TECHNISCHE UNIVERSITÄT DARMSTADT

New Cryptanalytic Results for RSA with Small Secret Exponents

Alexander May

Department of Computer Science

TU Darmstadt

Workshop on Crypto: Underlying Mathematics, Provability and Foundations

Outline of the talk

TECHNISCHE UNIVERSITÄT DARMSTADT

- Solve polynomial equations via lattices
  - Univariate modular case
  - Generalize: Multivariate and integer case
- Overview of known RSA results
  - e-th roots
  - Small RSA secret key
  - Factoring
- RSA with Small CRT exponents d_p, d_q
  - Attacks for Small e
  - Known difference: $d_p, d_q \leq N^{0.099}$
  - General case: $d_p, d_q \leq N^{0.073}$

A word about lattices

Let $v_1, \ldots, v_n \in \mathbb{Z}^n$ be linearly independent vectors.

$$L := \left\{ x \in \mathbb{Z}^n \mid x = \sum_{i=1}^{n} a_i v_i \text{ with } a_i \in \mathbb{Z}. \right\}$$

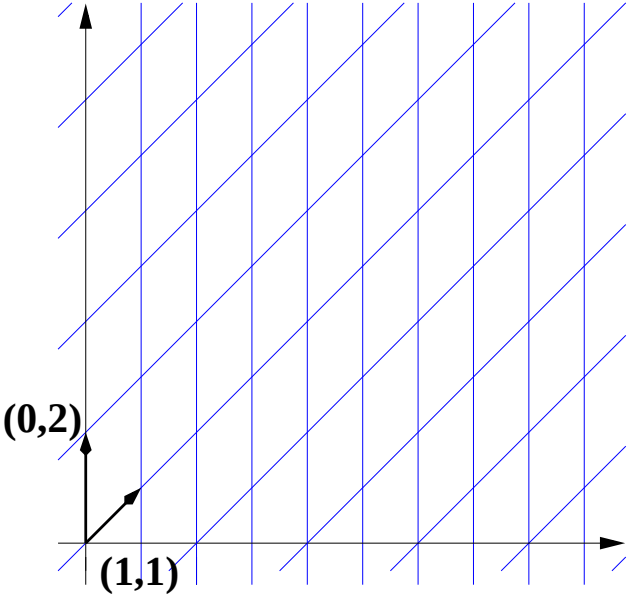


Fig.: Lattice spanned by $\{(0,2),(1,1)\}$

A word about lattices

Let $v_1, \ldots, v_n \in \mathbb{Z}^n$ be linearly independent vectors.

$$L := \left\{ x \in \mathbb{Z}^n \mid x = \sum_{i=1}^{n} a_i v_i \text{ with } a_i \in \mathbb{Z}. \right\}$$

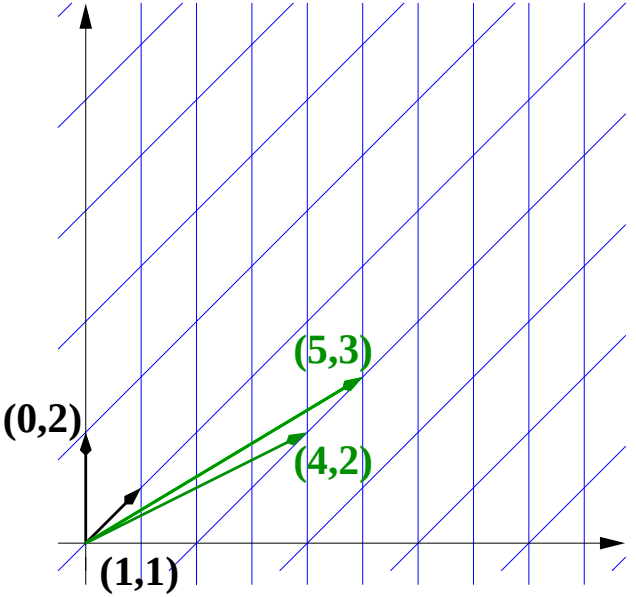


Fig.: Lattice spanned by $\{(0,2),(1,1)\}$

A word about lattices

Let $v_1, \ldots, v_n \in \mathbb{Z}^n$ be linearly independent vectors.

$$L := \left\{ x \in \mathbb{Z}^n \mid x = \sum_{i=1}^{n} a_i v_i \text{ with } a_i \in \mathbb{Z}. \right\}$$

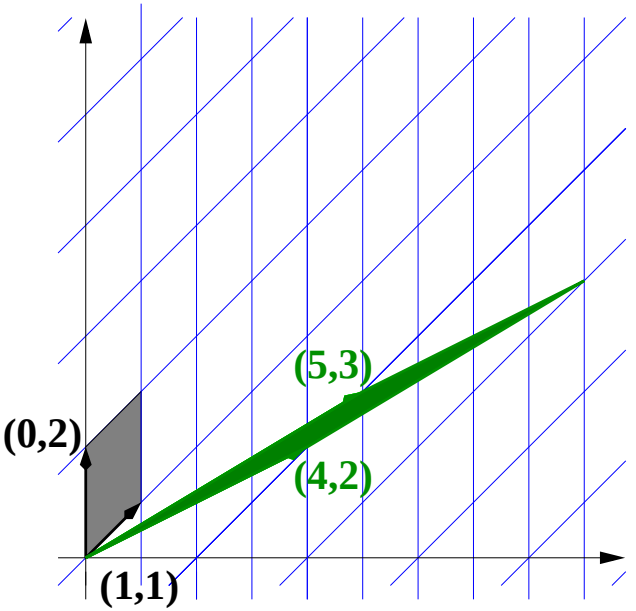


Fig.: Lattice spanned by $\{(0,2),(1,1)\}$

Lagrange- and L^3-algorithm

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem[Lagrange]: *Let L be a two-dimensional lattice. Then one can find in polynomial time a shortest vector $v \neq 0$ of L with*

$$\|v\| \leq \sqrt{2\det(L)}.$$

Lagrange- and L^3-algorithm

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem[Lagrange]: *Let L be a two-dimensional lattice. Then one can find in polynomial time a shortest vector $v \neq 0$ of L with*

$$\|v\| \leq \sqrt{2\det(L)}.$$

Theorem[Lenstra, Lenstra, Lovász]: *Let L be a lattice spanned by $v_1, \ldots, v_n$. Then one can find in polynomial time a basis $b_1, \ldots, b_n$ of L with*

$$\|b_1\| \leq 2^{\frac{n-1}{4}} \det(L)^{\frac{1}{n}}.$$

Coppersmith: modular case

TECHNISCHE UNIVERSITÄT DARMSTADT

- Let M be an integer of unknown factorization.
- Let b be a divisor of M (special case: $b = M$).

Problem: Given $f_b(x)$, find all solutions $|x_0| \leq X$ of

$$f_b(x) = 0 \mod b.$$

Coppersmith: modular case

TECHNISCHE UNIVERSITÄT DARMSTADT

- Let M be an integer of unknown factorization.
- Let b be a divisor of M (special case: $b = M$).

Problem: Given $f_b(x)$, find all solutions $|x_0| \leq X$ of

$$f_b(x) = 0 \mod b.$$

Idea:

- Collection of polynomials $f_1(x), f_2(x), \ldots, f_n(x)$ with the roots $|x_0| \leq X$ modulo b^m.
- Construct $f(x) = \sum_{i=1}^{n} a_i f_i$, $a_i \in \mathbb{Z}$ such that

$$f(x_0) = 0 \text{ over } \mathbb{Z}.$$

  Sufficient condition: $|f(x_0)| < b^m$.
- Solve $f(x)$ over the integers.

Howgrave Graham's theorem

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem: *Let* $f(x) \in \mathbb{Z}[x]$ *be a polynomial of degree* $n-1$ *with*

- $f(x_0) = 0 \bmod b^m$, *where* $|x_0| \leq X$
- $||f(xX)|| < \frac{b^m}{\sqrt{n}}$

Then $f(x_0) = 0$ *over the integers.*

Howgrave Graham's theorem

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem: *Let* $f(x) \in \mathbb{Z}[x]$ *be a polynomial of degree* $n-1$ *with*

- $f(x_0) = 0 \bmod b^m$, *where* $|x_0| \leq X$
- $||f(xX)|| < \frac{b^m}{\sqrt{n}}$

Then $f(x_0) = 0$ *over the integers.*

Example:

$$\begin{aligned} f(x) &= ax^2 + bx + c \\ f(xX) &= aX^2x^2 + bXx + c \\ ||f(xX)|| &= ||(aX^2, bX, c)|| \end{aligned}$$

Howgrave Graham's theorem

TECHNISCHE UNIVERSITÄT DARMSTADT

Theorem: *Let* $f(x) \in \mathbb{Z}[x]$ *be a polynomial of degree* $n-1$ *with*

- $f(x_0) = 0 \bmod b^m$, *where* $|x_0| \leq X$
- $||f(xX)|| < \frac{b^m}{\sqrt{n}}$

Then $f(x_0) = 0$ *over the integers.*

Proof:

$$
\begin{aligned}
|f(x_0)| &= \left| \sum_{i=0}^{n-1} a_i x_0^i \right| \leq \sum_{i=0}^{n-1} \left| a_i X^i \left(\frac{x_0}{X} \right)^i \right| \\
&\leq \sum_{i=0}^{n-1} |a_i X^i| \leq \sqrt{n} \cdot ||f(xX)|| < b^m
\end{aligned}
$$

Finding an Inverse $a^{-1} \bmod b$

TECHNISCHE UNIVERSITÄT DARMSTADT

Goal: Solve $ax = 1 \bmod b$, where $\gcd(a, b) = 1$.

Collection of polynomials:

- $f_1(x) = ax - 1$ with root $x_0 = a^{-1} \bmod b$ of size smaller than $X = b$.
- $f_2(x) = bx$ with same root $x_0 \bmod b$.
- Coeff-vector $f_1(xX), f_2(xX)$: $(aX, -1), (bX, 0)$

Finding an Inverse $a^{-1} \bmod b$

TECHNISCHE UNIVERSITÄT DARMSTADT

Goal: Solve $ax = 1 \bmod b$, where $\gcd(a, b) = 1$.

Collection of polynomials:

- $f_1(x) = ax - 1$ with root $x_0 = a^{-1} \bmod b$ of size smaller than $X = b$.
- $f_2(x) = bx$ with same root $x_0 \bmod b$.
- Coeff-vector $f_1(xX), f_2(xX)$: $(aX, -1), (bX, 0)$

Construct lattice L spanned by

$$B = \begin{bmatrix} bX & 0 \\ aX & -1 \end{bmatrix}$$

with $\det(L) = bX = b^2$.

Extracting the inverse

TECHNISCHE UNIVERSITÄT DARMSTADT

Lagrange Alg: vector $v = (c_0, c_1) \cdot \begin{bmatrix} bX & 0 \\ aX & -1 \end{bmatrix}$

with $\|v\| \leq \sqrt{2\det(L)} \approx b$.

v corresponds to

$$f(x) = c_0 bx + c_1(ax - 1)$$

Extracting the inverse

TECHNISCHE UNIVERSITÄT DARMSTADT

Lagrange Alg: vector $v = (c_0, c_1) \cdot \begin{bmatrix} bX & 0 \\ aX & -1 \end{bmatrix}$

with $\|v\| \leq \sqrt{2\det(L)} \approx b$.

v corresponds to

$$f(x) = c_0 bx + c_1(ax - 1)$$

According to Howgrave-Graham's theorem

$$f(x_0) = c_0 bx_0 + c_1(ax_0 - 1) = 0 \text{ over } \mathbb{Z}(!)$$

$$\Rightarrow x_0 = \frac{c_1}{c_0 b + c_1 a}$$

Extensions to more variables

TECHNISCHE UNIVERSITÄT DARMSTADT

Goal: Find roots of $f_b(x_1, \ldots, x_n) \bmod b$.

- Compute $f_1, f_2, \ldots, f_n$ with small roots over $\mathbb{Z}$.
- Eliminate variables by resultant computations.

Integer case: Find roots of $f(x_1, \ldots, x_n)$ over $\mathbb{Z}$.

- Compute $f_1, f_2, \ldots, f_{n-1}$ with small roots over $\mathbb{Z}$ s.t. f does not divide f_i for $1 \leq i < n$.
- Eliminate variables by resultant computations.

Extensions to more variables

TECHNISCHE UNIVERSITÄT DARMSTADT

Goal: Find roots of $f_b(x_1, \ldots, x_n) \bmod b$.

- Compute $f_1, f_2, \ldots, f_n$ with small roots over $\mathbb{Z}$.
- Eliminate variables by resultant computations.

Integer case: Find roots of $f(x_1, \ldots, x_n)$ over $\mathbb{Z}$.

- Compute $f_1, f_2, \ldots, f_{n-1}$ with small roots over $\mathbb{Z}$ s.t. f does not divide f_i for $1 \leq i < n$.
- Eliminate variables by resultant computations.

Problems:

- Finding optimal collection.
- Elimination requires algebraically independent f_i.

RSA Problem

TECHNISCHE UNIVERSITÄT DARMSTADT

Given: $N = pq$, $e \in \mathbb{Z}^*_{\phi(N)}$ and $c = m^e \bmod N$

Find: $m \in \mathbb{Z}_N$

- **Relaxation 1: Small e, m**
  Ptime if $m < N^{\frac{1}{e}}$: Compute $c^{\frac{1}{e}}$.
- **Relaxation 2: Small e, parts of m known**
  C '96: Ptime if $m = \tilde{m} + x$, $x < N^{\frac{1}{e}}$:

$$f(x) = (\tilde{m} + x)^e - c \bmod N$$

RSA Problem

TECHNISCHE UNIVERSITÄT DARMSTADT

Given: $N = pq$, $e \in \mathbb{Z}^*_{\phi(N)}$ and $c = m^e \bmod N$

Find: $m \in \mathbb{Z}_N$

- **Relaxation 1: Small e, m**
  Ptime if $m < N^{\frac{1}{e}}$: Compute $c^{\frac{1}{e}}$.
- **Relaxation 2: Small e, parts of m known**
  C '96: Ptime if $m = \tilde{m} + x$, $x < N^{\frac{1}{e}}$:
  $$f(x) = (\tilde{m} + x)^e - c \bmod N$$
- **Relaxation 3: Small, splittable $m < 2^k$**
  BJN '00: $m = m_1 \cdot m_2$ with $m_1 \approx m_2 < 2^{\frac{k}{2}}$.
  Check: $x^e = y^{-e} \cdot c \bmod N$, $x, y = 0, \ldots, 2^{\frac{k}{2}}$

RSA Secret Key Problem

TECHNISCHE UNIVERSITÄT DARMSTADT

Given: $N = pq$, $e \in \mathbb{Z}^*_{\phi(N)}$

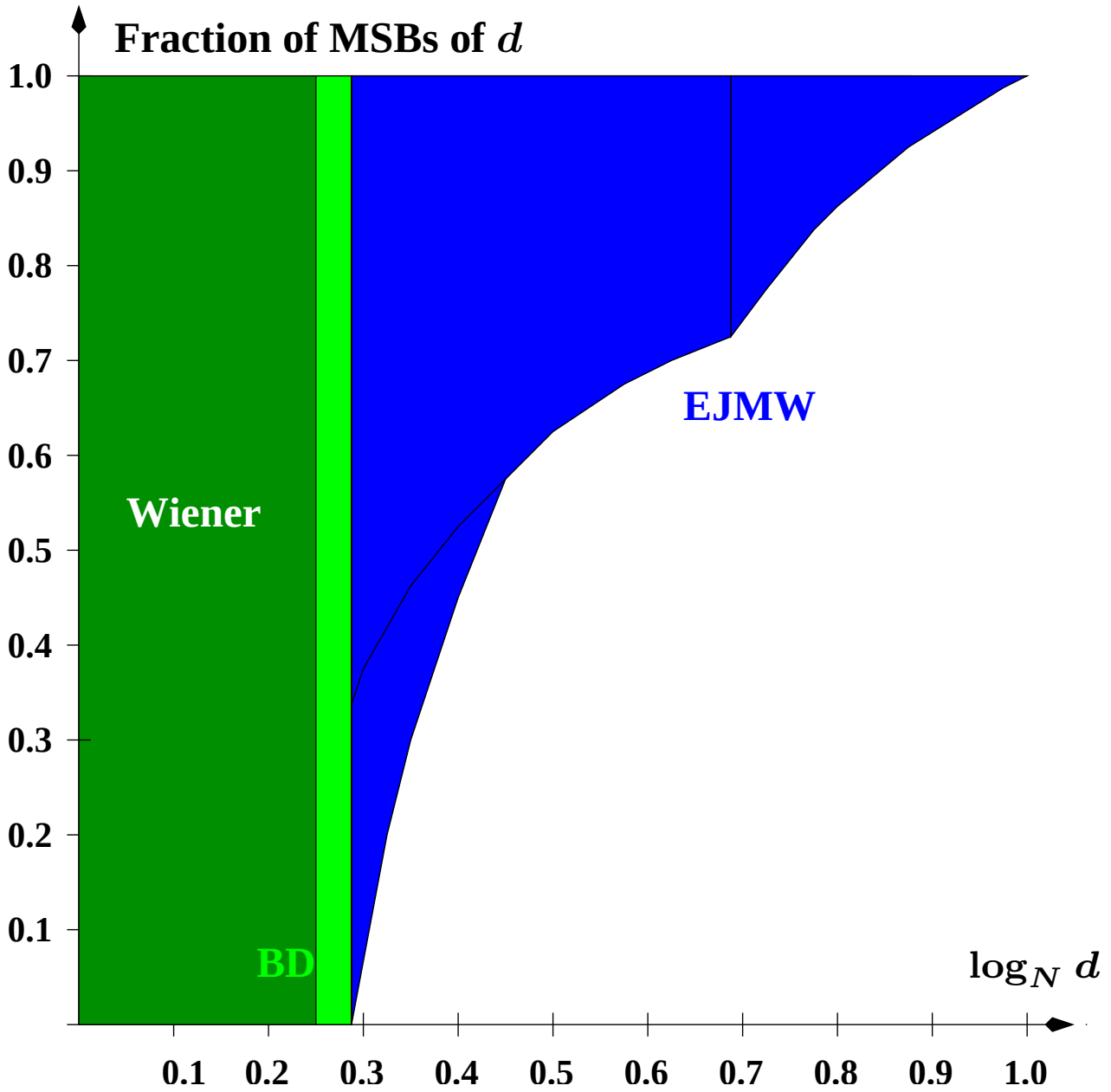
Find: d such that $ed = 1 \bmod \phi(N)$

$$f(d, k, p+q) = ed + k(N + 1 - (p+q)) - 1$$

- **Relaxation 1: Small d**
  Wiener '90, $\bmod N$: $d < N^{0.25}$
  BD '99, $\bmod e$: $d < N^{0.292}$
- **Relaxation 2: Known parts of d**
  EJMW '05: Extension to full range

Open: Use properties of d: Smoothness, splittable

RSA with Small Exponent d



Factoring Problem

TECHNISCHE UNIVERSITÄT DARMSTADT

Given: $N = pq$

Find: p

$$f(x) = x \bmod p$$

- **Relaxation 1: Known parts of p**
  C '96: $f(x) = \tilde{p} + x \bmod p$ with $x \leq N^{\frac{1}{4}}$.
- **Relaxation 2: "Small" p, q, i.e. $N = p^r q$**
  BDH '00: Solvable if $p = \tilde{p} + x$ with $x \leq N^{\frac{r}{(r+1)^2}}$.

Open Problems:

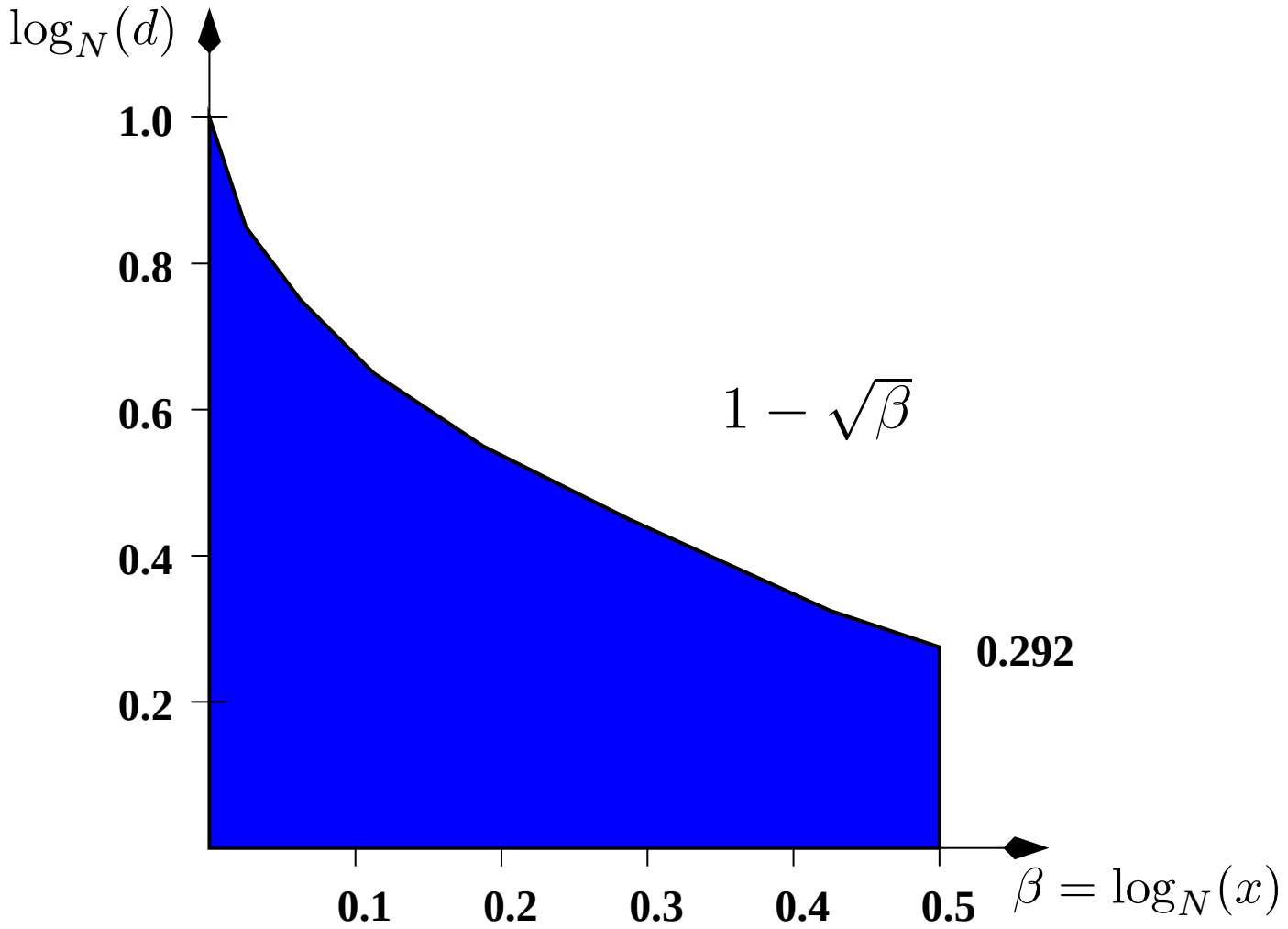
- Factor $N = p^r q^s$, $r \approx s$ with less bits.
- Factor $N = pqr$ with less bits.
- Factor using non-consecutive bits.

Combining relaxations

TECHNISCHE UNIVERSITÄT DARMSTADT

Small d and known bits of $p = \tilde{p} + x$.

$$f(k, (p+q)) = k(N + 1 - (p+q)) - 1 \bmod e$$

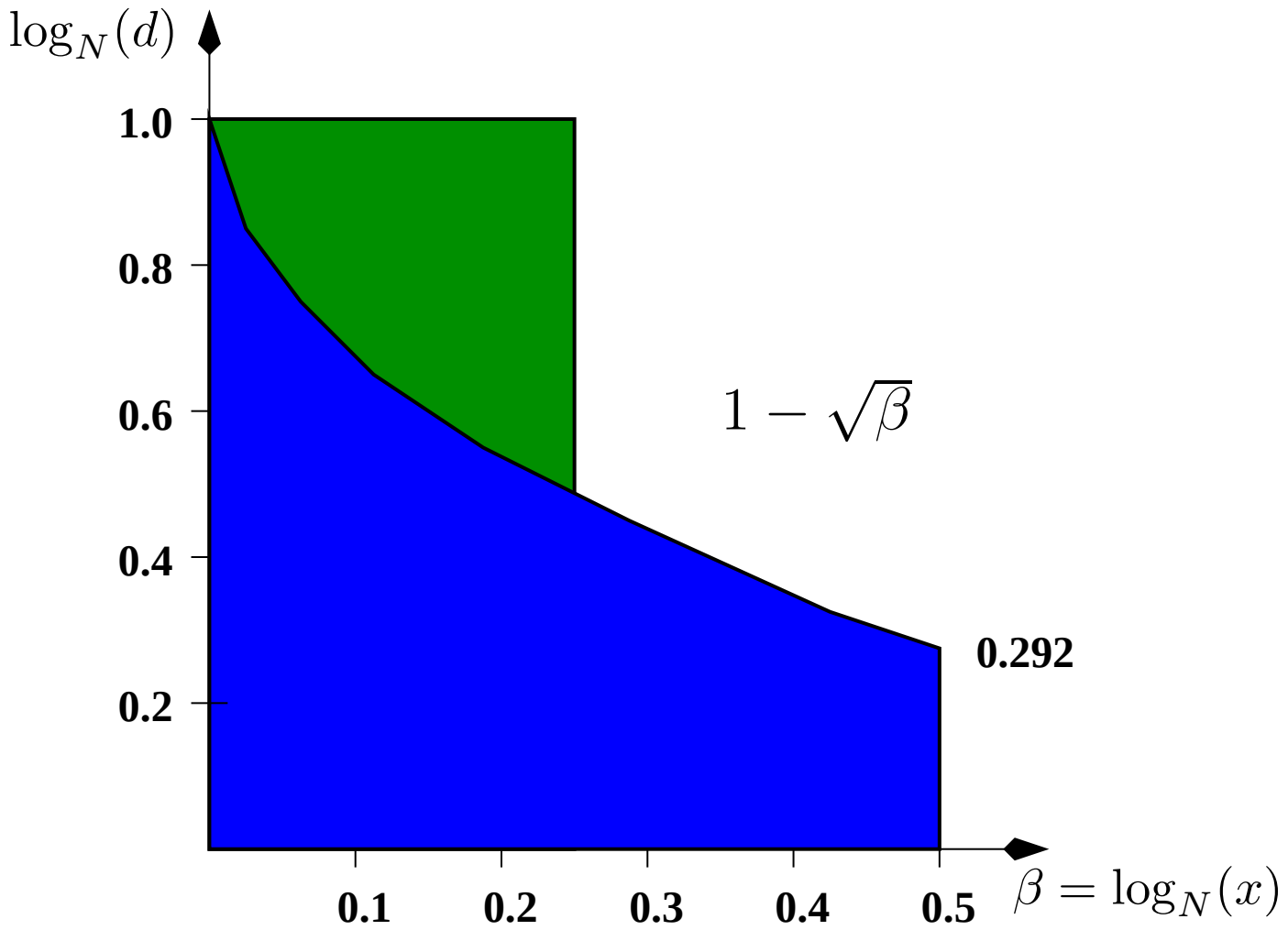


Combining relaxations

TECHNISCHE UNIVERSITÄT DARMSTADT

Small d and known bits of $p = \tilde{p} + x$.

$$f(k, (p+q)) = k(N + 1 - (p+q)) - 1 \bmod e$$



CRT-RSA problem

TECHNISCHE UNIVERSITÄT DARMSTADT

Setting: $d = (d_p \bmod p - 1, d_q \bmod q - 1)$ small

- **Relaxation 1: Imbalanced p, q**

  M '02: Works for $q < N^{0.382}$

  BM '06: Attack for $q < N^{0.468}$

- **Relaxation 2: Small e**

  BM '06: Cryptanalysis of two RSA-variants

  - Tunable-RSA (GHM, ACISP 2005)
  - Rebalanced-RSA (Sun, Wu, ePrint 2005)

CRT-RSA problem

TECHNISCHE UNIVERSITÄT DARMSTADT

Setting: $d = (d_p \bmod p - 1, d_q \bmod q - 1)$ small

- **Relaxation 1: Imbalanced p, q**

  M '02: Works for $q < N^{0.382}$

  BM '06: Attack for $q < N^{0.468}$

- **Relaxation 2: Small e**

  BM '06: Cryptanalysis of two RSA-variants
  - Tunable-RSA (GHM, ACISP 2005)
  - Rebalanced-RSA (Sun, Wu, ePrint 2005)

- **Relaxation 3: Known difference $d_p - d_q$**

  JM '06: $d_p, d_q \leq N^{0.099}$

- **Just small d_p, d_q**: $\mathcal{O}(\sqrt{\min\{d_p, d_q\}})$.

  JM '07: $d_p, d_q \leq N^{0.073}$

Linearization attack

TECHNISCHE UNIVERSITÄT DARMSTADT

Setting: e, $d_q \leq d_p$ small, p, q of same bit-size

Look at

$$\left| \begin{array}{rcl} ed_p & = & 1 + k(p-1) \\ ed_q & = & 1 + \ell(q-1) \end{array} \right|$$

with $k, \ell \leq \frac{ed_p}{\sqrt{N}}$. Rewrite as

$$\left| \begin{array}{rcl} ed_p + k - 1 & = & kp \\ ed_q + \ell - 1 & = & \ell q \end{array} \right|$$

Multiply

$$e^2 d_p d_q + e(d_p(\ell-1) + d_q(k-1)) + (1-N)k\ell = k + \ell - 1$$

Lattice technique

TECHNISCHE UNIVERSITÄT DARMSTADT

Linearize: $e^2 w + ex + (1-N)y = z$

Modulo e^2: $ex + (1-N)y = z \bmod e^2$

Lattice basis:

$$B = \begin{pmatrix} 1 & 0 & e \\ 0 & 1 & 1-N \\ 0 & 0 & e^2 \end{pmatrix}$$

Targetvektor $(x_0, y_0, w_0) \cdot B = (x_0, y_0, z_0)$.

Finding the solution

TECHNISCHE UNIVERSITÄT DARMSTADT

Sufficient condition: $|x_0 y_0 z_0| \leq e^2$.

$$
\begin{aligned}
z_0 &= k + \ell - 1 & &\leq \frac{e d_p}{\sqrt{N}}, \\
y_0 &= k\ell & &\leq \frac{e^2 d_p^2}{N}, \\
x_0 &= d_p(\ell - 1) + d_q(k - 1) & &\leq \frac{e d_p^2}{\sqrt{N}}.
\end{aligned}
$$

Finding the solution

TECHNISCHE UNIVERSITÄT DARMSTADT

Sufficient condition: $|x_0 y_0 z_0| \leq e^2$.

$$\begin{aligned} z_0 &= k + \ell - 1 & &\leq \frac{e d_p}{\sqrt{N}}, \\ y_0 &= k\ell & &\leq \frac{e^2 d_p^2}{N}, \\ x_0 &= d_p(\ell - 1) + d_q(k-1) & &\leq \frac{e d_p^2}{\sqrt{N}}. \end{aligned}$$

Yields condition

$$\frac{e d_p^2}{\sqrt{N}} \cdot \frac{e^2 d_p^2}{N} \cdot \frac{e d_p}{\sqrt{N}} \leq e^2$$

$$\Leftrightarrow \quad d_p^5 \leq \frac{N^2}{e^2} \Leftrightarrow d_p \leq \left(\frac{N}{e}\right)^{\frac{2}{5}}$$

Applications

TECHNISCHE UNIVERSITÄT DARMSTADT

Tunable-RSA (Galbraith, Heneghan, McKee 2005):
Parameters: 1024-bit N, 508-bit e and 200-bit d_p, d_q

Rebalanced-RSA (Sun, Wu 2005):
Parameters: 1024-bit N, 512-bit e and 199-bit d_p, d_q

Applications

TECHNISCHE UNIVERSITÄT DARMSTADT

Tunable-RSA (Galbraith, Heneghan, McKee 2005):
Parameters: 1024-bit N, 508-bit e and 200-bit d_p, d_q

Rebalanced-RSA (Sun, Wu 2005):
Parameters: 1024-bit N, 512-bit e and 199-bit d_p, d_q

1000 Experiments with:

- 1024-bit N, 512-bit e and 200-bit d_p, d_q
  Running time: 15 ms, Success rate 100%
- 1024-bit N, 512-bit e and 204-bit d_p, d_q
  Running time: 15 ms, Success rate 90%

Known difference

TECHNISCHE UNIVERSITÄT DARMSTADT

Scenario: $d_q - d_p = c$ **is known**

Recall that

$$e^2 d_p d_q + e(d_p(\ell - 1) + d_q(k-1)) + (1-N)k\ell - k - \ell = -1$$

Use $d_q = d_p + c$. Gives us polynomial equation

$$f(d_p, \ell, k) = 0 \text{ over } \mathbb{Z}.$$

Newton polytope is defined via

$$\{d_p^2, d_p, d_p\ell, d_pk, k\ell, k, \ell, 1\}.$$

Find roots using Coppersmith's method.

Analysis yields: $d_p, d_q \leq N^{0.099-\epsilon}$

Experimental results

TECHNISCHE UNIVERSITÄT DARMSTADT

N	d_p	dim	LLL-time
1000 bit	10 bit	54	32 min
2000 bit	22 bit	54	175 min
3000 bit	42 bit	54	487 min
4000 bit	60 bit	54	1015 min
5000 bit	85 bit	54	1803 min
500 bit	9 bit	99	105 min
1000 bit	18 bit	99	495 min
500 bit	13 bit	112	397 min

Attacking CRT-RSA

TECHNISCHE UNIVERSITÄT DARMSTADT

Scenario: Small d_p, d_q

Recall again that

$$e^2 d_p d_q + e(d_p(\ell - 1) + d_q(k-1)) + (1-N)k\ell - k - \ell = -1$$

Interpret as polynomial equation

$$f(d_p, d_q, k, \ell) = 0 \text{ over } \mathbb{Z}$$

Newton polytope is defined via

$$\{d_p d_q, d_p \ell, d_p, d_q k, d_q, k\ell, k, \ell, 1\}$$

Find roots using Coppersmith's method.

Analysis yields: $d_p, d_q \leq N^{0.073-\epsilon}$

Experimental results

TECHNISCHE UNIVERSITÄT DARMSTADT

e	d_p, d_q	dim	δ	asympt	LLL-time
250 bit	332 bit	27	0.227	0.287	2 sec
300 bit	299 bit	27	0.209	0.271	2 sec
400 bit	239 bit	27	0.173	0.243	2 sec
500 bit	199 bit	27	0.136	0.214	2 sec
577 bit	168 bit	27	0.108	0.192	2 sec
700 bit	116 bit	56	0.070	0.157	206 sec
800 bit	70 bit	95	0.044	0.128	834 sec
900 bit	35 bit	144	0.023	0.100	8198 sec
1000 bit	11 bit	144	0.004	0.073	7111 sec

Finding good bounds

TECHNISCHE UNIVERSITÄT DARMSTADT

Given : Polynomial $f(x_1, \ldots, x_n) \pmod N$

Find : Optimal bounds $|x_1| \leq X_1, \ldots, |x_n| \leq X_n$

Equiv. : Find optimal lattice basis.

Status quo:

- C '96: Solved for univariate modular case
- BM '05: Toolbox for bivariate integer case
- JM '06: Strategies for multivariate case

Work in progress (with Damien Stehlé):

- Algorithm that outputs optimal bound
- Uses Gröbner bases-like approach
- LLL reduction for selecting sub-lattice

Conclusion & Open problems

TECHNISCHE UNIVERSITÄT DARMSTADT

- First ptime attack for RSA-CRT
- Theoretical analysis worse than experiments
- Some bounds do not match
  - Small d vs. Factoring with known bits
  - Small d vs. CRT-RSA with $d_p = d_q$
- Give provable version of Coppersmith's method
- Need to automatize the bound analysis