Homework-I

Modular Encryption

Given a number *M* (which is not a power of 2 e.g. 5, 7 or 9), read a binary file in format of numbers from 0 to *M*-1. You can initially break the file into some blocks and then extract the numbers from each block. In this homework, all bits of the block is considered as a large positive integer and the numbers are the digits of that integer after representing based on *M*. For example given M=5, consider an 8-digit block 01010101 is equal to $85=(320)_5$. Maximum number of an 8-digit block is $255=(2010)_5$. Hence, each 8-digit block should be converted to a 4-digit number based on 5. This is why; 85 is equal to $(0320)_5$ not $(320)_5$.

Then convert each digit to another digit based on *M* by a one-to-one mapping. After that, you have a 4-digit number based on *M* which should be converted to a large number based on 2 in order to write in the new block. Maximum number based on 5 is $(4444)_5=624=(1001110000)_2$. Hence, each number should be coded as a 10-digit number in the new block. For example, $(0320)_5$ can be mapped to $(2012)_5=257=(0100000001)_2$. In other words, each 8-digit block is converted to a 10-digit one and the size of the file will be multiplied by 10/8.

In some cases, some bits should be added to the end of the bit stream to make the number of bits a multiple of the size of the block. Let's call this operation as bit padding. For example if the number of bits of the original file is 11 bytes equal to 88 bits and the size of the plain block is 9, it is desired to append 2 extra bits to the end of file. These bits can be selected arbitrarily. In addition, the crypto-blocks should be written in a file which is a multiplier of 8 bits (one byte) which may need bit padding. For decryption, by having *M*, the number of padding bits can be computed and ignored.

Write a program which has 4 options at the main menu:

1- Generate a key and store it in a file: In this mode, a key containing *M*, size of original block and the mapping function is generated and will be stored in a file called as the key-file if the user accepts it.
2- Encrypt a file: In this mode, a binary file and a key-file are given by the user and the encrypted file will be stored.
3- Decrypt a file: In this mode, an encrypted file and a key-file are given by the user and the original file will be generated and stored.

4- Discover the key: In this mode, the original file and its encrypted file are given by the user and the key-file will be discovered and stored. (Optional)

Regards,
M. Taheri