Trabalho 1/3 - IPC – Mario Gazziro – Turma: Fisica Computacional

Data de Entrega: dia 20/03 até meio-dia (individual)

Implementar uma função em linguagem C para cálculo da cifra SALSA20

Salsa20 é uma cifra de fluxo que foi submetida ao eSTREAM por Daniel Bernstein. Ela é construída através de uma função pseudorrandômica baseada na adição de 32-bits, adição bit a bit (XOR) e operações de rotação, que mapeia uma chave de 256 bits, um nonce de 64 bits e uma posição de fluxo de 64 bits em uma saída de 512 bits (existe ainda uma versão de 128 bits). Esse funcionamento dá ao Salsa20 uma vantagem incomum que permite ao usuário realizar buscas de maneira eficiente em qualquer parte do fluxo de saída. Desta forma, ela fornece uma velocidade de cerca de 4-14 ciclos por byte em software em processadores de arquitetura x86 bem como uma razoável performance de hardware. O Salsa20 por sua vez não é patenteado e Bernstein tem escrito diversas implementações de domínio público para as arquiteturas mais comuns.

Estrutura

Internamente, o algoritmo de cifragem utiliza a adição bit a bit (XOR), adição 32-bits mod 2^{32} e uma distância constante de operações de rotação em um estado interno de palavras de 32-bits. A escolha destas operações evita a possibilidade de ataques de tempo em implementações de software.

O estado inicial é formado por 8 palavras-chave, sendo estas: 2 palavras de posição de fluxo, 2 palavras de nonce e 4 palavras constantes. Como resultado de 20 rodadas de mistura, são produzidas 16 palavras da saída da cifra de fluxo.

Cada rodada de mistura consiste de 4 operações de quarto-de-rodada, realizadas nas colunas ou nas linhas do estado de 16 palavras, dispostas em uma matriz 4x4. A cada duas rodadas o padrão é repetido. O pseudocódigo para as rodadas é o seguinte: ⊞ é a adição modulo 2^{32}, <<<< é a operação de rotação à esquerda, ^ é o XOR e **x** ^= **y** é abreviação para **x** = **x** ^ **y**.

```
x[ 4] ^= (x[ 0] ⊞ x[12])<<<7;    x[ 9] ^= (x[ 5] ⊞ x[ 1])<<<7;
x[14] ^= (x[10] ⊞ x[ 6])<<<7;    x[ 3] ^= (x[15] ⊞ x[11])<<<7;
x[ 8] ^= (x[ 4] ⊞ x[ 0])<<<9;    x[13] ^= (x[ 9] ⊞ x[ 5])<<<9;
x[ 2] ^= (x[14] ⊞ x[10])<<<9;    x[ 7] ^= (x[ 3] ⊞ x[15])<<<9;
x[12] ^= (x[ 8] ⊞ x[ 4])<<<13;   x[ 1] ^= (x[13] ⊞ x[ 9])<<<13;
x[ 6] ^= (x[ 2] ⊞ x[14])<<<13;   x[11] ^= (x[ 7] ⊞ x[ 3])<<<13;
x[ 0] ^= (x[12] ⊞ x[ 8])<<<18;   x[ 5] ^= (x[ 1] ⊞ x[13])<<<18;
x[10] ^= (x[ 6] ⊞ x[ 2])<<<18;   x[15] ^= (x[11] ⊞ x[ 7])<<<18;

x[ 1] ^= (x[ 0] ⊞ x[ 3])<<<7;    x[ 6] ^= (x[ 5] ⊞ x[ 3])<<<7;
x[11] ^= (x[10] ⊞ x[ 9])<<<7;    x[12] ^= (x[15] ⊞ x[14])<<<7;
x[ 2] ^= (x[ 1] ⊞ x[ 0])<<<9;    x[ 7] ^= (x[ 6] ⊞ x[ 5])<<<9;
x[ 8] ^= (x[11] ⊞ x[10])<<<9;    x[13] ^= (x[12] ⊞ x[15])<<<9;
x[ 3] ^= (x[ 2] ⊞ x[ 1])<<<13;   x[ 4] ^= (x[ 7] ⊞ x[ 6])<<<13;
x[ 9] ^= (x[ 8] ⊞ x[11])<<<13;   x[14] ^= (x[13] ⊞ x[12])<<<13;
x[ 0] ^= (x[ 3] ⊞ x[ 2])<<<18;   x[ 5] ^= (x[ 4] ⊞ x[ 7])<<<18;
x[10] ^= (x[ 9] ⊞ x[ 8])<<<18;   x[15] ^= (x[14] ⊞ x[13])<<<18;
```