Small Roots of Modular Equations

James Davenport
(with thanks to Nick Howgrave-Graham, ex-Bath/NTL, and Paul Crouch, ex-Bath now BT)

University of Bath
(visiting Waterloo)

2 July 2009

Small roots $|x_0| < X$ of $p(x) = x^k + a_{k-1}x^{k-1} + \cdots a_0 \pmod{N}$ [HG97]

Pick a parameter $h \geq 2$.

Small roots $|x_0| < X$ of $p(x) = x^k + a_{k-1}x^{k-1} + \cdots a_0 \pmod{N}$ [HG97]

Pick a parameter $h \geq 2$.
Let $q_i = q_{u,v} = N^{h-1-v}x^u(p(x))^v$ where $v = \lfloor (i-1)/k \rfloor$ and u is the remainder. $q_{u,v}(x_0) \equiv 0 \pmod{N^{h-1}}$.

Small roots $|x_0| < X$ of $p(x) = x^k + a_{k-1}x^{k-1} + \cdots a_0 \pmod N$ [HG97]

Pick a parameter $h \geq 2$.
Let $q_i = q_{u,v} = N^{h-1-v}x^u(p(x))^v$ where $v = \lfloor (i-1)/k \rfloor$ and u is the remainder. $q_{u,v}(x_0) \equiv 0 \pmod{N^{h-1}}$.
Let M be the lower triangular $hk \times hk$ matrix $m_{i,j} = e_{i,j}X^{j-1}$ where $e_{i,j}$ is the coefficient of x^{j-1} in q_i.
$\mathrm{Det}(M) = X^{hk(hk-1)/2}N^{hk(h-1)/2}$.

Small roots $|x_0| < X$ of $p(x) = x^k + a_{k-1}x^{k-1} + \cdots a_0 \pmod{N}$ [HG97]

Pick a parameter $h \geq 2$.
Let $q_i = q_{u,v} = N^{h-1-v} x^u (p(x))^v$ where $v = \lfloor (i-1)/k \rfloor$ and u is the remainder. $q_{u,v}(x_0) \equiv 0 \pmod{N^{h-1}}$.
Let M be the lower triangular $hk \times hk$ matrix $m_{i,j} = e_{i,j} X^{j-1}$ where $e_{i,j}$ is the coefficient of x^{j-1} in q_i.
$\mathrm{Det}(M) = X^{hk(hk-1)/2} N^{hk(h-1)/2}$.
Write the LLL-"shortest" vector as $r(xX)$, then $r(x_0) \equiv 0 \pmod{N^{h-1}}$ and

$$|x| \leq X \Rightarrow |r(x)| \leq \left(2^{\frac{hk-1}{4}} \sqrt{hk}\right) X^{\frac{hk-1}{2}} N^{\frac{h-1}{2}}.$$

Small roots $|x_0| < X$ of $p(x) = x^k + a_{k-1}x^{k-1} + \cdots a_0 \pmod{N}$ [HG97]

Pick a parameter $h \geq 2$.
Let $q_i = q_{u,v} = N^{h-1-v}x^u(p(x))^v$ where $v = \lfloor (i-1)/k \rfloor$ and u is the remainder. $q_{u,v}(x_0) \equiv 0 \pmod{N^{h-1}}$.
Let M be the lower triangular $hk \times hk$ matrix $m_{i,j} = e_{i,j}X^{j-1}$ where $e_{i,j}$ is the coefficient of x^{j-1} in q_i.
$\mathrm{Det}(M) = X^{hk(hk-1)/2}N^{hk(h-1)/2}$.
Write the LLL-"shortest" vector as $r(xX)$, then $r(x_0) \equiv 0 \pmod{N^{h-1}}$ and

$$|x| \leq X \Rightarrow |r(x)| \leq \left(2^{\frac{hk-1}{4}}\sqrt{hk}\right) X^{\frac{hk-1}{2}} N^{\frac{h-1}{2}}.$$

$$X = \left\lceil \left(2^{-1/2}(hk)^{-1/(hk-1)}\right) N^{(h-1)/(hk-1)} \right\rceil - 1$$

means that $r(x) < N^{h-1}$ for $|x| \leq X$. So $r(x_0) = 0$.

How does this help?

I started out with a polynomial of degree k, often with small coefficients, and now I have one of degree hk with larger coefficients!

How does this help?

I started out with a polynomial of degree k, often with small coefficients, and now I have one of degree hk with larger coefficients!
But the root is now over the integers. This gives us two strategies.

How does this help?

I started out with a polynomial of degree k, often with small coefficients, and now I have one of degree hk with larger coefficients!

But the root is now over the integers. This gives us two strategies.

- Integer roots correspond to factors of the trailing coefficients (often cheap in practice).

How does this help?

I started out with a polynomial of degree k, often with small coefficients, and now I have one of degree hk with larger coefficients!

But the root is now over the integers. This gives us two strategies.

- Integer roots correspond to factors of the trailing coefficients (often cheap in practice).
- Factor modulo some small p and Hensel lift the linear factors (guaranteed non-dominant complexity).

How does this help?

I started out with a polynomial of degree k, often with small coefficients, and now I have one of degree hk with larger coefficients!

But the root is now over the integers. This gives us two strategies.

- Integer roots correspond to factors of the trailing coefficients (often cheap in practice).
- Factor modulo some small p and Hensel lift the linear factors (guaranteed non-dominant complexity).

How does this help?

I started out with a polynomial of degree k, often with small coefficients, and now I have one of degree hk with larger coefficients!

But the root is now over the integers. This gives us two strategies.

- Integer roots correspond to factors of the trailing coefficients (often cheap in practice).
- Factor modulo some small p and Hensel lift the linear factors (guaranteed non-dominant complexity).

As $h \to \infty$, $X \to 2^{-1/2} N^{1/k}$.

Complexity

(LLL) $O(h^9 k^6 \log^3 N)$.

Complexity

(LLL) $O(h^9 k^6 \log^3 N)$.
(LL) $O(h^6 k^4 \log^2 N)$ (I think!).

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1 \pmod{\phi(N) = (p-1)(q-1) = N - p - q + 1}$.

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1 \pmod{\phi(N) = (p-1)(q-1) = N - p - q + 1}$.
Let $s = p + q - 1$, so $\phi(N) = N - s$. $N - s$ is therefore the "approximate gcd" of $U = de - 1$ and "approximately N". [HG01]

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1 \pmod{\phi(N) = (p-1)(q-1) = N - p - q + 1}$.
Let $s = p + q - 1$, so $\phi(N) = N - s$. $N - s$ is therefore the "approximate gcd" of $U = de - 1$ and "approximately N". [HG01]
Write $s = s_0 X + x_0$, where $0 < x_0 < X$, where we will guess s_0.

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1 \pmod{\phi(N) = (p-1)(q-1) = N - p - q + 1}$.
Let $s = p + q - 1$, so $\phi(N) = N - s$. $N - s$ is therefore the "approximate gcd" of $U = de - 1$ and "approximately N". [HG01]
Write $s = s_0 X + x_0$, where $0 < x_0 < X$, where we will guess s_0.
Take polynomials $g_{ij}(x) = x^i (x - N + s_0 X)^j \cdot U^{m-j}$, so $g_{ij}(x_0) \equiv 0 \pmod{\phi(N)^m}$: a congruence to an *unknown* modulus.

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1 \pmod{\phi(N) = (p-1)(q-1) = N - p - q + 1}$.
Let $s = p + q - 1$, so $\phi(N) = N - s$. $N - s$ is therefore the "approximate gcd" of $U = de - 1$ and "approximately N". [HG01]
Write $s = s_0 X + x_0$, where $0 < x_0 < X$, where we will guess s_0.
Take polynomials $g_{ij}(x) = x^i(x - N + s_0 X)^j \cdot U^{m-j}$, so $g_{ij}(x_0) \equiv 0 \pmod{\phi(N)^m}$: a congruence to an *unknown* modulus.
However, if $h(xX)$ is the LLL-"smallest" vector in the corresponding lattice, and $||h(xX||$ is small enough ($< \phi(N)^m/2m$), we then have $h(x_0) = 0$.

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1$ $(\text{mod } \phi(N) = (p-1)(q-1) = N - p - q + 1)$.
Let $s = p + q - 1$, so $\phi(N) = N - s$. $N - s$ is therefore the "approximate gcd" of $U = de - 1$ and "approximately N". [HG01]
Write $s = s_0 X + x_0$, where $0 < x_0 < X$, where we will guess s_0.
Take polynomials $g_{ij}(x) = x^i (x - N + s_0 X)^j \cdot U^{m-j}$, so $g_{ij}(x_0) \equiv 0$ $(\text{mod } \phi(N)^m)$: a congruence to an *unknown* modulus.
However, if $h(xX)$ is the LLL-"smallest" vector in the corresponding lattice, and $||h(xX||$ is small enough $(< \phi(N)^m/2m)$, we then have $h(x_0) = 0$.
Complexity (LLL) $O(\log^9 N)$.

Application to RSA: balanced p, q [CM07]

Suppose we can break RSA $x \mapsto x^d \pmod{N = pq}$ *by exponentiation*, i.e. we know e such that $\left(x^d\right)^e \equiv x$, i.e. $de \equiv 1$ $(\text{mod } \phi(N) = (p-1)(q-1) = N - p - q + 1)$.
Let $s = p + q - 1$, so $\phi(N) = N - s$. $N - s$ is therefore the "approximate gcd" of $U = de - 1$ and "approximately N". [HG01]
Write $s = s_0 X + x_0$, where $0 < x_0 < X$, where we will guess s_0.
Take polynomials $g_{ij}(x) = x^i (x - N + s_0 X)^j \cdot U^{m-j}$, so $g_{ij}(x_0) \equiv 0$ $(\text{mod } \phi(N)^m)$: a congruence to an *unknown* modulus.
However, if $h(xX)$ is the LLL-"smallest" vector in the corresponding lattice, and $||h(xX||$ is small enough $(< \phi(N)^m / 2m)$, we then have $h(x_0) = 0$.
Complexity (LLL) $O(\log^9 N)$.
Complexity (LL) $O(\log^7 N)$ (I think!).

The general case

Write U for $de - 1$, ϕ for $\phi(N)$. Consider the polynomials

$$g_{ijk}(x, y) = x^i y^j U^{m-k}(x + y - (N + 1))^k \big|_{xy \mapsto N}$$

- No mixed monomials
- All $\equiv 0 \mod \phi^m$ when $(x, y) = (p, q)$.

Which equations?

$$g_{ijk}(x,y) = x^i y^j U^{m-k}(x+y-(N+1))^k\big|_{xy\mapsto N}$$

$m+1$ $i=0$, $j=0$, $0 \le k \le m$

$m+1$ $i=1$, $j=0$, $0 \le k \le m$

$a-1$ $1 < i \le a$, $j=0$, $k=m$

b $i=0$, $1 \le j \le b$, $k=m$

2m+a+b+1 total.

Given that there are no mixed monomials, we have $(a+m)+(b+m)+1$ monomials.

Assume high-order components known

Assume we know $p = p_0 X + x$, $q = q_0 Y + y$ with $0 \leq x < X$, $0 \leq y < Y$.

Assume high-order components known

Assume we know $p = p_0 X + x$, $q = q_0 Y + y$ with $0 \leq x < X$, $0 \leq y < Y$. Then write $t_{ijk}(x, y) = g_{ijk}(p_0 X + x, q_0 Y + y)$. Consider the lattice of coefficients of $t_{ijk}(xX, yY)$: e.g. ($m = 3$, $a = 2$, $b = 1$)

$$\begin{array}{c} t_{000} \\ t_{100} \\ t_{001} \\ t_{101} \\ t_{002} \\ t_{102} \\ t_{003} \\ t_{103} \\ t_{203} \\ t_{013} \end{array} \begin{pmatrix} U^3 & & & & & & & & & \\ * & U^3X & & & & & & & & \\ * & * & U^2Y & & & & & & & \\ * & * & * & U^2X^2 & & & & & & \\ * & * & * & * & UY^2 & & & & & \\ * & * & * & * & * & UX^3 & & & & \\ * & * & * & * & * & * & Y^3 & & & \\ * & * & * & * & * & * & * & X^4 & & \\ * & * & * & * & * & * & * & * & X^5 & \\ * & * & * & * & * & * & * & * & * & Y^4 \end{pmatrix}$$

Recovery?

As in the univariate case, if $h(x_0, y_0) \equiv 0 \pmod{\phi}^m$ and $||h(xX, yY|| < \phi^m/\sqrt{w}$ where h has w monomials, then $h(x_0, y_0) = 0$ exactly.

Recovery?

As in the univariate case, if $h(x_0, y_0) \equiv 0 \pmod{\phi}^m$ and $||h(xX, yY|| < \phi^m/\sqrt{w}$ where h has w monomials, then $h(x_0, y_0) = 0$ exactly.
In our case, $w = 2m + a + b + 1$ (no mixed monomials!).

Recovery?

As in the univariate case, if $h(x_0, y_0) \equiv 0 \pmod{\phi}^m$ and $||h(xX, yY|| < \phi^m/\sqrt{w}$ where h has w monomials, then $h(x_0, y_0) = 0$ exactly.
In our case, $w = 2m + a + b + 1$ (no mixed monomials!).
Complexity (LLL) $O(\log^{12} N)$.

Recovery?

As in the univariate case, if $h(x_0, y_0) \equiv 0 \pmod{\phi)^m}$ and $||h(xX, yY|| < \phi^m/\sqrt{w}$ where h has w monomials, then $h(x_0, y_0) = 0$ exactly.
In our case, $w = 2m + a + b + 1$ (no mixed monomials!).
Complexity (LLL) $O(\log^{12} N)$.
Complexity (LL) $O(\log^9 N)$ (I think!).

Recovery?

As in the univariate case, if $h(x_0, y_0) \equiv 0 \pmod{\phi}^m$ and $||h(xX, yY|| < \phi^m/\sqrt{w}$ where h has w monomials, then $h(x_0, y_0) = 0$ exactly.
In our case, $w = 2m + a + b + 1$ (no mixed monomials!).
Complexity (LLL) $O(\log^{12} N)$.
Complexity (LL) $O(\log^9 N)$ (I think!).
Roughly speaking the difference in complexity comes from the fact that we are doing 2-D guessing for p and q.

Unbalanced e, d: $ed < N^{3/2}$

Just to remind you that this is trivial (no lattices)

Unbalanced e, d: $ed < N^{3/2}$

Just to remind you that this is trivial (no lattices)
Suppose $ed = 1 + k\phi(N)$, and approximate k by $k' = (ed - 1)/N$.
Then $0 \leq k - k' \leq 6$, and we test all: $O(\log^2 N)$.

A different scenario

Let's *not* assume we know the secret exponent.

A different scenario

Let's *not* assume we know the secret exponent.
Let's consider the case of IP (an industry standard) packets being transmitted with low-exponent RSA encryption.

A different scenario

Let's *not* assume we know the secret exponent.
Let's consider the case of IP (an industry standard) packets being transmitted with low-exponent RSA encryption.
Our aim now is to recover *individual messages* rather than break the key as such.

IP

Figure: IP datagram, showing the fields in the IP header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Checksum $= -\sum w_i \pmod{65535}$: w_i the 16-bit words in the header.

Low-exponent RSA-encrypted IP [CD01]

Assume an IP packet m is sent as $m^d \pmod N$ for some small exponent d. If we can, e.g. denial of service, get two transmissions, where the `identification` differs by c, we have $m^d \pmod N$ and

$$\left(m + (2^{48} - 1)c2^{72}\right)^d \pmod N$$

(assuming no checksum wrapping).

Low-exponent RSA-encrypted IP [CD01]

Assume an IP packet m is sent as $m^d \pmod{N}$ for some small exponent d. If we can, e.g. denial of service, get two transmissions, where the `identification` differs by c, we have $m^d \pmod{N}$ and

$$\left(m + (2^{48} - 1)c2^{72}\right)^d \pmod{N}$$

(assuming no checksum wrapping).
Eliminating m gives a degree d^2 equation in c, where $c < 2^{16}$. In fact, this is a degree d equation in c^d.

Low-exponent RSA-encrypted IP [CD01]

Assume an IP packet m is sent as $m^d \pmod{N}$ for some small exponent d. If we can, e.g. denial of service, get two transmissions, where the `identification` differs by c, we have $m^d \pmod{N}$ and

$$\left(m + (2^{48} - 1)c2^{72}\right)^d \pmod{N}$$

(assuming no checksum wrapping).
Eliminating m gives a degree d^2 equation in c, where $c < 2^{16}$. In fact, this is a degree d equation in c^d.
Checksum wrapping gives us

$$\left(m + \left((2^{48} - 1)c - 1\right) 2^{72}\right)^d \pmod{N}$$

again a degree d^2 equation in c, but this doesn't collapse.

Timings from [CD01]

NTL Timings in seconds to lattice reduce RedHat Linux 6.2 on 1Ghz Pentium III with 500Mb RAM							
Public exponent		e=3			e=5		
RSA-type	wrapping	512	1024	2048	512	1024	2048
IP	Without	2	9	27	8068**	177	1386
	With	653	3413	3976	†	793465	§

† Not implemented due to software restrictions.

** Taking $\alpha \leq 2^{11}$ allowed $h = 2$, with $e = 5$ this formed a 10x10 matrix which reduced in 19 seconds.

Since $19 \times 32 \ll 8068$, this illustrates the power of guessing high-order bits.

Timings from [CD01]

NTL Timings in seconds to lattice reduce RedHat Linux 6.2 on 1Ghz Pentium III with 500Mb RAM							
Public exponent		e=3			e=5		
RSA-type	wrapping	512	1024	2048	512	1024	2048
IP	Without	2	9	27	8068**	177	1386
	With	653	3413	3976	†	793465	§

† Not implemented due to software restrictions.

** Taking $\alpha \leq 2^{11}$ allowed $h = 2$, with $e = 5$ this formed a 10x10 matrix which reduced in 19 seconds.

Since $19 \times 32 \ll 8068$, this illustrates the power of guessing high-order bits.

Once we have c, we recover m by a resultant calculation.

References

P.A. Crouch and J.H. Davenport.
Lattice Attacks on RSA-Encrypted IP and TCP.
In B. Honary, editor, *Proceedings 8th. IMA Conf. Cryptography and Coding*, pages 329–338, 2001.

J.-S. Coron and A. May.
Deterministic Polynomial-Time Equivalence of Computing the RSA Secret Key and Factoring.
J. Cryptology, 20:39–50, 2007.

N.A. Howgrave-Graham.
Finding Small Roots of Univariate Modular Equations Revisited.
Cryptography and Coding (Ed. M. Darnell), pages 131–142, 1997.

N.A. Howgrave-Graham.
Approximate Integer Common Divisors.
In J.H. Silverman, editor, *Proceedings CaLC 2001*, pages 51–66, 2001.