

1 Problem

Implement AES *encryption* as specified in the FIPS 197 standard or as described in Chapter 5 of your textbook in Java but only for 128 bit keys. Do not use *any* tables defined in the specification for e.g., the ByteSub table, the MixColumn matrix etc. You may, however, use the matrix for the affine transformation used with ByteSub. Do not also use the simplifications for multiplication when multiplying polynomials modulo the polynomials $x^8 + x^4 + x^3 + x + 1$ and $x^4 + 1$. You are encouraged to use JDK1.5 for the project but are not required to do so. The (tentative) deadline for submission is midnight March 12.

In order to better understand addition, multiplication, and division in $GF(2^8)$, you may want to experiment with the expression evaluator that I have a link to on the course web page. You may use my implementation of $GF(2^8)$ that uses the AES irreducible polynomial $x^8 + x^4 + x^3 + x + 1$ for the project. A link to its class file and javadoc that describes its interface can be found on the course web page.

2 Description

The input/output interface of **AES** should be as follows:

Running

```
java AES k=00000000000000000000000000000000 t=12345678123456781234567812345678
```

should generate an output approximating¹ the following:

t=12345678123456781234567812345678	←	128 bits of plaintext
k=00000000000000000000000000000000	←	128 bits of input key
c=e741356a719962e2c5ceb8d86171a9f0	←	128 bits of ciphertext
Rd	Round Key	State
0	00000000000000000000000000000000	12345678123456781234567812345678
1	62636363626363626363626363636363	ceee14e9ceee14e9ceee14e9ceee14e9
.
10	b4ef5b.....8e	e741356a719962e2c5ceb8d86171a9f0

There are eleven round keys (0 – 10) in all, and 11 state values to be output. Each is a sixteen byte quantity which you’ll output as hexadecimal nibbles. The first round key is the specified encryption key, while the rest are generated as per the AES key schedule. The state is the value of the 4×4 state “matrix” at the end of each round. To see the values of the internal “state” at each round of AES, try the URL <http://www-appn.comp.nus.edu.sg/~cs4236/cgi-bin/aes.cgi>. It will also be helpful in debugging your program.

3 An Example Transformation

Consider the following value of “state” at the beginning of a round:

$$S = 90dae4efa36ae74502801ca1c48a0879$$

¹ Very closely.

This is sixteen bytes long and is initialized to the 16 byte plaintext at the start of encryption. The “state” carries the transformations across the various encryption rounds. As described in the FIPS standard, it can be regarded as a 4×4 matrix written in column order thusly

$$\begin{pmatrix} 90 & a3 & 02 & c4 \\ da & 6a & 80 & 8a \\ e4 & e7 & 1c & 08 \\ ef & 45 & a1 & 79 \end{pmatrix}$$

3.1 ByteSub

The ByteSub transformation is the first transformation made in a round. Each byte of the state is individually and independently transformed by computing its inverse followed by a linear transformation. Consider the first byte of the state, 0x90. If you compute its inverse modulo the irreducible polynomial $(x^8 + x^4 + x^3 + x + 1)$, you get 0xde. If you apply the affine transformation to $de = 11011110_2$ as given on page 132 of your textbook, you get

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

which is 0x60. So the element $90 \rightarrow 60$ after ByteSub. The same transformation is applied to every byte in the “state”. After the transformation, the “state” matrix looks as follows.

$$\begin{pmatrix} 60 & 0a & 77 & 1c \\ 57 & 02 & cd & 7e \\ 69 & 94 & 9c & 30 \\ df & 6e & 32 & b6 \end{pmatrix}$$

3.2 ShiftRow

The ShiftRow transformation is simple enough. After the transformation the “state” matrix looks like

$$\begin{pmatrix} 60 & 0a & 77 & 1c \\ 02 & cd & 7e & 57 \\ 9c & 30 & 69 & 94 \\ b6 & df & 6e & 32 \end{pmatrix}$$

3.3 MixColumn

MixColumn is a little tricky so let’s do a complete transformation of the first column of the state matrix to see how it’s transformed from the first column vector to the second.

$$\begin{pmatrix} 60 \\ 02 \\ 9c \\ b6 \end{pmatrix} \xrightarrow{MC} \begin{pmatrix} ec \\ 6d \\ 80 \\ 49 \end{pmatrix}$$

the README file. Submit only java source files, not any .class files. Make sure that you package your source files so they'll unjar in the current directory and will *not* create any directories below. I should be able to compile your program by unjarring your submitted jar file and running `javac *.java` to create at least an AES.class. Your program should be written in the default package.

Just mail me the jar file by the submission date.