Nuevos Algoritmos de Factorización de Enteros para atacar RSA

Ekoparty

Buenos Aires, 3 de octubre de 2008

Hugo D.Scolnik
Departamento de Computación
Universidad de Buenos Aires

Esquema de la conferencia:

1) Las ideas fundamentales
2) Nuevos resultados
3) El algoritmo recursivo básico
4) Filtrando
5) Corrigiendo
6) Demostración del último algoritmo

Notación básica:

n un entero impar a factorizar

$a(m)$ denotará a módulo m o sea el resto de dividir a por m

Hoy en día todavía se ignora si el problema de descomponer a un entero en sus factores primos tiene complejidad subexponencial o polinomial (usando computadoras "normales", dejaremos las cuánticas para otro momento)

Aparte del interés teórico, existe uno muy práctico: si se pudiese factorizar enteros "grandes" entonces se quebrarían las firmas digitales hechas con el algoritmo más popular (RSA)

Veamos la razón:

El algoritmo RSA:

Seleccionar al azar dos números primos de "gran" longitud.
Calcular $n = pq$
Elegir un entero impar e primo relativo con $\varnothing(n) = (p-1)(q-1)$
Calcular d tal que $e.d\ (\varnothing(n)) = 1$ *(d existe y es único).*
Publicar el par $P = (e,n)$ *que es la clave pública del método RSA.*
Mantener en secreto el par $S = (d,n)$ *que es la clave privada del método RSA.*
Para encriptar un mensaje M con la clave privada se calcula
$S(M) = M^{d} \pmod n$ *y para desencriptarlo,*
$P(S(M)) = (S(M))^{c} \pmod n = M$

(si se factoriza n se obtienen p,q, de ahí $\varnothing(n) = (p-1)(q-1)$*, y al resolver* $e.d \pmod{\varnothing(n)} = 1$ *, se obtiene la clave privada a partir de la pública)*

La empresa RSA Data Security ***publicaba*** diversos desafíos conocidos como *RSAm* (donde *m* indica la longitud en bits del número). El logro más reciente (y último) es el *RSA640* =

3107418240490043721350750035888567930037346022842727545720161948823206440518081504556346829671723286782437916272838033415471073108501919548529007337724822783525742386454014691736602477652346609

Cuyos factores son:

163473364580925384844313388386509085984178367003309231218111085238933310010450815121211 8167511579

190087128166482211312685157393541397547189678996851549366663853908802710380210449895719 1261465571

Uno fundamental pues ese tamaño es un estándar

RSA1024 =

135066410865995223349603216278805969938881475605667027524485143851526510604859533833940287150571909441798207282164471551373680419703964191743046496589274256239341020864383202110372958725762358509643110564073501508187510676594629205563685529475213500852879416377328533906109750544334999811150056977236890927563

Recordemos algunas definiciones

Los enteros módulo n se denotan por $Z_n = \{0,1,...,n-1\}$

El grupo multiplicativo de Z_n es $Z_n^* = \{a \in Z_n \ / \text{MCD}(a,n) = 1\}$

Decimos que $a \in Z_n^*$, es un residuo cuadrático módulo n

si existe x^2 tal que $x^2 \equiv a(n)$

(o sea que hay valores λ tales que $a + \lambda n$ es un CP)

Al conjunto de todos los residuos cuadráticos módulo n

se lo denota por Q_n.

Notar que $0 \notin Q_n$ pero como lo necesitaremos, definimos

$RoS_n = Q_n \cup \{0\}$

Veamos el cálculo de residuos en forma eficiente

Un problema importante:

dados (a,n) determinar si

$$a \in RoS_n$$

Recordemos que el símbolo de Legendre se define para todos los enteros a y todos los primos p mediante

$$\left(\frac{a}{p}\right)=\begin{cases} 0 \text{ si } a\equiv 0(p) \\ 1 \text{ si } a\neq 0(p) \text{ y para algún entero } x \text{ es } x^2\equiv a(p) \\ -1 \text{ si no existe un } x \text{ tal que } x^2\equiv a(p) \end{cases}$$

Usando el criterio de Euler resulta $\left(\frac{a}{p}\right)=a^{\frac{(p-1)}{2}}(p)$

¿Qué sucede cuando calculamos $\left(\frac{a}{n}\right)$ para n impar compuesto ?

Si $n = p_1^{\alpha_1} \ldots p_k^{\alpha_k}$ se define al símbolo de Jacobi como el producto de los símbolos de Legendre de los factores primos, o sea:

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{\alpha_1} \ldots \left(\frac{a}{p_k}\right)^{\alpha_k}$$

Si $\left(\frac{a}{n}\right) = -1 \quad \Rightarrow \quad a \notin Q_n$

Si $a \in Q_n \qquad \Rightarrow \quad \left(\frac{a}{n}\right) = 1$

pero la recíproca no es cierta, o sea que puede ser que

$\left(\frac{a}{n}\right) = 1$ pero $a \notin Q_n$

Demostración de un nuevo algoritmo tal que dados (a,n) decide si

$$a \in RoS_n \quad \text{o} \quad a \notin RoS_n$$

Si $a \in Q_n$ sabemos que existen valores de x tales que $x^2 = a + nt$ para ciertos valores de t

Ejemplo: $a = 4,\ n = 7$

t	x^2
0	4
3	25
11	81
20	144
.	.
155	1089

Todos son de la forma

$x^2 = (7z + 2)^2$

Otro ejemplo:

Todos los cuadrados generados mediante

$x^2 = 81 + 532t$ están dados por las dos familias de parábolas

$(266z + 9)^2$ y $(266z + 47)^2$

(la teoría sobre generación y reducción de parábolas

y determinación de las equivalentes se debe a Martín Degrati)

El clásico método de Fermat

Sea $n = p.q$ el número a factorizar.

Si $x^2 = (\frac{p-q}{2})^2$, $y^2 = (\frac{p+q}{2})^2$ son enteros

entonces

$n + x^2 = y^2 \Rightarrow p = MCD(n,(x-y)),\ q = MCD(n,x+y)$

(salvo casos triviales)

La idea del viejo método de Fermat es lograr una representación $n + x^2 = y^2$ y esto sólo funciona si los factores primos están próximos.

Ejemplo:

N = 2581 que se puede escribir como

$2581 = 59^2 - 30^2 = (59+30)(59-30) =$

$= 89.29$

Pero en general es casi imposible conseguir una representación así. Por ese motivo un matemático ruso, Maurice Kraitchik , planteó buscar congruencias

$$x_i^2 \equiv y_i^2 (n)$$

Los métodos modernos más eficientes (quadratic sieve, number field sieve) tienen su origen en estas ideas, pero conducen a resolver sistemas de ecuaciones módulo 2 de millones de incógnitas.

Se considera que han llegado a su límite.

Pretendemos mirar al método de Fermat desde otro punto de vista.

Mi idea surgió del problema de decidir cuando un entero k es un cuadrado perfecto SIN calcular su raíz cuadrada.

El enfoque actual (ver por ejemplo GNU) es calcular $k(m)$ para valores elegidos de m y ver si se obtiene un valor en

RoS_m

Ejemplo: $k = 52$. Como $k(7)=3$ y $RoS(7) = \{0,1,2,4\}$ entonces 52 no es un cuadrado perfecto. Tampoco $k = 53$ es un cuadrado pero $53(7) = 4$ y entonces $m=7$ no "filtra" ese caso. Pero

$RoS_5 = \{0,1,4\}$ y $53(5) = 3$ o sea que se "filtra".

Veamos la idea central:

Sea por ejemplo $n = 507527$ y tomemos $m = 9$

$RoS_9 = \{0,1,4,7\}$ y como $507527(9) = 8$ resulta que

$(n + x^2)(9) = y^2(9)$

$8 + \{0,1,4,7\} = y^2(9)$

El único valor que da un elemento de RoS_9 es 1

y por lo tanto resulta que $x^2(9) = 1 \quad \Rightarrow \quad x^2 = 1 + 9t$

(realmente $x^2 = 64009$ y entonces $x^2 = 1 + 9t$ con $t = 7112$)

Más formalmente la idea es:

Para cualquier valor de m debe cumplirse que

$x^2(m) \in RoS_m$ pero de $n + x^2 = y^2$ resulta que debe ser

$(n + x^2)(m) \in ROS_m$ para obtener un cuadrado perfecto (y^2) .

Esto conduce a la siguiente definición:

Dado n diremos que $a \in RoS_c$ es un target para c

sí y sólo si $(n + a)(c) \in RoS_c$

Julia Picabea demostró que ciertos valores de m SIEMPRE

dan targets ÚNICOS para n impar.

Otro ejemplo:

$n = 2851$

$RoS_4 = \{0,1\},\ n(4) = 1$

$(n + x^2)(4) = (1 + \{0,1\})(4)$

o sea que $x^2(4) = 0$ *es el único valor posible.*

De hecho es $x^2 = 900$ *o sea que* $x^2(4) = 0$

Un nuevo resultado (debido al Dr. Eduardo Gil Moré):

El valor máximo de *m* que puede dar un target único es 1440

Nota: este resultado lo conocíamos experimentalmente. Curiosamente el enfoque utilizado para demostrarlo es similar al de Julia Picabea para demostrar que valores como $m = 3,4,8$ siempre dan targets únicos

Nota:

si $x^2 \equiv a(c) \Rightarrow x^2 = a + c.t$ para un cierto valor de t y

si $b = (n + x^2)(\ c) \in ROS_c \Rightarrow y^2 = b + c.u$ para un cierto valor de u resulta que

$$n + a + c.t = b + c.u \Rightarrow \Delta = \frac{n + a - b}{c} = u - t$$

La idea clave es usar a este último número como el nuevo número a factorizar

Enseguida veremos el cálculo de targets para algunos ejemplos

Un ejemplo más grande (RSA640):

3107418240490043721350750035888567930037346022842727545
7201619488232064405180815045563468296717232867824379162
7283803341547107310850191954852900733772482278352574238
6454014691736602477652346609

Algunos targets únicos:

$x^2 = 112+144.t, \quad y^2 = 81+144.u$ pero

$x^2 =$

1770731030469776370498858840496639153646058316918061728 3
1022179254224303389908271437986741603330808488012674224 7
1705314296463491272223064604055462199900979227120838917 7
8146715554754615318401 6

RSA 1024

135066410865995223349603216278805969938814756056670275244851438515265106048595338339402871505719094417982072821644715513736804197039641917430464965892742562393410208643832021103729587257623585096431105640735015081875106765946292055636855294752135008528794163773285339061097505443349998111500569772368909275 63

Algunos targets únicos para RSA1024

(57, 4, 96)
(81, 4, 120)
(9, 100, 144)
(121, 4, 160)
(81, 64, 180)
(201, 4, 240)
(153, 100, 288)
(81, 244, 360)
(441, 4, 480)
(441, 244, 720)
(441, 964, 1440) (máximo que la teoría indica)

Una conclusión:

$$rsa\ 1024 + (24z + 3)^2 = (36u + 10)^2$$

Volvamos al ejemplo:

$n = 507527 \quad x^2 = 64009 \quad y^2 = 571536$

a	b	c	*Delta = (n+a-b)/c*
1	0	3	169176
1	0	4	126882
4	1	5	105506
1	0	8	63441
89	216	360	1410
649	576	720	705

Comenzando con (649,576,720) y

delta =705, la siguiente iteración conduce a

$x^2 = 3529 + 30240t$ que da x^2 con $t = 2$

Entonces queda claro que hay que iterar

Sea (a_1, b_1, c_1) un target único.

Calculamos $\Delta_1 = \dfrac{n + a_1 - b_1}{c_1}$

y, si es impar (se puede y se debe usar deltas pares)

, un target único (a_2, b_2, c_2)

Un poco de álgebra indica como fórmulas posibles:

$x^2 = a_1 + c_1 a_2 + c_1 c_2 t_2$

$y^2 = b_1 + c_1 b_2 + c_1 c_2 u_2$

que a veces funcionan y a veces no

(luego veremos como "filtrarlas" ó "corregirlas")

Tn(649, 576, 720): D = 705

TD(0, 1, 4): (aa, bb, cc) = (649, 1296, 2880) -> OK <-.
TD(0, 1, 8): (aa, bb, cc) = (649, 1296, 5760) -> OK <-.
TD(0, 1, 16): (aa, bb, cc) = (649, 1296, 11520).

Las fórmulas anteriores dieron bien en los dos primeros casos pero no en el último.

Lo correcto sería (6409,7056,11520). Ahora,
6409 – 649 = 5760 = 7056 – 1296 = 8.720=8c_1

Esto no es casualidad, es un teorema

Resumiendo:

Dado un target único (a_1, b_1, c_1) calculamos

$\Delta_1 = \dfrac{n + a_1 - b_1}{c_1}$ y si es impar, otro TU (a_2, b_2, c_2)

La fórmula $x^2 = a_1 + a_2 c_1 + c_1 c_2 t_2$ puede ser o no correcta

Alternativas:

1) filtrarla

2) corregirla

Filtrando

Filtros de “primera generación”:

Tn(169, 96, 240): D = 2115

TD(1, 0, 4): (aa, bb, cc) = (409, 96, 960) F2 F3b F6.

TD(1, 4, 8): (aa, bb, cc) = (409, 1056, 1920) F2 F3b F6.

TD(1, 4, 16): (aa, bb, cc) = (409, 1056, 3840) F2 F3b F6.

TD(1, 4, 32): (aa, bb, cc) = (409, 1056, 7680) F2 F3b F6.

Los filtros eliminaron todos los casos, pero son “corregibles”.

Por ejemplo, el último debería ser (2569, 3216,7680)

Nuevamente si planteamos que $x^2 = 409 + \delta_x^2 + c_1 c_2 t$

δ_x^2 =2569 - 409 = 2160 = 9.240 = $\lambda_x^2 c_1$

Luego veremos que esto se deduce

Un nuevo resultado:

Todos los filtros anteriores pueden reducirse al problema de determinar si, dado el "candidato" (aa, bb, cc), se cumple que

$aa \in RoS_{cc}$, $bb \in RoS_{cc}$, $(n+aa)(cc) \in RoS_{cc}$

algo que ahora podemos hacer pues tenemos un algoritmo para decidir el problema de la residuosidad cuadrática.

Veremos ahora el último algoritmo que utiliza targets de orden 1,2,3,4,5 para filtrar y corregir, y su aplicación a RSA640


Finalizado el tiempo disponible…

Aquí terminamos

Muchas gracias !

E-MAIL

hscolnik@gmail.com