1

Lattice-based Integer Factorization – An Introduction to Coppersmith's Method

Alexander May

"I love this Coppersmith stuff, taking polynomials and shifting them around, it is so real! Way better than this cryptography meta-reduction non-sense."
— Arjen Lenstra, personal communication at Asiacrypt 2007

Abstract

Coppersmith's method is used to find unknowns in polynomial equations. In this chapter, our polynomial equations stem from cryptanalysis problems related to the RSA cryptosystem and integer factorization with the cryptographic secrets modeled as unknowns, but Coppersmith's method can be applied in a much more general context (see e.g. [8, 12], just to mention a few).

The beauty of Coppersmith's method comes from exploring exponentially-sized search spaces, while preserving polynomial time using the famous Lenstra-Lenstra-Lovasz (LLL) lattice reduction algorithm [29]. In Coppersmith-type literature, researchers are mainly focussing on maximizing the limits of the search space using an asymptotic lattice dimension analysis. This makes it quite cumbersome to follow a Coppersmith-type analysis for non-experts.

In this chapter, we follow a different approach to make the method more accessible to newcomers. We review some of the most famous applications of Coppersmith's approach, like RSA attacks with known parts of the message, Håstad's broadcast attack, factoring with known bits, certification of RSA keys, factorization via the RSA secret key, and RSA small secret key attacks. Instead of focussing on the method's full asymptotic extent, we provide small lattice bases that illustrate the applications, which can be easily and efficiently implemented. We hope that this approach results in a larger target audience, including security engineers, lecturers and students, and allows them to exper-

iment with and study the beauty of Coppersmith's LLL-based method and its applications.

1.1 Introduction to Coppersmith's Method

In 1996, Coppersmith [13, 14] proposed two lattice-based methods for finding small roots of polynomial equations, one method for polynomial equations *over the integers* and one for *modular* polynomial equations. In the following, we mainly focus on the latter *modular* technique, but in Section 1.5 we also discuss the relation to the *integer* technique.

Coppersmith's method is inspired by lattice-techniques from Håstad [19] and Girault,Toffin,Valleé [17], but contrarily to these methods, Coppersmith's method has the benefit that it provides *provable guarantees* to find *all* roots smaller than a certain bound X in *polynomial time* in the bitlength of e.g. the modulus of the polynomial equation. The polynomial run time is especially fascinating for two reasons.

First, the root bound X is in general of exponential size in the bitlength of the modulus. That is, we are able to find all roots in an interval of exponential size. Put differently, we are able to scan a search space as large as a polynomial fraction of the modulus in polynomial time.

Second, the use of LLL reduction [29] is sufficient, whereas former methods [19] often relied on finding shortest lattice vectors, which is in general a hard problem [1]. Hence, the efficiency of Coppersmith's methods is completely inherited from the efficieny of the famous Lenstra-Lenstra-Lovasz algorithm [29, 39]. In fact, using stronger lattice reduction instead of LLL would not increase the bound X up to which roots can be found.

Applications. The first applications of Coppersmith's method given in the original works [13, 14, 15] already impressively demonstrated the technique's power. Coppersmith showed in [13] that one can invert the RSA function $x \mapsto x^3 \bmod N$ in polynomial time given only a $\frac{2}{3}$ fraction of the bits of x (see Section 1.3.1). This has crucial implications for proving RSA security [38, 9], since under the assumption that inverting the RSA function is hard the Coppersmith result implies that even recovering a $2/3$-fraction must be hard.

In [14], Coppersmith showed that for an RSA modulus $N = pq$, having primes p, q of the same bit-size, recovering half of the bits of p leads to polynomial factorization (see Section 1.3.3 , and Lenstra [27] for an application). This result was used in many practical settings. In 2012, Lenstra et al [28]

showed that improper practical use of randomness for generating p, q is a severe real-world issue, by factoring many public RSA moduli by just doing gcd computations. This gdc-attack was later extended to nearby gcds using Coppersmith's attack [4, 35]. On the constructive side, the *factoring with known bits* method can be used to certify correctness of RSA public keys (N, e), in the sense that one can efficiently prove that e does not divide $\phi(N)$, see Section 1.3.3 and [25].

Certainly, the most prominent cryptanalysis application of Coppersmith's method is the improvement of Wiener's famous attack on RSA secret exponents $d < N^{\frac{1}{4}}$ [46] to $d < N^{0.292}$ by Boneh and Durfee [10] (see Section 1.4.2). Similar attacks have been designed for RSA secret exponents with small Chinese Remainder Theorem (CRT) representation $(d_p, d_q) = (d \bmod p-1, d \bmod q-1)$ (see Section 1.4.3 and [30, 5, 24, 44]). In 2017, Takayasu, Lu and Peng showed that one can efficiently factor N if $d_p, d_q \leq N^{0.122}$, thereby significantly improving over the previously best bound $d_p, d_q \leq N^{0.073}$ from [24].

We conclude our chapter in Section 1.5 by discussing some open questions related to the optimization of Coppersmith's method, and by giving an outlook for the promising Coppersmith-type research direction for polynomial systems. This direction was recently used in the amazing cryptanaltic result of Xu, Sarkar, Lu, Wang and Pan [47] in 2019 that fully breaks the so-called modular inversion hidden number problem [11].

1.2 Useful Coppersmith-type Theorems

Assume that we receive as input a polynomial $f(x)$ of degree δ over the ring $\mathbb{Z}_N$ for some integer N of unknown factorization, and we want to find all roots of $f(x)$ in a certain interval. The description length of f, N is $\Theta(\delta \log N)$. W.l.o.g. we may assume that f is monic, otherwise we multiply by the inverse coeffient of the leading monomial x^δ modulo N. If this inverse does not exist, then we can factor N and proceed recursively.

1.2.1 Idea of Coppersmith's method

Coppersmith's method constructs from $f(x)$ a polynomial $g(x)$ of usually larger degree such that every small modular root x_0 of f, i.e.

$$f(x_0) = 0 \bmod N \text{ with } |x_0| < X,$$

is also a root of g over $\mathbb{Z}$. Thus, we reduce *modular* univariate modular root finding to *integer* univariate root finding, for which we have standard methods.

Let us fix some integer $m \in \mathbb{Z}$. We construct g as an integer linear combination of multiples of

$$h_{i,j} = x^j N^i f^{m-i}(x).$$

Notice that every root x_0 of f satisfies $h_{i,j}(x_0) = 0 \bmod N^m$. Hence if g is an integer linear combination of the $h_{i,j}$'s then we have $g(x_0) = 0 \bmod N^m$ as well.

Let us identify the polynomials $h_{i,j}(x)$ with their coefficient vectors. The integer linear combinations of these vectors form an integer lattice L. A key observation is that small vectors in L correspond to linear combinations $g(x)$ with small coefficients. Moreover if $g(x)$ has small coefficients, and is evaluated at small points x_0 with $|x_0| \leq X$ only, then the result must also be (somewhat) small. More precisely, assume that $g(x_0)$ is in absolute value smaller than N^m for all $|x_0| \leq X$. Then we have for all x_0 that

$$g(x_0) = 0 \bmod N^m \text{ and } |g(x_0)| < N^m.$$

These two equations together imply that $g(x_0) = 0$, since the only multiple of N^m smaller in absolute value then N^m is $0 \cdot N^m = 0$. This implies that $g(x)$ has the desired roots over the integers!

Notice that our so far used criterion $|g(x_0)| < N^m$ for all $|x_0| \leq X$ is not efficiently checkable, since we do not know any root x_0, only their upper bound X. However if $g(x)$ has sufficiently small coefficients, this criterion should automatically be fulfilled. The following lemma makes this intuition precise. The lemma is usually contributed to Howgrave-Graham [22], but already appeared in the work of Håstad [19]. It provides an easily testable upper bound on the norm of g's coefficient vector that is sufficient to guarantee $|g(x_0)| < N^m$.

To this end let us introduce some useful notion. Let $g(x) = \sum_{i=0}^{n} c_i x^i$ be a univariate polynomial with coefficient vector $(c_0, c_1, \ldots, c_n)$. Then the polynomial $g(xX)$ has coefficient vector $\mathbf{v} = (c_0, c_1 X, \ldots, c_n X^n)$, and we denote by $\|g(xX)\|$ the Euclidean norm of $\mathbf{v}$.

Lemma 1.1 (Håstad/Howgrave-Graham) *Let $g(x)$ be a univariate polynomial with n monomials. Let m, X be positive integers. Suppose that*

1. $g(x_0) = 0 \bmod N^m$ *where* $|x_0| \leq X$,
2. $\|g(xX)\| < \frac{N^m}{\sqrt{n}}$.

Then $g(x_0) = 0$ holds over the integers.

Proof Property 2 implies

$$|g(x_0)| = \left|\sum_i c_i x_0^i\right| \leq \sum_i |c_i x_0^i|$$
$$\leq \sum_i |c_i| X^i \leq \sqrt{n}\,\|g(xX)\| < N^m.$$

By property 1 we know that $g(x_0)$ is a multiple of N^m, and therefore $g(x_0) = 0$. □

While in Coppersmith's method we construct polynomials $g(x)$ that automatically satisfy property 1 of Lemma 1.1, the norm property 2 of Lemma 1.1 guarantees that $g(x)$ has the same small roots as $f(x)$ over the integers (but it may have additional roots). Of course, our goal is to maximize X, i.e. the range of small roots that we can efficiently recover.

1.2.2 The crucial role of Lenstra-Lenstra-Lovasz reduction

Let L be the lattice defined by the coefficient vectors of $h_{i,j}(x)$. The following celebrated theorem relates the length of a shortest vector in an LLL reduced basis of L to the lattice determinant $\det(L)$, which is an invariant of L.

Theorem 1.2 (Lenstra, Lenstra, Lovász) *Let $L \in \mathbb{Z}^n$ be a lattice spanned by $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$. The LLL-algorithm outputs a lattice vector $\mathbf{v} \in L$ satisfying*

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

in time $O(n^6 \log^3 B_{max})$, where $B_{max} = \max_{i,j}\{|(\mathbf{b}_i)_j|\}$ is the largest basis entry.

Neumaier and Stehlé [36] showed that the same output quality as in Theorem 1.2 can be achieved in run time

$$O(n^{4+\epsilon} \log^{1+\epsilon} B_{\max}) \text{ for any constant } \epsilon > 0. \tag{1.1}$$

Notice that by Minkowski's first theorem [18], every lattice L contains a non-zero vector $\mathbf{v}' \leq \sqrt{n}\det(L)^{\frac{1}{n}}$. Hence, the smallest vector found by LLL-reduction may be longer by an exponential factor (in the dimension n) than the shortest lattice vector. This is however perfectly fine for Coppersmith's method. Recall that a vector $\mathbf{v}$ in a Coppermith-type lattice corresponds to some polynomial $g(x)$, for which we have to satify the second property of Lemma 1.1: $\|g(xX)\| < \frac{N^m}{\sqrt{n}}$. Thus, the LLL output vector $\mathbf{v}$ is short enough if

$$\|\mathbf{v}\| \leq 2^{\frac{n-1}{4}} \det(L)^{\frac{1}{n}} < \frac{N^m}{\sqrt{n}}.$$

We will see that in Coppersmith's method we have $\det(L) = N^{\Theta(m)}$ with $m \approx \log N$. Thus, for sufficiently large N, the terms $2^{\frac{n-1}{4}}$ and $\sqrt{n}$ can be neglected, which leads to the simplified so-called *enabling condition*

$$\det(L) \leq N^{mn}, \tag{1.2}$$

that plays a crucial role in all constructions using Coppersmith's method. The enabling condition is used to optimize X. Namely, we have to define a collection of polynomials $h_{i,j}$ such that their coefficient vectors span a lattice L with $\det(L)$ as small as possible.

1.2.3 Coppersmith-type Theorems

We are now ready to formulate Coppersmith's theorem for univariate polynomials.

Theorem 1.3 *Let N be an integer of unknown factorization. Let $f(x)$ be a univariate monic polynomial of constant degree δ. Then we can find all solutions x_0 of the equation*

$$f(x) = 0 \bmod N \quad \textit{with} \quad |x_0| \leq N^{\frac{1}{\delta}}$$

in time $O(\log^{6+\epsilon} N)$ for any $\epsilon > 0$.

Proof Let us just briefly sketch the proof, a full proof can be found in [31]. Choose $m \approx \frac{\log N}{\delta}$ and define the collection of polynomials

$$h_{i,j}(x) = x^j N^i f(x)^{m-i} \text{ for } 0 \leq i < m, 0 \leq j < \delta.$$

It is not hard to see that the coefficient vectors of $h_{i,j}(xX)$ form an $n = m\delta \approx \log N$-dimensional lattice basis B with $\det(L) = \det(B) \approx N^{\frac{\delta m^2}{2}} X^{\frac{n^2}{2}}$. Hence the enabling condition from Equation (1.2) translates to

$$N^{\frac{\delta m^2}{2}} X^{\frac{n^2}{2}} \leq N^{mn}.$$

Using $n = m\delta$ this is equivalent to $X^{\delta^2 m^2} \leq N^{\delta m^2}$, from which we easily derive the desired root bound $X \leq N^{\frac{1}{\delta}}$.

It remains to show the run time. We work in an $n \approx \log N$ dimensional lattice with largest entries of bit-size $\log B_{max} = O(m \log N) = O(\log^2 N)$. Using the runtime of the Neumeier-Stehlé LLL-variant of Equation (1.1) lattice reduction runs in time $O(\log^{6+\epsilon} N)$. □

Extending Coppersmith's bound. As already pointed out in Coppersmith's original work, any small root bound X can be be extended to cX for some real number c at the expense of an additional run time factor of c. It is an important open question whether this can be improved, or whether such a linear run time factor is unavoidable.

Theorem 1.4 *Let N be an integer of unknown factorization and $c \geq 1$. Let $f(x)$ be a univariate monic polynomial of constant degree δ. Then we can find all solutions x_0 of the equation*

$$f(x) = 0 \bmod N \quad \textit{with} \quad |x_0| \leq cN^{\frac{1}{\delta}}$$

in time $O(c \log^{6+\epsilon} N)$ for any $\epsilon > 0$.

Proof Split the interval $[-cN^{\frac{1}{\delta}}, cN^{\frac{1}{\delta}}]$ in c sub-intervals of size each $2N^{\frac{1}{\delta}}$, centered at some x_i. For each sub-interval with center x_i, we apply Theorem 1.3 to find all roots within this sub-interval. □

Theorem 1.4 has immediate consequences for cryptographic attacks (see Section 1.3), as it easily allows to extend Coppersmith-type root bounds.

Coppersmith method for divisor. Somewhat surpringly, one can also extend Coppersmith's method to find roots of $f(x)$ modulo b where $b \geq N^{\beta}$ is an *unknown* divisor of N. In principle, we keep the same strategy as before and simply work modulo b instead of N. E.g. $h_{i,j} = x^j N^i f(x)^{m-i}$ now has small roots modulo b^m. Moreover, the simplified *enabling condition* from Equation (1.2) becomes

$$\det(L) \leq N^{\beta mn}. \tag{1.3}$$

Working out the details yields the following theorem that was already stated in Coppersmith [13, 15] and Howgrave-Graham [22] for polynomial degree $\delta = 1$, and first appeared in its full form in [31].

Theorem 1.5 *Let N be an integer of unknown factorization, which has a divisor $b \geq N^{\beta}$, $0 < \beta \leq 1$. Let $c \geq 1$,and let $f(x)$ be a univariate monic polynomial of constant degree δ. Then we can find all solutions x_0 of the equation*

$$f(x) = 0 \bmod b \quad \textit{with} \quad |x_0| \leq cN^{\frac{\beta^2}{\delta}}$$

in time $O(c \log^{6+\epsilon} N)$ for any $\epsilon > 0$.

Notice that Theorem 1.4 is a special case of Theorem 1.5 with $\beta = 1$. For $\beta < 1$, the small root bound decreases polynomially with exponent β^2.

1.3 Applications in the Univariate Case

1.3.1 Inverting the RSA Function with Partial Knowledge.

Small Messages. Assume we work with RSA with small public encryption exponent e. As a simple example take $e = 3$, and let $c = m^3 \bmod N$ be an RSA ciphertext with $m < N^{\frac{1}{3}}$. Then $m^3 < N$ and therefore $c = m^3$ holds over the integers. Thus, we may compute $m = c^{\frac{1}{3}}$ over $\mathbb{Z}$ by standard efficient root finding methods such as Newton iteration.

Let us now formulate the same problem in terms of Coppersmith's method. Define the polynomial $f(x) = x^3 - c \bmod N$. Then an application of Theorem 1.3 yields that we find all roots $m < N^{\frac{1}{3}}$ in polynomial time. In fact, there is nothing to do, since Coppersmith's method constructs a polynomial equation with the same roots over the integers, but $f(x)$ already has the desired root over the integers.

The crucial advantage of Coppersmith's method is that it also covers the inhomogeneous case, in which the preimage m of the RSA function is not small, but we know it up to an additive term of size at most $N^{\frac{1}{3}}$ (or $N^{\frac{1}{e}}$ in general). This inhomogeneous case is addressed in the following.

Stereotyped Messages. Let us start with an illustrative example solvable in small lattice dimension. Let $c = m^3 \bmod N$, where we know an approximation m_1 of the message up to an additive error of size at most $m - m_1 < N^{\frac{5}{21}}$, i.e.

$$m_0 = m - m_1 \text{ for some unknown } m_0 < N^{\frac{5}{21}}.$$

One may think of m_1 as a $\frac{16}{21}$-fraction of the most significant bits. Our goal is to recover the $\frac{5}{21}$-bit fraction m_0 efficiently with Coppersmith's method.

Example: Stereotyped Messages	
Given:	$c = m^3 \bmod N$ and some m_1 satisfying $m - m_1 < N^{\frac{5}{21}}$.
Polynomial:	$f(x) = (x + m_1)^3 - c \bmod N$ with root $x_0 = m - m_1 < N^{\frac{5}{21}}$.
Parameters:	degree $\delta = 3$

Lattice basis. Define $m = 2$, i.e. all polynomials have root x_0 modulo N^2. Define the collection of seven polynomials

$$N^2,\ N^2x,\ N^2x^2,\ Nf(x),\ xNf(x),\ x^2Nf(x),\ f^2(x).$$

Let $X = N^{\frac{5}{21}}$ and $h_1(x), \ldots, h_7(x)$ denote the above collection. The coefficient vectors of $h_i(xX)$, $1 \le i \le 7$, define the following lattice basis

$$B = \begin{pmatrix} N^2 & & & & & & \\ & N^2X & & & & & \\ & & N^2X^2 & & & & \\ N(m_1^3-c) & 3Nm_1^2X & 3Nm_1X^2 & NX^3 & & & \\ & N(m_1^3-c)X & 3Nm_1^2X^2 & 3Nm_1X^3 & NX^4 & & \\ & & N(m_1^3-c)X^2 & 3Nm_1^2X^3 & 3Nm_1X^4 & NX^5 & \\ (m_1^3-c)^2 & 6(m_1^3-c)m_1^2X & (6(m_1^3-c)m_1+9m_1^4)X^2 & (20m_1^3-2c)X^3 & 15m_1^2X^4 & 6m_1X^5 & X^6 \end{pmatrix}.$$

Lattice basis B spans an $n = 7$ dimensional lattice L with $\det(L) = |\det(B)| = N^9X^{21}$. Using the enabling condition from Equation (1.2), we obtain

$$N^9X^{21} \leq N^{2\cdot 7} \text{ which implies } X \leq N^{\frac{5}{21}}.$$

Using the fast Neumaier-Stehlé variant of LLL reduction from Equation (1.1), we recover the lower $\frac{5}{21} \approx 0.238$-fraction of m in almost linear time $O(\log^{1+\epsilon} N)$.

An application of Theorem 1.4 yields a superior bound for the bits that one can recover at the cost of an increased running time.

Theorem 1.6 *Let $c' = m^e \bmod N$ with constant e. Assume we know some m_1 satisying $m - m_1 < cN^{\frac{1}{e}}$ for some $c \geq 1$. Then m can be found in time $O(c \log^{6+\epsilon} N)$.*

Proof Theorem 1.4 yields that in time $O(c \log^{6+\epsilon} N)$ we can recover all roots $x_0 = m - m_1$ of size

$$|x_0| \leq cN^{\frac{1}{\delta}} = cN^{\frac{1}{e}}.$$

□

Theorem 1.6 implies that e.g. for RSA exponent $e = 3$ given a $\frac{2}{3}$-fraction of the preimage m we find the remaining $\frac{1}{3}$-fraction in polynomial time. Maybe somewhat surprisingly, this has important applications for proving RSA security [38, 9]. Namely, if we assume that it is hard to invert the RSA function $m \mapsto m^e \bmod N$, then by Theorem 1.6 it must already be hard to recover an $\frac{e-1}{e}$-fraction of m.

1.3.2 Systems of Univariate Polynomial Equations

Polynomially Related Messages. The following is known as Håstad's RSA Broadcast Attack [19]. Assume that we obtain three textbook RSA ciphertexts c_1, c_2, c_3 under three different RSA public keys $(N_1, 3)$, $(N_2, 3)$, $(N_3, 3)$ for the same message m (without randomized padding function [3]). In other words

we obtain the following system of congruences

$$\begin{aligned} c_1 &= m^3 \bmod N_1 \\ c_2 &= m^3 \bmod N_2 \\ c_3 &= m^3 \bmod N_3. \end{aligned}$$

Let $N = \min_i\{N_i\}$, and let us assume $m < N$, which implies $m^3 < N^3$. Let us further assume w.l.o.g. that the public moduli N_i are pairwise coprime. By the Chinese Remainder Theorem (CRT) we can easily compute some

$$c = \mathrm{CRT}(c_1, c_2, c_3) = m^3 \bmod N_1 N_2 N_3.$$

Since $m^3 < N^3 < N_1 N_2 N_3$, we know that $c = m^3$ holds over the integers. Thus, we can simply compute $m = c^{\frac{1}{3}}$.

Now assume that we obtain the following system of congruences

$$\begin{aligned} c_1 &= m^2 \bmod N_1 \\ c_2 &= m^3 \bmod N_2 \\ c_3 &= m^3 \bmod N_3. \end{aligned} \tag{1.4}$$

Strictly speaking $c_1 = m^2 \bmod N_1$ is not a textbook RSA ciphertext (since squaring is not a bijection), but we may ignore that, since m is uniquely defined by the above system. Because the right-hand side of the equations in Equation (1.4) take different values, we may no longer simply compute $\mathrm{CRT}(c_1, c_2, c_3)$ as in Håstad's Broadcast attack.

Instead, we define the following degree-6 polynomials

$$f_1(x) = (x^2 - c_1)^3, f_2(x) = (x^3 - c_2)^2, f_3(x) = (x^3 - c_3)^2$$

that all share the same root $x_0 = m$ modulo N_1^3, N_2^2, respectively N_3^2. Via the Chinese Remainder Theorem we now compute a degree-6 polynomial $f = \mathrm{CRT}(f_1, f_2, f_3)$ that has the root $x_0 = m$ modulo $N_1^3 N_2^2 N_3^2$.

Example: Polynomially Related Messages	
Given:	Equations from (1.4). Let $M = N_1^3 N_2^2 N_3^2$ and $N = \min_i\{N_i\}$.
Polynomial:	$f(x)$ of degree 6 with root $x_0 = m$ modulo M, where $m < N$.
Parameters:	degree $\delta = 6$

Lattice basis: Let $m = 6$, i.e. all polynomials have root x_0 modulo M^6. We

define the following collection of 37 polynomials

$$\begin{array}{cccccc}
M^6, & M^6x, & M^6x^2, & M^6x^3, & M^6x^4, & M^6x^5, \\
M^5f(x), & M^4xf(x), & M^5x^2f(x), & M^5x^3f(x), & M^5x^4f(x), & M^5x^5f(x), \\
M^4f^2(x), & M^4xf^2(x), & M^4x^2f^2(x), & M^4x^3f^2(x), & M^4x^4f^2(x), & M^4x^5f^2(x), \\
M^3f^3(x), & M^3xf^3(x), & M^3x^2f^3(x), & M^3x^3f^3(x), & M^3x^4f^3(x), & M^3x^5f^3(x), \\
M^2f^4(x), & M^2xf^4(x), & M^2x^2f^4(x), & M^2x^3f^4(x), & M^2x^4f^4(x), & M^2x^5f^4(x), \\
Mf^5(x), & Mxf^5(x), & Mx^2f^5(x), & Mx^3f^5(x), & Mx^4f^5(x), & Mx^5f^5(x), \\
f^6(x). & & & & &
\end{array}$$

Let $h_1(x), \ldots, h_{37}(x)$ be the above collection, and let $X = N$. Then the coefficient vectors of $h_1(xX), \ldots, h_{37}(xX)$ define a lattice basis B of some lattice L with $\dim(L) = 37$ and

$$\det(L) = |\det(B)| = M^{6\sum_{i=1}^{6} i} X^{\sum_{i=1}^{36} i} = M^{126} X^{666}.$$

Thus, the enabling condition from Equation (1.2) yields

$$\det(L) \leq M^{6 \cdot 37} \text{ which is equivalent to } X \leq M^{\frac{16}{111}}.$$

Since $M = N_1^3 N_2^2 N_3^2 > N^8$, we have $M^{\frac{16}{111}} > N^{\frac{128}{111}} > N$. Thus, we find in time $O(\log^3 N)$ via LLL-reduction (or time $O(\log N^{1+\epsilon})$ via Equation (1.1)) the unique $m < N = \min_i\{N_i\}$.

Using Theorem 1.3 we can show a slightly stronger and more general result.

Theorem 1.7 ([32]) *Let $N_1, \ldots, N_k$ be pairwise co-prime RSA-moduli and let $m < \min_i\{N_i\}$. Let $g_i(x)$, $i = 1, \ldots, k$, be polynomials of degree δ_i. Assume we are given*

$$c_i = (g_i(m))^{e_i} \bmod N_i \text{ satisfying } \sum_{i=1}^{k} \frac{1}{\delta_i e_i} \geq 1.$$

Then m can be computed in time $O(\log^{6+\epsilon} N)$ for all $\epsilon > 0$.

Proof W.l.o.g. we assume that all $g_i(m)$ are monic. Otherwise, we multiply by the inverse of their leading coefficient. If this inverse computation fails, we obtain the factorization of some N_i which in turns enables us to compute m.

We define $\delta = \text{lcm}_i\{\delta_i e_i\}$ as the least common multiple. Furthermore, we define the degree-δ polynomials

$$f_i(x) = (g_i(x)^{e_i} - c_i)^{\frac{\delta}{\delta_i e_i}} \text{ with root } f_i(m) = 0 \bmod N_i^{\frac{\delta}{\delta_i e_i}}.$$

Let $M = \prod_{i=1}^{k} N_i^{\frac{\delta}{\delta_i e_i}}$ be the product of all moduli. Via Chinese Remaindering

we compute the polynomial

$$f(x) = \mathrm{CRT}(f_1, \ldots, f_k) = \sum_{i=1}^{k} b_i f_i(x) \bmod M, \text{ where } b_i \bmod N_j = \begin{cases} 1 \text{ if } i = j \\ 0 \text{ else} \end{cases}.$$

An application of Theorem 1.3 shows that we can find in time $O(\log^{6+\epsilon} N)$ all roots x_0 smaller than

$$|x_0| \leq M^{\frac{1}{\delta}}.$$

Using our condition $1 \leq \sum_{i=1}^{k} \frac{1}{\delta_i e_i}$, we know that our desired root m satisfies

$$m < \min_i\{N_i\} \leq \left(\min_i\{N_i\}\right)^{\sum_{i=1}^{k} \frac{1}{\delta_i e_i}} \leq \prod_{i=1}^{k} N_i^{\frac{1}{\delta_i e_i}} = M^{\frac{1}{\delta}}.$$

Thus, we recover the desired root via Coppersmith's method. □

An important special case of Theorem 1.8 is when all $g_i(x) = x$ and therefore $\delta_i = 1$.

Theorem 1.8 ([32]) *Let $N_1, \ldots, N_k$ be pairwise co-prime RSA-moduli and let $m < \min_i\{N_i\}$. Assume we are given*

$$c_i = m^{e_i} \bmod N_i \text{ satisfying } \sum_{i=1}^{k} \frac{1}{e_i} \geq 1.$$

Then m can be computed in time $O(\log^{6+\epsilon} N)$ for all $\epsilon > 0$.

Let us apply Theorem 1.8 to the equation system of Equation (1.4). Since $e_1 = 2, e_2 = e_3 = 3$ we have $\frac{1}{2} + \frac{1}{3} + \frac{1}{3} = \frac{7}{6} \geq 1$. Thus, we can find m in time $O(\log^{6+\epsilon} N)$. Recall that the analysis at the beginning of the section even allowed run time $O(\log^{1+\epsilon} N)$. The reason is that $\sum_i \frac{1}{e_i}$ is significantly larger than 1, which in turn allows for a smaller lattice basis, and therefore also for superior run time.

1.3.3 Factoring with Known Bits and RSA Key Certification

Let $N = pq$ be an RSA modulus, w.l.o.g. $p > q$ and therefore $p > N^{\frac{1}{2}}$. Assume that we know a good approximation p_1 of p up to an additive term of size at most $N^{\frac{1}{5}}$, i.e. we know some p_1 such that

$$p_0 = p - p_1 \text{ for some unknown } p_0 < N^{\frac{1}{5}}.$$

One may think of p_1 as a $\frac{4}{5}$-fraction of the most significant bits of p. Our task is to recover the $\frac{1}{5}$ least significant bits p_0.

Example: Factoring with Known Bits

Given:	$N = pq$, $p > N^{\frac{1}{2}}$ and some p_1 satisfying $p - p_1 < N^{\frac{1}{5}}$.
Polynomial:	$f(x) = x + p_1 \bmod p$ with root $x_0 = p - p_1 < N^{\frac{1}{5}}$.
Parameters:	degree $\delta = 1$, divisor size $\beta = \frac{1}{2}$

Lattice basis: Let $m = 2$, i.e. all polynomials have root x_0 modulo p^2. We define the collection of five polynomials

$$N^2,\ Nf(x),\ f^2(x),\ xf^2(x),\ x^2f^2(x).$$

Let $X = N^{\frac{1}{5}}$ and $h_1(x), \ldots, h_5(x)$ denote the above collection. Then the coefficient vectors of $h_i(xX)$, $i = 1, \ldots, 5$, form the lattice basis

$$B = \begin{pmatrix} N^2 & & & & \\ Np_1 & NX & & & \\ p_1^2 & 2p_1X & X^2 & & \\ & p_1^2X & 2p_1X^2 & X^3 & \\ & & p_1^2X^2 & 2p_1X^3 & X^4 \end{pmatrix}. \tag{1.5}$$

Notice that $n = \dim(L) = 5$ and $\det(L) = N^3X^{10}$. Thus the enabling condition from Equation (1.3) yields

$$N^3X^{10} \leq N^{\beta mn} = N^5 \text{ which implies } X \leq N^{\frac{1}{5}}.$$

LLL reduction runs on lattice basis B in time $O(\log^3 N)$ (and in $O(\log^{1+\epsilon} N)$ using Equation (1.1)).

Using Theorem 1.5, we may increase the bound $N^{\frac{1}{5}}$ at the cost of an increased run time.

Theorem 1.9 *Let N be composite with divisor $p > N^{\beta}$. Assume we are given p_1 satisfying $p - p_1 < cN^{\beta^2}$ for some $c \geq 1$. Then p can be found in time $O(c \log^{6+\epsilon} N)$ for any $\epsilon > 0$.*

Proof Theorem 1.5 implies that in time $O(c \log^{6+\epsilon} N)$ we find all $x_0 = p - p_1$ of size

$$|x_0| \leq cN^{\frac{\beta^2}{\delta}} = cN^{\beta^2}.$$

□

Let $N = pq$ be an RSA modulus with $N^{\frac{1}{2}} < p < 2N^{\frac{1}{2}}$, i.e. $\beta = \frac{1}{2}$. Theorem 1.9 implies that we can factor N in polynomial time given half of the bits of p, i.e. $p - p_1 < N^{\frac{1}{4}} < p^{\frac{1}{2}}$.

Moreover, if we set $p_1 = 0$ and $c = 2N^{\frac{1}{4}}$ then Theorem 1.9 gives us a deterministic factorization algorithm for RSA moduli with running time $\tilde{O}(N^{\frac{1}{4}})$, where the $\tilde{O}$-notion suppresses polylogarithmic factors.

Certifying RSA. For the correctness of many RSA-based protocols, we have to guarantee that the RSA parameters (N, e) are properly chosen. This means in particular that the RSA function $m \mapsto m^e \bmod N$ is a bijection, which is true iff

$$\gcd(e, \phi(N)) = 1.$$

However, the co-primality of e and the Euler totient function $\phi(N)$ is in general not publically checkable, since the knowledge of $\phi(N)$ implies the factorization of N (at least for RSA moduli).

However, let us for the moment assume that N is an RSA modulus, and that $e \geq \sqrt{2}N^{\frac{3}{10}}$ is prime. Primality of e can easily be checked in polynomial time [33]. Since $N = pq$ we have $\phi(N) = (p-1)(q-1)$. By primality of e we conclude that $\gcd(e, \phi(N)) \in \{1, e\}$. Therefore, (N, e) does not define a bijective RSA function iff $\gcd(e, \phi(N)) = e$, in which case we have $e|p-1$ or $e|q-1$. Wlog, let us assume $e|p-1$, i.e. there exists some unknown $k \in \mathbb{N}$ with

$$p = ek + 1. \tag{1.6}$$

W.l.o.g. we may also assume $\gcd(e, N) = 1$. On input e, N the Extended Euclidean Algorithms outputs Bézout coefficients $r, s \in \mathbb{Z}$ satisfying $er + Ns = 1$. Multiplying Equation (1.6) by r yields

$$pr = ekr + r = (1 - Ns)k + r.$$

Using $N = pq$, we obtain

$$p(r + qsk) = k + r.$$

Thus, the polynomial $f(x) = x + r$ satisfies $f(x_0) = 0 \bmod p$ with root $x_0 = k = \frac{p-1}{e}$. Notice that $r = e^{-1} \bmod N$ plays the role of p_1 from the previous *Factoring with Known Bits* approach. Namely, r is an approximation of some (unknown) multiple of p. Since N is an RSA modulus, we know that $\frac{1}{2}p < q$. Multiplication by p gives $p < \sqrt{2N}$. Thus, we obtain

$$x_0 = k = \frac{p-1}{e} < \frac{\sqrt{2N}}{\sqrt{2}N^{\frac{3}{10}}} = N^{\frac{1}{5}}. \tag{1.7}$$

Example: Certification of RSA Parameters

Given:	$N = pq$ and some prime $e \geq \sqrt{2}\, N^{\frac{3}{10}}$ with $e\|p-1$.
Polynomial:	$f(x) = x + (e^{-1} \bmod N) \bmod p$ with root $x_0 = \frac{p-1}{e} < N^{\frac{1}{5}}$.
Parameters:	degree $\delta = 1$, divisor size $\beta = \frac{1}{2}$

Lattice basis: Setting $p_1 = e^{-1} \bmod N$, we can directly use lattice basis B from Equation (1.5). Thus, we may in time $O(\log^3 N)$ via LLL (or $\log^{1+\epsilon} N$

via Equation (1.1)) compute the factorization of N if $e|p-1$. If Coppersmith's method fails to factor N, then by the provable guarantees of Coppersmith's method we certified that $\gcd(e, \phi(N) = 1$, and therefore that the RSA function is indeed a bijection.

Using Theorem 1.5, our certification procedure holds for even smaller public exponents $e \geq N^{\frac{1}{4}}$.

Theorem 1.10 *Given an RSA modulus $N = pq$ and some prime $e \geq N^{\frac{1}{4}}$. Then we can decide in time $O(\log^{6+\epsilon} N)$ for any $\epsilon > 0$ whether* $\gcd(e, \phi(N)) = 1$.

Proof We have $x_0 = k = \frac{p-1}{e} \leq \frac{\sqrt{2N}}{N^{\frac{1}{4}}} < 2N^{\frac{1}{4}}$. Thus, an application of Theorem 1.9 yields the desired result. □

Notice that Theorem 1.10 assumes that we start with some RSA modulus $N = pq$ that is a product of two primes p, q of equal bit-size. In fact, the proof of Theorem 1.10 builds on this property. Somewhat surprisingly, we can certify RSA parameters (N, e) for any N of unknown factorization, even for products of more than two primes.

Theorem 1.11 ([25]) *Given a composite N and some prime $e \geq N^{\frac{1}{4}}$. Then we can decide in time $O(\log^{6+\epsilon} N)$ for any $\epsilon > 0$ whether* $\gcd(e, \phi(N)) = 1$.

Proof sketch It is not hard to see that in the case $\gcd(e, \phi(N)) > 1$, we have $e|p-1$ for some prime divisor of p of N. But as opposed to the RSA case from Theorem 1.10 we now only have a trivial upper bound $p < N$ on p. This implies that an application of Coppersmith's method finds p in time $O(\log^{6+\epsilon} N)$, provided that $x_0 = k = \frac{p-1}{e} \leq \frac{N}{N^{\frac{1}{4}}} = N^{\frac{3}{4}}$.

Assume that $p \geq N^{\sqrt{3/4}}$. Then an application of Theorem 1.5 with $\beta = \sqrt{3/4}, \delta = 1$ yields that we find x_0 if

$$|x_0| \leq N^{\frac{\beta^2}{\delta}} = N^{\frac{3}{4}},$$

as desired. In summary, we find p iff $p \in [N^{\sqrt{3/4}}, N]$. Otherwise we know that $p < N^{\sqrt{3/4}} \approx N^{0.866}$. Therefore, we obtain a new upper bound on p, and may apply Coppersmith's method iteratively, finding a yet decreased lower bound on p, etc.

The proof in [25] shows that after $O(\log N)$ iterative applications of Coppersmith's method we succeed to cover the whole interval $[e, N]$ of all possible candidates for p. If all of these iterations fail to factor N, we certified that $\gcd(e, \phi(N)) = 1$. □

Computing the RSA Secret Key implies Factoring. Let (N, e) be an RSA public key. From the factorization of N one can derive the Euler function $\phi(N)$,

and compute the RSA secret key as $e^{-1} \bmod \phi(N)$. Thus, factoring implies computation of the RSA secret key.

A natural question is whether the converse is also true? If we are able to compute $d = e^{-1} \bmod \phi(N)$, does this lead to an efficient factorization algorithm? This question was already addresses in the original RSA paper [37], where the authors mentioned a probabilistic factorization algorithm due to Miller [33]. It remained open whether there also exists a deterministic factorization algorithm.

Let us first assume that $e \leq \frac{1}{3}N^{\frac{1}{2}}$. We will see that in this case we have a completely elementary factorization algorithm. Assume that we are able to compute the secret key d with $2 < d < \phi(N)$. Then there exists some unknown $k \in \mathbb{N}$ such that

$$ed = 1 + k\phi(N).$$

Let $\tilde{k} = \frac{ed-1}{N} < k$. Then

$$k - \tilde{k} = \frac{ed-1}{\phi(N)} - \frac{ed-1}{N} = \frac{(ed-1)(N-\phi(N))}{\phi(N)N} > 0.$$

Since p, q are of equal bit-size, we have

$$N - \phi(N) = p + q - 1 < 3N^{\frac{1}{2}} \text{ and } \frac{1}{2}N < \phi(N) < N.$$

Since $e \leq \frac{1}{3}N^{\frac{1}{2}}$, we obtain

$$0 < k - \tilde{k} < \frac{\frac{1}{3}N^{\frac{1}{2}}\phi(N) \cdot 3N^{\frac{1}{2}}}{\phi(N)N} = 1.$$

This implies that $k = \lceil \tilde{k} \rceil$. Knowledge of k gives us $\phi(N) = \frac{ed-1}{k}$. Solving the two equations $\phi(N) = (p-1)(q-1)$ and $N = pq$ eventually yields the factorization p, q.

What happens if $ed > N^{\frac{3}{2}}$? Let us model this case as a Coppersmith-type problem. Notice that $M = ed - 1$ is a multiple of the (unknown) $\phi(N)$, and the polynomial

$$f(x) = N - x \bmod \phi(N) \text{ has a small root } x_0 = p + q - 1 < 3N^{\frac{1}{2}}.$$

Notice here the analogy to *Factoring with Known Bits*: M plays the role of N, $\phi(N)$ plays the role of p, but this time we know by construction a good approximation N of $\phi(N)$.

	Example: Factoring via RSA Secret Key
Given:	$N = pq$ and e, d satisfying $ed = 1 \bmod \phi(N)$, $ed < (9N)^{\frac{5}{3}}$.
Polynomial:	$f(x) = N - x \bmod \phi(N)$ with root $x_0 = p + q - 1 < 3N^{\frac{1}{2}}$.
Parameters:	degree $\delta = 1$, divisor size $\beta = \log_{ed-1} \phi(N) \geq \frac{3}{5}$

Lattice basis: Set $m = 2$, then all polynomials have root x_0 modulo $\phi(N)^2$. Set $M = ed - 1 = (9N)^\alpha$ for some $\alpha < \frac{5}{3}$. Let $\beta = \frac{1}{\alpha}$ and $X = 3N^{\frac{1}{2}}$. We define the collection of four polynomials

$$h_1(x) = M^2,\ h_2(x) = Mf(x),\ h_3(x) = f^2(x),\ h_4(x) = xf^2(x).$$

The integer linear combinations of the coefficient vectors of $h_1(xX), \ldots, h_4(xX)$ form a lattice L with basis

$$B = \begin{pmatrix} M^2 & & & \\ MN & -MX & & \\ N^2 & -2NX & X^2 & \\ & N^2X & -2NX^2 & X^3 \end{pmatrix}.$$

We have $n = \dim(L) = 4$ and $\det(L) = |\det(B)| = M^3X^6$. Thus the enabling condition from Equation (1.3) becomes $M^3X^6 \leq M^{\beta mn} = M^{\frac{8}{\alpha}}$. Using $M = ed - 1 = (9N)^\alpha$, this implies

$$X \leq M^{\frac{4}{3\alpha} - \frac{1}{2}} = (9N)^{\frac{4}{3} - \frac{\alpha}{2}}.$$

Since $\alpha < \frac{5}{3}$, the exponent satisfies $\frac{4}{3} - \frac{\alpha}{2} > \frac{1}{2}$. Therefore, we find the root $x_0 = p + q - 1 < 3N^{\frac{1}{2}}$ in time $O(\log^{1+\epsilon} N)$ via LLL reduction. From x_0 we easily derive the factorization of N.
An application of Theorem 1.5 yields the following Theorem.

Theorem 1.12 *Let $N = pq$ be an RSA modulus with key pair $2 < e, d < \phi(N)$ satisfying $ed = 1 \bmod \phi(N)$. On input N, e, d one can compute the factorization of N in time $O(\log^{6+\epsilon} N)$ for any $\epsilon > 0$.*

Proof Define $M = ed - 1 = \phi(N)^\alpha$ for some $\alpha < 2$. Let $f(x) = N - x \bmod \phi(N)$ with root $x_0 = p + q - 1 < 3N^{\frac{1}{2}}$. Let $\delta = 1, \beta = \frac{1}{\alpha}, c = 6$. Via Theorem 1.5 we find x_0 in time $O(\log^{6+\epsilon} N)$ if

$$|x_0| \leq 6M^{\beta^2} = 6\phi(N)^{\frac{1}{\alpha}}.$$

Since $\alpha < 2$ and $\phi(N) > \frac{N}{2}$ we obtain

$$6\phi(N)^{\frac{1}{\alpha}} > 6\phi(N)^{\frac{1}{2}} > 3N^{\frac{1}{2}}.$$

Thus, Coppersmith's method succeeds to recover $x_0 = p + q - 1$, from which we derive N's factorization. □

1.4 Multivariate Applications – Small Secret Exponent RSA

Throughout this section we assume that e is approximately of size N, denoted $e \approx N$. If we choose some small secret key d, then $e = d^{-1} \bmod \phi(N)$ should be randomly distributed, i.e $e \geq \phi(N)/2 \gg N/4$ with probability $\frac{1}{2}$. In fact, all attacks that we study in this section become even more effective (in the sense of larger achievable root bounds), if e is significantly smaller than N. For the ease of notation, we ignore the case of smaller e.

1.4.1 Wiener's Attack.

In 1990, Wiener [46] discovered that for RSA public keys (N, e) with corresponding secret $d < \frac{1}{3}N^{\frac{1}{4}}$ the factorization of N can be found via continued fractions in time $O(\log^3 N)$. We formulate Wiener's result in terms of Coppersmith's method. We have $ed = 1 + k(N + 1 - p - q)$, and therefore the bivariate polynomial

$$f(x, y) = x(N - y) + 1 \bmod e \text{ has root } (x_0, y_0) = (k, p + q - 1).$$

We already know that for RSA moduli with primes p, q of same bit-size we have $y_0 = p + q - 1 < 3N^{\frac{1}{2}}$. Moreover, since $e < \phi(N)$ we have

$$x_0 = k = \frac{ed - 1}{\phi(N)} < d.$$

Notice at this point that $e \ll N$ would imply a smaller upper bound on x_0.

By setting

$$u = 1 - xy \tag{1.8}$$

we linearize $f(x, y)$ as

$f(u, x) = u + xN \bmod e$ with root $(u_0, x_0) = (1 - x_0 y_0, x_0) = (1 - k(p + q - 1), k)$.

Notice that $|u_0| < k(p + q - 1) < 3dN^{\frac{1}{2}}$.

Example: RSA with Small Secret Exponent d

Given:	$N = pq$, e satisfying $ed = 1 \bmod \phi(N)$ with $d < \frac{1}{3}N^{\frac{1}{4}}$
Polynomial:	$f(u, x) = u + xN \bmod \phi(N)$ with root $(u_0, x_0) = (1 - k(p + q - 1), k)$.

Lattice basis: Set $m = 1$. Then all polynomials have root (x_0, y_0) modulo e. Define $X = \frac{1}{3}N^{\frac{1}{4}}$ and $U = 3N^{\frac{3}{4}}$. We also define the collection of polynomials

$$h_1(u, x) = ex, \; h_2(u, x) = f(u, x).$$

The integer linear combinations of the coefficient vectors of $h_1(uU, xX)$ and $h_2(uU, xX)$ form a lattice L with basis

$$B = \begin{pmatrix} eX & \\ NX & U \end{pmatrix}.$$

We have $n = \dim(L) = 2$ and $\det(L) = eUX$. The enabling condition from Equation (1.2) becomes

$$eUX \leq e^2, \text{ which implies } X \leq \frac{e}{3N^{\frac{3}{4}}}.$$

Assuming $e \approx N$, we obtain $X \leq \frac{1}{3}N^{\frac{1}{4}}$. Thus, we recover d in time $O(\log^{1+\epsilon} N)$.

Notice that Coppersmith's method guarantees only that we find a polynomial $g(u, x) = g_0u + g_1x$ from an LLL-reduced shortest lattice vector such that $g(u_0, x_0) = 0$. In our bivariate case, it remains to recover the root (u_0, x_0). We conclude from $g(u_0, x_0) = 0$ that

$$g_0u_0 = -g_1x_0.$$

Since $\gcd(u_0, x_0) = 1$, it follows that $|g_0| = x_0$ and $-|g_1| = u_0$.

1.4.2 Boneh-Durfee Attack with Unraveled Linearization.

Let us start with the polynomial $f(u, x) = u + xN$ as in the description of Wiener's attack. Our goal is to improve on Wiener's $N^{\frac{1}{4}}$-bound.

Example: RSA with Small Secret Exponent d	
Given:	$N = pq$, e satisfying $ed = 1 \bmod \phi(N)$ with $d < N^{0.256}$
Polynomial:	$f(u, x) = u + xN \bmod \phi(N)$ with root $(u_0, x_0) = (1 - k(p + q - 1), k)$.

Lattice basis (1^{st} try): Instead of choosing $m = 1$ as in Wiener's attack, we take $m = 2$ and define the following collection of polynomials with root modulo e^2

$$h_1 = e^2x,\ h_2 = ef(u, x),\ h_3 = e^2x^2,\ h_4 = exf(u, x),\ h_5 = f^2(u, x).$$

This leads to lattice L with basis

$$B = \begin{pmatrix} e^2X & & & & \\ eNX & eU & & & \\ & & e^2X^2 & & \\ & & eNX^2 & eUX & \\ & & N^2X^2 & 2NUX & U^2 \end{pmatrix}.$$

We obtain enabling condition

$$\det(L) = e^6U^4X^4 \leq e^{10}.$$

This is equivalent to the enabling condition $UX \leq e$ from Wiener's attack. It seems that Wiener's bound is stable under increasing m. However, we may also include in our collection the polynomial $h_6 = yf^2(u,x)$. This polynomial has monomials

$$x^2y,\ uxy,\ u^2y.$$

Using Equation (1.8) substitute each occurence of xy by $1-u$. Such a substitution is called *unraveled linearization* [20, 21], and leaves us with monomials

$$x,\ ux,\ u,\ u^2,\ u^2y,$$

of which only the last monomial u^2y does not appear in $h_1, \ldots, h_5$.

Unfortunately, the new collection is not yet good enough for improving the $\frac{1}{4}$-bound. However, we succeed with *unraveled linearization* when using larger m.

Lattice basis (2^{nd} try): Let us construct a similar lattice basis with the choice $m = 4$. Let $d < N^\delta$ with $\delta = 0.256$, set $X = N^\delta$ and $U = Y = N^{\frac{1}{2}+\delta}$.

We take the powers of $f^i(u,x)$ for $i = 3, 4$ and in addition the polynomial $yf^4(u,x)$. That is, we take the 10 polynomials

$$\begin{array}{ccccc} e^4x^3, & e^3x^2f, & e^2xf^2, & ef^3, & e^4x^4, \\ e^3x^3f, & e^2x^2f^2, & exf^3, & f^4, & yf^4. \end{array}$$

This leads to lattice L with basis $B =$

$$\begin{pmatrix} e^4X^3 & & & & & & & & & \\ e^3NX^3 & e^3UX^2 & & & & & & & & \\ e^2N^2X^3 & 2e^2NUX^2 & e^2U^2X & & & & & & & \\ eN^3X^3 & 3eN^2UX^2 & 3eNU^2X & eU^3 & & & & & & \\ & & & & e^4X^4 & & & & & \\ & & & & e^3NX^4 & e^3UX^3 & & & & \\ & & & & e^2N^2X^4 & 2e^2NUX^3 & e^2U^2X^2 & & & \\ & & & & eN^3X^4 & 3eN^2UX^3 & 3eNU^2X^2 & eU^3X & & \\ & & & & N^4X^4 & 4N^3UX^3 & 10N^2U^2X^2 & 4NU^3X & U^4 & \\ N^4X^3 & 4N^3UX^2 & 10N^2U^2X & 4NU^3 & & -N^4UX^3 & -4N^3U^2X^2 & -10N^2U^3X & -4NU^3 & U^4Y \end{pmatrix}.$$

We obtain $\dim(L) = 10$ and $\det(L) = e^{20}U^{20}X^{16}Y$. For $e \approx N$, our enabling condition becomes

$$e^{20}U^{20}X^{16}Y \leq e^{40} \text{ which implies } N^\delta \leq N^{\frac{19}{74}} \approx N^{0.256}.$$

Thus, we find via lattice reduction a polynomial with root (u_0, x_0, y_0) over $\mathbb{Z}$ in time $O(\log^{1+\epsilon} N)$. Using $u = 1 - xy$ from Equation (1.8) gives us a polynomial in x, y only. In fact, one can show that under the same enabling condition LLL reduction provides two polynomials with the desired root over the integers. We

then compute $y_0 = p + q - 1$ from both polynomials via resultant computation. From y_0 we easily obtain the factorization of N.

In general, one can improve the bound for d to $N^{1-\sqrt{\frac{1}{2}}} \approx N^{0.292}$.

Theorem 1.13 (Boneh-Durfee '99 [10]) *Let (N, e) be an RSA modulus with $ed = 1 \bmod \phi(N)$, $e \approx N$ and $d < N^{0.292}$. Then N can be factored in time $O(\log^{6+\epsilon} N)$ for any $\epsilon > 0$.*

Proof For a proof we directly refer to Boneh, Durfee [10]. A slightly simpler proof using the unraveled linearization strategy from above can be found in [21]. □

If e is significantly smaller than N, then one obtains larger bounds on d in Theorem 1.13. On the other hand, if $e > N^{1.875}$ then the attack of Theorem 1.13 no longer works for any secret d.

Whether the $N^{0.292}$ bound can be improved, or whether it is optimal is one of the major open problems in Coppersmith-type cryptanalysis. Boneh and Durfee [10] argued that their polynomial has a unique solution for all $d < N^{\frac{1}{2}}$. Clearly, it is a necessary requirement for the efficiency of Coppersmith's method that there are not exponentially many solutions, since we have to output all of them. However, even a unique solution does not imply that we can efficiently find it.

In fact, there have been serious efforts to improve the Boneh-Durfee attack [2, 21, 26, 42] indicating that the attack is either optimal, or that significantly new techniques have to be developed for an improvement.

Ernst et al. [16] and Takayasu, Kunihiro [40, 41, 43] showed that the Boneh-Durfee attack admits a smooth extension to Partial Key Exposure attacks, where one gets a constant fraction of d's most significant bits. If $d \leq N^{0.292}$ then we need no bits at all, while with growing d we need a larger known fraction for recovering d (and the factorization) in polynomial time. Eventually, for full size d we need all bits of d, coinciding with the result of Theorem 1.12.

1.4.3 Small CRT exponents

Since Wiener's small d attack in 1990, it was an open problem whether there exist polynomial time attacks for RSA secret keys d with a small Chinese Remainder representation, i.e. $d_p = d \bmod p - 1$ and $d_q = d \bmod q - 1$ are both small. This is a question with practical significance, since for performance reasons almost all real-world RSA implementations compute the RSA decryption function using CRT-exponents (d_p, d_q).

The first polynomial time attack on CRT-exponents [30] was described in

2003, but only worked for RSA with (significanty) imbalanced prime factors p, q. This was improved in 2005 by Bleichenbacher and May [5], but their attack still required (less) imbalanced prime factors or small e. Here, we review the idea of the Bleichenbacher-May attack, since it is the basis for later improvements.

We start with the set of equations

$$\begin{aligned} ed_p &= 1 + k(p-1) \\ ed_q &= 1 + \ell(q-1). \end{aligned} \tag{1.9}$$

and rewrite these as

$$\begin{aligned} ed_p + k - 1 &= kp \\ ed_q + \ell - 1 &= \ell q. \end{aligned}$$

Multiplication of both equations leads to the identity

$$e^2 d_p d_q + e(d_p(\ell-1) + d_q(k-1)) + k\ell(1-N) - (k+\ell-1) = 0 \tag{1.10}$$

with four unknowns d_p, d_q, k, ℓ. Let $d_p, d_q < N^\delta$. Then k (and analogously ℓ) can be upper-bounded as

$$k = \frac{ed_p - 1}{p-1} \leq N^{\frac{1}{2}+\delta}.$$

Let us take Equation (1.10) modulo e^2 and work with the linear polynomial equation

$$f(x,y,z) = ex + y(1-N) - z \text{ with root } (x_0, y_0, z_0) = (d_p(\ell-1) + d_q(k-1), k\ell, k+\ell-1). \tag{1.11}$$

Choose $X = N^{\frac{1}{2}+2\delta}, Y = N^{1+2\delta}, Z = N^{\frac{1}{2}+\delta}$. We take the collection of three polynomials

$$e^2 x,\ e^2 y,\ f(x,y,z)$$

that lead to a lattice L with basis

$$\begin{pmatrix} e^2 X & & \\ & e^2 Y & \\ eX & (1-N)Y & -Z \end{pmatrix}.$$

Our enabling condition gives us

$$e^4 XYZ \leq e^6, \text{ resulting in } \delta \leq 0.$$

Thus, the Bleichenbacher-May attack does not succeed for full-size e (but for e that are significantly smaller than N).

However, the identity of Equation (1.10) was used in 2007 by Jochemsz and

May [24] to show an attack on CRT exponents $d_p, d_q \leq N^{0.073}$. Unfortunately, to the best of our knowledge dimension 30 is the smallest lattice dimension for which the Jochemsz-May attack provides a positive bound, and the attack uses Coppersmith's method over the integers (rather than the modular approach that we used in this work). Therefore, we omit a concrete lattice basis here.

Theorem 1.14 (Jochemsz-May 2005[24]) *Let (N, e) be an RSA modulus with $ed = 1 \bmod \phi(N)$ satisfying $d_p = d \bmod p - 1, d_q = d \bmod q - 1 \leq N^{0.073}$. Then the factorization of N can be found in time $\tilde{O}(\log^{6+\epsilon} N)$ for any $\epsilon > 0$.*

The Jochemsz-May attack was optimized in [21] using smaller lattice bases that still asymptotically achieve the same bound $N^{0.073}$. This optimization indicated that there is no possible improvement using solely Equation (1.10).

However in 2017, Takayasu, Lu and Peng used all three polynomial equations from Equation (1.9) and Equation (1.10) to further improve the bound to $N^{0.091}$ [44], and later even to an impressive $N^{0.122}$ [45].

Theorem 1.15 (Takayasu, Lu and Peng [45]) *Let (N, e) be an RSA modulus with $ed = 1 \bmod \phi(N)$ satisfying $d_p = d \bmod p-1, d_q = d \bmod q-1 \leq N^{0.122}$. Then the factorization of N can be found in time $\tilde{O}(\log^{6+\epsilon} N)$ for any $\epsilon > 0$.*

1.5 Open Problems and Further Directions

1.5.1 Optimal Use of Coppersmith's Method.

We have seen that Coppersmith's method leads to powerful results, where all roots within an exponentially-sized search space can be found in polynomial time. In the last sections we derived polynomial equations f, defined certain polynomial collections using algebraic shifts of f, and sometimes further optimized using back-substitution via unraveled linearization.

All this is currently handcrafted, and it is unclear how to find optimal strategies for any of the above steps.

Initial polynomial selection. The most important step is to define the initial polynomial(s) to work with. The results of Section 1.4.3 illustrate the importance of the polynomial selection. If one takes into account the polynomial from Equation (1.11) only, then there seems to be no hope to improve upon the $N^{0.073}$ bound from Theorem 1.14. However, if we also take the polynomials from Equation (1.9) then we get the significantly improved $N^{0.122}$ bound from Theorem 1.15.

In general, we are not aware of any strategy for properly selecting the initial polynomial(s). Different polynomial choices for the same problem lead to different Newton polytopes of the polynomial, which in turn lead to different optimization in Coppersmith's method, as shown in [6].

In fact, some improved bounds in literature stem (solely) from a more clever polynomial selection.

Shift selection, theoretically. After polynomial selection of f, one has to find an optimal collection of polynomials. This is usually a major part of the analysis of Coppersmith's method. The only strategy we are aware of is due to Jochemsz and May [23], and is based on f's Newton polytope. A polynomial's Newton polytope is the convex hull of the exponent vectors of f's monomials. E.g. if $f(x,y) = ax^2 + bxy + cy^2 + d$ then the convex hull of its exponent vectors $(2,0),(1,1),(0,2),(0,0)$ forms a triangle in $\mathbb{Z}^2$.

In a nutshell, the tradeoff that has to be optimized in Coppersmith's method is that one has to introduce as many shifts as possible, while keeping the newly introduced monomials as small as possible. This is realized in the Jochemsz-May method by shifting with the points in the interior of the Newton polytope.

However, to obtain optimal bounds one also has to shift in the direction of smaller unknown. E.g. let X, Y be upper bounds for the root x_0, y_0 of a polynomial $f(x,y)$. If $X \ll Y$ then it is beneficial to additional shift in x-direction.

This optimization is by now a quite handcrafted process. Can we derive an optimal algorithm that on input of a polynomial including (parameterized) upper bounds on the desired root outputs an optimal collection of shifts together with the resulting maximized root bound parameter(s)?

The ingenuity of LLL reduction: Shift selection, practically. From a practical perspective, the optimal selection of polynomials can be delegated to lattice reduction [34]. To this end, we use as many shifts as possible, and look which vectors are chosen by LLL reduction as a linear combination of the shortest vectors. These vectors usually form a proper sub-lattice that might admit a better root bound.

There are many examples in the literature, where authors reported that experimentally they were able to find larger roots than were predicted by theory. Usually, such a behaviour is explained by a sub-lattice structure. As an example, in the Boneh-Durfee attack on small secret d one may find experimentally via LLL reduction within the original lattice basis that led to the $N^{0.284}$ bound the sub-lattice that admits the $N^{0.292}$ bound.

While it is nice that LLL automatically optimizes the shift selection, it also

leads to some open questions. In order to find better root bounds (asymptotically), we have to find an optimal shift selection (asymptotically). How do we derive an optimal asymptotic shift selection for arbitrary parameter m in the Coppersmith method, from some optimized shift selection for fixed m? Even if we are capable of finding an optimal asymptotic shift selection, how do we analyze the resulting root bound? The standard approach only handles asymptotic determinant calculation for triangular lattice bases, but in general we do not have triangularity (see [10]).

Use of Unraveled Linearization. Sometimes we know additional relations involving the desired root. E.g. in the analysis of Boneh-Durfee attack (Section 1.4.2) we used a polynomial $f(u, x)$, where $u = 1 - xy$ was related to x. Using this relation improved the root bound, we called this technique Unraveled Linearization.

At the moment, we are not aware of any general strategy that exploits the power of Unraveled Linearization in a systematic way.

Coppersmith's Method: Modular or Integer? Usually, we derive a polynomial equation (system) that we somehow artifically transform into a modular polynomial. As an example, take the system of equations from Equation (1.9) that led to the integer equation from Equation (1.10). For our analysis, we looked for a polynomial root of Equation (1.10) modulo e^2, but we could as well have worked modulo e, or modulo N.

Equation (1.10) is an especially interesting example, since all unknowns are linked. If we work modulo e^2, then the unknown $d_p d_q$ vanishes, but d_p, d_q still appear in the unknown coefficient $(kd_p + \ell d_q)$ of e. So untuitively, by working modulo e^2 we lose information from Equation (1.10). As a consequence, Jochemsz and May chose to work with Coppersmith's method over the integers directly on the polynomial equation from Equation (1.10), without any modular reduction.

Nevertheless, the currently best bound on CRT-exponents from Takayasu, Lu and Peng (see Theorem 1.15) works with Coppersmith's method modulo e^2. Can we also express the Takayasu-Lu-Peng attack [45] in terms of Coppersmith's method over the integers? Does it lead to the same attack bound, or does it even improve? More general, can we express all modular attacks as integer attacks? A result of Blömer and May [6] gives some indication that there are modular attacks that might not be convertable to integer attacks with the same bound, even for the univariate modular case.

1.5.2 Further Directions in Coppersmith-type Cryptanalysis

Optimality of RSA Small Secret Exponent Attacks. Without any doubt, small secret RSA exponent attacks are the best studied cryptanalysis application of Coppersmith's method, and are considered the most exciting results in this area. Yet, we do not have strong guarantees for the optimality of the bounds.

While the $N^{0.292}$ Boneh-Durfee bound consistently resisted serious efforts for improvements within the last two decades [2, 21, 26, 42], a first positive CRT-bound of $N^{0.073}$ (Theorem 1.14) was derived in 2007, and further improved in 2017 to $N^{0.122}$ (Theorem 1.15). Since the interaction of Equation (1.9) and Equation (1.10) is much more involved than for the simple RSA key equation $ed = 1 \bmod \phi(N)$, the optimality of the $N^{0.122}$-bound is much harder to study. Takayasu, Lu, Peng [45] experimentally reported that they found sublattices that admit practically better bounds than theoretically predicted. However, whether this also leads to better asymptotic bounds remains an open problem.

If we do not develop methods to directly prove the optimality of an attack, including the optimality of polynomial selection and shift selection, then one might be at least able to link attacks. E.g. can we derive the Boneh-Durfee bound as a special case of the Takayasu-Lu-Peng attack for the setting $d_p = d_q = d$? This would give us some more confidence in the optimality of Takayasu-Lu-Peng.

Systems of Polynomial Equations. Even for a single polynomial we do not yet fully understand how to optimize Coppersmith's method. The situation becomes much more challenging when we move to systems of polynomials. However, information-theoretically speaking polynomial systems also provide more power to an attacker.

We take as an example (yet again) the RSA secret exponent attack. Although we do not know how to improve the $N^{0.292}$ bound, it was already argued in the original work of Boneh and Durfee [10] that we cannot beat the $N^{0.5}$ bound, since beyond this bound we expect exponentially many roots. Thus, with single polynomials we usually can only find roots within a polynomial fraction of the modulus.

Another example are the RSA Stereotyped Messages from Section 1.3.1, where we recover in polynomial time an unknown $\frac{1}{e}$-fraction of the messages. The situation changes if we move to polynomial equation systems. From the results for univariate polynomial systems in Ritzenhofen, May [32], we derived in Theorem 1.8 a result that basically states that *every* RSA equation with

public exponent e_i yields a $\frac{1}{e_i}$-fraction of the message, and these fractions add up! Thus, if we have sufficiently many equations, we can fully recover the whole message in polynomial time, not just a fraction of the message.

A natural example, where an arbitrary number of polynomial equations appear are Pseudo Random Number generators. In 2009, Hermann and May [20] looked at power generators of the form $s_i = s_{i-1}^3 \bmod N$ and showed that these generators can successfully be attacked if one outputs a $(1 - \frac{1}{e})$-fraction of all $\log N$ bits in each iteration. This implies that the Blum-Blum-Shub generator [7] with $e = 2$ should only output significantly less than half of its bits per iteration.

In 2001, Boneh, Halevi and Howgrave-Graham [11] constructed the so-called Inverse Congruential Pseudo Random Number generator with security based on the Modular Inversion Hidden Number Problem (MIHNP). In the MIHNP one obtains samples of the form

$$(t_i, \mathrm{MSB}_\delta(\alpha + t_i)^{-1} \bmod p)$$

for some random $t_i \in \mathbb{Z}_p$, where MSB_δ reveals the δ most significant bits. The goal in MIHNP is to recover the hidden number $\alpha \in \mathbb{Z}_p$.

In 2019, Xu, Sarkar, Lu, Wang and Pan [47] showed the really impressive result that for any constant fraction $\frac{1}{d}$ of output bits per iteration, given n^d MIHNP samples one can solve MIHNP in polynomial time.

Theorem 1.16 (Xu, Sarkar, Lu, Wang, Pan [47]) *The Modular Inversion Hidden Number Problem can be solved for any constant fraction $\frac{\delta}{\log p}$ in polynomial time.*

Notice that this result has a completely new quality from a cryptanalytic perspective. While other cryptanalysis results using Coppersmith's method work in polynomial time only if especially small parameters are chosen, or if some side-channel information reveals parts of the secret, the result of Xu et al. completely breaks MIHNP for any parameter setting. Before [47], it was conjectured in [11] that MIHNP is hard whenever one outputs less than a $\frac{\delta}{\log p} = \frac{1}{3}$-fraction of the bits, a quite typical result using Coppersmith's method.

We strongly believe that a careful understanding of Coppersmith's method applied to systems of polynomial equations might lead to similar groundbreaking cryptanalysis results in future.

Acknowledgement. The author thanks the anonymous reviewer for a thorough proofreading.

References

[1] Ajtai, Miklós. 1998. The Shortest Vector Problem in L2 is NP-hard for Randomized Reductions (Extended Abstract). Pages 10–19 of: *30th ACM STOC*. Dallas, TX, USA: ACM Press.

[2] Bauer, Aurélie, and Joux, Antoine. 2007. Toward a Rigorous Variation of Coppersmith's Algorithm on Three Variables. Pages 361–378 of: Naor, Moni (ed), *EUROCRYPT 2007*. LNCS, vol. 4515. Barcelona, Spain: Springer, Heidelberg, Germany.

[3] Bellare, Mihir, and Rogaway, Phillip. 1995. Optimal Asymmetric Encryption. Pages 92–111 of: Santis, Alfredo De (ed), *EUROCRYPT'94*. LNCS, vol. 950. Perugia, Italy: Springer, Heidelberg, Germany.

[4] Bernstein, Daniel J., Chang, Yun-An, Cheng, Chen-Mou, Chou, Li-Ping, Heninger, Nadia, Lange, Tanja, and van Someren, Nicko. 2013. Factoring RSA Keys from Certified Smart Cards: Coppersmith in the Wild. Pages 341–360 of: Sako, Kazue, and Sarkar, Palash (eds), *ASIACRYPT 2013, Part II*. LNCS, vol. 8270. Bengalore, India: Springer, Heidelberg, Germany.

[5] Bleichenbacher, Daniel, and May, Alexander. 2006. New Attacks on RSA with Small Secret CRT-Exponents. Pages 1–13 of: Yung, Moti, Dodis, Yevgeniy, Kiayias, Aggelos, and Malkin, Tal (eds), *PKC 2006*. LNCS, vol. 3958. New York, NY, USA: Springer, Heidelberg, Germany.

[6] Blömer, Johannes, and May, Alexander. 2005. A Tool Kit for Finding Small Roots of Bivariate Polynomials over the Integers. Pages 251–267 of: Cramer, Ronald (ed), *EUROCRYPT 2005*. LNCS, vol. 3494. Aarhus, Denmark: Springer, Heidelberg, Germany.

[7] Blum, Lenore, Blum, Manuel, and Shub, Mike. 1986. A Simple Unpredictable Pseudo-Random Number Generator. *SIAM Journal on Computing*, **15**(2), 364–383.

[8] Boneh, Dan. 2000. Finding smooth integers in short intervals using CRT decoding. Pages 265–272 of: *32nd ACM STOC*. Portland, OR, USA: ACM Press.

[9] Boneh, Dan. 2001. Simplified OAEP for the RSA and Rabin Functions. Pages 275–291 of: Kilian, Joe (ed), *CRYPTO 2001*. LNCS, vol. 2139. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[10] Boneh, Dan, and Durfee, Glenn. 1999. Cryptanalysis of RSA with Private Key d Less than $N^{0.292}$. Pages 1–11 of: Stern, Jacques (ed), *EUROCRYPT'99*. LNCS, vol. 1592. Prague, Czech Republic: Springer, Heidelberg, Germany.

[11] Boneh, Dan, Halevi, Shai, and Howgrave-Graham, Nick. 2001. The Modular Inversion Hidden Number Problem. Pages 36–51 of: Boyd, Colin (ed), *ASIACRYPT 2001*. LNCS, vol. 2248. Gold Coast, Australia: Springer, Heidelberg, Germany.

[12] Cohn, Henry, and Heninger, Nadia. 2011. Ideal forms of Coppersmith's theorem and Guruswami-Sudan list decoding. Pages 298–308 of: Chazelle, Bernard (ed), *ICS 2011*. Tsinghua University, Beijing, China: Tsinghua University Press.

[13] Coppersmith, Don. 1996a. Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. Pages 178–189 of: Maurer, Ueli M. (ed), *EUROCRYPT'96*. LNCS, vol. 1070. Saragossa, Spain: Springer, Heidelberg, Germany.

[14] Coppersmith, Don. 1996b. Finding a Small Root of a Univariate Modular Equation. Pages 155–165 of: Maurer, Ueli M. (ed), *EUROCRYPT'96*. LNCS, vol. 1070. Saragossa, Spain: Springer, Heidelberg, Germany.

[15] Coppersmith, Don. 1997. Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. *Journal of Cryptology*, **10**(4), 233–260.

[16] Ernst, Matthias, Jochemsz, Ellen, May, Alexander, and de Weger, Benne. 2005. Partial Key Exposure Attacks on RSA up to Full Size Exponents. Pages 371–386 of: Cramer, Ronald (ed), *EUROCRYPT 2005*. LNCS, vol. 3494. Aarhus, Denmark: Springer, Heidelberg, Germany.

[17] Girault, Marc, Toffin, Philippe, and Vallée, Brigitte. 1990. Computation of Approximate L-th Roots Modulo n and Application to Cryptography. Pages 100–117 of: Goldwasser, Shafi (ed), *CRYPTO'88*. LNCS, vol. 403. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[18] Gruber, Peter M, and Lekkerkerker, Cornelis Gerrit. 1987. *Geometry of numbers*. North-Holland.

[19] Håstad, Johan. 1986. On Using RSA with Low Exponent in a Public Key Network. Pages 403–408 of: Williams, Hugh C. (ed), *CRYPTO'85*. LNCS, vol. 218. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[20] Herrmann, Mathias, and May, Alexander. 2009. Attacking Power Generators Using Unravelled Linearization: When Do We Output Too Much? Pages 487–504 of: Matsui, Mitsuru (ed), *ASIACRYPT 2009*. LNCS, vol. 5912. Tokyo, Japan: Springer, Heidelberg, Germany.

[21] Herrmann, Mathias, and May, Alexander. 2010. Maximizing Small Root Bounds by Linearization and Applications to Small Secret Exponent RSA. Pages 53–69 of: Nguyen, Phong Q., and Pointcheval, David (eds), *PKC 2010*. LNCS, vol. 6056. Paris, France: Springer, Heidelberg, Germany.

[22] Howgrave-Graham, Nick. 1997. Finding Small Roots of Univariate Modular Equations Revisited. Pages 131–142 of: Darnell, Michael (ed), *6th IMA International Conference on Cryptography and Coding*. LNCS, vol. 1355. Cirencester, UK: Springer, Heidelberg, Germany.

[23] Jochemsz, Ellen, and May, Alexander. 2006. A Strategy for Finding Roots of Multivariate Polynomials with New Applications in Attacking RSA Variants. Pages 267–282 of: Lai, Xuejia, and Chen, Kefei (eds), *ASIACRYPT 2006*. LNCS, vol. 4284. Shanghai, China: Springer, Heidelberg, Germany.

[24] Jochemsz, Ellen, and May, Alexander. 2007. A Polynomial Time Attack on RSA with Private CRT-Exponents Smaller Than N^{0}.073. Pages 395–411 of: Menezes, Alfred (ed), *CRYPTO 2007*. LNCS, vol. 4622. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[25] Kakvi, Saqib A., Kiltz, Eike, and May, Alexander. 2012. Certifying RSA. Pages 404–414 of: Wang, Xiaoyun, and Sako, Kazue (eds), *ASIACRYPT 2012*. LNCS, vol. 7658. Beijing, China: Springer, Heidelberg, Germany.

[26] Kunihiro, Noboru, Shinohara, Naoyuki, and Izu, Tetsuya. 2012. A Unified Framework for Small Secret Exponent Attack on RSA. Pages 260–277 of: Miri, Ali, and Vaudenay, Serge (eds), *SAC 2011*. LNCS, vol. 7118. Toronto, Ontario, Canada: Springer, Heidelberg, Germany.

[27] Lenstra, Arjen K. 1998. Generating RSA Moduli with a Predetermined Portion.

Pages 1–10 of: Ohta, Kazuo, and Pei, Dingyi (eds), *ASIACRYPT'98*. LNCS, vol. 1514. Beijing, China: Springer, Heidelberg, Germany.

[28] Lenstra, Arjen K., Hughes, James P., Augier, Maxime, Bos, Joppe W., Kleinjung, Thorsten, and Wachter, Christophe. 2012. Public Keys. Pages 626–642 of: Safavi-Naini, Reihaneh, and Canetti, Ran (eds), *CRYPTO 2012*. LNCS, vol. 7417. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[29] Lenstra, Hendrik Willem, Lenstra, Arjen K, Lovfiasz, L, et al. 1982. Factoring polynomials with rational coeficients.

[30] May, Alexander. 2002. Cryptanalysis of Unbalanced RSA with Small CRT-Exponent. Pages 242–256 of: Yung, Moti (ed), *CRYPTO 2002*. LNCS, vol. 2442. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[31] May, Alexander. 2009. Using LLL-reduction for solving RSA and factorization problems. Pages 315–348 of: *The LLL algorithm*. Springer.

[32] May, Alexander, and Ritzenhofen, Maike. 2008. Solving Systems of Modular Equations in One Variable: How Many RSA-Encrypted Messages Does Eve Need to Know? Pages 37–46 of: Cramer, Ronald (ed), *PKC 2008*. LNCS, vol. 4939. Barcelona, Spain: Springer, Heidelberg, Germany.

[33] Miller, Gary L. 1976. Riemann's hypothesis and tests for primality. *Journal of computer and system sciences*, **13**(3), 300–317.

[34] Miller, Stephen D., Narayanan, Bhargav, and Venkatesan, Ramarathnam. 2017. *Coppersmith's lattices and "focus groups": an attack on small-exponent RSA*. Cryptology ePrint Archive, Report 2017/835. `http://eprint.iacr.org/2017/835`.

[35] Nemec, Matús, Sýs, Marek, Svenda, Petr, Klinec, Dusan, and Matyas, Vashek. 2017. The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli. Pages 1631–1648 of: Thuraisingham, Bhavani M., Evans, David, Malkin, Tal, and Xu, Dongyan (eds), *ACM CCS 2017*. Dallas, TX, USA: ACM Press.

[36] Neumaier, Arnold, and Stehlé, Damien. 2016. Faster LLL-type reduction of lattice bases. Pages 373–380 of: *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*.

[37] Rivest, Ronald L., Shamir, Adi, and Adleman, Leonard M. 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the Association for Computing Machinery*, **21**(2), 120–126.

[38] Shoup, Victor. 2001. OAEP Reconsidered. Pages 239–259 of: Kilian, Joe (ed), *CRYPTO 2001*. LNCS, vol. 2139. Santa Barbara, CA, USA: Springer, Heidelberg, Germany.

[39] Smeets, Ionica, Lenstra, Arjen K., Lenstra, Hendrik, Lovász, László, and van Emde Boas, Peter. 2010. The History of the LLL-Algorithm. ISC. Springer, Heidelberg, Germany.

[40] Takayasu, Atsushi, and Kunihiro, Noboru. 2014. Partial Key Exposure Attacks on RSA: Achieving the Boneh-Durfee Bound. Pages 345–362 of: Joux, Antoine, and Youssef, Amr M. (eds), *SAC 2014*. LNCS, vol. 8781. Montreal, QC, Canada: Springer, Heidelberg, Germany.

[41] Takayasu, Atsushi, and Kunihiro, Noboru. 2015. Partial Key Exposure Attacks on CRT-RSA: Better Cryptanalysis to Full Size Encryption Exponents. Pages

518–537 of: Malkin, Tal, Kolesnikov, Vladimir, Lewko, Allison Bishop, and Polychronakis, Michalis (eds), *ACNS 15*. LNCS, vol. 9092. New York, NY, USA: Springer, Heidelberg, Germany.

[42] Takayasu, Atsushi, and Kunihiro, Noboru. 2016. How to Generalize RSA Cryptanalyses. Pages 67–97 of: Cheng, Chen-Mou, Chung, Kai-Min, Persiano, Giuseppe, and Yang, Bo-Yin (eds), *PKC 2016, Part II*. LNCS, vol. 9615. Taipei, Taiwan: Springer, Heidelberg, Germany.

[43] Takayasu, Atsushi, and Kunihiro, Noboru. 2017. A Tool Kit for Partial Key Exposure Attacks on RSA. Pages 58–73 of: Handschuh, Helena (ed), *CT-RSA 2017*. LNCS, vol. 10159. San Francisco, CA, USA: Springer, Heidelberg, Germany.

[44] Takayasu, Atsushi, Lu, Yao, and Peng, Liqiang. 2017. Small CRT-Exponent RSA Revisited. Pages 130–159 of: Coron, Jean-Sébastien, and Nielsen, Jesper Buus (eds), *EUROCRYPT 2017, Part II*. LNCS, vol. 10211. Paris, France: Springer, Heidelberg, Germany.

[45] Takayasu, Atsushi, Lu, Yao, and Peng, Liqiang. 2019. Small CRT-exponent RSA revisited. *Journal of Cryptology*, **32**(4), 1337–1382.

[46] Wiener, Michael J. 1990. Cryptanalysis of Short RSA Secret Exponents (Abstract). Page 372 of: Quisquater, Jean-Jacques, and Vandewalle, Joos (eds), *EUROCRYPT'89*. LNCS, vol. 434. Houthalen, Belgium: Springer, Heidelberg, Germany.

[47] Xu, Jun, Sarkar, Santanu, Hu, Lei, Wang, Huaxiong, and Pan, Yanbin. 2019. New Results on Modular Inversion Hidden Number Problem and Inversive Congruential Generator. Pages 297–321 of: Boldyreva, Alexandra, and Micciancio, Daniele (eds), *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part I*. Lecture Notes in Computer Science, vol. 11692. Springer.