1. RC5

Routo Terada

Este algoritmo é de Ron Rivest (The RC5 encryption algorithm, Fast Software Encryption, 2nd. International Workshop, Lec. Notes in Comp. Sci. 1008, pp 86-96, Springer-Verlag, 1995).

RC5 foi projetado para qualquer computador de 16 ou 32 ou 64 bits. Possui uma descrição compacta e é adequado para implementações em software ou hardware. Como o DES, RC5 possui várias iterações e as várias subchaves são utilizadas em uma função de iteração. Ao contrário do DES, o número de iterações e o número de bytes na chave são variáveis. Baseia-se na operação de rotação (i.e., deslocamento circular) de um número variável de posições, e esse número depende de quase todos os bits resultantes da iteração anterior e do valor da subchave em cada iteração.

Após alguns anos de análise dos especialistas sabe-se que após 8 iterações todo bit da entrada legível afeta pelo menos uma rotacão. Há uma criptanálise diferencial para o RC5 com 64 bits de entrada que necessita 2^{24} textos legíveis escolhidos para 5 iterações, e 2^{68} para 15 iterações. E contra a criptanálise linear o RC5 é considerado seguro após 6 iterações.

1.1. Parâmetros do RC5

1. $w = 16, 32$, ou 64 bits, é o número de bits em cada variável (i.e, cada posição de memória);
2. r é o número de iterações (rounds);
3. b é o número de bytes na chave.

O algoritmo genérico é representado por RC5-w/r/b. De fato é uma família de algoritmos. No RC5-32 o autor recomenda 12 iterações, e para o RC5-64, 16 iterações.

1.2. Operações básicas do RC5

As operações básicas usadas são todas sobre operandos de w bits, e são:

1. $v \boxplus u$ é a soma dos inteiros v, u de w bits, resultando um valor de w bits (i.e., soma $\bmod 2^w$);

2. $v \boxminus u$ é a subtração dos inteiros v, u de w bits, resultando um valor de w bits (i.e., subtração mod 2^w);

3. $v \oplus u$ é o ou-exclusivo (XOR) de v, u de w bits, resultando um valor de w bits;

4. $v \ll t$ é o deslocamento circular (i.e. rotação) de t posições para a esquerda dos bits em v. Se t exceder w bits, pode-se considerar os w bits menos significativos sem alterar o resultado.

5. $v \gg t$ é o deslocamento circular (i.e. rotação) de t posições para a direita dos bits em v. Se t exceder w bits, pode-se considerar os w bits menos significativos sem alterar o resultado.

A entrada é de $2w$ bits, e a saída também. As duas metades da entrada A e B são alteradas a cada iteração. São usadas 2 subchaves antes da primeira iteração, e 2 subchaves em cada iteração, tendo-se no total $2r + 2$ subchaves. A chave K é de b bytes, da qual são geradas as subchaves K_j de w bits cada, como descrito a seguir.

1.3. Algoritmo de geração de subchaves RC5

O algoritmo expande os b bytes da chave K para $2r + 2$ subchaves de w bits.

Inicialmente este algoritmo descreve como a chave K é armazenada nas variáveis $L[0], L[1], \ldots$.

Algoritmo de subchaves RC5
Entrada: r (número de iterações), chave K de b bytes armazenados em $K[0]$, $K[1]$, ... $K[b-1]$;
Saída: $2r + 2$ subchaves de w bits $K_0, K_1, K_3, \ldots K_{2r+1}$

1. Seja $u = w/8$ (i.e. número de bytes em cada variável de w bits; e.g. $u = 32/4 = 4$ para $w = 32$ bits);

2. Seja $c = \lceil b/u \rceil$ (i.e. número de posições de w bits ocupados por K, e.g. $c = \lceil 10/4 \rceil = 3$, para $b = 10$ bytes);

3. Se necessário, preencher K à direita de $K[b-1]$ até completar um comprimento total em bytes divisível por u, i.e.

$$\text{para } j = b, b+1, \ldots c \times u - 1 \text{ faça: } K[j] \leftarrow 0;$$

4. **para** $j = 0, 1, ...c - 1$ **faça**:

$$L_j \leftarrow \sum_{t=0}^{u-1} 2^{8t} K[j \times u + t]$$

   i.e. preencher os u bytes, da direita para a esquerda, de cada L_j com os bytes $K[0], K[1], K[2], ...K[c \times u - 1]$;

5. $K_0 \leftarrow P_w$; (veja o valor de P_w na tabela a seguir)

6. **para** $j = 1, 2, ...2r+1$ **faça**: $K_j \leftarrow K_{j-1} \boxplus Q_w$; (veja o valor de Q_w na tabela a seguir)

7. $i \leftarrow 0; j \leftarrow 0; A \leftarrow 0; B \leftarrow 0; t \leftarrow \max\{c, 2r + 2\}$;

8. **para** $s = 1, 2, 3, ...3t$ **faça** {

   1. $K_i \leftarrow (K_i \boxplus A \boxplus B) \lll 3; A \leftarrow K_i; i \leftarrow i + 1 \bmod (2r + 2)$;
   2. $L_j \leftarrow (L_j \boxplus A \boxplus B) \lll (A \boxplus B); B \leftarrow L_j; j \leftarrow j + 1 \bmod c$;
   3. }

9. A saída é $K_0, K_1, K_2, ...K_{2r+1}$

As constantes P_w e Q_w são representações binárias das constantes matemáticas e e Φ. Foram escolhidas pelo autor por serem constantes que não foram definidas por ele e portanto não levanta qualquer suspeita de incorporar algum conhecimento prévio que facilite a quebra do algoritmo.

w	16	32	64
P_w	$b7e1$	$b7e15163$	$b7e15162 \quad 8aed2a6b$
Q_w	$9e37$	$9e3779b9$	$9e3779b9 \quad 7f4a7c15$
Valores em hexadecimal			

1.4. Algoritmo de criptografia RC5

Tanto na criptografia como na decriptografia este algoritmo supõe que o computador armazena os bytes em modo little-endian, ou seja tanto o texto legível como o ilegível são armazenados da seguinte forma nas variáveis A e B, para $w = 32$: os primeiros quatro bytes do texto (i)legível (x_0, x_1, x_2, x_3) são armazenados em A na seqüência (x_3, x_2, x_1, x_0), e os quatro bytes seguintes do texto (i)legível (x_4, x_5, x_6, x_7) são armazenados em B na seqüência (x_7, x_6, x_5, x_4).

Algoritmo RC5
Entrada: chave K de b bytes, texto legível de $2w$ bits (A, B);
Saída: ilegível de $2w$ bits (A, B)

1. Calcular $2r + 2$ subchaves $K_0, K_1, K_3, ...K_{2r+1}$
2. $A \leftarrow A \boxplus K_0; B \leftarrow B \boxplus K_1;$
3. **para** $j = 1, 2, 3, ...r$ **faça**:
$$A \leftarrow ((A \oplus B) \ll B) \boxplus K_{2j}; B \leftarrow ((B \oplus A) \ll A) \boxplus K_{2j+1}$$
4. A saída é o valor (A, B).

1.5. Algoritmo de decriptografia RC5

Algoritmo de decriptografia RC5
Entrada: chave K de b bytes, texto ilegível de $2w$ bits (A, B);
Saída: ilegível de $2w$ bits (A, B)

1. Calcular $2r + 2$ subchaves $K_0, K_1, K_3, ...K_{2r+1}$
2. **para** $j = r, r - 1, ...1$ **faça**:
$$B \leftarrow ((B \boxminus K_{2j+1}) \gg A) \oplus A, A \leftarrow ((A \boxminus K_{2j}) \gg B) \oplus B$$
3. A saída é o valor $(A \boxminus K_0, B \boxminus K_1)$.

1.6. Dados para testes do RC5

Para o RC5-32/12/16 tem-se os seguintes dados:
Texto legível: b278 c165 cc97 d184.
Chave: 5269 f149 d41b a015 2497 574d 7f15 3125.
Texto ilegível: 15e4 44eb 2498 31da

2. RC6

Este algoritmo é um dos cinco candidatos finalistas a AES. É de autoria de R. Rivest, M.J.B. Robshaw, R. Sidney, e Y.L. Yin.

RC6 é naturalmente inspirado no RC5, mas o objetivo segundo os autores foram de torná-lo mais seguro contra criptanálise e mais veloz que o RC5. Como o RC5, RC6 foi projetado para qualquer computador de 16 ou 32 ou 64 bits. Possui também uma descrição compacta e é adequado para implementações em software ou hardware. Como o DES e o RC5, RC6 possui várias iterações e as várias subchaves são utilizadas em uma função de iteração. Como no RC5 o número de iterações e o número de bytes na chave são variáveis. Como no RC5, baseia-se também na operação de rotação, mas a operação de multiplicação foi introduzida.

2.1. Parâmetros do RC6

Como no RC5, w é o número de bits do computador; utilizamos $w = 32$. r é o número de iterações (rounds); é recomendado um mínimo de 20 iterações, e usamos $r = 20$. b é o número de bytes da chave, $0 < b < 256$. Genericamente o algoritmo recebe o nome RC6-$w/r/b$. Nesta seção tem-se então a descrição do RC6-$32/20/b$ Os autores afirmam que implementações em linguagem C em computadores de 32 bits a 200Mhz atingem velocidades da ordem de cinco Mbytes por segundo.

2.2. Operações básicas do RC6

As operações básicas usadas são todas sobre operandos de w bits, e são descritas a seguir.

O logaritmo na base 2 de w é representado por $\lg w$.

As operações básicas são:

1. $a \boxplus b$ soma de inteiros módulo 2^w;
2. $a \boxminus b$ subtração de inteiros módulo 2^w;
3. $a \oplus b$ ou-exclusivo bit a bit sobre operandos de w bits;
4. $a \boxtimes b$ multiplicação de inteiros módulo 2^w;
5. $a \lll b$ deslocamento circular (rotação) de a para a esquerda de ℓ posições, onde ℓ é o inteiro igual aos $\lg w$ bits menos significativos de b;

6. $a \ggg b$ deslocamento circular (rotação) de a para a direita de ℓ posições, onde ℓ é o inteiro igual aos $\lg w$ bits menos significativos de b;

2.3. Algoritmo de criptografia RC6

Tanto na criptografia como na decriptografia este algoritmo supõe que o computador armazena os bytes em modo little-endian, ou seja tanto o texto legível como o ilegível são armazenados da seguinte forma nas variáveis A, B, C e D, para $w = 32$: os primeiros quatro bytes do texto (i)legível (x_0, x_1, x_2, x_3) são armazenados em A na seqüência (x_3, x_2, x_1, x_0), e os quatro bytes seguintes do texto (i)legível (x_4, x_5, x_6, x_7) são armazenados em B na seqüência (x_7, x_6, x_5, x_4), etc..

No que segue $(A, B, C, D) \leftarrow (B, C, D, A)$ significa atribuir em paralelo os valores à direita para as variáveis correspondentes à esquerda.

Algoritmo RC6
Entrada: Texto legível armazenado em quatro variáveis de w-bits: A, B, C, D; número r de iterações; $2r + 4$ subchaves armazenadas em variáveis de w-bts: $S[0], ...S[2r + 3]$;
Saída: Texto ilegível armazenado em A, B, C, D;

1. $B \leftarrow B \boxplus S[0]$;
2. $D \leftarrow D \boxplus S[1]$;
3. **para** $i = 1$ **até** r **faça** {
    1. $t \leftarrow (B \boxtimes (2B \boxplus 1)) \lll \lg w$;
    2. $u \leftarrow (D \boxtimes (2D \boxplus 1)) \lll \lg w$;
    3. $A \leftarrow ((A \oplus t) \lll u) \boxplus S[2i]$;
    4. $C \leftarrow ((C \oplus u) \lll t) \boxplus S[2i + 1]$;
    5. $(A, B, C, D) \leftarrow (B, C, D, A)$;
    6. }
4. $A \leftarrow A \boxplus S[2r + 2]$;
5. $C \leftarrow C \boxplus S[2r + 3]$;

2.4. Decriptografia RC6

Algoritmo de decriptografia RC6
Entrada: Texto ilegível armazenado em quatro variáveis de w-bits: A, B, C, D; número r de iterações; $2r + 4$ subchaves armazenadas em variáveis de w-bts: $S[0], ...S[2r + 3]$;
Saída: Texto legível armazenado em A, B, C, D;

1. $C \leftarrow C \boxminus S[2r + 3]$;
2. $A \leftarrow A \boxminus S[2r + 2]$;
3. **para** $i = r$ **até** 1 **faça** {
    1. $(A, B, C, D) \leftarrow (D, A, B, C)$;
    2. $u \leftarrow (D \boxtimes (2D \boxplus 1)) \lll \lg w$;
    3. $t \leftarrow (B \boxtimes (2B \boxplus 1)) \lll \lg w$;
    4. $C \leftarrow ((C - S[2i + 1]) \ggg t) \oplus u$;
    5. $A \leftarrow ((A - S[2i]) \ggg u) \oplus t$;
    6. }
4. $D \leftarrow D \boxminus S[1]$;
5. $B \leftarrow B \boxminus S[0]$;

2.5. Geração de subchaves RC6

Este algoritmo é essencialmente o mesmo do Algoritmo RC5. O algoritmo expande os b bytes da chave K para $2r + 4$ subchaves de w bits: $S[0], S[1], ...$

Inicialmente este algoritmo descreve como a chave K é armazenada nas variáveis $L[0], L[1], ...$.

Algoritmo de geração de subchaves RC6
Entrada: Número de iterações r, número de bits do computador w, número de bytes na chave b, chave de b bytes $K[0], K[1], ...K[b]$;
Saída: $2r + 4$ subchaves de w-bits armazenados em $S[0], ...S[2r + 3]$;

1. Seja $u = w/8$ (i.e. número de bytes em cada variável de w bits; e.g. $u = 32/4 = 4$ para $w = 32$ bits);

2. Seja $c = \lceil b/u \rceil$ (i.e. número de posições de w bits ocupados por K, e.g. $c = \lceil 10/4 \rceil = 3$, para $b = 10$ bytes);

3. Se necessário, preencher K à direita de $K[b-1]$ até completar um comprimento total em bytes divisível por u, i.e.

$$\text{para } j = b, b+1, ...c \times u - 1 \text{ faça: } K[j] \leftarrow 0;$$

4. **para** $j = 0, 1, ...c - 1$ **faça**:

$$L[j] \leftarrow \sum_{t=0}^{u-1} 2^{8t} K[j \times u + t]$$

   i.e. preencher os u bytes, da direita para a esquerda, de cada $L[j]$ com os bytes $K[0], K[1], K[2], ...K[c \times u - 1]$;

5. $S[0] = P_w$; /* $P_{32} = B7E15163$ na base 16 */

6. **para** $i = 1$ **até** $2r + 3$ **faça** {

   1. $S[i] = S[i-1] \boxplus Q_w$; /* $Q_w = 9E3779B9$ na base 16 */
   2. }

7. $A = B = i = j = 0$;

8. $v = 3 \times \max\{c, 2r + 4\}$;

9. **para** $s = 1$ **até** v **faça** {

   1. $A = S[i] = (S[i] \boxplus A \boxplus B) \lll 3$;
   2. $B = L[j] = (L[j] \boxplus A \boxplus B) \lll (A \boxplus B)$;
   3. $i = (i + 1) \bmod (2r + 4)$;
   4. $j = (j + 1) \bmod c$;
   5. }

2.6. Dados para teste do RC6

Damos a seguir os valores do texto legível, ilegível e da chave para RC32/20/16.

2.6.1. Dados de criptografia RC6

Texto legível: 0000 0000 0000 0000 0000 0000 0000 0000

Chave: 8000 0000 0000 0000 0000 0000 0000 0000

Subchaves:

S[0]=7192b7cf S[1]=2a620299 S[2]= e4401d7 S[3]= 38579d3 S[4]=c9d48f80
S[5]=a74af2ab S[6]=cabaec57 S[7]=739d4ce3 S[8]=351dd1af S[9]=80116b95
S[10]=6b6fbf82 S[11]=31a5f473 S[12]=5d5a45df S[13]= 2eefe39 S[14]=8546986f
S[15]=302f7b46 S[16]=c566bd06 S[17]=1adf0f39 S[18]=575a6bd0 S[19]=68517c87
S[20]=5aeec93a S[21]=cc76a3b4 S[22]=88e15dc6 S[23]=804f98c5 S[24]=55480901
S[25]=56b7004c S[26]= c6c42f7 S[27]=59bf24a6 S[28]=2981de63 S[29]=7fcfa5e4
S[30]=a6ddaaa1 S[31]=ceab987e S[32]=98e92951 S[33]= 78d856e S[34]=39e053f9
S[35]=6b1d57d9 S[36]=f914ae91 S[37]=c6196f07 S[38]=97764d20 S[39]=65e3f5d0
S[40]=ba549018 S[41]=fc1d2fa3 S[42]=86c0de2d S[43]= 565963f

Valores intermediários (após cada iteração) da criptografia:

[0]=000000007192b7cf000000002a620299
[1]=7192b7cff3b4eb692a620299b2957151
[2]=f3b4eb69185abefeb29571519b3563f1
[3]=185abefec4f4db589b3563f12b183172
[4]=c4f4db584298c9f32b183172f97beadc
[5]=4298c9f3a80c04caf97beadc4ef2ec7d
[6]=a80c04ca5fbfe0af4ef2ec7d4c044c71
[7]=5fbfe0af07a484f24c044c71040201b8
[8]=07a484f2f3bc4d7b040201b8854e4f4e
[9]=f3bc4d7b23ab89f4854e4f4e33abea4c
[10]=23ab89f48d6d145633abea4ca6e7c7dc
[11]=8d6d14563b4200ffa6e7c7dc931ed017
[12]=3b4200ffabbad0cf931ed01767fc854b
[13]=abbad0cf63c9139d67fc854b05cd85e3
[14]=63c9139df39ad64305cd85e3381c84fd
[15]=f39ad64357f3aab3381c84fd4bc27cd3
[16]=57f3aab31190d6a04bc27cd3c7c7ebd4
[17]=1190d6a04174a9fdc7c7ebd4185873a4
[18]=4174a9fd446e442a185873a4221b45d7
[19]=446e442aabaf1c81221b45d76e80f30e

[20]=abaf1c812dcb63e96e80f30eacdd03b5

Texto ilegível: 326f faae 2dcb 63e9 73e6 894d acdd 03b5

2.6.2. Dados de decriptografia RC6

Texto ilegível: 3f11 c73c c933 686e 07ee b4bf a588 1437

Chave: 0000 0000 0000 0000 0000 0000 0000 0000

Subchaves:
S[0]=2a66311c S[1]=9b17852d S[2]=8108b207 S[3]=39d14185 S[4]=9c64df5f
S[5]=4bed6bcd S[6]=b1d88726 S[7]=4e6ee8c6 S[8]=66f7fa9c S[9]=429c2724
S[10]=c955b6bf S[11]=9306e49a S[12]=75524dd9 S[13]=56f4da3c S[14]=5ec06b9b
S[15]=abbc779b S[16]=15ee39a1 S[17]=66e7755f S[18]= 25068d S[19]=7cb6d760
S[20]=a6b4a4be S[21]=11b98fb8 S[22]=512b019f S[23]=5a199fce S[24]=bd468f9e
S[25]=1db66d3a S[26]=61b5d390 S[27]=84bfed42 S[28]=8690aee8 S[29]=774b9ee5
S[30]=9fd534b1 S[31]=848939e0 S[32]=62a03756 S[33]=fd977349 S[34]=59abcdca
S[35]=c8af1e91 S[36]=8e6d2107 S[37]=359b2c7d S[38]=63018004 S[39]=c7b27ae3
S[40]=55fcc1b3 S[41]=c5bd8db7 S[42]= 5b4016a S[43]=4b38ac83 I=1

Valores intermediários (após cada iteração) da decriptografia:
[20]=395dc5d2c933686ebcb6083ca5881437
[19]=2fb433e8395dc5d2e01a5e34bcb6083c
[18]=69d5c0b02fb433e8e5749d97e01a5e34
[17]=fb9d027969d5c0b0e4ed89b4e5749d97
[16]=835b43cffb9d027920792eb0e4ed89b4
[15]=fa12db0d835b43cfcd9b26f020792eb0
[14]=99149402fa12db0d62ef65e3cd9b26f0
[13]=915e890e99149402e3e7ed2662ef65e3
[12]=5fb075f7915e890e0d74a8fde3e7ed26
[11]=df4ad7ee5fb075f789d154ef0d74a8fd
[10]=448df3bfdf4ad7ee07134b9b89d154ef
[9]=264ea443448df3bffa64218507134b9b
[8]=6f3cc2ce264ea443403b8782fa642185
[7]=f10003186f3cc2ce5fecce30403b8782
[6]=eda6053df10003186d1ce6155fecce30
[5]=03c36387eda6053dbae8e0b76d1ce615

[4]=93e5e7b303c3638730bec103bae8e0b7
[3]=c0d8145093e5e7b35005b0c130bec103
[2]=35777384c0d81450dee325d55005b0c1
[1]=2a66311c357773849b17852ddee325d5
[0]=800000002a66311c000000009b17852d

Texto legível: 8000 0000 0000 0000 0000 0000 0000 0000