Message Encryption

Qcamp 2018, Experimental Session 01/06/2018

1 Introduction

In QKD, messages are encrypted with symmetric key, contrary to the asymmetrical classical cryptography which requires public and private key pairs. Symmetric keys are 'theoretically' more secure, as long the distribution process is secure in itself, noting all the necessary assumptions [1]. This is because in asymmetric cryptography, if one has a very powerful computer (i.e. quantum computer), one 'technically' can obtain the private key from the public key [2]. However, it does not mean that messages encrypted in any "QKD-ish" way are always secure, as you will see below. As people always say: with a great power comes great responsibility.

1.1 Main Idea

Ideally, to obtain the 'information-theoretic security', the key length should be as long as the message. However, similar to the experiment, key rates generated with today QKD systems are still pretty low [3]. Thus, most QKD systems employs a hybrid approach: the short key is expanded using classical means [4].

In our experiment, the key generated with QKD serves as the seed to the Mersenne Twister pseudo-random number generator (PRNG), where the output is the expanded key. This expanded key has the same length with the message. To encrypt the message, an XOR operation (refer to Table 1) is performed between each bit of the expanded key and the message.

A	B	XOR(A,B)
0	0	0
0	1	1
1	0	1
1	1	0

Table 1: XOR (exclusive OR) truth table

To decrypt the message, one needs the same key (which is distributed through QKD), and perform the key expansion by using the same method. The expanded key can then be XOR-ed with the encrypted text to produce the original message.

[1] https://arxiv.org/abs/0902.2839

[2] This statement is true until today, but it does not mean that it will always be true: google "post quantum cryptography".

[3] https://www.nature.com/articles/npjqi201625

[4] Usually, key expansion is performed using ciphers such as Advanced Encryption Systems (AES).

2 Assignment

Task 1 [2 pts] Show that for any binary values of A and B:

$$\text{XOR}(\text{XOR}(A, B), B) = A$$

i.e. performing XOR operation twice on A gives back the value of A.

Task 2 [2 pts] One simple way to produce PRNG is with Linear Feedback Shift Register (LFSR) [5]. LFSR uses a simple, iterative procedures to produce a string of pseudo-random bits, and hence very easy to be implemented by software and machines. Similar to other PRNGs, the generated bit sequences will repeat itself after a number of times [6].

For the context of this exercise, we are using LFSR particularly because the repetition sequence is very short for small number of bits. With this in consideration, by using a short sequence LFSR to expand the key, we can crack the message very easily.

The maximum length LFSR sequence for $n = 7$ bits (127 numbers in total) is given below:
1 1 1 1 1 1 1 0 1 0 1 0 1 0 0 1 1 0 0 1 1 1 0 1 1 1 0 1 0 0 1 0 1 1 0 0 0 1 1 0 1 1 1 1 0 1 1 0 1 0 1 1 0 1 1 0
0 1 0 0 1 0 0 0 1 1 1 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 1 0 1 1 1 0 0 1 1 0 1 0 0 0 1 0 0 1 1 1 1 0 0 0 1 0 1 0 0 0
0 1 1 0 0 0 0 0 1 0 0 0 0 0 0

You will encode a 16-bit message with 16 bits of the sequence. Choose one bit from the sequence randomly (as a seed), and copy the 16-bit numbers that comes afterwards! This will be the key [7].

[5]https://en.wikipedia.org/wiki/Linear-feedback_shift_register

[6]This is also another weakness of PRNGs, but don't worry; the ones used in your daily cryptographic applications has extremely long repetition sequences.

[7]The analogy is that, the chosen location is the seed (your original key), and the sequence that comes afterwards is the expanded key.

Task 3 [2 pts] The 16-bit messages to encode is “QC” in ASCII representation: 01010001 01100011. By using the key obtained earlier, encrypt the message (by performing XOR on each bit of the message and the key). Write down the encrypted message below!

Task 4 [2 pts] A particular cryptographic weakness of any XOR-ed ASCII string is that, right off the bat, we can already deduce some information of the key. For example, if you know that the message only contains letters [A-Z and a-z], then you know for certain that some bits in the message have some particular values. Write down what the binary representation of each message character looks like! Hint: out of 8 bits in a character, two bits are certain. You might want to look at the ASCII table!

Task 5 [2 pts] You obtain a 3-character encrypted text below. You also know that the messages are ASCII letters. By using the strategy in Task 4, and by looking closely at the LFSR sequence, predict where the seed is located, and decrypt the message!

0 0 0 1 0 1 0 0   0 1 0 0 1 1 1 0   1 1 1 0 1 1 1 0