VE475 Introduction to Cryptography
Homework 3

Jiang, Sifan
jasperrice@sjtu.edu.cn
515370910040

June 7, 2019

Ex. 1 - Finite fields

1. Assume X^2+1 is reducible in $\mathbb{F}_3[X]$, then X^2+1 can be written as the product of two polynomials of lower degree. The possible factors of X^2+1 are X, $X+1$, and $X+2$.

$$\begin{aligned} X \cdot X &= X^2 \\ X \cdot (X+1) &= X^2 + X \\ X \cdot (X+2) &= X^2 + 2X \\ (X+1) \cdot (X+1) &= X^2 + 2X + 1 \\ (X+1) \cdot (X+2) &= X^2 + 2 \\ (X+2) \cdot (X+2) &= X^2 + X + 1 \end{aligned}$$

   Since none of them is equal to X^2+1, it is irreducible in $\mathbb{F}_3[X]$.

2. Since X^2+1 is irreducible in $\mathbb{F}_3[X]$ and $1+2X$ is a polynomial in $\mathbb{F}_3[X]$, there exists a polynomial $B(X)$ such that

$$(1+2X) \cdot B(X) \equiv 1 \mod (X^2+1)$$

   So, $B(X)$ is the multiplicative inverse of $1+2X \mod X^2+1$ in $F_3[X]$.

3. Using the extended Euclidean algorithm.

   Initially, $r_0 = X^2+1$, $r_1 = 2X+1$, $s_0 = 0$, $s_1 = 1$, $t_0 = 1$, and $t_1 = 0$.

q	r_0	r_1	s_0	s_1	t_0	t_1
0	$2X+1$	X^2+1	1	0	0	1
$2X$	$X+1$	$2X+1$	X	1	1	0
2	2	$X+1$	$X+1$	X	1	1
$2X$	1	2	X^2+2X	$X+1$	$X+1$	1
2	0	1	X^2+1	X^2+2X	$X+2$	$X+1$

   So, the multiplicative inverse of $1+2X \mod X^2+1$ in $\mathbb{F}_3[X]$ is $X^2+2X = 2+2X$.

Ex. 2 - AES

1. (a) *InvShiftRows* should be described as: cyclically shift to the right row i by offset i, $0 \leq i \leq 3$.

   (b) Since in the $AddRoundKey$ layer, each byte of the state is combined with a byte from the round key using the XOR operation ($\oplus$). Also, since $(a \oplus k) \oplus k = a$, where $a \in \{0, 1\}$, and $k \in \{0, 1\}$, the inverse of the layer $AddRoundKey$ is just applying $AddRoundKey$ again.

   (c) In *MixColumns* layer, we have $B = C(X) \times A$, where B is the output matrix and A is the state matrix. So the transformation *InvMixColumns* is given by $C^{-1}(X) \times C(X) \times A = A = C^{-1}(X) \times B$. We then need to verify if the result of the multiplication of the two matrices is an identity matrix or not.

$$\begin{aligned}
&\begin{pmatrix} 00001110 & 00001011 & 00001101 & 00001001 \\ 00001001 & 00001110 & 00001011 & 00001101 \\ 00001101 & 00001001 & 00001110 & 00001011 \\ 00001011 & 00001101 & 00001001 & 00001110 \end{pmatrix} \\
\times &\begin{pmatrix} 00000010 & 00000011 & 00000001 & 00000001 \\ 00000001 & 00000010 & 00000011 & 00000001 \\ 00000001 & 00000001 & 00000010 & 00000011 \\ 00000011 & 00000001 & 00000001 & 00000010 \end{pmatrix} \\
= &\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = I
\end{aligned}$$

   The detailed calculation, take the element of first row and first column as the example, is

$$\begin{aligned}
I_{0,0} =&(00001110 \times 00000010) \oplus (00001011 \times 00000001) \oplus \\
&(00001101 \times 00000001) \oplus (00001001 \times 00000011) \\
=&00011100 \oplus 00001011 \oplus 00001101 \oplus (00010010 \oplus 00001001) \\
=&00011100 \oplus 00001011 \oplus 00001101 \oplus 00011011 \\
=&1
\end{aligned}$$

   So, the matrix is the inverse matrix of $C(X)$, and the transformation *InvMixColumns* is given by multiplication by the matrix.

2. In the first round, generate a round key according to the original 128 bits key. Then, apply the $AddRoundKey$ layer with columns $K(40), \cdots, K(43)$. And then apply *InvShiftRows* and *InvSubBytes* layers sequentially.

   In each round of the second to tenth rounds, sequentially apply layer $AddRoundKey$, *InvMixColumns*, *InvShiftRows*, and *InvSubBytes*. The columns used in $AddRoundKey$ layer in round i, where $i \in [2, 10]$, are $K(4(11-i)), \cdots, K(4(11-i)+3)$.

   In the last step, apply $AddRoundKey$ layer with columns $K_0, \cdots, K_3$.

3. Because *InvShiftRows* and *InvSubBytes* apply changes on each element "independently" and "linearly": *InvShiftRows* only shift the order of the elements but not changing the values; *InvSubBytes* map the state value to the combination of row and column numbers. So the order of applying these two layers won't matter the result, thus can be applied on reverse order.

4. (a) Since $AddRoundKey$ apply operation element-wise while $InvMixColumns$ apply operation on the whole matrix, meaning they are using totally different methods and cannot be viewed as "linear". So, reversing the order of application of $AddRoundKey$ and $InvMixColumns$ would give different results.

   (b) The result of applying first $MixColumns$ and then $AddRoundKey$ is

$$(a_{i,j}) \longrightarrow [(m_{i,j})(a_{i,j})] \oplus (k_{i,j})$$

   (c)

   (d)

5.

6.

Ex. 3 - DES

1.

2.

3.

4.

Ex. 4 - Programming