And we call that probability to be say delta, for all pair of messages m_0, m_1 in the plain text space, and for all ciphertext c belonging to the cipher text space. Given this, we will prove that the original condition of the Shannon's perfect security is also true for the encryption process.

Namely, the conditional probability that the message is m in the ciphertext c is same as the probability that plain text is not right. So what we are going to do is assume we have an arbitrary plain text and arbitrary cipher text character. Now let us try to compute the conditional

probability that message is m given the cipher text c . So what I am going to do here is I am going to apply the Bayes theorem here.

And by applying the Bayes theorem contain the numerator and denominator of the conditional probabilities. So in the left hand side, the numerator is message being m and the denominator is ciphertext being c . So I am applying the Bayes theorem and rewriting the same expression in lhs, where I am changing the numerator and denominator. And as for the Bayes theorem, I will get the right hand side part right.

So, now let us try to compute the value of probability what is c equal to little c . The probability that your ciphertext is little c is going to be this summation, this summation state that you take all possible messages from the plain text space and find out what is the probability that the message is that candidate plain text namely m dash and given that the message is m dash, what is the probability that m dash is encrypted in c .

That will give you the probability distribution over the cipher text t little c right. Because you have all you what you have to basically do is imagine you have the plane text space you take each of the plaintext candidate plaintext that is precisely the first term in this summation. And once you have fixed that candidate plaintext you just compute what is the probability that that candidate plaintext will take you to the ciphertext little c , that is his second term.

And if you multiply all this, if these 2 probabilities and take the summation over all candidate plain text, that will give you the probability distribution of ciphertext being little c . Now, as part of our hypothesis of the condition, we know that the conditional probability that ciphertext is little c given the plain text is m dash is same for all m dash namely, it is δ , because that is what is the assumption that we are making.

We are making the assumption that our encryption process satisfies the alternate condition. So for every m dash, I can replace the second term in the summation by δ . And as a result, I get this simplified form of the summation and I cannot take out δ common and what I will be left

with summation of probabilities over the plain text space that plain text is little m dash. And if I as per the rules of probability, if I sum up all those probabilities I will get 1.

So that means I can say that the probability that ciphertext is little c is nothing but δ , okay, so we have computed the value of probability of c equal to little c , that is δ . Now, let us try to compute the conditional probability that ciphertext is little c given the plain text is m , that is the numerator part of this rhs expression. And again, as per our hypothesis of this, theorem, or this lemma, we already know that the given the plain text is little m , the probability that ciphertext will be c is nothing but δ .

And that is irrespective of what is my little m , it could be m_0 , it could be m_1 it could be m_2 for any candidate m from the plaintext base the probability that it could lead to the ciphertext when little c is δ . So, that means the numerator part of this rhs expression is also δ . Hence, I can say that my original lhs namely, the conditional probability that cipher text a plain text is m , little m given the ciphertext is little c is nothing but this equality.

And in the numerator as well as in denominator, I have the δ so I can cancel it out. And hence, I obtain that this conditional probability is nothing but the probability that plaintext is little m , which is precisely what exactly we wanted to prove. That means we have proved that if the encryption process satisfies the condition of the alternate definition, then it also has to satisfy the condition of original Shannon's definition right. So, now let us do the proof in the reverse direction.

(Refer Slide Time: 27:54)

Perfect Security : First Equivalent Definition

$\square$ If $\Pr[\mathbf{M} = m \mid \mathbf{C} = c] = \Pr[\mathbf{M} = m], \forall m \in \mathcal{M}, c \in \mathcal{C}$ Then $\Pr[\mathbf{C} = c \mid \mathbf{M} = m_0] = \Pr[\mathbf{C} = c \mid \mathbf{M} = m_1], \forall m_0, m_1 \in \mathcal{M}, c \in \mathcal{C}$
$\square$ Proof: Let $m_0, m_1 \in \mathcal{M}, c \in \mathcal{C}$ be arbitrary plain-texts and cipher-text $\square$ Given that $\Pr[\mathbf{M} = m_0 \mid \mathbf{C} = c] = \Pr[\mathbf{M} = m_0]$ $\Rightarrow \frac{\Pr[\mathbf{C} = c \mid \mathbf{M} = m_0] \cdot \Pr[\mathbf{M} = m_0]}{\Pr[\mathbf{C} = c]} = \Pr[\mathbf{M} = m_0]$ (Expanding LHS by Bayes theorem) $\therefore \Pr[\mathbf{C} = c \mid \mathbf{M} = m_0] = \Pr[\mathbf{C} = c]$ $\square$ Similarly, given that $\Pr[\mathbf{M} = m_1 \mid \mathbf{C} = c] = \Pr[\mathbf{M} = m_1]$ $\therefore \Pr[\mathbf{C} = c \mid \mathbf{M} = m_1] = \Pr[\mathbf{C} = c]$ (Expanding LHS by Bayes theorem and simplifying as above)

Namely, we assume that we have an encryption process which satisfies the condition of the original Shannon's definition. That is the conditional probability that message is little m given the cipher text this little c is same as the probability that messages little m , for all messages in the plain text space and all cipher text in the cipher text space. Given this will prove that the distribution of the cipher text is independent of the underlying plaintext.

Namely, it does not matter whether the plain text is m_0 or whether the plain text is m_1 with equal probability, it could lead to the cipher text little c , and this holds for every pair of messages m_0, m_1 in from the plain text space and every cipher text c from the cipher text space right. So let us first try to compute the probability that ciphertext is little c given the plain text is m_0 . For this we are going to use the fact that as per the given condition, since the encryption scheme satisfies the original channel and condition.

We know that the conditional probability that message is m_0 given ciphertext is c is same as the probability that messages m_0 , right. So now what I am going to do is I am going to expand my lhs part here using the Bayes theorem, where I am just changing the numerator and the denominator of the conditional probability. And by applying the Bayes theorem, I get this equality.

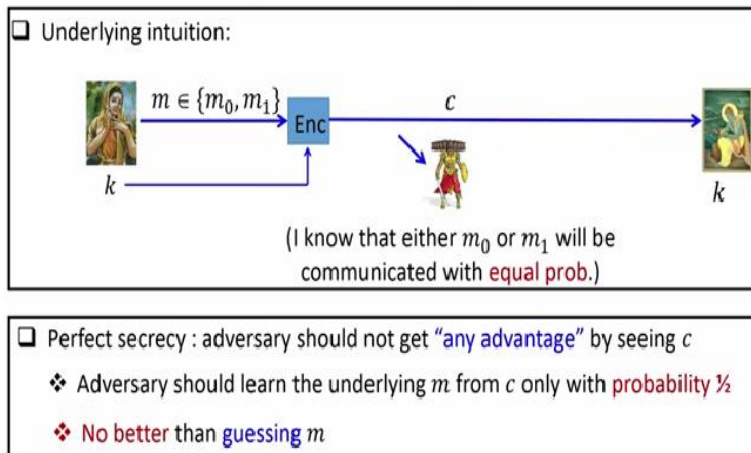
Now, what I can do is I can cancel out this common term from both the lhs and rhs. And hence I obtained that the conditional probability that m_0 will lead to an encryption of c is same as the probability that ciphertext is c , by applying the same logic, I can conclude that the conditional probability that ciphertext is c given the plain text is m_1 is same as the probability that ciphertext is c .

As a result, both this conditional probability that m_0 will lead to an encryption of c and the probability that m_1 will lead to an encryption of c are same namely, it is probability that ciphertext is equal to c . That means, we have proved that if the original Shannon's condition is satisfied, then the encryption process also has to satisfy the condition for the alternate definition. That means, both these definitions are equivalent to each other.

So, if you are given an encryption process and if you are asked to prove whether it is perfectly secure or not, then you can either prove it as per the original Shannon's condition or you can prove it as per the first alternate definition.

(Refer Slide Time: 30:30)

Perfect Security : Second Equivalent Definition



Now, let us see another equivalent definition of perfect security. And before going into this definition, let us first try to understand the underlying intuition that we want to capture through this definition. So, the goal of the perfect security is the following. Imagine a scenario where a key for the encryption process has been indeed upon between the sender and the receiver. And

suppose the adversary knows that the candidate plaintext for the sender is going to be either m_0 or m_1 and that to with equal probability.

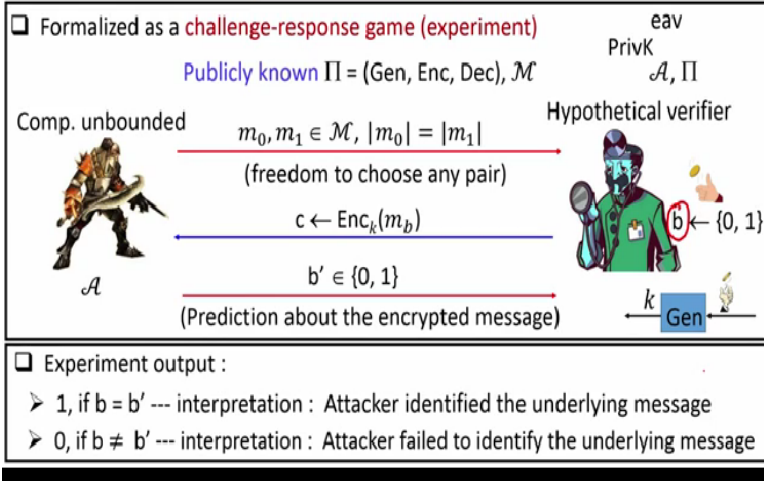
And suppose indeed sender is going to encrypt message m_0 or m_1 with equal probability. And it encrypts 1 of these messages randomly using the key k as for the encryption process, computes the cipher text c and sent it over the channel and say the attacker intercepts the cipher text c , and the attacker has unbounded computing power. Intuitively perfect secrecy demands that the adversaries knowledge about the underlying plain text should remain the same even after seeing the ciphertext c .

So what was this information about the plain text before any cipher text was communicated in this particular case, with probability half from the viewpoint of the attacker, the plain text could be m_0 , and with probability half the plain text would be m_1 . Now perfect secrecy demands that even after seeing the cipher text c and even after knowing the steps of the encryption process, and even if the adversary has unbounded computing power, the advantage that adversary gets after seeing the cipher text and learning the underlying plain text should be 0.

That means, even after seeing the cipher text c the probability that the underlying plain text would be m_0 or m_1 should be half. That means adversary can do nothing better than guessing the underlying predicts. That is the basic underlying intuition that we are now going to formalize.

(Refer Slide Time: 32:21)

Perfect Security : Second Equivalent Definition



And this intuition is formalized by a challenge response game, which we call this experiment. And we are going to see that in the rest of this course, every security definition is going to be formalized by this kind of challenge response experiment or a game which model something which we want which can happen in reality, right. So the experiment is as follows. So we assume that we are given a publicly known cipher namely a triplet of algorithms and the message space, and the game is played between 2 entities.

The first entity here is an adversary or an algorithm for the attacker which we denote by this algorithm $\mathcal{A}$ and the algorithm or the adversary is computationally unbounded. And this models the fact that we are trying to capture the notion of perfect secrecy where the adversary is computationally unbound. And the second entity here is the hypothetical verifier which is going to model the sender right.

Now, the nomenclature that we are going to follow while naming these kind of experiments is as follows. So, this particular experiment is denoted by this name. And now let us try to decode each of the individual parts of this complicated name. So, the name prefix PrivK denotes here that we are trying to model an experiment for a symmetric encryption process or a private key encryption process.

That is why the name privK , later on when we will try to model the security requirement for public key primitives. This privK will be replaced by pubK . The name of the string em denotes here we are considering an adversary where the adversary is only allowed to eavesdrop out just listen the cipher text because we are in the cipher text only attack model. The name a here denotes the name of the algorithm here and pi denotes the name of the encryption process with respect to which this game is going to be played.

So, that is the nomenclature we are going to follow to denote this particular experiment right. So now what are the rules of this game. This experiment is going to be a randomized experiment. So the first step here is that adversary selects a pair of messages from the plain text space, say m_0 and m_1 with the restriction that their size has to be same. An adversary can choose any pair of message there is absolutely no restriction which we put that on the pair of message that it can submit to the verifier.

The only restriction is that their length have to be same, so adversary can use adversary can deterministically pick up a pair of message, it could randomly pick a pair of message, it can use any strategy to pick the pair of message that it wants to send to the verifier. Now, once the pair of messages are communicated to the verifier, what the verifier is going to do is the following, it is going to run the key generation algorithm which will output a uniformly random key or a key as per the steps of the key generation algorithm for the verifier.

And the verifier is going to randomly pick 1 of these 2 messages for encryption. So the index of the message which is going to pick for encryption is denoted by b which is picked uniformly randomly. That means, with probability half, it might be picking the message m_0 for encryption, or with probability half it might be picking the message m_1 for encryption. Once it has decided to index of the message, it encrypts that message m_b using the key k which is known only to the verifier and the ciphertext is given to the adversary.

And now the goal of the adversary is to find out the index of the message which has been encrypted in c , that means the adversary has to find out whether the c that it is c is an encryption of m_0 or whether it is an encryption of m_1 . So after analyzing the cipher text c as per whatever

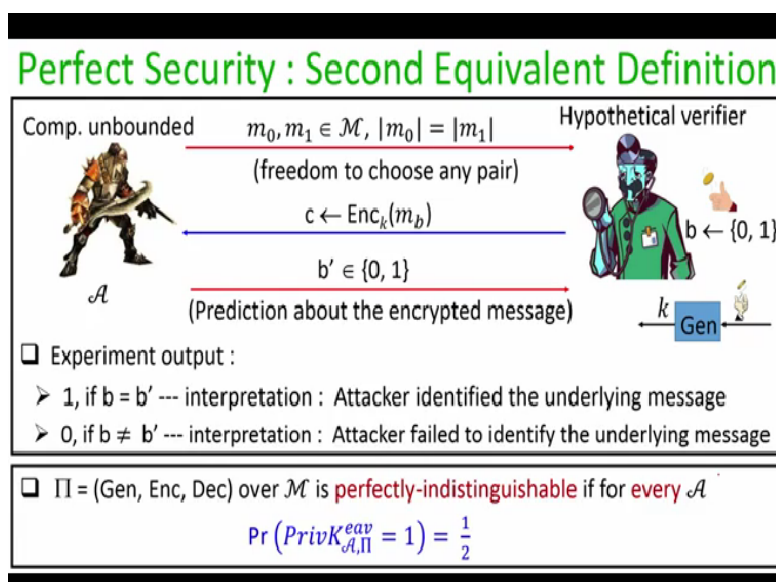
strategy adversary wants to follow, it gives them prediction. Namely, it outputs a bit, which is either 0 or 1 corresponding to the index, which it feels that that particular message has been encrypted in the cipher text c .

That means b dash denotes the index, which according to the adversary is the index of the message which is encrypted in the ciphertext c . That is the experiment and as you can see, this is a randomized experiment, because adversary could pick any pair of message as per its random s in the same way the verifier is going to pick any random message out of this pair for encryption, and the key could be any random key as per the key generation algorithm.

Now the output of the experiment is decided as follows. We say that output of the experiment is 1 or which is interpreted as adversary has won the game. If it has correctly predicted what exactly is the message which is encrypted in the ciphertext c . That means if it correctly output b dash equal to b then we say that the adversary has won this experiment or the adversaries experiments output is 1.

On the other hand, if the adversary incorrectly identifies what is encrypted in that cipher text c that means it outputs b dash which is different from b , then we say that output of the experiment is 0, which is interpreted as adversary has lost the game right.

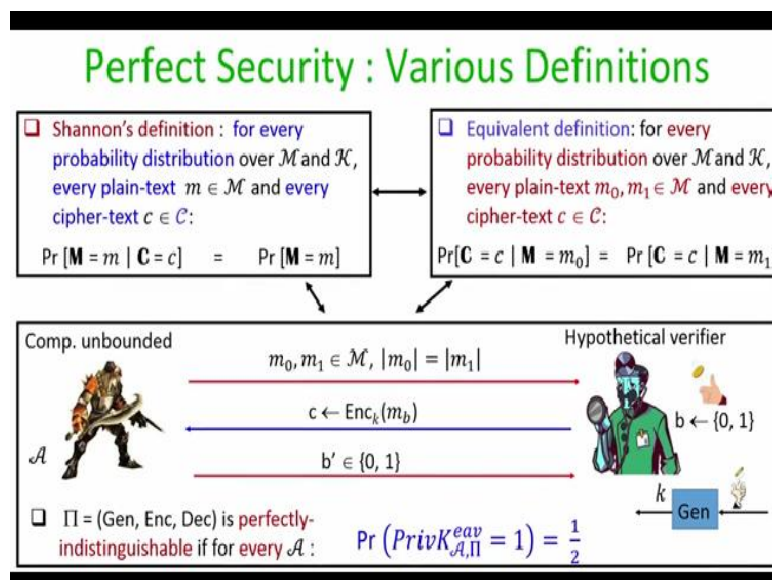
(Refer Slide Time: 37:41)



Now, this is the experiment and now we have a definition we say is that an encryption process π with respect to which this game has been played, is perfectly indistinguishable. If for every attacker who participates in this game, the probability that the output of the experiment is 1 that means probability that the adversary outputs b equal to b dash in the experiment is half. If it is more than half, then we say that the encryption process is not perfectly indistinguishable.

Also, the probability here is taken over the randomness of an adversary and what the randomness of the verifier namely the value of b and the value of k used output by the key generation right. So, that is our third equivalent definition of perfect secrecy.

(Refer Slide Time: 38:30)



So, the original definition of perfect secrecy as given by Shannon was this, we saw the first equivalent definition and now we have this game based definition right. So we have now 3 different definitions. And it can be proved that all these 3 definitions are equivalent to each other. We already proved that the first 2 definitions are equivalent to each other. We can also prove that this third definition is equivalent to the first definition.

And the third definition is also equivalent to the second definition. That means, if you are given an encryption process, we can prove it to be secure as per any of these 3 conditions, we can directly prove it using the Shannon's original definition, we can try to prove it using the

equivalent definition, or we can try to prove it using the game based definition right. So these are the 3 equivalent definitions of perfect security.

So now I would like to stress a little bit on this game based definition. So this game based definition describes the notion of what we call as perfect indistinguishability. That means the ciphertext is completely indistinguishable from the viewpoint of the attacker, and it leaks no information whatsoever, whether it is an encryption of m_0 or whether it is an encryption of m_1 and this notion of indistinguishability is very important.

Because for the rest of the course, we will see that all the definitions that we are going to formulate will be in terms of the notion of indistinguishability right.

(Refer Slide Time: 39:59)

Vignere Cipher is Not Perfectly-secure

☐ Consider an instance Π of Vignere cipher: $\mathcal{M} = \mathcal{C} = \{0, 1, \dots, 25\}^2$
☐ Gen outputs a uniformly random $t \in \{1, 2\}$ (period) and random key $k \in \{0, 1, \dots, 25\}^t$



$\mathcal{A}$

$m_0 = 00, m_1 = 01$

$c = (c_1, c_2) \leftarrow \text{Enc}_k(m_b)$

$b' = 0, \text{ if } c_1 = c_2$

$b' = 1, \text{ if } c_1 \neq c_2$



Gen

$b \leftarrow \{0, 1\}$

k

☐ Claim $\Pr(\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}} = 1) = 0.75$

$$\begin{aligned}
 \diamond \Pr(\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}} = 1) &= \frac{1}{2} \cdot \Pr[\mathcal{A} \text{ outputs } b' = 0 \mid b = 0] + \frac{1}{2} \cdot \Pr[\mathcal{A} \text{ outputs } b' = 1 \mid b = 1] \\
 &= \frac{1}{2} \cdot \left(\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{26} + 1 - \frac{1}{2} \cdot \frac{1}{26} \right) = \frac{3}{4}
 \end{aligned}$$

So now let us do an example here. And we will prove that the Vignere cipher which we have discussed in the last lecture is not perfectly secure. So, we are going to consider an instance of Vignere cipher, where the message space and the cipher text space are going to be of length 2, namely, the plaintext is going to be a 2 character string and the cipher text is going to be a 2 character string that is why the message space and the cipher text space are strings over the set 0 to 25 cross 0 to 25.

Now, the key generation algorithm for this instance of bigness cipher is going to do the following. Since the message string, plain text string is of length 2 then the key which is going to be used should be of length either 1 or it should be either length 2. So that means the period of the key which is the value t is a uniformly random value, which could be either 1 or 2. And once the period of the key has been selected, we select the key of that much size uniformly, randomly from the set 0 to 25 okay.

So that is a key generation algorithm. And once the key has been picked, we encrypt the plain text that is available with the sender as per the encryption process of the Vigenere cipher. Namely, we divide the plain text into blocks of size t . If t is equal to 1, that means the same t is used to encrypt both the characters of the plain text. Whereas if t is equal to 2, then we divide the plain text into 1 single block of size 2 and, encrypt the whole block in 1 row using the 2 different characters using the 2 characters of the key. That is encryption process.

And the decryption process is just the reverse operation of the corresponding encryption process. So now, we are going to prove that this instance of Vigenere cipher is not perfectly secure. That means we will give a concrete attack where the attacker can significantly find out what exactly is the underlying message which has been encrypted and we are going to use the game based definition here.

Namely, we will show an instance of the game here we will show that the distinguishing advantage of the attacker is significantly better than half. So the instance of the experiment here is as follows, so adversary submits the following pair of message, it submits a message m_0 , which consists of the same characters 00. And it submits another plain text m_1 which consists of 2 different plain text characters right.

Now, as for the rules of the game, the verifier is going to randomly encrypt 1 of these 2 messages as per the key generation algorithm. So it decides whether it is going to encrypt the message m_0 or whether it is going to encrypt the message m_1 with probably half and half. And once it decided what is going to encrypt, it runs a key generation algorithm which is going to output a period which could be either 1 or 2.

And once the period is generated a key of that much size is uniformly randomly picked and given to the verifier. And now using that key the verifier encrypts and sends the ciphertext to the attacker. So let the cipher text character be c_1 and c_2 . Since in this case the plain text which is encrypted consist of 2 characters, the cipher text is also going to contain 2 characters. Now the goal of the attacker is to find out after seeing this cipher text c .

Whether the cipher text c corresponds to an encryption of 00, or whether it corresponds to an encryption of 01. And the adversary does that by applying the following strategy. It compares the 2 cipher text characters in c , namely c_1 and c_2 . And if it finds that c_1 and c_2 are same, then it predicts that the message m_0 is encrypted in c , whereas if the cipher text characters c_1 and c_2 are different than it predicts that the message m_1 is encrypted in ciphertext.

That is analysis of the attack right. So now let us see what is the distinguishing advantage of the attacker, whether it can significantly distinguish apart, whether the c that it is seeing is an encryption of 00 or whether it is an encryption of 01. We claim that the probability that adversary can win this experiment, namely, it can correctly identify what has been encrypted in the cipher text c is 0.75, which is significantly better than half.

Then that means that this scheme, this instance of Vigenere cipher, is not perfectly secure. So now let us prove our claim. So what is the probability that adversary wins the experiment, the probability adversary wins to experiment is the following. We can view this instance of the experiment as 2 individual experiments, the experiment where the index b is equal to 0, namely the verifier has selected the message m_0 for encryption and c is the encryption of m_0 .

And the second experiment here is the version of the experiment where the value of b is equal to 1 that means the verifier has selected the message m_1 for encryption and c is the encryption of m_1 . So the probability that the adversary correctly wins the experiment is what is the probability that b is equal to 0 and probability of b equal to 0 is half. And given that b equal to 0 adversary, indeed outputs p dash equal to 0.

That is one case in which adversary can win the experiment. And the second case is when b is equal to 1, which can occur with probability half. And given that b is equal to 1, what is the probability that adversary correctly output the dash equal to 1. If we sum these 2 things, then that gives the winning probability of the adversary for this experiment. So now let us compute each of these 2 individual probabilities one by one right.

So let us first compute the probability that adversaries analysis outputs b dash equal to 0, given that indeed b equal to 0, that means we now want to analyze what is the probability that if m_0 is encrypted in c adversaries analysis indeed outputs b dash equal to 0, right. So if you see the adversaries analysis, adversary outputs b dash equal to 0 only if both the ciphertext characters are same. And if indeed 00 is encrypted, then there are 2 possibilities right.

The key could be either of period 1, or the key could be of period 2. So if the plain text to 00 is encrypted, right. If the plain text 00 is encrypted with a period of size 1, that means that the key size is 1 and the same character of the keys used to encrypt 0 as well as a second 0, then of course the both the cipher text characters are going to be same. Whereas if the key is of size 2 right, then the probability that both the cipher text characters are same as same as probability with which the key of size 2 satisfies the following condition.

Namely, the first character of the key is the same as the second character of the key. If that happens, then again, the encryption of the message m_0 will produce the ciphertext in both the characters being the same. So these are the 2 individual events under which an encryption of m_0 encryption of the message 00 will lead to a cipher text we are both the cipher text characters are same. So now what is the probability that the key generation algorithm of the verifier outputs a key of size 1.

Well, it is $1/2$. And what is the probability that the key generation algorithm for the verifier outputs a random key of size 2 where both the key characters are the same. The probability of that is $1/2$ into $1/26$ right. So if we sum these 2 probabilities, then we get the probability that adversaries analysis correctly output b dash equal to 0, given that verifier has indeed encrypted the message m_0 right.

So that is the first probability we have computed. Now let us try to compute the second probability here. Namely, what is the probability that adversary outputs $b' = 1$, given that indeed $b = 1$, that means we are now trying to analyze that assume the verifier has encrypted the message 0, 1, what is the probability that adversary's analysis also outputs $b' = 1$.

And for this we are actually going to subtract the complementary probability namely, we will compute the probability that adversary incorrectly outputs $b' = 0$ given that $b = 1$ and we are going to subtract this probability from 1 and that will give us the required probability. Now let us try to compute the probability that adversary's analysis or adversary's algorithm outputs $b' = 0$ given that $b = 1$, right.

So the probability of this event is same as a key of size period 2 is selected with the first character being 1 more than the second character. Because, for example, if the key is k_1, k_2 where k_1 and k_2 differs by 1 character, or whether the difference in the sense that say, the first character is the 1 and the second character is 0 of the key, or say the first character of the key is 2 and the second character of the keys 1 and so on.

If this kind of relationship holds between the 2 characters of the key, and if indeed the message 0, 1 is encrypted then after encryption both the cipher text characters will be same. Only if this event happens then adversary will incorrectly end up outputting $b' = 0$, even though b is equal to 1. Now what is the probability that a key period 2 is selected with the first character being 1 more than the second character.

And the probability of this is $\frac{1}{2}$ into $\frac{1}{26}$ right. Because with probability $\frac{1}{2}$, a key of period 2 could be selected, because the probability half the key could be a period 1 probability half the key could be period 2. So the $\frac{1}{2}$ here denotes the probability that key is of period 2. And given this, what is the probability that the first character of the key is 1 more than the second character of the key well, it is same as $\frac{1}{26}$ right.

So if we multiply these 2 probabilities, we get this complimentary probability. And if we now subtract this complimentary probability from 1, we get our required probability. So now we have both this individual probabilities computed, and if we sum them together, and we have outside 1 by 2 as the common thing, we get that the probability with which adversary analysis in this case could correctly identify what is encrypted is 3 by 4, which is significantly more than 1 by 2. And that means this instance of Vigenere cipher is not perfectly secure.

So that brings me to the end of this lecture. To summarize, in this lecture we discussed the notion of perfect secrecy, which is the first formal notion of secrecy, and which is also the most strongest form of secrecy because this notion of security is against an adversary which is computationally unbounded. So, if you achieve perfect secrecy, then that is the best possible notion of secrecy that you can think of.

We also discuss some alternate definitions of perfect secrecy, namely, the distribution of the ciphertext should be independent of the underlying plaintext. And we also saw the game based definition, which models perfect indistinguishability. I hope you enjoyed this lecture. Thank you.