Birla Institute of Technology and Science

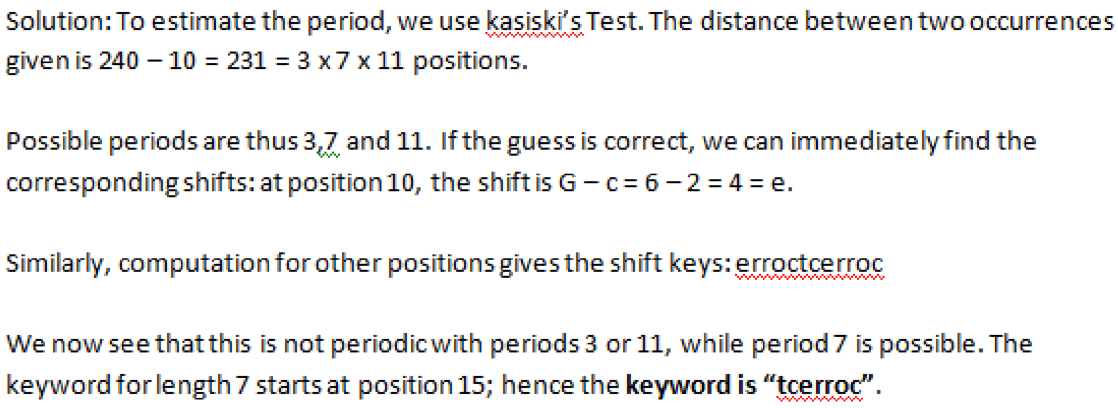
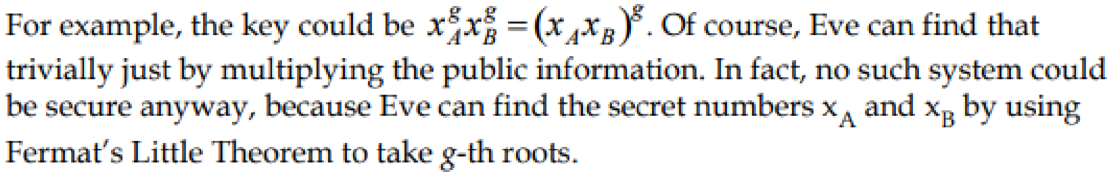
Second Semester 2022-2023

CRYPTOGRAPHY (BITS-F463)

Final Exam (Open Book)

Duration: 110 Min **Maximum Marks:20**

Instructions: Answer all questions. Answers without any explanation or justification will not be given any marks even when the final answer is correct. Write your assumptions clearly.

Q1.	**(a)** You have found an old ciphertext, where you know that the plaintext discusses cryptographic methods. You suspect that a Vigenere Cipher has been used and therefore look for repeated strings in the ciphertext.You find that the string GIPDVHIVRGVA occurs twice in the ciphertext. The first occurrence starts at character position 10 in the text and the second at character position 241 (start counting from 1). You make the inspired guess that this ciphertext sequence is the encryption of the plaintext word "cryptography". If this guess is correct, what is the key?  Solution: To estimate the period, we use kasiski's Test. The distance between two occurrences given is 240 – 10 = 231 = 3 x 7 x 11 positions. Possible periods are thus 3,7 and 11. If the guess is correct, we can immediately find the corresponding shifts: at position 10, the shift is G – c = 6 – 2 = 4 = e. Similarly, computation for other positions gives the shift keys: erroctcerroc We now see that this is not periodic with periods 3 or 11, while period 7 is possible. The keyword for length 7 starts at position 15; hence the **keyword is "tcerroc".**	**[3M]**
.	**(b)** In Diffie-Helman key exchange protocol both the parties generate a public value (g^{α} *mod p*) where *g* is a generator, *p* is a prime number and α *is* a locally generated random number. Based on the public value of each other both the parties can calculate a common secret value. Consider a modified Diffie-Helman key exchange in which the public value is calculated as α^{g} *mod p* . How two parties can use this modified mechanism to establish a common secret among themselves. How can an adversary get the key without knowing the secret values of both the parties? Can adversary also find the secret values. Solution:  For example, the key could be $x_A^g x_B^g = (x_A x_B)^g$. Of course, Eve can find that trivially just by multiplying the public information. In fact, no such system could be secure anyway, because Eve can find the secret numbers x_A and x_B by using Fermat's Little Theorem to take *g*-th roots.	**[3M]**

[2M]

(c) In elliptic curve arithmetic what would be the value of 2Q and 3Q where Q is a point on the elliptic curve if the tangent line at Q is vertical?

Solution:

For a vertical tangent line, the point of intersection is infinity. Therefore $2Q = O$. $3Q = 2Q + Q = O + Q = Q$.

[2M]

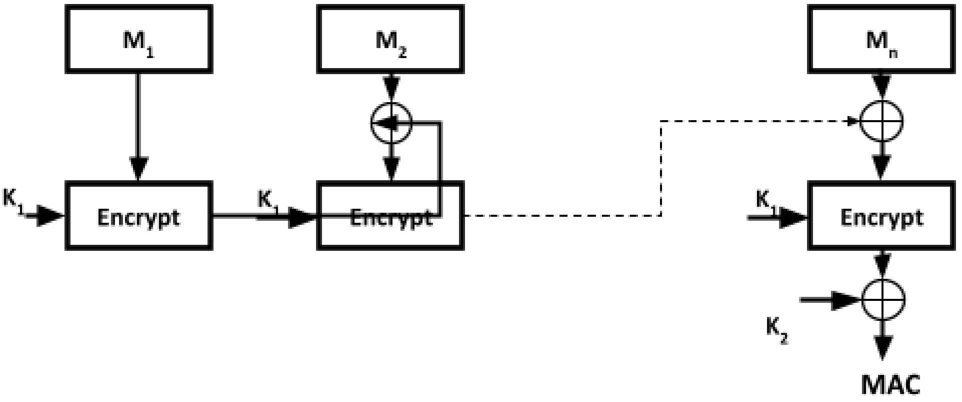
(d) Consider the elliptic curve define by $y^2 = x^3 + 2x + 1$ with a modulus of $p = 11$. Compute the value of 2G and 3G for the point G = (3,2).

Solution:

correction: G = (3,1)
Answer:

Q2. **(a)** Using a chosen plaintext attack, show that the following MAC is not secured. Assume the message length is the multiple of block size.

[4M]

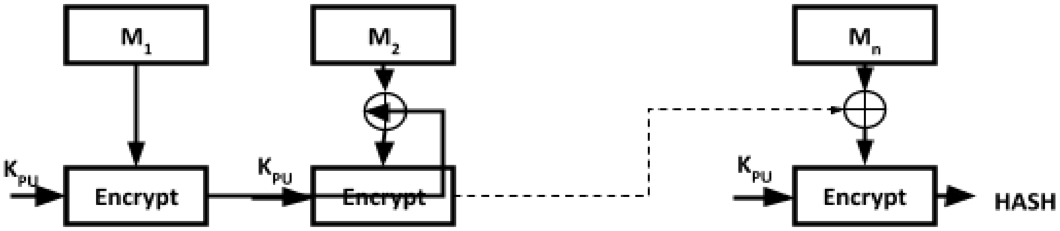


Solution: Suppose an adversary is able to ask for the MACs of three messages $0 = 0^n$, where n is the cipher block size; the message $1 = 1^n$ and the message 1||0. As a result of these three queries the adversary gets

T_0 = CBC(K,**0**) $\oplus$ K_1; T_1 = CBC(K,**1**) $\oplus$ K_1, and T_2 = CBC(K,[CBC(K,1)]) $\oplus K_1$.

There are two blocks to process. The output of the encryption of the first message block is E(k, 0) = CBC(k, 0) = $T_0 \oplus K_1$. This is XORed with the second message block ($T_0 + T_1$), so the output of the second encryption is ($T_1 \oplus K_1$) = CBC(k,1) = E(k,1). So the output of the second encryption E(k, [E(k,1)]) = CBC(k, [CBC(k,1)]) = $T_2 \oplus K_1$. After the final XOR with k_1, we get VMAC(k, 0 || ($T_0 \oplus T_1$)) = T_2.

(b) Consider the following hash function based on asymmetric key encryption with known public and private keys. Either the public key or the private key can be used for the encryption purpose. **[4M]**



Show that the above hash does not satisfy the weak collision resistant property.

Solution: Given a two block message B1, B2 and its hash RSAH (B1, B2) = RSA (RSA (B1) ⊕ B2)
Given an arbitrary block C1, choose C2 so that RSAH (C1, C2) = RSAH (B1, B2). Thus, the hash function does not satisfy weak collision resistance.
The opponent has the two-block message B1, B2 and its hash RSAH (B1, B2). The following attack will work. Choose an arbitrary C1 and choose C2 such that:

C2 = RSA (C1) ⊕ RSA (B1) ⊕ B2

Then RSA (C1) ⊕ C2 = RSA (C1) ⊕ RSA (C1) ⊕ RSA (B1) ⊕ B2
= RSA (B1) ⊕ B2

So RSAH (C1, C2) = RSA [RSA (C1) ⊕ C2)] = RSA [RSA (B1) ⊕ B2]
= RSAH (B1, B2)

Q. 5 Consider a family of functions $F: KXD \rightarrow R$. Imagine that you are in a room which contains a terminal connected to a computer outside your room. You can type something into your terminal and send it out, and an answer will come back. The allowed questions you can type must be elements of the domain D, and the answers you get back will be elements of the range R. The computer outside your room implements a function $g: D \rightarrow R$ so that whenever you type a value X you get back g(X). However, your only access to g is via this interface, so the only thing you can see is the input-output behavior of g. We consider two different ways in which g will be chosen, giving rise to two different worlds. **[6M]**

World 0: The function g is drawn at random from the set Func(D,R), where Func(D,R) is the set of all one to one functions from D to R.
World 1: The function g is drawn at random from F. This means that a key is chosen randomly from K and then g is set to F_K.

You are not told which of the two worlds was chosen. The choice of world, and of the corresponding function g, is made before you enter the room, meaning before you start typing questions. Once made, however, these choices are fixed until your "session" is over. Your job is to discover which world you are in. To do this, the only resource available to you is your link enabling you to provide values X and get back g(X). After trying some number of values of your choice, you must make a decision regarding which world you are in. The quality of pseudorandom family F can be thought of as measured by the difficulty of telling, in the above game, whether you are in World 0 or in World 1. Eventually, your algorithm A, would output a bit b which is its decision as to which world you are interacting with. Outputting the bit "1" means that A "thinks" it is in world 1; outputting the bit "0" means that A thinks it is in world 0.

Let $F: K\,X\,D \rightarrow R$ be a family of functions, and let A be an algorithm that takes an oracle for a function $g: D \rightarrow R$, and returns a bit.

The *prp-advantage* of A is defined as $ADv_{F}^{prp}(A) = \Pr Pr\ [EXP(1) = 1]\ - \Pr Pr\ [EXP(0)]\ = 1]$, ,where EXP(1) and EXP(0) corresponds to interacting with world 1 and world 0 respectively.

Now, consider the following block cipher (family of functions) $E: \{0, 1\}^3\ X\ \{0, 1\}^2 \rightarrow \{0, 1\}^2$:

Key	0	1	2	3
0	0	1	2	3
1	3	0	1	2
2	2	3	0	1
3	1	2	3	0
4	0	3	2	1
5	1	0	3	2
6	2	1	0	3
7	3	2	1	0

The eight possible keys are the eight rows, and each row shows where the points to which 0, 1, 2, and 3 map. Compute the maximal **PRP-advantage** an adversary can get **(a) with one oracle query (b) with two oracle queries, and (c) with four oracle queries.**

By oracle query, we mean that the adversary can choose a message (say x) from his side and get the value of E(key,x) for a key value randomly chosen by the challenger before the start of the game and not known to the adversary. Also give the algorithms of adversary for each case.

Ans: 0, 1/3, 2/3

Q6. **[8M]**

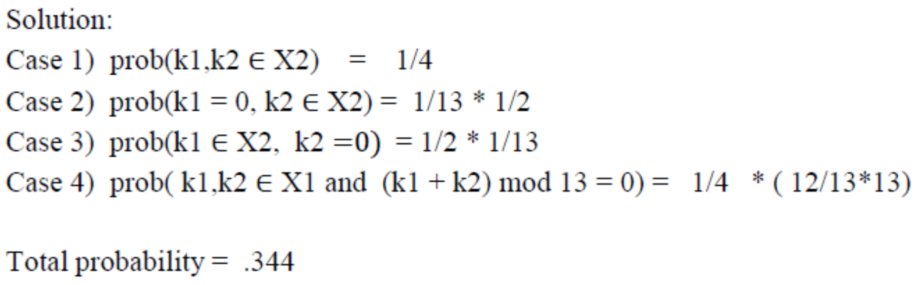
Consider a variant of Caesar cipher defined as follows. Let X1 = {0,1,…., 12} and X2 = {13,14,…., 25}. Let M=C=K=X=X1 ∪ X2. Define

$E_k(m) = \ \{(m\ +\ k) mod\ 13 \qquad if\ k \in X_1 \wedge m \in X_1\ m \qquad if\ k \in X_1 \wedge m \in X_2\ r$

Now consider, a double encryption scheme$(E^2,\ D^2)$, as $E^2_{k1,k2}(m) = E_{k2}(E_{k1}(m))$. Assume k1 and k2 are independent. Answer the following questions.

Let $c_0\ = 7$ be the observed ciphertext generated using E^2. What is the probability that c_0 is the encryption of plaintext $m\ = m_0\ s.t. m_0\ = c_0\ i.e. Prob[m\ = m_0|c\ = c_0]$.

Is $\left(E^2,\ D^2\right)$ perfectly secure? Why or why not?



Solution:
Case 1) prob(k1,k2 ∈ X2) = 1/4
Case 2) prob(k1 = 0, k2 ∈ X2) = 1/13 * 1/2
Case 3) prob(k1 ∈ X2, k2 =0) = 1/2 * 1/13
Case 4) prob(k1,k2 ∈ X1 and (k1 + k2) mod 13 = 0) = 1/4 * (12/13*13)

Total probability = .344

No, clear from part a). The probability of Prob[m = 7] = 1/26, whereas the Prob[m =7| c = 7] = .60

Q8. a) Why is the authentication protocol as shown in Fig. 1 insecure even when instantiated and used correctly (i.e., both r_a and r_b are random numbers and K_{AB} was securely shared in the first place)?

[8M]

b) Present two ways to modify the authentication protocol as given in Fig 1. Write down the modified protocols in details and state your assumptions clearly.

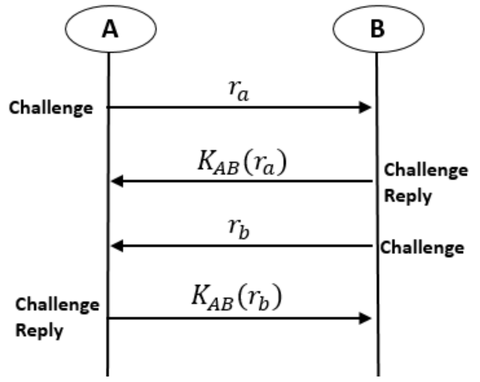


Fig. 1

Ans, a)

Solution:

An adversary can bypass this authentication protocol without knowing K_{AB}. This attack can be carried out as follows:

1. Eve (as an adversary) initiates a connection to Bob by sending r_1 and receiving r_2 and $K_{AB}(r_1)$ back.
2. She opens a second connection to Alice by sending the challenge that she got from Bob in the previous connection, i.e., r_2. As a result, she receives r_3 and $K_{AB}(r_2)$ as a response from Bob.
3. She then can send $K_{AB}(r_2)$ back to Bob on the original connection, leading to the completion of the first authentication protocol with Bob.

Ans b)

Solution:

One way to do it is to change $K_{AB}(r_a)$ to $K_{AB}(r_a||B)$ and $K_{AB}(r_b)$ to $K_{AB}(r_b||A)$. The resulting protocol will be:

1. $A \rightarrow B$: r_a
2. $B \rightarrow A$: $K_{AB}(r_a||B)$
3. $B \rightarrow A$: r_b
4. $A \rightarrow B$: $K_{AB}(r_b||A)$

Or use distinct keys: K_{AB} and K_{BA} in different directions, i.e., changing $K_{AB}(r_b)$ to $K_{BA}(r_b)$. The resulting protocol will be:

1. $A \rightarrow B$: r_a
2. $B \rightarrow A$: $K_{AB}(r_a)$
3. $B \rightarrow A$: r_b
4. $A \rightarrow B$: $K_{BA}(r_b)$

Or use sequence numbers or timestamps instead of nonces in the above protocols.