Docs do sucesso

Questão 3.1

- Dado $y = SPN(x, \pi_S, \pi_P, (K^1, ..., K^{Nr+1}))$, ache $\pi_{S'}, \pi_{P'}$ tal que $x = SPN(x, \pi_{S'}, \pi_{P'}, (L^{Nr+1}, ..., L^1))$, onde L^i é uma permutação de K^i

Temos que $y = Enc_K(x)$. O processo inverso $Dec_K(y)$ é simplesmente a inversão das funções π_S e π_P e as *round keys* $L^(Nr - i) = K^i$. $\diamond$

Questão 3.2

- Prove que a decriptação em uma cifra de Feistel pode ser feita aplicando o algorítimo à cifra com o *key schedule* invertido.

Uma rede de Feistel é um caso especial de uma cifra iterada onde $f : M \times K \to C$ possui a forma $g(L^{i-1}, R^{i-1}, K^i)$, onde

$L^i = R^{i-1}, R^i = L^{i-1} \oplus f(R^{i-1}, K^i)$

Nesse processo, o estado w^i se quebra em duas metades, L^i, R^i, do mesmo tamanho.

Seja w^{i-1} e w^i estados do processo de decriptação. Temos que:

$w^i = L^i||R^i = R^{i-1}||L^{i-1} \oplus f(R^{i-1}, K^i)$

Queremos encontrar wi^{i-1}:

$R^{i-1} = L^i$

Aplicamos $\oplus f(R^{i-1}, K^i)$ em ambos os lados de $R^i = L^{i-1} \oplus f(R^{i-1}, K^i)$:

$L^{i-1} = R^i \oplus f(R^{i-1}, K^i) = R^i \oplus f(L^i, K^i)$

Como podemos observar pelas definições de R^{i-1} e L^{i-1}, basta aplicarmos a rede de Feistel sobre a cifra, usando o *key schedule* em ordem reversa.

$\diamond$

Questão 3.3

- Seja $DES(x, K)$ a encritapção do text x com a chave K usando o criptosistema DES. Suponha que $y = DES(x, K)$ e $y' = DES(c(x), c(K))$, onde $c(\dot{})$ denota o complemento bitwise de seu argumento. Prove que $y' = c(y)$, isto é, que se complementarmos o texto puro e a chave, a cifra também será complementada. Note que isso pode ser provado usando somente a descrição *high level* do DES, ou seja, a estrutura das *S-boxes* e outros componentes são irrelevantes.

Temos que o DES é basicamente uma rede Feistel com 16 *rounds*. Portanto, a cada *round* o estado w^i de 64 *bits* é quebrado em dois blocos, L^i, R^i, de 32 bits.

Seja x a entrada do algorítimo, ou seja, $w^0 = x = L^0||R^0$. Se complementarmos x, cada uma de suas metadas está sendo igualmente complementada, ou seja $c(x) = c(L^0||R^0) = c(L^0)||c(R^0)$. Isso é verdade devido ao fato de que o complemente de um binário é obtido simplesmente trocando os 0s por 1s e vice-versa.

Seja agora K a chave do algorítimo que possui 56 *bits*. Cada chave K^i do *key schedule* é gerada a partir de rotações nos bits da chave K, que nada mais são que *shifts* binários. Portanto, ao usarmos $c(K)$ em vez de K, cada *round key* K'^i pode ser escrito como $K'^i = (c(K^i)$.

Por fim, seja a $R'^1 = L'^0 \oplus f(c(R^0), c(K^0))$. Nós já sabemos que $L'^0 = c(L^0)$. No entanto, como o primeiro passo da função f involve um ou-exclusivo entre as entradas, a propriedade de complemento é eliminada. Logo, $f(c(R^0), c(K^0)) = f(R^0, K^0)$. Pelo mesmo motivo, no entanto, temos que $c(L^0) \oplus f(R^0, K^0) = c(L^0 \oplus f(R^0, K^0))$. Logo, $R'^1 = c(L^0 \oplus f(R^0, K^0)) = c(R^1)$. Portanto, $c(DES(x, K)) = DES(c(x), c(K))$. $\diamond$

Questão 3.7

- Suponha que uma sequência de texto puro $x_1, ..., x_n$ dê uma sequência de cifra $y_1, ..., y_n$. Suponha agora que um bloco de cifra, y_i, é transmitido incorrentamente, ou seja, algum 1 foi trocado por um 0 ou vice-versa. Mostre que o número de blocos de texto puro que serão decriptados incorretamente é igual a um se os modos ECB ou OFB forem usados na encriptação, e igual a 2 se os modos CBC ou CFB forem usados.

Primeira parte - ECB e OCB

O modo ECB funciona de acordo com a função $y_i = Enc_K(x_i)$, ou seja, é a encriptação direta de x usando a chave K. Neste caso, seja x_{i_j} o bit j pertencente ao bloco i da entrada. Como cada bloco é independente do outro, somente o bloco i que possui o bit trocado será afetado na desencriptação. Do mesmo modo, no modo OFB o bloco i não é propagado para outros blocos, dado que ele é *xor'd* com o *keystream* z_i somente após a propagação de z_i para a próxima iteração.

Segunda parte - CBC e CFB

O modo CFB é caracterizado por $y_i = e_K(y_{i-1} \oplus x_i)$. Assumindo que o erro ocorre no bloco x_i, temos que o bloco em que o erro ocorreu será decifrado incorretamente, assim como o bloco adjacente (a cifra do bloco i corrompido é

utilizada tanto para decifrar o bloco i, quanto o bloco $i+1$, onde é utilizada como `IV`) – porém, o erro não é mais propagado após decifragem de ambos os blocos.

No modo CBC, caso o erro tenha ocorrido no bloco i a cifra corrompida será utilizada para decifrar apenas o próprio bloco i e o bloco adjacente $i+1$. O erro não é propagado para outros blocos, já que o bloco $i+2$ recebe apenas a cifra do bloco $i+1$ como entrada. $\diamond$

Questão 4.6

- Suponha que $f : (0,1)^m \to (0,1)^m$ é uma bijeção resistente à primeira pré-imagem. Defina $h : (0,1)^{2m} \to (0,1)^m$ como segue. Dado que $x \in (0,1)^{2m}$:

  $$x = x'||x''$$

  onde $x', x'' \in (0,1)^m$. Então:

  $$h(x) = f(x' \oplus x'')$$

  Prove que h não é resistente a segunda pré-imagem.

Temos, pela definição de resistência à primeira pré-imagem, que dado $y \in Y$, deve ser difícil de achar $x \in X$ tal que $y = h(x)$. A definição de segunda pré-imagem segue de que para dado $x \in X$, deve ser difícil de achar $x' \in X$ tal que $x' \neq x$ e $h(x') = h(x)$.

Da definição de h, temos:

$$h = f(x) = f(x' \oplus x'')$$

Sejam $x_1, x_2 \in X$ tais que $x_1 = x'||x''$ e $x_2 = x''||x'$. Temos que:

$$h(x_1) = f(x_1) = f(x' \oplus x'')h(x_2) = f(x_2) = f(x'' \oplus x')$$

Mas, $f(x' \oplus x'') = f(x'' \oplus x')$. Portanto, h não é resistente à segunda imagem. $\diamond$

Questão 4.9

- Suponha $h_1 : (0,1)^{2m} \to (0,1)^m$ é uma função hash resistente à colisão.
  - Defina $h_2 : (0,1)^{4m} \to (0,1)^m$ como segue:
    - Escreva $x \in (0,1)^{4m}$ como $x = x_1||x_2$, onde $x_1, x_2 \in (0,1)^{2m}$
    - Defina $h_2(x) = h_1(h_1(x_1)||h_1(x_2))$
      Prove que h_2 é resistente à colisão.

– Para um inteiro $i \geq 2$, defina a função hash $h_i : (0,1)^{2^i m} \to (0,1)^m$ recursivamente a partir de h_{i-1} como segue:
  * Escreva $x \in (0,1)^{2^i m}$ como $x = x_1||x_2$, onde $x_1, x_2 \in (0,1)^{2^{i-1} m}$
  * Defina $h_i(x) = h_{i-1}(h_{i-1}(x_1)||h_{i-1}(x_2))$
    Prove que h_i é resistente a colisão.

Primeira parte

Devemos provar que h_2 é resistente colisão, ou seja, que não existem $x', x'' \in (0,1)^{2m}$ tais que $h_2(x') = h_2(x'')$. Assuma que $h_2(x') = h_2(x'')$ e que $x' = x_1||x_2$ e $x'' = x_2||x_1$. Portanto, temos que $h_2(x') = h_2(h_1(x_1)||h_1(x_2))$ e $h_2(x'') = h_2(h_1(x_2)||h_1(x_1))$. No entanto, sabemos que h_1 é resistente a colisão. Portanto, $h_2(h_1(x_1)||h_1(x_2)) = h_2(h_1(x_2)||h_1(x_1))$ é uma contradição, pois deve ser difícil de achar a, b tais que $h_1(a) = h_1(b)$.

Segunda parte

Devemos provar uma recorrência agora. Para isso, usamos o resultado do item anterior e aplicamos indução em i.

Caso base: $h_2(x) = h_1(h_1(x_1)||h_1(x_2))$ é resistente a colisão, dado que h_1 seja igualmente resistente a colisão. **Hipótese indutiva**: $h_{i-1}(x) = h_{i-2}(h_1(x_1)||h_{i-2}(x_2))$ é resistente a colisão. **Passo indutivo**: Devemos provar que $h_i(x)$ é resistente a colisão. Da definição de $h_i(x)$, temos que $h_i(x) = h_i(h_{i-1}(x_1)||h_{i-1}(x_2))$. Assuma que h_i não é resistente a colisão. Isso implica que $h_i(h_{i-1}(x_1)||h_{i-1}(x_2))$ não é resistente a colisão, o que significa que é fácil achar colisões em $h_{i-1}(x)$. No entanto, sabemos pela hipótese indutiva que h_{i-1} é resistente a colisão. Portanto achamos uma contradição e h_i deve ser resistente a colisão.