CS 5602 Introduction to Cryptography
Lecture 20
Information Theoretic Security

George Markowsky
Computer Science Department
Missouri University of Science & Technology

1

Goals

- To introduce the concept of perfect secrecy.
- To discuss the security of the one-time pad.
- To introduce the concept of entropy.
- To explain the notion of key equivocation, spurious keys and unicity distance.
- To use these tools to understand why the prior historical encryption algorithms are weak.
- Slides are taken right from the book

2

Introduction

- We first need to overview the difference between information theoretic security and computational security.
- Informally, a cryptographic system is called computationally secure if the best possible algorithm for breaking it requires N operations, where N is such a large number that it is infeasible to carry out this many operations.
- With current computing power we assume that 2^{80} operations is an infeasible number of operations to carry out.
- Hence, a value of N larger than 2^{80} would imply that the system is computationally secure.
- No actual system can be proved secure under this definition, since we never know whether there is a better algorithm than the one known.
- In practice we say a system is computationally secure if the best known algorithm for breaking it requires an unreasonably large amount of computational resources.

3

Introduction

- Another practical approach, related to computational security, is to reduce breaking the system to solving some well-studied hard problem.
- For example, we can try to show that a given system is secure if a given integer N cannot be factored.
- Systems of this form are often called provably secure.
- However, we only have a proof relative to some hard problem, hence this does not provide an absolute proof.

4

Introduction

- Essentially, a computationally secure scheme, or one which is provably secure, is only secure when we consider an adversary whose computational resources are bounded.
- Even if the adversary has large, but limited, resources she still will not break the system.
- We need to be careful about the key sizes etc. If the key size is small then our adversary may have enough computational resources to break the system.
- We need to keep abreast of current algorithmic developments and developments in computer hardware.

5

Introduction

- At some point in the future we should expect our system to become broken, either through an improvement in computing power or an algorithmic breakthrough.
- Quantum computing?
- It turns out that most schemes in use today are computationally secure, and so every chapter in this book (except this one) will solely be interested in computationally secure systems.

6

Unconditional Security

- On the other hand, a system is said to be unconditionally secure when we place no limit on the computational power of the adversary.
- In other words a system is unconditionally secure if it cannot be broken even with infinite computing power.
- Hence, no matter what algorithmic improvements are made or what improvements in computing technology occur, an unconditionally secure scheme will never be broken.
- Other names for unconditional security you find in the literature are perfect security or information theoretic security.

7

Crytographic Hierarchy

- You have already seen that the following systems are not computationally secure, since we already know how to break them with very limited computing resources:
  - Shift cipher
  - Substitution cipher
  - Ballaso (Vigenere) cipher
- Of the systems we shall meet later, the following are computationally secure but are not unconditionally secure:
  - DES and Rijndael,
  - RSA,
  - ElGamal encryption
- However, the one-time pad which we shall meet in this chapter is unconditionally secure, but only if it is used correctly.

8

Probability and Ciphers

- Let P denote the set of possible plaintexts.
- Let K denote the set of possible keys.
- Let C denote the set of ciphertexts.
- Each of these can be thought of as a probability distribution, where we denote the probabilities by p(P = m), p(K = k), p(C = c).
- We will often write P, K, and C for random variables associated with P, K, and C
- So for example, if our message space is P= {α, β, γ } and the message α occurs with probability 1/4 then we write p(P = α) = 1/4.

9

Encryption & Decryption Functions

- We write e_k to represent encryption using the key k
- We write d_k to represent decryption using the key k
- We assume that P and K are independent variables
- The user does not select a key based on the content of a message (very bad)
- The set of ciphertexts under a specific key k is defined by

  C(k) = {e_k(x) | x ∈ P}
- We then have the relationship $$p(C = c) = \sum_{k : c \in \mathbb{C}(k)} p(K = k) \cdot p(P = d_k(c)).$$

10

Key Example

- P= {a, b, c, d}
- p(P = a) = 1/4
- p(P = b) = 3/10
- p(P = c) = 3/20
- p(P = d) = 3/10
- K= {k_1, k_2, k_3}
- p(K = k_1) = 1/4
- p(K = k_2) = 1/2
- p(K = k_3) = 1/4
- C= {1, 2, 3, 4}

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

- Time for some calculations

11

Some Calculations

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

$$p(C = 1) = p(K = k_1)p(P = d) + p(K = k_2)p(P = b) + p(K = k_3)p(P = c) = 0.2625,$$

$$p(C = 2) = p(K = k_1)p(P = c) + p(K = k_2)p(P = d) + p(K = k_3)p(P = d) = 0.2625,$$

$$p(C = 3) = p(K = k_1)p(P = a) + p(K = k_2)p(P = a) + p(K = k_3)p(P = b) = 0.2625,$$

$$p(C = 4) = p(K = k_1)p(P = b) + p(K = k_2)p(P = c) + p(K = k_3)p(P = a) = 0.2125.$$

		a	b	c	d	
		0.2500	0.3000	0.1500	0.3000	1.0000
k1	0.2500	0.0625	0.0750	0.0375	0.0750	0.2500
k2	0.5000	0.1250	0.1500	0.0750	0.1500	0.5000
k3	0.2500	0.0625	0.0750	0.0375	0.0750	0.2500
		0.2500	0.3000	0.1500	0.3000	

P(C = 1)	0.2625
P(C = 2)	0.2625
P(C = 3)	0.2625
P(C = 4)	0.2125
	1.0000

12

Some Calculations

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

$$p(C = c|P = m) = \sum_{k:m=d_k(c)} p(K = k)$$

$p(C = 1|P = a) = 0,\quad p(C = 2|P = a) = 0,$
$p(C = 3|P = a) = 0.75,\quad p(C = 4|P = a) = 0.25,$

$p(C = 1|P = b) = 0.5,\quad p(C = 2|P = b) = 0,$
$p(C = 3|P = b) = 0.25,\quad p(C = 4|P = b) = 0.25,$

$p(C = 1|P = c) = 0.25,\quad p(C = 2|P = c) = 0.25,$
$p(C = 3|P = c) = 0,\quad p(C = 4|P = c) = 0.5,$

$p(C = 1|P = d) = 0.25,\quad p(C = 2|P = d) = 0.75,$
$p(C = 3|P = d) = 0,\quad p(C = 4|P = d) = 0.$

P= {a, b, c, d}
p(P = a) = 1/4
p(P = b) = 3/10
p(P = c) = 3/20
p(P = d) = 3/10
K= $\{k_1, k_2, k_3\}$
p(K = k_1) = 1/4
p(K = k_2) = 1/2
p(K = k_3) = 1/4

13

Some Calculations

Anybody recognize this?

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

$$p(P = m|C = c) = \frac{p(P = m)p(C = c|P = m)}{p(C = c)}$$

$p(P = a|C = 1) = 0,\quad p(P = b|C = 1) = 0.571,$
$p(P = c|C = 1) = 0.143,\quad p(P = d|C = 1) = 0.286,$

$p(P = a|C = 2) = 0,\quad p(P = b|C = 2) = 0,$
$p(P = c|C = 2) = 0.143,\quad p(P = d|C = 2) = 0.857,$

$p(P = a|C = 3) = 0.714,\quad p(P = b|C = 3) = 0.286,$
$p(P = c|C = 3) = 0,\quad p(P = d|C = 3) = 0,$

$p(P = a|C = 4) = 0.294,\quad p(P = b|C = 4) = 0.352,$
$p(P = c|C = 4) = 0.352,\quad p(P = d|C = 4) = 0.$

P= {a, b, c, d}
p(P = a) = 1/4
p(P = b) = 3/10
p(P = c) = 3/20
p(P = d) = 3/10
K= $\{k_1, k_2, k_3\}$
p(K = k_1) = 1/4
p(K = k_2) = 1/2
p(K = k_3) = 1/4

14

Some Conclusions

- If we see the ciphertext 1 then we know the message is not equal to a. We also can guess that it is more likely to be b rather than c or d.
- If we see the ciphertext 2 then we know the message is not equal to a or b. We also can be pretty certain that the message is equal to d.
- If we see the ciphertext 3 then we know the message is not equal to c or d and have a good chance that it is equal to a.
- If we see the ciphertext 4 then we know the message is not equal to d, but cannot really guess with certainty as to whether the message is a, b or c.

$p(P = a|C = 1) = 0,\quad p(P = b|C = 1) = 0.571,$
$p(P = c|C = 1) = 0.143,\quad p(P = d|C = 1) = 0.286,$

$p(P = a|C = 2) = 0,\quad p(P = b|C = 2) = 0,$
$p(P = c|C = 2) = 0.143,\quad p(P = d|C = 2) = 0.857,$

$p(P = a|C = 3) = 0.714,\quad p(P = b|C = 3) = 0.286,$
$p(P = c|C = 3) = 0,\quad p(P = d|C = 3) = 0,$

$p(P = a|C = 4) = 0.294,\quad p(P = b|C = 4) = 0.352,$
$p(P = c|C = 4) = 0.352,\quad p(P = d|C = 4) = 0.$

15

Perfect Security (Secrecy)

- So in our previous example the ciphertext does reveal a lot of information about the plaintext. But this is exactly what we wish to avoid, we want the ciphertext to give no information about the plaintext.
- A system with this property, that the ciphertext reveals nothing about the plaintext, is said to be perfectly secure.
- Definition 5.1 (Perfect Secrecy). A cryptosystem has perfect secrecy if

  p(P = m|C = c) = p(P = m)
- for all plaintexts m and all ciphertexts c.

16

Lemma 5.2+

- Lemma 5.2+. A cryptosystem has perfect secrecy iff p(C = c|P = m) = p(C = c) for all m and c.
- Use Bayes's Theorem

PROOF. This trivially follows from the definition

$$p(P = m|C = c) = \frac{p(P = m)p(C = c|P = m)}{p(C = c)}$$

and the fact that perfect secrecy means $p(P = m|C = c) = p(P = m)$.

- Note that if p(P=m|C=c) = p(P=m) then p(C=c|P=m) = p(C=c)p(P=m|C=c)/p(P=m) = P(C|c)

17

Lemma 5.3

- Lemma 5.3. Assume the cryptosystem is perfectly secure, then |K| ≥ |C| ≥ |P| (Smart uses #K, etc., instead of |K|.)
- Proof: For any cryptosystem you must have |C| ≥ |P| since you must be able to recover the plaintext from the ciphertext
- We assume that every ciphertext can occur, i.e. p(C = c) > 0 for all c ∈ C, since if this does not hold then we can alter our definition of C.
- Then for any message m and any ciphertext c we have from Lemma 5.2+ that p(C = c|P = m) = p(C = c) > 0.
- This means for each m, that for all c there must be a key k such that $e_k(m)$ = c.
- Thus, |K| ≥ |C|

18

Shannon's Theorem

- Theorem 5.4 (Shannon). Let (P, C, K, $e_k(\cdot)$, $d_k(\cdot)$) denote a cryptosystem with |P| = |C| = |K|. Then the cryptosystem provides perfect secrecy iff
  - every key is used with equal probability 1/|K|,
  - for each m ∈ P and c ∈ C there is a unique key k such that $e_k(m) = c$.
- Proof: (perfect secrecy → two conditions)
- Suppose the system gives perfect secrecy. Then we have already seen for all m ∈ P and c ∈ C there is a key k such that $e_k(m) = c$. Now, since we have assumed |C| = |K| we have $|\{e_k(m) \mid k \in K\}| = |K|$

19

Shannon's Theorem

We need to show that every key is used with equal probability, i.e.

$$p(K = k) = 1/\#\mathbb{K} \text{ for all } k \in \mathbb{K}.$$

Let $n = \#\mathbb{K}$ and $\mathbb{P} = \{m_i : 1 \le i \le n\}$, fix $c \in \mathbb{C}$ and label the keys $k_1, \ldots, k_n$ such that

$$e_{k_i}(m_i) = c \text{ for } 1 \le i \le n.$$

We then have, noting that due to perfect secrecy $p(P = m_i | C = c) = p(P = m_i)$,

$$\begin{aligned} p(P = m_i) &= p(P = m_i | C = c) \\ &= \frac{p(C = c | P = m_i)p(P = m_i)}{p(C = c)} \\ &= \frac{p(K = k_i)p(P = m_i)}{p(C = c)}. \end{aligned}$$

Hence we obtain, for all $1 \le i \le n$,

$$p(C = c) = p(K = k_i).$$

This says that the keys are used with equal probability and hence

$$p(K = k) = 1/\#\mathbb{K} \text{ for all } k \in \mathbb{K}.$$

20

Shannon's Theorem (← Part)

Now we need to prove the result in the other direction. Namely, if

- $\#\mathbb{K} = \#\mathbb{C} = \#\mathbb{P}$,
- every key is used with equal probability $1/\#\mathbb{K}$,
- for each $m \in \mathbb{P}$ and $c \in \mathbb{C}$ there is a unique key k such that $e_k(m) = c$,

then we need to show the system is perfectly secure, i.e.

$$p(P = m | C = c) = p(P = m).$$

We have, since each key is used with equal probability,

$$\begin{aligned} p(C = c) &= \sum_k p(K = k)p(P = d_k(c)) \\ &= \frac{1}{\#\mathbb{K}} \sum_k p(P = d_k(c)). \end{aligned}$$

Also, since for each m and c there is a unique key k with $e_k(m) = c$, we must have

$$\sum_k p(P = d_k(c)) = \sum_m p(P = m) = 1.$$

Hence, $p(C = c) = 1/\#\mathbb{K}$. In addition, if $c = e_k(m)$ then $p(C = c | P = m) = p(K = k) = 1/\#\mathbb{K}$. So using Bayes' Theorem we have

$$\begin{aligned} p(P = m | C = c) &= \frac{p(P = m)p(C = c | P = m)}{p(C = c)} \\ &= \frac{p(P = m)\frac{1}{\#\mathbb{K}}}{\frac{1}{\#\mathbb{K}}} \\ &= p(P = m). \end{aligned}$$

21

Example 1

- Modified Shift Cipher. Recall the shift cipher is one in which we "add" a given letter (the key) onto each letter of the plaintext to obtain the ciphertext.
- We now modify this cipher by using a different key for each plaintext letter.
- For example, to encrypt the message HELLO, we choose five random keys, say FUIAT.
- We then add the key onto the plaintext, modulo 26, to obtain the ciphertext MYTLH.
- Notice, how the plaintext letter L encrypts to different letters in the ciphertext.
- When we use the shift cipher with a different random key for each letter, we obtain a perfectly secure system.
- To see why this is so, consider the situation of encrypting a message of length n.
- Then the total number of keys, ciphertexts and plaintexts are all equal, namely: #K= #C= #P= 26^n.
- In addition each key will occur with equal probability: p(K = k) = $1/26^n$, and for each m and c there is a unique k such that ek(m) = c. Hence, by Shannon's Theorem this modified shift cipher is perfectly secure.
- This is a lame example in my opinion, because this is the one-time pad in a convoluted way

22

Example 2 (one-Time Pad)

- The above modified shift cipher basically uses addition modulo 26.
- One problem with this is that in a computer, or any electrical device, mod 26 arithmetic is hard, ut binary arithmetic is easy.
- We are particularly interested in the addition operation, which is denoted by ⊕ and is equal to the logical exclusive-or, or XOR, operation:

⊕	0	1
0	0	1
1	1	0

- In 1917 Gilbert Vernam patented a cipher which used these principles, called the Vernam cipher or one-time pad.

23

Example 2 (one-Time Pad)

- To send a binary string you need a key, which is a binary string as long as the message.
- To encrypt a message we XOR each bit of the plaintext with each bit of the key to produce the ciphertext.
- Each key is only allowed to be used once, hence the term one-time pad.
- This means that key distribution is a pain, a problem which we shall come back to again and again.
- To see why we cannot get away with using a key twice, consider the following chosen plaintext attack.

24

Example 2 (one-Time Pad)

- We assume that Alice always uses the same key k to encrypt a message to Bob.
- Eve wishes to determine this key and so carries out the following attack:
  - Eve generates m and asks Alice to encrypt it.
  - Eve obtains $c = m \oplus k$.
  - Eve now computes $k = c \oplus m$.
- You may object to this attack since it requires Alice to be particularly stupid, in that she encrypts a message for Eve.
- *I disagree that Alice needs to be particularly stupid*
- But in designing our cryptosystems we should try and make systems which are secure even against stupid users.

25

Example 2 (one-Time Pad)

- Another problem with using the same key twice is the following.
- Suppose Eve can intercept two messages encrypted with the same key
- $c_1 = m_1 \oplus k$,
- $c_2 = m_2 \oplus k$.
- Eve can now determine some partial information about the pair of messages m_1 and m_2 since she can compute $c_1 \oplus c_2 = (m_1 \oplus k) \oplus (m_2 \oplus k) = m_1 \oplus m_2$.
- Despite the problems associated with key distribution, the one-time pad has been used in the past in military and diplomatic contexts.

26

Entropy (Introduction)

- If every message we send requires a key as long as the message, and we never encrypt two messages with the same key, then encryption will not be very useful in everyday applications such as Internet transactions.
- This is because getting the key from one person to another will be an impossible task.
- After all one cannot encrypt it since that would require another key.
- This problem is called the key distribution problem.

27

Entropy (Introduction)

- To simplify the key distribution problem we need to turn from perfectly secure encryption algorithms to ones which are, hopefully, computationally secure.
- This is the goal of modern cryptographers, where one aims to build systems such that
  - one key can be used many times,
  - one small key can encrypt a long message.
- Such systems will not be unconditionally secure, by Shannon's Theorem, and so must be at best only computationally secure.

28

Entropy (Introduction)

- The word entropy is another name for uncertainty, and the basic tenet of information theory is that uncertainty and information are essentially the same thing.
- This takes some getting used to, but consider that if you are uncertain what something means then revealing the meaning gives you information.
- As a cryptographic application suppose you want to determine the information in a ciphertext, in other words you want to know what its true meaning is,
  - you are uncertain what the ciphertext means,
  - you could guess the plaintext,
  - the level of uncertainty you have about the plaintext is the amount of information contained
- in the ciphertext.

29

Entropy (Introduction)

- If X is a random variable, the amount of entropy (in bits) associated with X is denoted by H(X), we shall define this quantity formally in a second. First let us look at a simple example to help clarify ideas.
- Suppose X is the answer to some question, i.e. Yes or No.
- If you know I will always say Yes then my answer gives you no information.
- So the information contained in X should be zero, i.e. $H(X) = 0$.
- There is no uncertainty about what I will say, hence no information is given by me saying it, hence there is zero entropy.
- If you have no idea what I will say and I reply Yes with equal probability to replying No then I am revealing one bit of information.
- Hence, we should have $H(X) = 1$.
- Note that entropy does not depend on the length of the actual message; in the above case we have a message of length at most three letters but the amount of information is at most one bit.

30

Entropy

Definition 5.5 (Entropy). *Let X be a random variable which takes on a finite set of values x_i, with $1 \le i \le n$, and has probability distribution $p_i = p(X = x_i)$. The entropy of X is defined to be*

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i.$$

Why - ?

We make the convention that if $p_i = 0$ then $p_i \log_2 p_i = 0$.

Let us return to our *Yes* or *No* question above and show that this definition of entropy coincides with our intuition. Recall, X is the answer to some question with responses *Yes* or *No*. If you know I will always say *Yes* then

$$p_1 = 1 \text{ and } p_2 = 0.$$

We compute

$$H(X) = -1 \cdot \log_2 1 - 0 \cdot \log_2 0 = 0.$$

Hence, my answer reveals no information to you.

If you have no idea what I will say and I reply *Yes* with equal probability to replying *No* then

$$p_1 = p_2 = 1/2.$$

We now compute

$$H(X) = -\frac{\log_2 \frac{1}{2}}{2} - \frac{\log_2 \frac{1}{2}}{2} = 1.$$

Hence, my answer reveals one bit of information to you.

31

Entropy

- We always have H(X) ≥ 0.
- The only way to obtain H(X) = 0 is if for some i we have $p_i = 1$ and $p_j = 0$ when i ≠ j.
- If $p_i = 1/n$ for all i then H(X) = $\log_2 n$.
- Another way of looking at entropy is that it measures by how much one can compress the information.
- If I send a single ASCII character to signal Yes or No, for example I could simply send Y or N, I am actually sending 8 bits of data, but I am only sending one bit of information.
- If I wanted to I could compress the data down to 1/8th of its original size.
- Hence, naively if a message of length n can be compressed to a proportion ϵ of its original size then it contains $\epsilon \cdot n$ bits of information in it.

32

Baby Cryptosystem

Let us return to our baby cryptosystem considered in the previous section. Recall we had the probability spaces

$$\mathbb{P} = \{a, b, c, d\},\ \mathbb{K} = \{k_1, k_2, k_3\} \text{ and } \mathbb{C} = \{1, 2, 3, 4\},$$

with the associated probabilities:

- $p(P = a) = 0.25$, $p(P = b) = p(P = d) = 0.3$ and $p(P = c) = 0.15$,
- $p(K = k_1) = p(K = k_3) = 0.25$ and $p(K = k_2) = 0.5$,
- $p(C = 1) = p(C = 2) = p(C = 3) = 0.2625$ and $p(C = 4) = 0.2125$.

We can then calculate the relevant entropies as:

$$H(P) \approx 1.9527,$$
$$H(K) \approx 1.5,$$
$$H(C) \approx 1.9944.$$

Hence the ciphertext 'leaks' about two bits of information about the key and plaintext, since that is how much information is contained in a single ciphertext. Later we will calculate how much of this information is about the key and how much about the plaintext.

33

Upper Bound for Entropy

Theorem 5.6 (Jensen's Inequality). *Suppose*

$$\sum_{i=1}^{n} a_i = 1$$

with $a_i > 0$ for $1 \le i \le n$. Then, for $x_i > 0$,

$$\sum_{i=1}^{n} a_i \log_2 x_i \le \log_2 \left(\sum_{i=1}^{n} a_i x_i \right).$$

With equality occurring if and only if $x_1 = x_2 = \ldots = x_n$.

Using this we can now prove the following theorem:

Theorem 5.7. *If X is a random variable which takes n possible values then*

$$0 \le H(X) \le \log_2 n.$$

The lower bound is obtained if one value occurs with probability one, the upper bound is obtained if all values are equally likely.

34

Proof of Upper Bound

$$\begin{aligned} H(X) &= -\sum_{i=1}^{n} p_i \log_2 p_i \\ &= \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} \\ &\le \log_2 \left(\sum_{i=1}^{n} \left(p_i \times \frac{1}{p_i} \right) \right) \quad \text{by Jensen's inequality} \\ &= \log_2 n. \end{aligned}$$

35