جامعة بيرزيت
BIRZEIT UNIVERSITY

Faculty of Engineering and Technology
Electrical and Computer Engineering Department

ENCS4320, Applied Cryptography

Midterm Exam (V3)

Wednesday, 4/01/2023

Name:______________________________ **ID**:______________________

1. (**2 pts**) Let $M = C = K = \{0, 1, 2, \ldots, 255\}$ and consider the following cipher defined over (K, M, C): $E(k, m) = m + k \pmod{256}$; $D(k, c) = c - k \pmod{256}$. Does this cipher have perfect secrecy? Explain your answer.

   Yes, it does have perfect secrecy.

   Perfect Secrecy essentially means these notions:

   1. P(M=m|C=c)=P(M=m) i.e. seeing a ciphertext doesn't give you any extra information about the plaintext. The probability of seeing a message m after the ciphertext has been observed is the same as the probability of the message without the ciphertext.
   2. P(C=c|M=m$_0$)=P(C=c|M=m$_1$) i.e. the probability of ciphertext c is equally likely for 2 different messages.
   3. The key is as long as the message and a key should be used uniquely with a probability 1/|K| where |K| is the key space.

2. (**2 pts**) In a Feistel cipher, how does the encryption in one rounds look like? How does decryption work?

   Let us describe one round of a Feistel cipher which takes m and produces $R_k(m)$. Here, k is the round key.

   - Split the plaintext m into two halves (L_0, R_0)
   - $L_1 = R_0$
   - $R_1 = L_0 \oplus f_k(R_0)$

   - Then, $R_k(m)$ is (L_1, R_1).

   To obtain $m = (L_0, R_0)$ from $R_k(m) = (L_1, R_1)$, we set
   $R_0 = L_1$ and then compute
   $L_0 = R_1 \oplus f_k(R_0)$

3. (**4pts**) Consider a block cipher with 5-bit block size and 5-bit key size such that

$$E_k(b_1b_2b_3b_4b_5) = (b_1b_2b_3b_4b_5)\oplus k$$

Encrypt $m = (0101001010)_2$ using k=$(10001)_2$ and **CTR**-mode (IV=(10011)$_2$).

$m = m_1m_2$ with $m_1 = 01010,\ m_2 = 01010.$

$c_0 = ctr = \mathbf{10011}$
$ctr + 1 = 10100$
$ctr + 2 = 10101$

$c_1 = (ctr+1)\oplus k \oplus m_1 = 10100 \oplus 10001 \oplus\ 01010 = \mathbf{01111}$

$c_2 = (ctr+2)\oplus k \oplus m_2 = 10101 \oplus 10001 \oplus\ 01010 = \mathbf{01110}$

$c = c_0c_1c_2 = \mathbf{10101\ \ 01111\ \ 01110}$

4. (**3 pts**) For AES, what are the four layers that each round consists of? Which layer makes AES highly nonlinear?

The 4 layers are:

1. **ByteSub** (each byte gets substituted with another byte (like a single S-box in DES); provides confusion)
2. **ShiftRow** (the 16 bytes are permuted (like a P-box in DES but on bytes, not bits); provides diffusion)
3. **MixCol** (each column in the 4x4 matrix is linearly transformed; provides diffusion)
4. **AddRoundKey** (the state is xored with a 128 bit round key)

The ByteSub layer is highly nonlinear

5. (**5 pts**) In DES, What is the key size? How many rounds? What does each S-box do? Why is there no 2DES? To (naively) brute-force DES, how much data must we encrypt?

DES uses 56-bit keys and has a 64 bit block size. It consists of 16 rounds.

Each S-box has a 6-bit input and a 4-bit output. (nonlinear)

The meet-in-the-middle attack is also the reason why 2DES does not provide significantly increased security over DES.

Hence, given m and c, to make a list of all possible $E_k(m)$ (to check for which k we have $E_k(m) = c$, we need to encrypt 2^{56} times 64 bits.

This is $2^{56}.8 = 2^{59}$ bytes, or 512 pebibyte (binary analog of petabyte) or 576 petabyte (since$2^{59}\ \approx 5.76.10^{7}$).

6. (**4 pts**) Assume we are using AES, consider the following encryption scheme for 256-bit messages: to encrypt message $M = m_1 \parallel m_2$ using key k (where $|m_1| = |m_2|$, choose random 128-bit r and compute the ciphertext $C = r \parallel \ (F_k(r) \oplus m_2) \parallel m_1$. Sate wither this encryption scheme is chosen-plaintext attack (CPA) or not?

Let m_1 and m_2 be arbitrary but distinct.

Using the encryption oracle, obtain an encryption $r \| c_1 \| c_2$ of $m_1 \| m_2$.

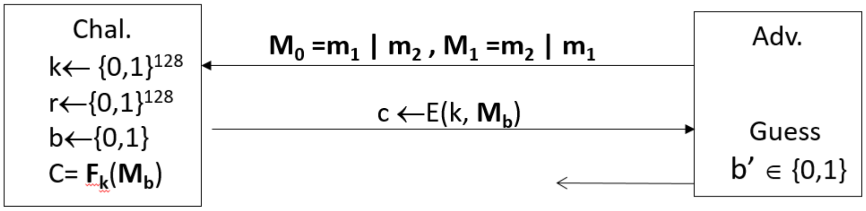
Output messages $M_0=m_1\|m_2$ and $M_1=m_2\|m_1$.

Not that the last block is not encrypted. Therefore,

if $c_2 = m_2$ → the challenge cipher for M_0

if $c_2 = m_1$ → the challenge cipher for M_1

The attacker win the game with probability 1.


Chal.
$k \leftarrow \{0,1\}^{128}$
$r \leftarrow \{0,1\}^{128}$
$b \leftarrow \{0,1\}$
$C = F_k(M_b)$
$M_0 = m_1 \mid m_2 , M_1 = m_2 \mid m_1$
$c \leftarrow E(k, M_b)$
Adv.
Guess
$b' \in \{0,1\}$

7. (**4 pts**) Let F be a pseudorandom function. Show that the following MACs is **insecure**, even if used to authenticate fixed-length messages. (**Gen** outputs a uniform $k =\in \{0,1\}^n$. Let $\langle i \rangle$ denote an $n/2$-bit encoding of the integer i.)
To authenticate a message $m = m_1, \ldots, m_\ell$, where $m_i = \in \{0,1\}^{n/2}$, compute

$$t \coloneqq F_k(\langle 1 \rangle \parallel m_1) \oplus \ldots \oplus F_k(\langle \ell \rangle \parallel m_\ell)$$

For simplicity we fix $\ell = 2$. Let $\mathcal{A}$ be an adversary in the MAC-forge game

Let $m_1, m_1', m_2, m_2' \in \{0,1\}^{n/2}$ with $m_1 \neq m_1'$ and $m_2 \neq m_2'$. The attacker obtains

tag t_1 on the message $M_1 = m_1, m_2$,

$$\boldsymbol{t_1 = F_k(\langle 1 \rangle \parallel m_1) \oplus F_k(\langle 2 \rangle \parallel m_2)}$$

tag t_2 on the message $M_2 = m_1, m_2'$

$$\boldsymbol{t_2 = F_k(\langle 1 \rangle \parallel m_1) \oplus F_k(\langle 2 \rangle \parallel m_2')}$$

tag t_3 on the message $M_3 = m_1', m_2$

$$\boldsymbol{t_3 = F_k(\langle 1 \rangle \parallel m_1') \oplus F_k(\langle 2 \rangle \parallel m_2)}$$

One can then verify that $t_1 \oplus t_2 \oplus t_3$ is a valid forgery tag for the new message $M_4 = m_1', m_2'$.

$$\boldsymbol{t_4 = F_k(\langle 1 \rangle \parallel m_1') \oplus F_k(\langle 2 \rangle \parallel m_2')}$$

8. (**4 pts**) What security goals the AES-GCM can achieve? Show how you can use AES-GCM to avoid replay attacks?

Security goals:

- Data privacy: adversary should not be able to read message **M**
- Data integrity: adversary should not be able to modify message **M**
- Data authenticity: message M really originated from Alice

You add sequence number (sn) in the Associated Data (AD) field of AES-GCM. The attacker cannot change the sn.

9. (**4 pts**) Let H be a collision-resistant hash function (e.g., SHA256), and

(a) Let p be a random 20-bit string. Assume an attacker knows the only the length of p and learns $H(p)$. Can the attacker recover p. Explain your answer.

Yes. By brute force on p.

It is feasible to do 2^{20}

(b) Using the hash function alone does not protect the integrity of the message against an active attacker. Show how it is possible to mount man-in-the-middle attack.

The active attacker can stop the message and send his own message along with its hash value. The recipient will not be able to verify that the message and the corresponding hash comes from an adversary.