Prova 1 de Criptografia

Aluno: Matheus Tararam de Laurentys

NUSP: 9793714

Questão 1: Considero K global, X_{ij} é o Byte j de X no round i.

Item 1:

Primeiro passo: f_1 (round i)

Inversa f_1^{-1}:

Entrada C_i

Saída B_i

$$B_{ij} = C_{ij} \text{ (XOR) } K^j_{2i-1}, \quad j = 1, 4, 5, 8$$

$$B_{ij} = C_{ij} - K^j_{2i-1}, \quad j = 2, 3, 6, 7 \text{ e}$$

$x - y \Rightarrow x - y \bmod 257$

Demonstração

(1) $\forall a, b: b = a \text{ (XOR) } b \text{ (XOR) } a.$

No caso: $B_{ij} = \underbrace{K^j_{2i-1} \text{ (XOR) } B_{ij}}_{C_{ij}} \text{ (XOR) } K^j_{2i-1}$

(2) $C = B + K \bmod 257 \Leftrightarrow C \bmod 257 = B + K \bmod 257$

$\Leftrightarrow C - K \bmod 257 = B \bmod 257$

$\Leftrightarrow C - K = B$

Digitalizado com

Segundo Passo f_2: (round i)

Inversa f_2^{-1}:

Entrada D_i

Saída C_i

$C_{ij} = \log_{45}(D_{ij})$, $j = 1, 4, 5, 8$

$C_{ij} = 45^{(D_{ij})}$, $j = 2, 3, 6, 7$

Demostração:

(3) O enunciado define $\log_{45}(x)$ como a inversa de $45^x \bmod 257$. Assim, $\log_{45}(45^{(x)} \bmod 257) = x$. O contrário, $45^{(\log_{45}(x))} \bmod 257 = x$, também vale pela propriedade de funções inversas.

Terceiro Passo f_3: (round i)

Inversa f_3^{-1}:

Entrada E_i

Saída D_i

$D_{ij} = E_{ij} - K_{2i}^{j}$, $j = 1, 4, 5, 8$

$D_{ij} = E_{ij} (XOR)\ K_{2i}^{j}$, $j = 2, 3, 6, 7$

Demostração:

Iguais a (1) e (2)

Digitalizado com

Item 2:

Definimos, no item anterior, $f_1^{-1}, f_2^{-1}, f_3^{-1}, f_4^{-1}$.

Assim sendo, a inversa de um roundo é definida por: $B = f_1^{-1}(f_2^{-1}(f_3^{-1}(f_4^{-1}(F)))))^{(\dagger)}$.

F foi calculado com: $F = f_4(f_3(f_2(f_1(B)))))^{(*)}$.

Substituindo (*) em (†) temos:

$$B = f_1^{-1}(f_2^{-1}(f_3^{-1}(f_4^{-1}(f_4(f_3(f_2(f_1(B))))))))$$

que é verdadeiro pelas propriedades de funções.

Item 3.

Trasformação final T:

Inversa T^{-1}:

Entrada G

Saída F

$$G_j = F_j \text{ (XOR) } K^j_{2R+1}, \quad j = 1, 4, 5, 8$$

$$G_j = F_j - K^j_{2R+1}, \quad j = 2, 3, 6, 7$$

Demonstração

Iguais a (1) e (2)

Digitalizado com

Quarto Passo f_4

Inversa f_4^{-1}:

Entrada F_i

Saída E_i

$$E_{ij} = F_{ij} - F_{i;1+j} \bmod 256, \quad j = 1,3,5,7$$

$$E_{ij} = 2F_{ij} - F_{i;j-1} \bmod 256, \quad j = 2,4,6,8$$

Demonstração

(4) No algoritmo original temos

$$F_{ij} = 2E_{ij} + E_{i;j+1} \bmod 256, \quad j = 1,3,5,7$$

$$F_{ij} = E_{i;j-1} + E_{ij} \bmod 256, \quad j = 2,4,6,8$$

Temos:

$$\begin{cases} 2E_{ij} + E_{i;j+1} = F_{ij} \\ E_{ij} + E_{i;j+1} = F_{i;j+1} \end{cases} \quad ; \underline{j = 1,3,5,7}$$

$$E_{ij} = F_{ij} - F_{i;j+1}$$

$$E_{i;j+1} = 2F_{i;j+1} - F_{ij}$$

Digitalizado com

Questão 2.

Item 1.

Usando K NONCE, a segurança do algoritmo aumenta. O objetivo desse K é proteger contra ataques de texto legível e conhecido, pois, mesmo que fosse secreto, seria possível quebrar a cifra do algoritmo com esses ataques.

Item 2.

No algoritmo de assinatura C e D estão no intervalo mostrado ($0 < C, D < 0$). Dessa forma não é possível que, após a operação de módulo, C e D sejam legítimos fora do intervalo

Item 4.

Mostrar $g^{xD^{-1}} \cdot (g^S \bmod p)^{CD^{-1}} \bmod p \bmod q = g^K \bmod p \bmod q$

$\Rightarrow$ Assinatura verdadeira sobre x e x não foi alterado.

$$g^{xD^{-1}} \cdot g^{SCD^{-1}} \bmod p \bmod q = g^K \bmod p \bmod q$$

$$\therefore K = xD^{-1} + SCD^{-1}$$

$$DK = x + SC \Leftrightarrow D = (x + SC)K^{-1}$$

~~$(x+SC)K^{-1} \cdot K = x + SC$~~

$\Downarrow$

Assinatura é verdadeira e x não foi alterado

Digitalizado com