密码学原理·hw2

1 实现keccak算法、AES算法（CBC模式）

1.1 Keccak算法

Keccak算法是SHA-3的标准算法，其思路简述如下：

1. 给定一个可变长度的输入串，将其等分为长度为`r`bits的块（需pad以应对不能整除的情况）；
2. 遍历所有`r`bits块，将其补充`c`bits，使得$r+c=1600$，得到长度为`1600`bits的状态`S`；
3. `S`初始化为0，每次与当前块先异或，再划作5×5的`lanes`，每个lane占64bits；
4. 对`lanes`进行$\theta\to\rho\to\pi\to\chi\to\zeta$步的打乱操作；
5. 所有块遍历完毕后，依据输出要求，每次对`S`进行与第4步相同的操作，直到输出完毕。

官网给出的伪代码如下（笔者使用C++语言实现，代码见`codes/keccak/`目录）：

```
Keccak-f[b](A) {
  for i in 0…n-1
    A = Round[b](A, RC[i])
  return A
}

Round[b](A,RC) {
  # θ step
  C[x] = A[x,0] xor A[x,1] xor A[x,2] xor A[x,3] xor A[x,4],   for x in 0…4
  D[x] = C[x-1] xor rot(C[x+1],1),                             for x in 0…4
  A[x,y] = A[x,y] xor D[x],                           for (x,y) in (0…4,0…4)

  # ρ and π steps
  B[y,2*x+3*y] = rot(A[x,y], r[x,y]),                 for (x,y) in (0…4,0…4)

  # χ step
  A[x,y] = B[x,y] xor ((not B[x+1,y]) and B[x+2,y]),  for (x,y) in (0…4,0…4)

  # ι step
  A[0,0] = A[0,0] xor RC

  return A
}
```

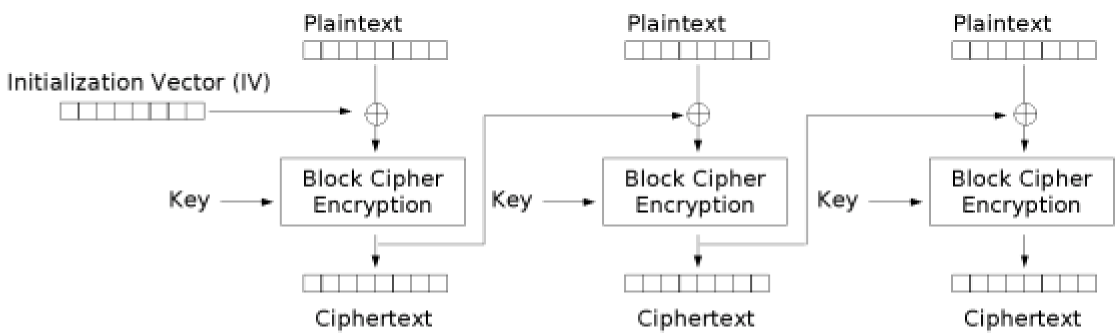
```
Keccak[r,c](Mbytes || Mbits) {
  # Padding
  d = 2^|Mbits| + sum for i=0..|Mbits|-1 of 2^i*Mbits[i]
  P = Mbytes || d || 0x00 || … || 0x00
  P = P xor (0x00 || … || 0x00 || 0x80)

  # Initialization

  S[x,y] = 0,                               for (x,y) in (0…4,0…4)
```

```
  # Absorbing phase
  for each block Pi in P
    S[x,y] = S[x,y] xor Pi[x+5*y],          for (x,y) such that x+5*y < r/w
    S = Keccak-f[r+c](S)

  # Squeezing phase
  Z = empty string
  while output is requested
    Z = Z || S[x,y],                        for (x,y) such that x+5*y < r/w
    S = Keccak-f[r+c](S)

  return Z
}
```

1.2 AES算法（CBC模式）

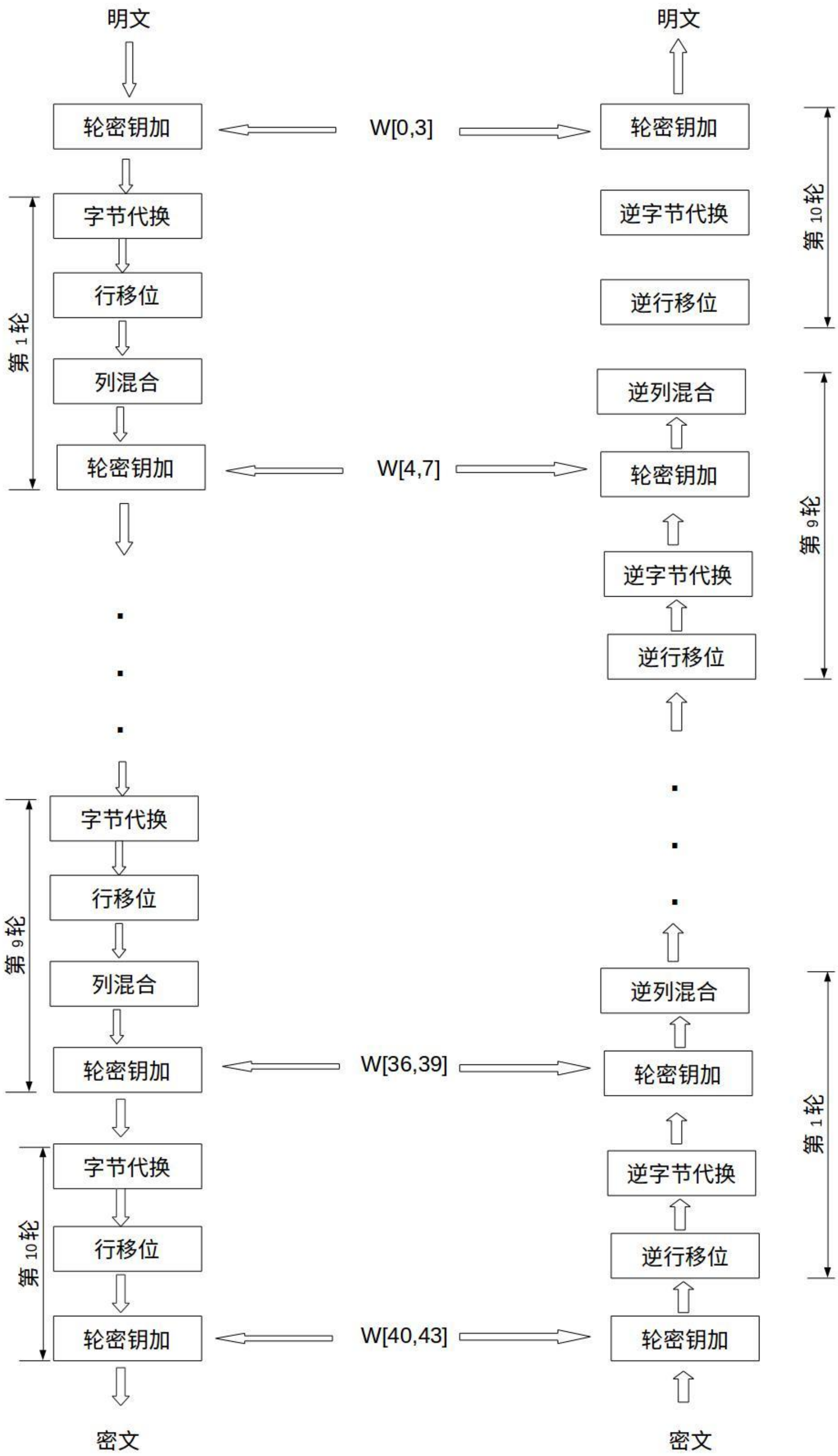
AES算法是一种对称加密算法，即加密和解密使用同一密钥。其思路简述如下：

1. CBC模式：由于AES算法是一种分组加密算法，即必须将输入16字节为一组进行加密，如果组和组之间独立加密，则密文会暴露明文的统计信息。因此，CBC模式使得每个分组的明文都与上一分组的密文先进行XOR（第一分组的明文与一个初始向量 `IV` 进行XOR， `IV` 可以公开传输），从而使得AES可以对抗统计分析。



Cipher Block Chaining (CBC) mode encryption

2. 分组加密：首先将输入按16字节为一组分块，然后以组为单位进行如下的加密-解密操作：



明文
轮密钥加
W[0,3]
字节代换
行移位
列混合
轮密钥加
W[4,7]
第1轮
字节代换
行移位
列混合
轮密钥加
W[36,39]
第9轮
字节代换
行移位
轮密钥加
W[40,43]
第10轮
密文
明文
轮密钥加
逆字节代换
逆行移位
第10轮
逆列混合
轮密钥加
逆字节代换
逆行移位
第9轮
逆列混合
轮密钥加
逆字节代换
逆行移位
第1轮
轮密钥加
密文

3. 轮密钥加：将该分组与密钥逐位XOR即可；
4. 字节代换：利用S盒（加密）和逆S盒（解密）进行查表，每个字节的高位为行，低位为列，将该分组中的各个字节逐个替换即可；
5. 行移位：将该分组（16B）按列主序排成一个4×4矩阵，第0~3行依次左移（加密）或右移（解密）0~3位即可；
6. 列混合：对上述4×4矩阵，在$GF(2^8)$域上作矩阵乘法。

   加密：

$$\begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

   解密：

$$\begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

7. 密钥扩展：将密钥按一定规则扩展为10个（详见代码实现）。

笔者使用C++实现，代码详见`codes/aes/`目录。

2 找出生成0 0 1 0 1 0 1 0 0 1 0 0 0 1的最短线性反馈移位寄存器

解：设$a_0a_1a_2a_3a_4a_5a_6a_7a_8a_9a_{10}a_{11}a_{12}a_{13}=00101010010001$。

a_0	a_1	a_2	a_3	a_4	a_5	a_6
0	0	1	0	1	0	1
a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}
0	0	1	0	0	0	1

1. 初值$f_0(x)=1,l_0=0$，$d_0=c_0a_0=0$，故$f_1(x)=1,l_1=0$
2. $d_1=c_0a_1=0$，故$f_2(x)=1,l_2=0$
3. $d_2=c_0a_2=1$，因$l_0=l_1=l_2=0$，故$f_3(x)=1+x^{2+1}=1+x^3,l_3=3$
4. $d_3=c_0a_3+c_1a_2+c_2a_1+c_3a_0=0$，故$f_4(x)=f_3(x)=1+x^3,l_4=3$
5. $d_4=c_0a_4+c_1a_3+c_2a_2+c_3a_1=1$，因$l_2<l_3=l_4$，故
   $f_5(x)=f_4(x)+x^{4-2}f_2(x)=1+x^2+x^3,l_5=\max\{l_4,4+1-l_4\}=3$

6. $d_5 = c_0a_5 + c_1a_4 + c_2a_3 + c_3a_2 = 0$，故$f_6(x) = 1 + x^2 + x^3, l_6 = 3$
7. $d_6 = a_6 + a_4 + a_3 = 0$，故$f_7(x) = 1 + x^2 + x^3, l_7 = 3$
8. $d_7 = a_7 + a_5 + a_4 = 1$，因$l_2 < l_3 = ... = l_7$，故
   $f_8(x) = f_7(x) + x^{7-2}f_2(x) = 1 + x^2 + x^3 + x^5, l_8 = \max\{l_7, 7 + 1 - l_7\} = 5$
9. $d_8 = a_8 + a_6 + a_5 + a_3 = 1$，因$l_7 < l_8$，故
   $f_9(x) = f_8(x) + x^{8-7}f_7(x) = 1 + x^2 + x^3 + x^5 + x(1 + x^2 + x^3) = 1 + x + x^2 + x^4 + x^5,$
   $l_9 = \max\{l_8, 8 + 1 - l_8\} = 5$
10. $d_9 = a_9 + a_8 + a_6 + a_5 + a_4 = 1$，因$l_7 < l_8 = l_9$，故
    $f_{10}(x) = f_9(x) + x^{9-7}f_7(x) = 1 + x + x^2 + x^4 + x^5 + x^2(1 + x^2 + x^3)$
    $= 1 + x,$
    $l_{10} = \max\{l_9, 9 + 1 - l_9\} = 5$
11. $d_{10} = a_{10} + a_9 = 1$，因$l_7 < l_8 = l_9 = l_{10}$，故
    $f_{11}(x) = f_{10}(x) + x^{10-7}f_7(x) = 1 + x + x^3(1 + x^2 + x^3)$
    $= 1 + x + x^3 + x^5 + x^6,$
    $l_{11} = \max\{l_{10}, 10 + 1 - l_{10}\} = 6$
12. $d_{11} = a_{11} + a_{10} + a_8 + a_6 + a_5 = 1$，因$l_{10} < l_{11}$，故
    $f_{12} = f_{11}(x) + x^{11-10}f_{10}(x) = (1 + x + x^3 + x^5 + x^6) + x(1 + x)$
    $= 1 + x^2 + x^3 + x^5 + x^6,$
    $l_{12} = \max\{l_{11}, 11 + 1 - l_{11}\} = 6$
13. $d_{12} = a_{12} + a_{11} + a_9 + a_8 + a_6 = 0$，故$f_{13}(x) = 1 + x^2 + x^3 + x^5 + x^6, l_{13} = 6$
14. $d_{13} = a_{13} + a_{12} + a_{10} + a_9 + a_7 = 0$，故$f_{14}(x) = 1 + x^2 + x^3 + x^5 + x^6, l_{14} = 6$

故$\langle 1 + x^2 + x^3 + x^5 + x^6, 6\rangle$即为产生所给序列一个周期的最短线性移位寄存器。

参考资料

1. https://keccak.team/keccak_specs_summary.html
2. https://blog.csdn.net/qq_28205153/article/details/55798628
3. https://blog.csdn.net/charleslei/article/details/48710293