Correção de Erros nas Chaves Privadas RSA

Reynaldo Cáceres Villena
Orientador: Routo Terada

DCC - IME - USP

Agosto de 2011

RSA

O criptosistema RSA foi proposto publicamente em 1978 e nomeado assim pelos seus criadores Ron **R**ivest, Adi **S**hamir e Leonard **A**dleman. RSA é o criptosistema de chave pública (Cripto-sistema assimétrico) mais utilizado até a data.

Criptosistema RSA

Definimos $N = pq$ onde p e q são primos. Por definição da função Totiente do Euler temos que $\phi(N) = (p-1)(q-1)$.

Keygen: Escolher um e co-primo a $\phi(N)$ e procure um d tal que que $ed = 1 \bmod \phi(N)$

Keydist: Publicar a chave $\text{pk}\langle N, e\rangle$ e manter em segredo $\text{sk}\langle N, d\rangle$

Encrypt: Para uma mensagem $M \in \mathbb{Z}_N$, o criptograma é $C = M^e \bmod N$

Decrypt: Para um criptograma C, a mensagem é $M = C^d \bmod N$

PKCS - Public Key Cryptography Standards

O PCKS é um padrão com recomendações e definições para implementação dum criptosistema de chave pública.
O PCKS 1 especifica para o RSA que a chave secreta sk deve conter as seguintes informações:

- O expoente para decriptagrar d.
- Os fatores primos p e q de N.
- Os valores de $d_p = 1 \bmod(p-1)$ e $d_q = 1 \bmod(q-1)$.
- e a inversa de q modulo p, denotado por $q_p^{-1} = q^{-1} \bmod p$. Os valores d_p, d_q e q_p^{-1} são necessarios para decriptografar usando o TCR (teorema chinês do resto).

Modificação no Criptosistema RSA

Keydist: Publicamos a chave pk$\langle N, e\rangle$, calculamos d_p, d_q e q_p^{-1}, e mantemos em segredo sk$\langle d, p, q, d_p, d_q, q_p^{-1}\rangle$

Ataques Side-channel

Na criptografia, um ataque Side-channel é qualquer ataque onde a informação é obtida pela implementação física de um cripto-sistema (Hardware), para realizar esses ataques requiere un conhecimento técnico das operações internas do sistema onde o cripto-sistema foi implantado.

Clases:

- Ataques de temporização
- Análise de energia
- Criptanálise acústica
- Ataque de cache (DRAM)
- Ataque eletromagnético

Com estes ataques podemos obter $\tilde{sk}$ $\langle \tilde{d}, \tilde{p}, \tilde{q}, \tilde{d}_p, \tilde{d}_q \rangle$ com um valor determinado δ, onde δ é o rango de erro.

Geração de Candidatos

Temos os seguintes dados $\tilde{sk}\langle \tilde{d}, \tilde{p}, \tilde{q}, \tilde{d}_p, \tilde{d}_q \rangle$ com δ>0,27 e $pk\langle N, e \rangle$. Sabemos que a formula RSA é:

Equação RSA

$$ed = 1 \bmod (p-1)(q-1)$$
$$ed = 1 + k(p-1)(q-1)$$

sabemos que o d e e pertencem ao grupo $\phi(N)$, pelo qual temos que $e, d < (p-1)(q-1)$, e já que $d < (p-1)(q-1)$ obtemos que o valor de k esta entre 1 e e, $1 < k < e$.
Cada candidato para este programa é obtido pela assignação de um valor a k, então o número de candidatos esta dado pelo valor $e - 1$

Dados

Na atualidade, o valor mais usado para o expoente e é $2^{16} + 1$ portanto temos $2^{16} = 65536$ candidatos.

Determinação dos valores de k, k_p e k_q

Para cada candidato agora tenemos $\tilde{sk}\langle \tilde{d}, \tilde{p}, \tilde{q}, \tilde{d}_p, \tilde{d}_q \rangle$ e k

Tenemos as seguintes equações:

$ed = 1 + k(N + 1 - p - q)$ onde $d < N + 1 - p - q$

$ed_p = 1 + k_p(q - 1)$ onde $d_p < q - 1$

$ed_q = 1 + k_q(p - 1)$ onde $d_q < p - 1$

portanto k, k_p e $k_q < e$

As tres equações de acima podem ser resolvidas modulo e, onde o numero de soluções para (k_p, k_q) esta dado por 2^m, onde m é o numero de fatores primos de e.

Dados

para $e = 2^{16} + 1$ que é primo, temos $2^1 = 2$ soluções para (k_p, k_q).

Correção dos bits superiores de $\tilde{d}$

Para cada candidato $\tilde{sk}\langle \tilde{d}, \tilde{p}, \tilde{q}, \tilde{d}_p, \tilde{d}_q \rangle, k,$. Sabemos que:

Para um e pequeno, podemos calcular facilmente os bits superiores de d com a seguinte equação

$$d = \frac{1+k(N+1-p-q)}{e}$$
$$d = \frac{1+k(N+1)}{e} - \frac{k(p+q)}{e}$$
$$d_k = \frac{1+k(N+1)}{e}, \text{ onde } |d - d_k| = \frac{k(p+q)}{e} \leq 2^{\lceil \frac{n}{2} \rceil + 2}$$

Com d_k podemos corrigir a metade dos bits superiores de $\tilde{d}$ de nosso candidato $\tilde{sk}\ \langle \tilde{d}, \tilde{p}, \tilde{q}, \tilde{d}_p, \tilde{d}_q \rangle$.

para i desde $\lceil \frac{n}{2} \rceil + 2$ até n
    $\tilde{d}[i] = d_k[i]$

Correção dos bits inferiores de $\tilde{d}_p$ e $\tilde{d}_q$

Seja s[i] o i-ésimo bit de s.

Sabemos que p e q são primos portanto corrigimos os bit:

$p[0] = 1$
$q[0] = 1$

seja $\tau(x)$ o maior expoente da potência de 2 que divide x.

sabemos que $2|p-1$, então temos que $2^{1+\tau(k_p)}|k_p(p-1)$

$ed_p = 1 \bmod k_p(p-1)$
$ed_p = 1 \bmod 2^{1+\tau(k_p)}$

Agora podemos corrigir os $1+\tau(k_p)$ primeiros bits de $\tilde{d}_p$ com 1.
Usamos o mesmo conceito para corrigir os $1+\tau(k_q)$ primeiros bits de $\tilde{d}_q$.

Planteamento das Restrições

Podemos fácilmente olhar o seguinte:

- Uma troca de valor no $p[i]$ afeta ao valor de $d_p[1 + \tau(k_p)]$
- Uma troca de valor no $q[i]$ afeta ao valor de $d_q[1 + \tau(k_q)]$
- Uma troca de valor no $q[i]$ ou $p[i]$ afeta ao valor de $d[1 + \tau(k)]$

Restrições

$$p[i] + q[i] \equiv c1$$
$$d[i + \tau(k)] + p[i] + q[i] \equiv c2$$
$$d_p[i + \tau(k_p)] + p[i] \equiv c3$$
$$d_q[i + \tau(k_q)] + q[i] \equiv c4$$

Podemos ver que temos 5 variáveis

$p[i]$ $q[i]$ $d[i + \tau(k)]$ $d_p[i + \tau(k_p)]$ $d_p[i + \tau(k_p)]$

Ingenuamente podemos pensar que temos 2^5 possibilidades, mas de fato só temos como máximo 2.

Algoritmo de Reconstrução

LEMA DE HENSEL.- Uma raiz $r = (r_1, r_2, ...r_n)$ de um polimonio $f(x_1, x_2, ...x_n)$ mod π^i podemos obter uma raiz $r + b$ mod π^{i+1} se $b = (b_1\pi^i, b_2\pi^i, ...b_n\pi^i)$ é uma solução com $0 \leq b_i \leq \pi - 1$.
$f(r + b) = f(r) + \sum b_j\pi^i f_{x_j}(r) \equiv 0 \pmod{\pi^{i+1}}$
Podemos reescrever o lema com r em base $\pi = 2$ e assumimos que os i primeiros bits são conhecidos. Portanto o lema disse que o seguinte bit de $r, r[i] = (r_1[i], r_2[i], ...)$
$f(r)[i] + \sum f_{x_j}(r) r_j[i] \equiv 0 \pmod 2$

Aplicando o lema de Hensel a nossas últimas equações com uma solução parcial (p', q', d', d'_p, d'_q) até o bit $i - 1$, podemos obter os bits i

$$p[i]+q[i] \equiv (N - p'q')[i] \text{ -mod 2}$$
$$d[i+\tau(k)]+p[i]+q[i] \equiv (k(N+1)+1-k(p'+q')-ed')[i+\tau(k)] \text{ mod 2}$$
$$-d_p[i+\tau(k_p)]+p[i] \equiv (k_p(p'-1)+1-ed'_p)[i+\tau(k_p)] \text{ mod 2}$$
$$-d_q[i+\tau(k_p)]+p[i] \equiv (k_q(q'-1)+1-ed'_p)[i+\tau(k_q)] \text{ mod 2}$$

Algoritmo Error-Correction

Entrada:(N, e), e um erróneo $\tilde{sk}$ $\langle \tilde{d}, \tilde{p}, \tilde{q}, \tilde{d}_p, \tilde{d}_q \rangle$ com δ,

Fase de Iniciação:

$(k, k_p, k_q) \leftarrow Init(N, e)$

$Slice(0) \leftarrow Mount(e, k, k_p, k_q)$

Para $i = 1$ **to** $\lceil \frac{n/2-1}{t} \rceil$**:**

Fase de Expansão: Para cada candidato (p', q', d', d'_p, d'_q) con Slices $0, 1, ..., (i-1)t$ expandimos t times o modulo principal de Expand(.) do algoritmo de Heninger-Shacham, de este obtemos 2^t novos candidatos

Fase de Filtrado: Para cada novo candidato (p', q', d', d'_p, d'_q) contamos o total de bits que correspondem a $\tilde{sk}$. Se o numero é menor ao parámetro C, o candidato é descartado

Fase de Finalização: Para cada candidato $sk' = (p', q', d', d'_p, d'_q)$ verificamos as equações RSA, se cumplem então sk' é a saida

SAIDA: $sk = (p, q, d, d_p, d_q)$

Valores dos parámetros t e C

Segundo as Probabilidades de Hoeffding os valores para t, C e γ_0

$t = \lceil \frac{ln(n)}{10\epsilon^2} \rceil$

$\gamma_0 = \sqrt{(1 + \frac{1}{t})\frac{ln(2)}{10}}$

$C = 5t(\frac{1}{2} + \gamma_0)$

Sabemos que: $\delta \leq \frac{1}{2} - \gamma_0 - \epsilon$. Nota que para un n e t grande nossa variável γ_0 converge a $\sqrt{\frac{ln(2)}{10}} \approx 0.263$. Este significa que o algoritmo só trabalha para rango de erro

$\delta \leq \frac{1}{2} - 0.263 - \epsilon = 0.237 - \epsilon$

Probabilidade de sucesso do algoritmo

Teorema

Seja $X_1, ..., X_n$ uma sequencia de eventos independentes com uma probabilidade de succeso idêntica $Pr[X_i = 1] = p$ para todo i. Definimos $X = \sum_{i=1}^{n} X_i$. então para todo $0 < \gamma < 1$

- $Pr[X \geq (p + \gamma)] \leq e^{-2\gamma^2}$
- $Pr[X \leq (p + \gamma)] \leq e^{-2\gamma^2}$

A probabilidade de eliminar um solução esta dada por:

$Pr[X < C] = Pr[X < 5t(\frac{1}{2} + \gamma_0)] \leq Pr[X < 5t(1 - \delta - \epsilon)]$

$- \leq e^{-10t\epsilon^2} \leq \frac{1}{n}$

O número de execução do algoritmo esta dado por $\beta = \frac{n}{2t} + 1$. A probabilidade de sucesso dada por:

$Pr[Sucesso] = (1 - Pr[X < C])^{\beta} \geq (1 - \frac{1}{n})^{\beta} \geq 1 - \frac{\beta}{n}$

$- \geq 1 - (\frac{1}{2t} + \frac{1}{n}) \geq 1 - (\frac{5\epsilon^2}{ln(n)} + \frac{1}{n})$

Ordem do Algoritmo

O tempo de execução do algoritmo esta dado por:

$$T = T_{init} + O(e)(T_{mount} + T_{main})$$

Implementação

Table 1. Parameters for varied RSA scenarios

sk	m	Equations	δ
(p, q)	2	(8)	0.084
(p, q, d)	3	(8),(9)	0.160
(p, q, d, d_p)	4	(8)-(10)	0.206
(p, q, d, d_q)	4	(8),(9),(11)	0.206
(p, q, d, d_p, d_q)	5	(8)-(11)	0.237

Implementação

- A implementação deste algoritmo foi testada sobre uma Intel Seon processador QuadCore de 2.66 Ghz com 8 Gb de DDR2 SDRAM de 800 Mhz
- Os experimentos foram feitos para chaves de tamanho de 1024 bits e com expoente de criptação $e = 2^{16} + 1$
- Os experimentos foram testados para os rangos de erro $\delta \in [0.05, 0.2]$

Table 2. Experimental results for $n = 1024$ and $\mathsf{sk} = (p, q, d, d_p, d_q)$

δ	0.05	0.06	0.07	0.08	0.09	0.10	0.11	0.12	0.13	0.14	0.15	0.16	0.17	0.18	0.19	0.20
t	3	4	5	6	7	9	9	10	10	11	12	12	13	16	16	20
C	12	16	20	24	28	36	36	39	39	42	46	45	48	59	59	74
Pr theoretical	0.39	0.48	0.51	0.49	0.44	0.50	0.27	0.49	0.28	0.44	0.28	0.35	0.43	0.47	0.26	0.23
experimental	0.40	0.48	0.52	0.50	0.45	0.51	0.27	0.50	0.28	0.45	0.28	0.35	0.44	0.50	0.24	0.21
time	< 1s					...						< 1s	3.7s	23s	25s	3m

Referências

- Correcting Errors in RSA Private Keys
  Wilko Henecka e Alexander Meurer (CRYPTO 2010)
- Reconstructing rsa private keys from random key bit
  N. Heninger e H. Shacham (CRYPTO 2009)
- Factoring with an oracle
  U.M. Maurer (EUROCRYPT 1992)
- Efficient factoring based on partial information
  R.L. Rivest e A. Shamir (EUROCRYPT 1985)
- Pobrability Inequalities for Sums of Bounded Random Variables
  W. Hoeffding (1963)
- RSA Laboratories. PCKS #1 v2.1
  Online: http://www.rsa.com/rsalabs/node.asp?id=2125

???