

## CS 134: MIDTERM SOLUTION

### 10. Problem 1:

What are the four main **types of attacks** on encryption algorithms? Describe each.

SOLUTION:

- (a) Ciphertext only: The attacker has only access to a set of ciphertexts and the attack is successful if the corresponding plaintexts or the encryption key is recovered.
- (b) Known plaintext: The attacker has access to a set of plaintexts and their corresponding ciphertexts.
- (c) Chosen plaintext: The attacker can choose arbitrary plaintexts to be encrypted and obtain the corresponding ciphertexts. After that, the attacker tries to decrypt a ciphertext that differs from the ones he got.
- (d) Chosen ciphertext: The attacker chooses a set of ciphertexts and causes it to be decrypted with an unknown key. This is called "lunchtime" or "midnight" attack, referring to a scenario in which an attacker gains access to an unattended decryption machine, while the owner is at lunch or sleeping. After that, the attacker tries to decrypt a ciphertext that differs from the ones he chose.

### 10. Problem 2:

What are the common **modes of operation** for a conventional cipher, such as DES or AES? Explain how each one works.

SOLUTION:

- ECB: Each block of plaintext is encrypted with the key.
  - o Pros: local error; parallel encryption
  - o Cons: permutation attack; identical plaintext blocks are encrypted into identical ciphertext blocks
- CBC:  $C_i = E(K, C_{i-1} \text{ XOR } P_i)$

- Pros: a change in the message affects all ciphertext blocks after the change as well as the original block
- Cons: IV required; a single error affects two consecutive blocks of ciphertext; no parallel encryption
- OFB:  $V_i = E(K, V_{i-1})$        $C_i = P_i \text{ XOR } V_i$ 
  - Pros: stream cipher; local error; parallel encryption; offline computation of  $V_i$
  - Cons: IV required; easy to modify (known plaintext attack)
- CFB:  $C_i = P_i \text{ XOR } E(K, C_{i-1})$ 
  - Pros: parallel decryption
  - Cons: sequential encryption
- MAC:  $P_i, P_{i+1}, \dots, C_n$  in CBC mode

### 10. Problem 3:

You're at a geek party where the host plays the following game:

- He asks each guest to dip a hand in a barrel which has 400 tickets and pick one
- Each ticket has a number between 1 and 400 (inclusive)
- Each guest remembers the number on the ticket and puts the ticket back into the barrel
- Each guest is then supposed to find a partner – a person whose ticket number PLUS the guest's own ticket number EQUAL 401.

What does the number of guests at the party need to be for you to have at least 50% chance of not finding a partner?

SOLUTION:

Suppose  $K$  is the number of guests to have 50% chance of finding a partner: solution to our problem is  $K-1$ .

For each ticket  $X$  there's only one ticket  $Y = 401 - X$  that would make a match.

Prob. of having no matchers  $P_0$  is

$$P_0 = 1 * (1 - (1/400)) * (1 - (2/400)) * (1 - (3/400)) \dots$$

Prob. of having no matchers  $P_1$  is

$$P_1 = 1 - P_0$$

Set  $P_1 = 0.5$  and solving for  $K$

$$K = 1.17 * \text{SQRT}(400) = 23.4$$

With 23 guests there's 50% chance of not finding a partner

10 (3+3+4). Problem 4:

1. Calculate  $\phi(77)$  and  $\phi(\phi(77))$
2. What is the order of the group  $Z_{35}^*$ ?
3. What is the order of each element in  $Z_7^*$ ?

SOLUTION:

- $77 = 11 * 7$ , so  $\phi(77) = (11-1) * (7-1) = 10 * 6 = 60$
- $\phi(\phi(77)) = \phi(60) = 60 * (1 - (1/2)) * (1 - (1/3)) * (1 - (1/5)) = 16$
- the order of  $Z_{35}^*$  is 24 because  $35 = 5 * 7$ , and  $(5-1) * (7-1) = 4 * 6 = 24$
- in  $Z_7^*$ 
  - $\text{ord}(1) = 1$
  - $\text{ord}(2) = 3$
  - $\text{ord}(3) = 6$
  - $\text{ord}(4) = 3$
  - $\text{ord}(5) = 6$
  - $\text{ord}(6) = 2$

10. Problem 5:

Alice creates a new hash function from DES. She takes message  $M$  and splits it into 64-bit blocks  $M_1 \dots M_t$ . She then defines a hash of  $M$  as the XOR of all blocks encrypted with DES with some known (not secret) key  $K$ . In other words  $H(M) = \text{DES}(K, M_1) \text{ XOR } \text{DES}(K, M_2) \text{ XOR } \dots \text{ XOR } \text{DES}(K, M_t)$ . You can assume that Alice then sends  $M, H(M)$  to Bob.

Is this a good hash function? In either case, explain/justify your answer.

SOLUTION:

- As the key is known, an attacker can forge her own message/hash pairs. Anyway, this is not a crucial point, as we are only interested in using DES as a hash function and not as a cipher.
- Message XYZ and its permutation will collide. Thus the attacker can perform a permutation attack
- Any block that appear an even number of time in the message can be removed without detection by the recipient. That is XYXZ will have the same hash of YZ

10 (5+5). Problem 6:

1. What is a better cipher: Affine or Caesar? Why?
2. What (if any) are the problems with the Permutation cipher? (Assume we're encrypting one alphabet letter at a time and the ciphertext is also alphabetic).

SOLUTION

- Both the affine and the Caesar ciphers are monoalphabetic ciphers. Still an affine cipher is (a little) stronger against bruteforce attacks ( $12 \cdot 26$  keys compared to 26 keys).
- Frequency analysis.

10. (4+3+3). Problem 7:

Recall El Gamal encryption (where  $y = b^x \pmod p$  is the public key of the decryptor – Bob)

1. generate random  $r$
2. ciphertext is:  $c = my^r \pmod p$ ,  $k = b^r \pmod p$

What happens if:

- a) Alice re-uses the same  $r$  in encrypting 2 different messages  $m_1$  and  $m_2$ ?
- b) Alice accidentally reveals  $r$  used in encrypting a message  $m$ ?
- c) Alice encrypts the same message twice (but with different  $r$ -s)

SOLUTION:

- a)  $c_1/c_2$  reveals  $m_1/m_2$
- b) the message can be recovered
- c) nothing happens

10 (5+5). Problem 8:

- (a) Bob chooses an RSA modulus  $n = 13 \times 7 = 91$ . He wants an easy-to-remember encryption exponent, so he wants to use either  $e = 25$  (his age) or  $e = 12$  (the size of his shoes). Which one(s) won't work and why?
- (b) Alice picks another RSA modulus  $n = 5 \times 11 = 55$ . She picks  $e=3$ . What is Alice's decryption exponent  $d$ ?

SOLUTION:

- For RSA to work we must have  $e*d = 1 \pmod{(p-1)(q-1)}$ 
  - 25 has inverse 49 mod 72
  - 12 has no inverse mod 72
- 27 because 27 is the invers of 3 mod 40