Universidade de São Paulo
Instituto de Matemática e Estatística
Departamento de Ciência da Computação

Aplicação do ataque χ^2 sobre o cifrador RC6

Aluno: Eduardo Takeo Uêda
Orientador: Prof. Dr. Routo Terada
Área de concentração: Criptografia

São Paulo – Novembro de 2003

Eduardo Takeo Uêda
Mestrando em Ciência da Computação
edutakeo@ime.usp.br

Routo Terada
Professor do DCC/IME/USP
rt@ime.usp.br

Relatório de estudos apresentado como requisito à aprovação na disciplina Tópicos em Ciência da Computação, do Programa de Pós-Graduação em Ciência da Computação.

Professor responsável pela disciplina:

Yoshiharu Kohayakawa

Sumário

1 Introdução

O RC6 é um algoritmo de criptografia simétrica, sendo uma evolução do RC5, seu antecessor. Ele foi submetido como candidato ao AES (Advanced Encryption Standard), e para isso foi desenvolvido de forma a satisfazer os requisitos dessa competição norte americana. O RC6 também foi submetido ao NESSIE (New European Schemes for Signatures, Integrity, and Encryption), algo como um AES europeu, só que mais amplo. O NESSIE envolve algoritmos simétricos, algoritmos assimétricos e esquemas de assinaturas digitais.

O ataque χ^2 foi originalmente proposto por Knudsen e Meier como um ataque de texto escolhido sobre o RC6. Eles se concentraram sobre correlações entre bits de entrada (texto legível) e bits de saida (texto cifrado), medidas por teste χ^2.

O objetivo desse relatório é apresentar o ataque χ^2 sobre o algoritmo criptográfico RC6, já que atualmente esse é uma das técnicas de criptanálise mais bem sucedidas contra esse algoritmo. Para isso, inicialmente apresentamos o algoritmo RC6, depois passamos à descrição e análise do ataque χ^2 sobre o RC6. Por último discutimos o ataque χ^2 aplicado a uma variante do RC6 denominada RC6W.

2 Cifrador RC6

2.1 Descrição do cifrador

Como o RC5, RC6 é uma família parametrizada completa de algoritmos de criptografia. Uma versão do RC6 é especificada como RC6-$w/r/b$ onde o tamanho da palavra é w bits, a criptografia consiste de um número não negativo de r rodadas, e b denota o tamanho da chave de criptografia em bytes. A submissão ao AES foi feita com $w = 32$ e $r = 20$, nós usamos apenas RC6 para referenciar tais versões. Quando qualquer outro valor de w ou r é desejado, os valores de parâmetro serão especificados como RC6-w/r. Devido ao AES as versões do RC6 se concentram sobre chaves de 16, 24, e 32 bytes.

Para todas as variantes, RC6-$w/r/b$ opera sobre quatro palavras de w bits cada, usando as seis operações básicas seguintes. O logaritmo na base dois de w é denotado por $\lg w$.

$a + b$: adição inteira módulo 2^w;

$a - b$: subtração inteira módulo 2^w;

$a \times b$: multiplicação inteira módulo 2^w;

$a \oplus b$: ou-exclusivo;

$a \lll b$: rotação da palavra a de w bits para a esquerda pela quantidade dada pelos $\lg w$ bits menos significativos de b;

$a \ggg b$: rotação da palavra a de w bits para a direita pela quantidade dada pelos $\lg w$ bits menos significativos de b.

RC6 explora operações dependentes de dados tal que a multiplicação inteira de 32 bits seja eficientemente implementada em muitos processadores. Essa multiplicação é uma primitiva de difusão muito efetiva, e é usada para computar quantidades de rotações, de modo que sejam dependentes de todos os bits de uma palavra.

2.2 Geração da chave

A geração da chave do RC6-$w/r/b$ é praticamente idêntica a geração da chave do RC5-$w/r/b$ e é apresentada aqui. A única diferença é que mais palavras são derivadas da chave fornecida pelo usuário para uso durante a criptografia e decriptografia. Suficientes bytes com valor zero são anexados para dar um tamanho de chave igual a um número não nulo de palavras. Os bytes da chave são lidos em modo *little-endian* em um vetor de c palavras de w bits $L[0], \cdots, L[c-1]$. O número de palavras de w bits que serão geradas para a adição de chave em cada rodada é $2r + 4$ e estas são armazenadas em um vetor $S[0, \cdots, 2r + 3]$.

As contantes $P_{32} =$ `B7E15163` e $Q_{32} =$ `9E3779B9` (hexadecimal) são algumas "constantes mágicas" como usado na geração da chave do RC5. O valor de P_{32} é derivado da expansão binária de $e - 2$, onde e é a base da função logaritmo natural. O valor de Q_{32} é a expansão de $\phi - 1$, onde ϕ é a razão de ouro. Definições similares de RC5 para P_{64}, etc, podem ser usadas para versões do RC6 com outros tamanhos de palavras. Estes valores são um tanto arbitrários, e outros valores poderiam ser escolhidos para dar "gosto pessoal" ou versões proprietárias ao RC6.

Geração de chave para o RC6-$w/r/b$

Entrada:
Chave de b bytes fornecida pelo usuário em um vetor de c palavras $L[0, \cdots, c-1]$; número r de rodadas

Saida:
Chaves de w bits para cada rodada $S[0, \cdots, 2r+3]$

Procedimento:
$S[0] = P_w$

para $i = 1$ até $2r + 3$ faça
$S[i] = S[i-1] + Q_w$

$A = B = i = j = 0$

$v = 3 \times max\{c, 2r+4\}$
para $s = 1$ até v faça
{
$A = S[i] = (S[i] + A + B) \lll 3$
$B = L[j] = (L[j] + A + B) \lll (A + B)$
$i = (i+1) \bmod (2r+4)$
$j = (j+1) \bmod c$
}

2.3 Criptografia e decriptografia

RC6 trabalha com quatro registradores de w bits A, B, C, D que contém um texto de entrada *"legível"* inicial bem como um texto cifrado no fim da criptografia. O primeiro byte do texto de entrada *"legível"* ou texto cifrado é colocado no byte menos significativo de A; o último byte do texto de entrada ou texto cifrado é colocado no byte mais significativo de D. Nós usamos $(A, B, C, D) = (B, C, D, A)$ para indicar atribuição paralela de valores da direita à registradores da esquerda.

Criptografia com RC6-$w/r/b$

Entrada:
Texto *"legível"* armazenado em quatro registradores de w bits A, B, C, D; número r de rodadas; vetor de sub-chaves de w bits $S[0, \cdots, 2r+3]$

Saida:
Texto cifrado armazenado em A, B, C, D

Procedimento:
$B = B + S[0]$
$D = D + S[1]$
para $i = 1$ até r faça
{
$t = (B \times (2B + 1)) \lll \lg w$
$u = (D \times (2D + 1)) \lll \lg w$
$A = ((A \oplus t) \lll u) + S[2i]$
$C = ((C \oplus u) \lll t) + S[2i + 1]$
$(A, B, C, D) = (B, C, D, A)$
}
$A = A + S[2r + 2]$
$C = C + S[2r + 3]$

Decriptografia com RC6-$w/r/b$

Entrada:
Texto cifrado armazenado em quatro registradores de w bits A, B, C, D; número r de rodadas; vetor de sub-chaves de w bits $S[0, \cdots, 2r + 3]$

Saida:
Texto *"legível"* armazenado em A, B, C, D

Procedimento:
$C = C + S[2r + 3]$
$A = A + S[2r + 2]$
para $i = r$ até 1 faça
{
$(A, B, C, D) = (D, A, B, C)$
$u = (D \times (2D + 1)) \lll \lg w$
$t = (B \times (2B + 1)) \lll \lg w$
$C = ((C - S[2i + 1]) \ggg t) \oplus u$
$A = ((A - S[2i]) \ggg u) \oplus t$
}

$D = D - S[1]$
$B = B - S[0]$

3 Ataque χ^2 sobre o RC6

3.1 Testes χ^2

Nesta seção recordaremos como distinguir um código aleatório com distribuição de probabilidade p_X desconhecida de um código com distribuição p_U uniforme. Uma ferramenta comum para essa tarefa é o teste χ^2, que é brevemente recordado juntamente com alguns fatos úteis. Usaremos os testes χ^2 para detectar correlação entre entradas especificas e sub-blocos de saida para r rodadas do RC6.

Seja $X = X_0, X_1, \cdots, X_{n-1}$ variáveis aleatórias independentes e identicamente distribuídas com valores em um conjunto $\{a_0, a_1, \cdots, a_{m-1}\}$ com distribuição de probabilidade desconhecida. Então o teste χ^2 é usado para decidir se a observação $X_0, X_1, \cdots, X_{n-1}$ é consistente com as hipóteses $Pr\{X = a_j\} = p(j)$ para $0 \leq j < m$, onde $p_X = \{p(j)\}$ é uma distribuição de probabilidade (discreta) sobre o conjunto de m elementos. $N_{a_j}(X)$ denota o número de vezes da observação X sobre o valor a_j. Então obviamente $\sum_i N_{a_j}(X) = n$. A estatística χ^2 é uma variável aleatória definida por

$$\chi^2 = \sum_{j=1}^{m} \left(N_{a_j}(X) - np(j)\right)^2 / np(j)$$

Para a distribuição uniforme p_U, a estatística χ^2 é justamente $m/n \sum_i \left(N_{a_j}(X) - n/m\right)^2$. Em um teste χ^2, a estatística χ^2 é comparada a $\chi^2_{a,m-1}$, o valor para o teste χ^2 com $m-1$ graus de liberdade com nível de significância a. Em nossa investigação do RC6, precisamos especificamente do valor para 1023 graus de liberdade, como mostra as tabelas 1 e 2. Por exemplo, a entrada 1131 para 0.99 na tabela 1 diz que a expressão $m/n \sum_i \left(N_{a_j}(X) - n/m\right)^2$ para n grande excederá 1131 somente 1% das vezes, desde que a distribuição da observação X seja realmente uniforme.

Nível	0.50	0.60	0.70	0.80	0.90	0. 95	0.99	0.999	0.9999
χ^2	1022	1033	1046	1060	1081	1098	1131	1168	1200

Tabela 1: Valores da distribuição χ^2 com 1023 graus de liberdade.

Nível	$1-2^{-16}$	$1-2^{-24}$	$1-2^{-32}$	$1-2^{-48}$	$1-2^{-64}$
χ^2	1222	1280	1330	1414	1474

Tabela 2: Valores da distribuição χ^2 com 1023 graus de liberdade.

Para experimentos práticos a questão levantada é quão grande é o tamanho n da observação para detectar que a distribuição p_X é não-uniforme. Para estimar n, considere o desvio de uma distribuição de probabilidade p_X definida pela medida da distância

$$||p_X - p_U|| = \sum_j (p_X(j) - p_U(j))^2$$

De [3] nós citamos o valor esperado da estatística χ^2 de uma distribuição p_X, bem como algumas conclusões úteis:

$$E_X\chi^2 = nm||p_X - p_U|| + m - m||p_X||$$

Para o caso da distribuição uniforme isto implica $E_X\chi^2 = m - 1$. Além disso, segue que para $n = c/||p_X - p_U||$ o valor esperado é $E_X\chi^2 = cm + m - m||p_x||$. Em casos práticos frequentemente $||p_X|| \approx ||p_U||$, isto simplifica que $E_X\chi^2 \approx (c+1)m - 1$. Assim $E_X\chi^2$ difere de $E_U\chi^2$ significativamente, se $c = \Omega(1)$. Como uma conlusão, o tamanho $n = c/||p_X - p_U||$ de uma observação basta para distinguir um código com distribuição p_X de um código com distribuição uniforme. Claramente, a constante c necessita ser grande para um alto nível de significância a.

3.2 Correlações no RC6

Aqui, investigamos a aleatoriedade do RC6. Esta análise é baseada em experimentos sistemáticos, aumentando o número de rodadas do RC6 como descrito em [2]. Esse método é usado para demonstrar que detectar aleatoriedade é possível até seis rodadas do RC6.

Para esse propósito, os $\lg w$ bits menos significativos das palavras A e C da entrada são fixados como zero. Além disso, as palavras B e D são escolhidas de forma que produzam rotação zero na primeira rodada. O teste χ^2 é aplicado na concatenação dos

$\lg w$ bits menos significativos de A'' e C'', que são partes do resultado da criptografia de (A, B, C, D).

Nos preocupamos aqui com o caso onde $w = 32$ bits. A tabela 3 mostra os resultados da implementação dos testes para $r = 2$ e 4 rodadas. Lembramos que para inteiros de dez bits o valor esperado da estatística χ^2 é 1023, e de acordo com a tabela 1 a 95% de nível de significância é 1098 e a 99% de nível de significância é 1131.

r	#textos	χ^2	#testes
2	2^{13}	1096	20
2	2^{14}	1196	20
2	2^{15}	1332	20
2	2^{16}	1649	20
2	2^{17}	1208	20
4	2^{29}	1096	20
4	2^{30}	1163	20
4	2^{31}	1314	20
4	2^{32}	1527	20
4	2^{33}	1054	20

Tabela 3: χ^2 esperado para uma função aleatória de 1023 graus de liberdade.

3.3 Recuperação da chave de criptografia

Como confirmado pelos experimentos, o valor χ^2 é significativamente alto se entradas são escolhidas de maneira que as rotações dependentes de dados na primeira rodada do RC6 sejam zero. Claramente, a escolha de uma entrada depende do conhecimento da sub-chabe $S[0]$ (ou $S[1]$, respectivamente).

Mais uma vez lembramos que estamos descrevendo o caso onde $w = 32$ bits. O ataque para recuperação da chave é como descrito em [2]. Escolhemos textos tal que os cinco bits menos significativos da primeira e terceira palavras(A e C) sejam zero. Preparamos um vetor com 2^{10} entradas para cada valor da sub-chave $S[1]$. Para cada texto de entrada usamos uma tabela T pré-calculada com todos os 2^{27} valores que produzem rotação zero, para determinar o valor de $S[1]$ que induz rotação zero na palavra D. Para cada valor, atualizamos cada vetor incrementando a entrada correspondente para o valor obtido dos dez bits "marcados" do texto cifrado. Repita o ataque suficientemente muitas vezes, até um vetor ter um valor altamente significante no teste χ^2.

Para uma estimativa da complexidade de recuperação da sub-chave $S[1]$, consideramos

versões do RC6 com até r rodadas. Para cada experimento da chave $S[1]$, o teste χ^2 é executado com

$$2^{13} \times (2^{16.2})^{\frac{r-2}{2}} \times 2^{-8.1}$$

textos de entrada. Então para a escolha correta de $S[1]$ o valor χ^2 esperado está em torno de 1100, que é, significativamente alto com relação a 1023. Para cada chave que produz um valor χ^2, repetimos o ataque com textos de entrada adicionais.

Depois de recuperar a sub-chave $S[1]$, a sub-chave $S[0]$ pode ser determinada com uma quantidade pequena de textos e trabalho. Conhecidas $S[0]$ e $S[1]$ as rotações dependentes de dados da primeira rodada podem ser fixadas para zero sem esforço. Então testes χ^2 podem agora ser aplicados para controlar entradas para a segunda rodada. Assim é possível determinar as sub-chaves $S[2]$ e $S[3]$ do mesmo modo que foi feito com $S[0]$ e $S[1]$. Com isso, também podemos achar outras sub-chaves.

4 Ataque χ^2 sobre o RC6W

4.1 Preliminares

Em [4] é discutido o ataque χ^2 sobre uma variante do algoritmo RC6, denominada RC6W, que difere do RC6 original por não apresentar a adição das sub-chaves $S[0]$ e $S[1]$ no início do algoritmo e das sub-chaves $S[2r+2]$ e $S[2r+3]$ no final.

Apresentamos a seguir algumas notações, elaboradas com o objetivo de facilitar o entendimento dos vários aspectos que envolvem o ataque χ^2 a partir desse ponto.

$lsb_n(X)$: n bits menos signifivativos de X;

S_i : i-ésima subchave;

X^i : i-ésimo bit de X;

$f(X) : X \times (2X + 1)$;

$F(X) : f(X)(\bmod 2^w) \lll \lg w$.

Denotamos o bit menos significativo (LSB) como o primeiro bit, e o bit mais significativo (MSB) como o w-ésimo bit para qualquer elemento de w bits. Aqui estamos mais interessados no caso onde $w = 32$ bits.

4.2 Estatística χ^2 do RC6

Nesta seção, investigamos como conseguir um desvio maior com menos restrição de textos. Em [2], se textos de entrada (A_0, B_0, C_0, D_0) são escolhidos de tal modo que $lsb_n(A_0)$ e $lsb_n(C_0)$ são fixados, e B_0 e D_0 induzem rotação zero na primeira rodada, então os textos cifrados apresentam desvios elevados. Entretanto, tal condição é restritiva por causa do número de textos que satisfazem tais condições é reduzido a 2^{108}. Em [4] foram analizadas outras condições que tem quase o mesmo efeito. Daí, os seguintes experimentos são considerados para atingir uma correlação maior entre textos de entrada e saida.

Teste 1 : Teste χ^2 sobre $lsb_5(A_{r+1})||lsb_5(C_{r+1})$ no caso onde B_0 e D_0 induzem rotação zero na primeira rodada; $lsb_5(A_0) = 0$; e $lsb_5(C_0) = 0$.

Teste 2 : Teste χ^2 sobre $lsb_5(A_{r+1})||lsb_5(C_{r+1})$ no caso onde B_0 e D_0 induzem rotação zero na primeira rodada; $lsb_5(A_0) = 0, \cdots, 31$; e $lsb_5(C_0) = 0$.

Teste 3 : Teste χ^2 sobre $lsb_n(A_{r+1})||lsb_n(C_{r+1})$ para $n = 3, 4, 5$ no caso onde $lsb_5(A_0)$ e $lsb_5(C_0)$ são 0, e B_0 e D_0 induzem rotação zero na primeira rodada.

Teste 4 : Teste χ^2 sobre (quaisquer 5 bits consecutivos de $A_{r+1})||lsb_5(C_{r+1})$ no caso onde $lsb_5(A_0)$ e $lsb_5(C_0)$ são 0, e B_0 e D_0 induzem rotação zero na primeira rodada.

Teste 5 : Teste χ^2 sobre $lsb_5(A_{r+1})||lsb_5(C_{r+1})$ do RC6 sem adição das sub-chaves $S[0]$ e $S[1]$ com $lsb_5((A_0 \oplus F(B_0)) \lll F(D_0)) = 0$ e $lsb_5((C_0 \oplus F(B_0)) \lll F(D_0)) = 0$.

Teste 6 : Teste χ^2 sobre $lsb_5(A_{r+1})||lsb_5(C_{r+1})$ do RC6 sem adição das sub-chaves $S[0]$ e $S[1]$ com $lsb_5((C_0 \oplus F(D_0)) \lll F(B_0)) = 0$ e $lsb_5((A_0 \oplus F(B_0)) \lll F(D_0)) = 0, \cdots, 31$.

4.3 Algoritmos de ataque sobre o RC6W

Nesta seção são apresentados três algoritmos de ataque para a recuperação da chave de criptografia do algoritmo RC6W. Os dois primeiros deles são do tipo texto escolhido e o terceiro é do tipo texto conhecido.

Todos os três algoritmos recuperam $lsb_2(S_{2r})$ e $lsb_2(S_{2r+1})$ do RC6W, mas mais bits da chave podem ser estimados repetindo o algoritmo sequencialmente. Para simpli-

ficar um pouco a descrição dos algoritmos considere $(lsb_3(B_{r+1}), lsb_3(D_{r+1})) = (y_b, y_d)$, $(lsb_2(S_{2r}, lsb_2(S_{2r+1}) = (s_a, s_c)$ e $(lsb_5(F(A_{r+1})), lsb_5(F(D_{r+1}))) = (x_c, x_a)$, onde x_a e x_c são quantidades de rotações sobre A_r e C_r na r-ésima rodada, respectivamente.

O algoritmo 1 considera os testes 3 e 4 vistos anteriormente. O algoritmo 2 é uma evolução do algoritmo 1 a partir da aplicação do teste 2, conseguindo um resultado melhor devido a redução da variância. Por último o algoritmo 3 é ainda melhor do que os algoritmos 1 e 2, considerando os testes 5 e 6 mostrados.

Algoritmo de ataque #1

1: Escolha um texto de entrada (A_0, B_0, C_0, D_0) com $(lsb_5(A_0), lsb_5(C_0), lsb_5(F(B_0)), lsb_5($ $F(D_0))) = (0, 0, 0, 0)$, e criptografe ele;

2: Para cada $s_a, s_c = 0, 1, 2, 3$, seja $s = s_a||s_c$ e $S^3_{2r}, S^3_{2r+1} = 0$, decriptografe $y_d||y_b$ com a chave $(S^3_{2r}||s_a, S^3_{2r+1}||s_c)$ por uma rodada usando as quantidades de rotação x_a e s_c da r-ésima rodada. As decriptografias de $y_d||y_b$ são os valores $z_a||z_c = z$;

3:Para cada s, xa, xc, e z, atualizamos cada vetor incrementando $count[s][x_a][x_c][z]$;

4:Para cada s, x_a, e x_c, calcule $\chi^2[s][x_a][x_c]$;

5: Calcule a média $ave[s]$ de $\chi^2[s][x_a][x_c]$ para cada s, e devolva o s com o mais elevado $ave[s]$ como $lsb_2(S_{2r})||lsb_2(S_{2r+1})$.

Algoritmo de ataque #2

1: Escolha um texto de entrada (A_0, B_0, C_0, D_0) com $(lsb_5(F(B_0)), lsb_5(F(D_0)), lsb_5(C_0)) =$ $(0, 0, 0)$, considere $lsb_5(A_0) = t$, e criptografe ele;

2: Para cada $(s_a, s_c)(s_a, s_c = 0, 1, 2, 3)$, seja um inteiro de 4 bits $s = s_a||s_c$, $S^3_{2r}, S^3_{2r+1} = 0$, e decriptografe $y_d||y_b$ com a chave $(S^3_{2r}||s_a, S^3_{2r+1}||s_c)$ por uma rodada. Consideramos a decriptografia de y_d, y_b como z_a, z_c, que são inteiros de 3 bits. Também consideramos um inteiro de 6 bits $z = z_a||z_c$;

3:Para cada s, t, xa, xc, e z, atualizamos cada vetor incrementando $count[s][t][x_a][x_c][z]$;

4:Para cada s, t, x_a, e x_c, calcule $\chi^2[s][x_a][x_c]$;

5: Calcule a média $ave[s]$ de $\chi^2[s][t][x_a][x_c]$ para cada s, e devolva o s com o mais elevado $ave[s]$ como $lsb_2(S_{2r})||lsb_2(S_{2r+1})$.

Algoritmo de ataque #3

1: Dado um texto de entrada (A_0, B_0, C_0, D_0), considere $lsb_3((A_0 \oplus F(B_0)) \lll F(D_0)) = t_a$, $lsb_3((C_0 \oplus F(D_0)) \lll F(B_0)) = t_c$, e criptografe ele;

2: Para cada $(s_a, s_c)(s_a, s_c = 0, 1, 2, 3)$, seja um inteiro de 4 bits $s = s_a||s_c$, $S^3_{2r}, S^3_{2r+1} = 0$, e decriptografe $y_d||y_b$ com a chave $(S^3_{2r}||s_a, S^3_{2r+1}||s_c)$ por uma rodada. Consideramos a decriptografia de y_d, y_b como z_a, z_c, que são inteiros de 3 bits. Também consideramos um inteiro de 6 bits $z = z_a||z_c$;

3:Para cada s, t_a, t_c, xa, xc, e z, atualizamos cada vetor incrementando $count[s][t_a][t_c][x_a][x_c][z]$;

4:Para cada s, t_a, t_c, x_a, e x_c, calcule $\chi^2[s][t_a][t_c][x_a][x_c]$;

5: Calcule a média $ave[s]$ de $\chi^2[s][t_a][t_c][x_a][x_c]$ para cada s, e devolva o s com o mais elevado $ave[s]$ como $lsb_2(S_{2r})||lsb_2(S_{2r+1})$.

As tabelas a seguir mostram a probabilidade de sucesso e o valor χ^2 dos algoritmos 2, 3 e 4, respectivamente, para cinco rodadas.

#textos	#chaves	$valor\chi^2(63graus)$		
		Média	Nível	Variância
2^{17}	12	63.106	0.527	0.165
2^{18}	8	63.076	0.526	0.122
2^{19}	16	63.216	0.531	0.109
2^{20}	32	63.492	0.541	0.107
2^{21}	71	64.049	0.561	0.102
2^{22}	99	65.119	0.597	0.133
2^{23}	100	67.321	0.668	0.218

Tabela 4: Probabilidade de sucesso e valores χ^2 do algoritmo 1 (em 100 experimentos).

#textos	#chaves	$valor\chi^2(63graus)$		
		Média	Nível	Variância
2^{22}	21	63.067	0.526	0.003
2^{23}	54	63.135	0.528	0.003
2^{24}	93	63.267	0.533	0.005

Tabela 5: Probabilidade de sucesso e valores χ^2 do algoritmo 2 (em 100 experimentos).

#textos	#chaves	$valor\chi^2(63graus)$		
		Média	Nível	Variância
2^{25}	26	63.057	0.526	0.0003
2^{26}	59	63.108	0.528	0.0005
2^{27}	100	63.230	0.532	0.0007

Tabela 6: Probabilidade de sucesso e valores χ^2 do algoritmo 3 (em 100 experimentos).

5 Conclusão

Apresentamos o ataque χ^2 sobre o algoritmo RC6 e uma de suas variações, o RC6W, que difere do RC6 original apenas por não apresentar as adições das sub-chaves $S[0]$ e $S[1]$ no início do algoritmo e $S[2r+2]$ e $S[2r+3]$ no final.

A estimativa é de que seja possível quebrar o RC6 com até 15 rodadas, por isso esse é considerado no momento uma das técnicas de criptanálise bem mais sucedidas contra esse algoritmo. Já no caso do RC6W a estimativa é de que consigamos quebrar até 17 rodadas. Só para lembrar, o RC6 foi submetido à competição AES com $r = 20$ rodadas.

Uma contribuição interessante com relação ao que foi estudado seria tentar achar uma forma de fortalecer o algoritmo RC6 contra esse ataque, pois até o momento não temos notícia de alguém que tenha tentando realizar tal feito. O ataque χ^2 é recente na literatura de criptanálise, e sua aplicação a outros algoritmos talvez possa gerar outras contribuições.

Referências

[1] Isogai, N.; Matsunaka, T.; Miyaji, A.. *Optimized χ^2-Attack against RC6*, Artigo , 2003

[2] Knudsen, L.R.; Meier, W.. *Correlations in RC6 with a Reduced Number of Rounds*, Artigo, 2001

[3] Kelsey, J.; Schneier, B.; Wagner, D. *Mod n cryptanalysis, with applications against RC5P and M6*, Artigo, 1999

[4] Miyaji, A.; Nonaka, M.. *Cryptanalysis of Reduced-Round without Whitening*, Artigo, 2003

[5] Rivest, R.L.; Robshaw, M.J.B.; Sidney, R.; Yin, Y.L.. *The RC6 Block Cipher* , Artigo, 1998

[6] Takenaka, M.; Shimoyama, T.; Koshiba, T.. *Theoretical Analysis of "Correlations in RC6"* , Artigo, 2002

[7] Terada, R.. *Segurança de Dados - Criptografia em Redes de Computador*, Livro, 2000