Zheng Ma

Mar 1, 2005

Solutions to Midterm Examination

Problem 1: Caeser cipher (20 points)

Grading Key Points: *4 points each.*

(a) Describe the Caeser Cipher.
Solution: Let the message space M and ciphertext space C consist of strings of uppercase letters from the Roman alphabet and spaces. Number the letters of the alphabet starting at 0, so $A = 0, B = 1, ..., Z = 25$. The key space $K = \{0, ..., 25\}$. To encrypt, E_k replaces each letter m by the letter $(m + k) \bmod 26$. To decrypt, D_k replaces each letter c by the letter $(c - k) \bmod 26$.

(b) What is the size of the key space?
Solution: 26.

(c) Describe what it means for a cipher to be information-theoretically secure.
Solution: *Information-theoretically secure* means that there is no statistical correlation between the plaintext and the ciphertext. The distribution of c does not depend on m, so knowing c gives no information about m. (**Note:** You need to mention the distribution of c and m or the statistical correlation between them to get full points of this problem.)

(d) Under what conditions is the Caeser Cipher information-theoretically secure?
Solution: When the Caeser cipher is used with a one-letter message.

(e) Describe at least two ways of breaking a Caeser cipher on an English-language message.
Solution:

i. Brute force attack: Try each of the 26 possible keys, checking each of the possible plaintexts with an English dictionary to see whether it is meaningful.
ii. Frequency analysis: As we did in the problem set, sort the letters by frequency and try to match them with the frequency table of English.
iii. Known plaintext attack: If we know any plaintext-ciphertext pair, then we can compute k and break the system.

Problem 2: Feistel network (20 points)

Grading Key Points: *Feistel network structure, computation of the function f, intermediate result, final result.*

Consider a block cipher using 8-bit blocks that is based on the basic DES architecture (Feistel network) with two rounds and no initial or final permutation. The scrambling function for round i is $f_i(x, K) = (2i \cdot K)^x \bmod 15$, for $i = 1, 2$, where the key K is a member of $\mathbf{Z}_{15}$.

If $K = 7$ and the ciphertext is 00111111, what is the plaintext? Draw the picture of the Feistel Cipher network to help you, and show your intermediate results.

Solution: The picture of the Feistel Cipher Network is showed in Fig.1. We know L_2 and R_2. The computation of $f_i(x)$ in the i^{th} round is $(2i \cdot 7)^x \bmod 15$. All the intermediate results are shown in Fig.1. So the plaintext is 00101000.

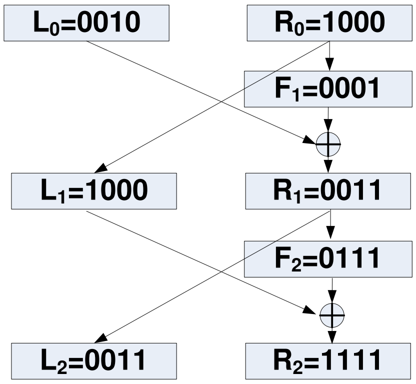

Figure 1: Feistel Network

Problem 3: Message authentication code (20 points)

Grading Key Points: *(a) With key, easy to compute and verify; without key, hard. (b) DES as MAC generation function; multiple message blocks; ciphertext chaining mode, initialization vector.*

(a) What is a Message Authentication Code (MAC)?
Solution: A MAC is generated by a function $C_k(m)$ that can be computed by anyone knowing a secret key k. However, it should be hard for an attacker to find any pair (m, ξ) such that $\xi = C_k(m)$ without knowing k.

(b) Describe how you would design a way to compute a MAC using the block cipher of problem 2 above.
Solution: One way to use a block cipher to compute a MAC is to use one of the ciphertext chaining modes, CBC or CFB. Here, we use the above DES function as the block cipher. Suppose we split the message into t blocks, namely $m_1, m_2, ..., m_t$. Then $c_i = \text{DES}_k(m_i \oplus c_{i-1})$, c_0 is a fixed *initialization vector* The last ciphertext block c_t depends on all t message blocks $m_1, m_2, ..., m_t$. Therefore, we define $C_k(m) = c_t$. The result of this process is reputed to be a good MAC generation function. Note that the MAC is only a single block long, which in general is much shorter than the message. A MAC acts like a checksum for preserving data integrity, but it has the advantage that an adversary cannot compute a valid MAC for an altered message.

Problem 4: Chinese remainder theorem (20 points)

Grading Key Points: *(a) Chinese remainder theorem, formula, inverse calculation, final result; (b) Show bijection, surjection,injection.*

Let $n = 13 \times 9 = 117$.

(a) Find a number $x \in \mathbf{Z}_n$ such that: $\begin{cases} x \equiv 6 \pmod{13} \\ x \equiv 3 \pmod{9}. \end{cases}$

Solution: Here $n_1 = 13$ and $n_2 = 9$:

$$\begin{aligned}
&n = n_1 \times n_2 = 117 \\
&N_1 = n/n_1 = 9, M_1 \equiv N_1^{-1} \equiv 3 \pmod{13} \\
&N_2 = n/n_2 = 13, M_2 \equiv N_2^{-1} \equiv 7 \pmod{9} \\
&x = a_1 M_1 N_1 + a_2 M_2 N_2 = 6 \cdot 9 \cdot 3 + 3 \cdot 13 \cdot 7 \equiv 84 \pmod{117}
\end{aligned}$$

(b) Explain why the number x you found in part (a) is unique in $\mathbf{Z}_n$.

Solution 1:
To see that the solution is unique in Z_{117}, let χ be the mapping $x \mapsto (x \bmod 13, x \bmod 9)$. χ is a surjection from Z_{117} to $Z_{13} \times Z_9$ since we have just shown for all $(a_1, a_2) \in Z_{13} \times Z_9$ that there exists $x \in Z_{117}$ such that $\chi(x) = (a_1, a_2)$. χ is also an injection since $|Z_{117}| = |Z_{13} \times Z_9|$. Hence, χ is a bijection, so the solution is unique in Z_{117}.

Solution 2 (Due to Melody Chan):
Suppose that we have y other than above x satisfy the equations: $\begin{cases} y \equiv x \equiv 6 \pmod{13} \\ y \equiv x \equiv 3 \pmod{9}. \end{cases}$
Then, we show that $y \equiv x \pmod{117}$. From above equations, we know that $13 \mid (y - x)$ and $9 \mid (y - x)$, but $\gcd(13, 9) = 1$, so $(13 \times 9) \mid (y - x)$. So $y \equiv x \pmod{117}$

Problem 5: Euler's theorem (20 points)

Grading Key Points: *(a) Euler function of the products of two primes, general Euler's function, final result; (b) Order of an elements, Euler's theorem, final result.*

(a) Calculate $\phi(77)$ and $\phi(\phi(77))$.
Solution:
$\phi(77) = \phi(7) \times \phi(11) = 6 \times 10 = 60$
$\phi(\phi(77)) = \phi(60) = \phi(2^2 \times 3 \times 5) = (2 - 1) \times 2^{2-1} \times (3 - 1) \times (5 - 1) = 16$

(b) Find a positive integer $x < 101$ such that $10^{5^{101}} \equiv 10^{5^x} \pmod{77}$.
Solution:
First, we note that $\gcd(10, 77) = 1$, so $10 \in \mathbf{Z}_{77}^*$. By Euler's Theorem, we know that $10^{\phi(77)} = 10^{60} \equiv 1 \pmod{77}$. So we only need to find x such that $5^{101} \equiv 5^x \pmod{60}$. Note that $\gcd(5, 60) \neq 1$ so $5 \notin \mathbf{Z}_{60}^*$, and you can't use Euler's Theorem again! Also, you can't use Chinese Remainder Theorem either because x is at the exponent! Luckily, we know another important concept, order of an element in a group. We can easily see that $5^1 \equiv 5 \pmod{60}$, $5^2 \equiv 25 \pmod{60}$, $5^3 \equiv 5 \equiv 5^1 \pmod{60}$. So $\mathrm{ord}(5) = 2$. Because $101 \equiv 1 \pmod{2}$, $5^{101} \equiv 5 \pmod{60}$. So $x \equiv 1 \pmod{2}$. Also $x < 101$, so any x in $\{1, 3, 5, 7, ..., 99\}$ solves the problem.

Problem 6: Discrete logarithm (20 points)

Grading Key Points: *(a) Any of the eight primitive roots of 17, Lucas test or explicit test; (b) use the primitive roots to generate* 5.

(a) Find a primitive root of 17. Justify your answer.
Solution: Let's try $g = 3$. Because $\phi(17) = 16$ we need to test g^2, g^4, g^8 to see whether they are $1 \in \mathbf{Z}_{17}^*$. We can easily verify that:
$$\begin{cases} g^2 \equiv 9 \neq 1 \pmod{17} \\ g^4 \equiv 81 \equiv -4 \neq 1 \pmod{17} \\ g^8 \equiv 16 \neq 1 \pmod{17}. \end{cases}$$
So $g = 3$ is a primitive root by the Lucas test.

(b) Find $\log_b 5 \pmod{17}$, where b is the primitive root you found in part (a).
Solution: We seek x such that $3^x \equiv 5 \pmod{17}$. We have to use 3 to generate the whole group until we get $5 \in \mathbf{Z}_{17}^*$, so $3^0 = 1, 3^1 = 3, 3^2 = 9, 3^3 = 10, 3^4 = 13, 3^5 = 5 \in \mathbf{Z}_{17}^*$. So $\log_3 5 \equiv 5 \pmod{17}$

Note: Other solutions like $g = 5, \log_5 5 \equiv 1 \pmod{17}$ are also graded in a similar way.

Problem 7: Simple block cipher (20 points)

Grading Key Points: *(a) Understanding the encryption and decryption, decryption algorithm, calculation and the final result. (b) Known plaintext attack, Brute force attack, information leakage.*

Alice and Bob have designed a very simple block cipher with the following encryption protocol:

1. The message is a binary string. It is padded at the right-most end with 0's so that it's length is divisible by 8.

2. The padded message is split into blocks of 8 bits each.

3. A further block of 8 bits is appended at the right-most end which contains the length of the original message, in bits. (It is assumed that messages are of length less than 2^8 so that the length will fit into an 8-bit block)

4. Alice and Bob agree on a symmetric key of 8 bits.

5. Starting with the left-most message block:

   i. The key is XORed with the message block to obtain the ciphertext block.

   ii. The message block and the key are then added together (using ordinary binary integer addition) and the 8 most significant bits form the key for the next block.

Step 5 is repeated until all of the message blocks have been encrypted. For example, the message

$$11010011 \;\; 11001101 \;\; 10001010 \;\; 1101$$

is padded with 4 zeros and length byte 00011100 to become

$$11010011 \;\; 11001101 \;\; 10001010 \;\; 11010000 \;\; 00011100$$

Using the key 01100110, the message is encrypted to produce the following ciphertext:

$$10110101 \;\; 01010001 \;\; 00111110 \;\; 01001111 \;\; 10101011.$$

Questions:

(a) Write the corresponding decryption protocol and demonstrate it by decrypting the ciphertext

$$11001100\ \ 01111000\ \ 11101100\ \ 10010010$$

using the same key 01100110.
Solution:
Let σ denote the operation of getting the highest 8 bits of a binary number. The above encryption algorithm can be described as:
$k_0 = k$
$c_i = k_{i-1} \oplus m_i$
$k_i = \sigma(k_{i-1} + m_i)$
So, the corresponding decryption algorithm is:
$k_0 = k$
$m_i = k_{i-1} \oplus c_i$
$k_i = \sigma(k_{i-1} + m_i)$
Thus we can find m by applying the algorithm to all of the ciphertext, then use the last block to determine the length of the plaintext. Then we can eliminate the padding zeros during the encryption and get the plaintext. Having said that, we have the following intermediate results:
$m_1 = c_1 \oplus k_0 = 11001100 \oplus 01100110 = 10101010$
$k_1 = \sigma(m_1 + k_0) = \sigma(10101010 + 01100110) = \sigma(100010000) = 10001000$
$m_2 = c_2 \oplus k_1 = 01111000 \oplus 10001000 = 11110000$
$k_2 = \sigma(m_2 + k_1) = \sigma(10111100 + 10001000) = \sigma(101111000) = 10111100$
$m_3 = c_3 \oplus k_2 = 11101100 \oplus 10111100 = 01010000$
$k_3 = \sigma(m_3 + k_2) = \sigma(01010000 + 10111100) = \sigma(100001100) = 10000110$
$m_4 = c_4 \oplus k_3 = 10010010 \oplus 10000110 = 00010100$
From $m_4 = 20$, we know that the plaintext has length 20, so the plaintext is $m =$ 10101010 11110000 0101.

(b) Discuss the security of Alice and Bob's symmetric key cryptosystem.
Solution:
At first sight, the system may seem secure to you: different blocks are encrypted under different keys, different combinations of blocks using the same key will generate different ciphertext. All of those are good properties. However, the system may not be that secure as you imagined. We actually want to show you how easy it is to build an insecure system which looks secure. Here are some security holes of the system (you'll get all the points if you get the first two):

- We know that the key space is only $2^8 = 256$, which is relatively small. So we can use brute-force attack to it.
- The system suffers from known-plaintext attack. We can apply XOR to the first block to get the initial key and break the system.
- Because there is no MAC, an attacker still has a chance to change the last eight bits to give Bob wrong information to determine the length of string and let him treat the padding zero as a message bit.
- We know that the last eight bits are the ciphertext of the length of the plaintext. We already know the range of the length by counting the length of the cipher. This can

reveal several bits of the key. For instance, in (a), we know that the length of the plaintext is in the range $[16 \dots 24]$. So the highest 3 bits of m_4 must be 000, so we know that the highest 3 bits of k_3 are 100.

- Once a key has a 1 in the leftmost bit than every key thereafter also will, since we are taking the 8 most significant bits of the sum. This of course exposes every 8th bit of most of the message, even if the key is totally unknown.