1. SAFER K-64 (Secure And Fast Encryption Routine)

RT março de 1998

Este algoritmo com chave de 64 bits foi desenvolvido para a empresa Cylink Corp. por J. Massey (ETH, Suiça).Os blocos de entrada e saída também são de 64 bits.

Todas as operações, como soma, subtração, logaritmo, são sobre operandos em bytes de 8 bits, resultando um byte, como veremos mais tarde. Por exemplo, sendo x e y valores inteiros positivos armazenados cada um em um byte, $x + y$ será um valor de 8 bits.

Uma mesma função é aplicada um certo número de rounds $r = 1, 2, ..., R$, e em cada round são utilizadas duas sub-chaves K_{2r-1} e K_{2r}, sendo cada uma de 64 bits e geradas da chave principal K. J. Massey recomenda pelo menos $R = 6$ rounds.

Um bloco B de entrada, de 64 bits, é dividido em 8 sub-blocos $B_1, B_2, B_3, ..., B_8$. E então, R rounds são aplicados a estes sub-blocos, e depois uma transformação final T é aplicada, obtendo se a saída final.

1.1. Descrição de um round

Cada round utiliza duas sub-chaves K_{2r-1} e K_{2r}, sendo cada uma de 64 bits e geradas da chave principal K como veremos mais tarde.

1.1.1. Primeiro passo

Conforme a figura em anexo, primeiramente cada sub-bloco é submetido a XOR (ou-exclusivo) ou somado a bytes da sub-chave K_{2r-1} da seguinte forma (sendo $K^1_{2r-1}, K^2_{2r-1}, K^3_{2r-1}, K^4_{2r-1}, K^5_{2r-1}, K^6_{2r-1}, K^7_{2r-1}, K^8_{2r-1}$ os 8 bytes de K_{2r-1},):

$B_1(XOR)K^1_{2r-1} = C_1, B_2 + K^2_{2r-1} = C_2, B_3 + K^3_{2r-1} = C_3, B_4(XOR)K^4_{2r-1} = C_4$,

$B_5(XOR)K^5_{2r-1} = C_5, B_6 + K^6_{2r-1} = C_6, B_7 + K^7_{2r-1} = C_7, B_8(XOR)K^8_{2r-1} = C_8$

1.1.2. Segundo passo

Conforme a figura em anexo, os 8 bytes C_j são submetidos a dois tipos de operações:

$$y = 45^x \bmod 257 \ (y = 0 \text{ se } x = 128, \text{ pois } 45^{128} \bmod 257 = 256)$$

Observe que 257 é primo e 45 é elemento primitivo do corpo $GF(257)$, i.e., $45^x \bmod 257$ para $x = 0, 1, 2, ..., 256$ gera todos os elementos de $GF(257)$. A segunda operação é a inversa da anterior, i.e., $\log_{45}(45^x \bmod 257) = x$.

$x = \log_{45} y$ ($x = 128$ se $y = 0$, para ser consistente com a operação anterior)

Recomendamos que esta duas operações sejam previamente calculadas e tabeladas na forma $exp[x] = y$ e $log[y] = x$ onde $exp[]$ e $log[]$ são vetores de 256 posições, para $x, y = 0, 1, 2, ..., 255$. Desta forma, economiza-se tempo pois consulta a estes vetores é mais rápida que calcular toda vez que necessitar um valor. Note que uma vez calculado o valor de $exp[i]$, podemos definir $log[exp[i]] = i$.

1.1.3. Terceiro passo

Os blocos de um byte obtidos no Passo dois são agora submetidos a operações de soma e XOR com os bytes da chave K_{2r} conforme a figura em anexo.

1.1.4. Quarto passo

Finalmente, os 8 bytes são sumetidos à operação HT2 de 2 entradas a_1, a_2 e duas saídas b_1, b_2definida por:

$$\begin{aligned} b_1 &= (2a_1 + a_2) \bmod 256 \\ b_2 &= (a_1 + a_2) \bmod 256 \end{aligned}$$

conforme a figura em anexo.

1.2. Descrição da transformação final T

Depois de R rounds, os 8 blocos são submetidos a uma transformação final T que é exatamente igual ao Primeiro Passo descrito acima, só que a sub-chave utilizada deve ser a última: K_{2R+1}. E o resultado é a saída final do algoritmo SAFER.

1.3. Descrição da geração das sub-chaves

A primeira sub-chave, K_1, é a própria chave principal K de 64 bits. Para $r = 1, 2, 3, ..., 2R$ as sub-chaves seguintes são geradas a partir de K_1 pela fórmula:

$$K_{r+1} = (K_1 <<< 3r) + c_{r+1}$$

onde $v <<< t$ significa deslocamento circular (dentro de cada byte) de v para a esquerda de t posições de bits, e os c_{r+1} são constantes calculadas como explicado a seguir. Denotando os 8 bytes de cada c_i por $c_{i1}, c_{i2}, c_{i3}, c_{i4}, c_{i5}, c_{i6}, c_{i7}, c_{i8}$, eles devem satisfazer, para $i = 2, 3, 4, ..., 2R + 1$:

$$\text{para } j = 1, 2, 3, ..., 8 : c_{ij} = 45^{[45\hat{}(9i+j) \bmod 257]} \bmod 257$$

onde $v\hat{}t$ denota v elevado a t.

2. Descrição do algoritmo inverso do SAFER

Primeiramente os 8 bytes são submetidos à transformação inversa de T que foi descrita anteriormente, isto é, com subtrações no lugar de somas.

Depois tem-se R rounds inversos em que a operação $IHT2$, inversa da $HT2$, é definida por:

$$\begin{aligned} a_1 &= (b_1 - b_2) \bmod 256 \\ a_2 &= (-b_1 + 2b_2) \bmod 256 \end{aligned}$$

3. Dados para verificação

$45^{127} \bmod 257 = 217$
$45^{128} \bmod 257 = 256$
$45^{129} \bmod 257 = 212$

4. Modo CBC – Cipher Block Chaining

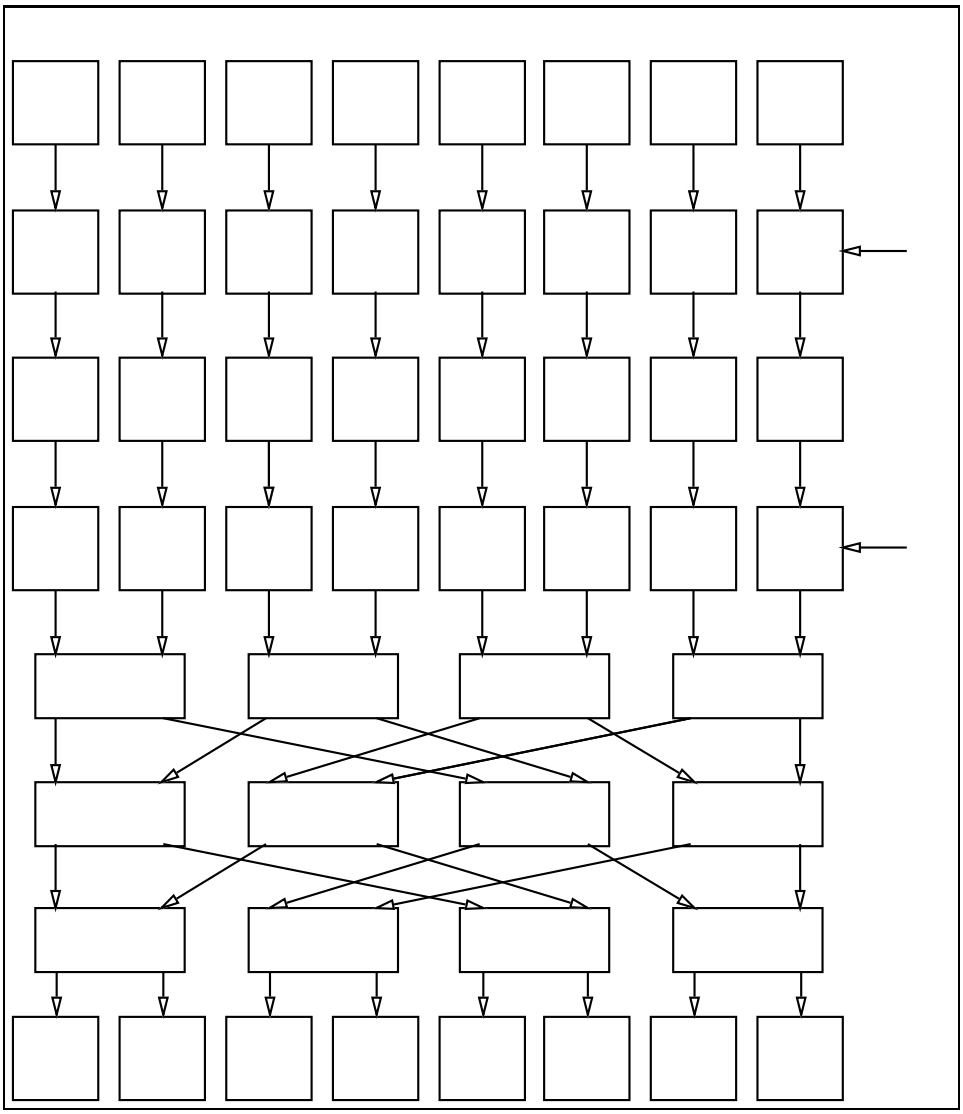

Figure 3.1: Figura 1 - SAFER K-64

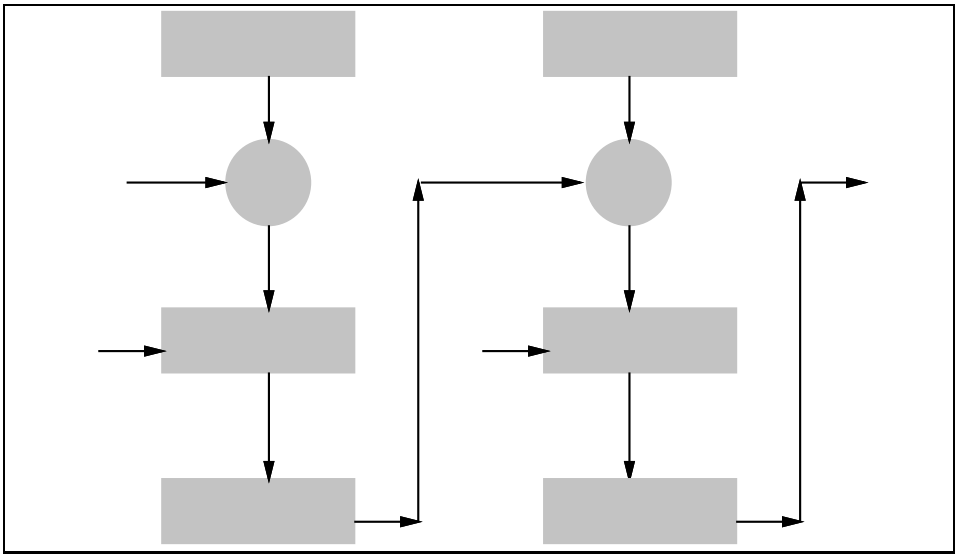

Figure 4.1: Figura 2 - Modo CBC