**Applications of Cryptography CSCE 4050/5050 (Spring 2025)**

**Homework 3**

**Name; Pradip Sapkota**

**Student ID: 11821781**

1. [**Chacha20 encryption/decryption: implementation**] Write a program, which encrypts a message “Pay $2000 to Bob” (without quotes) using the Chacha20 cipher. Your program must print on the screen the following: the plaintext in text and in hex, the key, nonce, and the resulting ciphertext, all in hex. In a practical application, the key and nonce are chosen at random. In this assignment, for simplicity, they will be hardcoded into your program. The key and nonce will be selected as follows: take the last two letters of your student ID, for example, if your student is 12345678, then the last two letters are “78”. The key will be “0x78..78” and the nonce will be “0x87..87”. The ciphertext (in hex) will be saved into a text file “ciphertext.txt”. Next, your program will decrypt the ciphertext and print the resulting plaintext to the screen. Note that in all the above operations, you will use the same key and nonce, which are chosen as described above.

**IMPORTANT NOTE:** Make sure to select the key and nonce correctly (they will be different for all students). An incorrect selection will result in reduced grades.

**Answer:**

A screenshot of a computer

Description automatically generated

1. [**Chacha20’s keystream quality: implementation**] Use the same key and nonce as in the previous question. Write a program which stores the first 256 bytes of the Chacha20 keystream in the file “keystream.txt” (in the hex format), this file will be included into your submission. Also, the program will test the keystream quality using the Frequency (Monobit) Test from NIST SP 800-22 (rev. 1a):  
   <https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762>  
   (see Sec. 2.1). Print the values computed as described in that section. Use the decision rule at the 1% level (see Sec. 2.1.5) to conclude if the keystream is random or not. Explain your decision.

**Answer:**

A computer screen with white text

Description automatically generated

The NIST Frequency Test determines if the keystream's 1s and 0s are balanced, as would be expected in a random sequence. The P-value, in this case 0.507387, is determined by the test and compared with the significance level of 1% (0.01). Since 0.507387>= 0.01, we can conclude the sequence as random that means ChaCha20 keystream passes the test based on NIST frequency Test at 1% significant level.

1. [**Malleability of stream ciphers: implementation**] Copy the file “ciphertext.txt” (it was produced in Question 1) to a file “ciphertext1.txt”. Then manually modify it so that the decryption would result in “Pay $7000 to Bob”. Print the modified “ciphertext1.txt” to the screen (in hex). Run your decryption program from Question 1 and confirm that the decrypted plaintext is now modified as expected. Explain how you modified the ciphertext and why your attack was successful.

**Answer:**

**A black screen with white text

Description automatically generated**

Well, I manually modified the 6th byte of the original ciphertext (9a0c22668598a425bb9c125285dc6717) i.e. **98** to **9d** making modified ciphertext as 9a0c2266859da425bb9c125285dc6717. Using the following equation

Keystream = ciphertext XOR plaintext = 0x98 XOR 0x32 = 0xAA

New ciphertext = keystream XOR new plaintext = 0xAA XOR 0X37 = 0X9D

The attack was successful because stream ciphers like ChaCha20, encrypt data by XORing a keystream with plaintext. Flipping the ciphertext's bits instantly switches the plaintext's bits as well because the keystream is reused for decryption. By changing the byte which corresponds to ‘5’ to ‘7’ without a key, we altered the decrypted plaintext.

1. [**Predicting LCG**] Consider a linear congruential generator (LCG) defined in Lecture 3-1.   
   Suppose that it is known that p = 11, and that you have observed the first three output values:  
   r[1] = 6, r[2] = 2, and r[3] = 1.

Predict the next value r[4]. Explain your answer.

**Hints and remarks:** You will find r[4] by solving a linear system of equations over Z11 according to the approach shown in Lecture 3-1. Other methods will not be given credit.

**Answer:**

Here,

We have,

r [1] = 6, r [2] = 2 and r[3] =1

r [4] =?

now, r[i] = (a ⋅ r[i-1] + b) mod p

Two equations we have,

6a + b = 2 mod 11

2a + b = 1 mod 11

On subtraction, 6a + b – 2a – b = 2-1 mod 11

4a = 1 mod 11

4 ⋅ 3 = 12 = 1 mod 11 ( The multiplicative inverse of 4 mod 11 is 3)

Thus, a = 3

Now from equation 2;

2 ⋅ 3 + b = 1 mod 11

b = -5 mod 11

b = 6 mod 11

Now,

r [4] = (a ⋅ r[3] + b) mod p

r [4] = 3 ⋅ 1 + 6 mod 11

r [4] = 9 mod 11

**Therefore, the value of r[4] found using the requested method is 9 .**

1. [**Advantage (distinguishing)**] Let G : K → {0,1}n be a PRG that outputs the keystream of RC4 (stream cipher). We know from Lecture 2-2 that G has some weaknesses. Define a statistical test for distinguishing the output of G from a uniformly random sequence and compute the respective advantage.

Hints: Follow the example in Lecture 3-1. Remember that the RC4 PRG outputs byte values.   
Also, for a uniform distribution, Pr[ any byte is 0 ] = 1/256. Your statistical test can exploit either of the two weaknesses presented in Lecture 3-1.

**Answer:**

Let G : K → {0,1}n be a PRG that outputs the keystream of RC4 (stream cipher).

**Statistical Definition:**

Here, test A checks if the second byte of the sequence is 0. If it is, its RC4(output 1); otherwise, its random (output 0).

In RC4, Pr[2nd byte] = 0] = 2/256 //instead of 1/256 due to the RC4’s bias.

i.e. P[(G(k)) = 1] = 2/256

For a random sequence, Pr[any byte = 0] = 1/256 = P[(A(r) = 1]

**The advantage of the test A given by;**

AdvPRG[A,G] =| P[(G(k)) = 1] – P[(A(r) = 1] | ∈ [0,1]

AdvPRG[A,G] = | 2\256 – 1/256| = 1/256

which is a constant and greater than negligible value 1/(2)^256 , that means test A can distinguish RC4 output from truly random sequence.

1. [**Definition of semantic security**] Consider Slide 12 of Lecture 3-2 and suppose that the adversary B selects the messages as follows: m0 such that lsb(m0) = 1 and m1 such that lsb(m1) = 0.   
   Compute the advantage AdvSS[B,E] in this case. Explain your answer.

**Answer:**

Given adversary B selects the messages as follows: m0 such that lsb(m0) = 1 and m1 such that lsb(m1) = 0,

The advantage AdvSS[B,E] still remains 1.

Because, adversary B selects messages m0 and m1 such that lsb(m0) = 1 and lsb(m1) = 0, then adversary A can always deduce the LSB of the plaintext from the ciphertext, B uses A’s output to determine b.

If lsb(mb) = 1, B concludes b = 0 as m0  has least significant bit 1

If lsb(mb) = 0, B concludes b = 1 as m1  has least significant bit 0

This way B can distinguish between b=0 and b=1

Also,

AdvSS[B,E] = | Pr[EXP(0) = 1] – Pr[EXP(1) = 1] | = | 0 – 1 | = 1

Because B can deduce b = 0 correctly making Pr[EXP(0) = 1] = 0 and b = 1 making Pr[EXP(1) = 1] =1.