On the Security of Some Variants of RSA

by

M. Jason Hinek

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2007

Abstract

The RSA cryptosystem, named after its inventors, Rivest, Shamir and Adleman, is the most widely known and widely used public-key cryptosystem in the world today. Compared to other public-key cryptosystems, such as elliptic curve cryptography, RSA requires longer keylengths and is computationally more expensive. In order to address these shortcomings, many variants of RSA have been proposed over the years. While the security of RSA has been well studied since it was proposed in 1977, many of these variants have not. In this thesis, we investigate the security of five of these variants of RSA. In particular, we provide detailed analyses of the best known algebraic attacks (including some new attacks) on instances of RSA with certain special private exponents, multiple instances of RSA sharing a common small private exponent, Multi-prime RSA, Common Prime RSA and Dual RSA.

Acknowledgements

There are many people that I would to thank for their support, encouragement and friendship throughout my doctoral studies and in particular during the making of this thesis.

First I would like to thank my supervisors, Doug Stinson and Mark Giesbrecht, for supporting me both academically and financially throughout my Ph.D. program. I am especially thankful for all the advice and help that they have offered.

Next I would like to thank the other members of my examination committee, Ian F. Blake, David Jao, Alfred Menezes and Jeffrey O. Shallit, for their numerous comments and helpful suggestions that have improved the quality of this thesis.

My doctoral studies have been partly supported by an Ontario Graduate Scholarship, a GO-Bell scholarship and a University of Waterloo Thesis Completion Award. In addition, I have also received a Graduate Studies Office Research Travel Assistantship and numerous teaching assistantships from the David R. Cheriton School of Computer Science. I would like to gratefully acknowledge all of this assistance.

I have been very fortunate to have the support and friendship of many different people from different communities on campus. From the Centre for Applied Cryptographic Research, I would like thank Omran Ahmadi, Ken Giuliani, Charles Lam, Atefeh Mashatan and Berkant Ustaoğlu. From Minota Hagey Residence, I would like to thank Shalini Aggarwal, Yongtao Hu, Anmar Khadra, Zhenpeng Qu, Claude-Guy Quimper and Antoine Vella. From St. Paul's Graduate Apartments, I would like to thank Misato Sekita-Krislock and Nathan Krislock. And for friendship throughout, I thank Lisa Mac Donald and Sam Hoda.

For the majority of my Ph.D. studies, Petr Olsar was my office mate and I couldn't have asked for a better person to share 3501 with. I would like to thank Petr for being a good friend and for the many tennis games.

Once again a special thank you goes to Tom Holly and Sam Lam for their friendship and all the emails that brightened up each day. An additional thank you goes to Sam for his quick editing and the excellent artwork that he provided for my thesis defence (again!).

My family has always been very supportive of my education and I thank them all for this. Special mention, however, must be given to Annie and Randy Cecile who have always made the trip up for my convocations; Marion and Ron Waxman for encouragement and all the vans for my many moves; Patrick Wansbrough for always being enthusiastic about higher education, being supportive of me and my studies and for his friendship; and to my grandmothers, Maria Hinek and Theresa Wansbrough, for their unending love and support.

From my extended family, I would like to thank Catherine (Aunt Kay) McCarthy. Her love of reading and knowledge has left a lasting impression on me from a very young age. Even though she is no longer with us, she lives on in my memory and I thank her for being a role model and for the friendship and love she always showed me.

Closest to my heart has been Sunita Sarkar. As my family in Waterloo, she has been my partner and my best friend. I thank her for all the love and support and perhaps most importantly the patience that she has always shown me. And of course there is Chyanne, who has brightened my life since she has entered it. I thank her for the love that only a cat can give.

As always, I am indebted to my parents, Nancy and Steve Hinek, for the constant love and support that they have always given me. I cannot thank them enough.

To my parents, Nancy and Steve

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Cryptographic Preliminaries

The RSA cryptosystem, named after its inventors Rivest, Shamir and Adleman [102], is the most widely known and widely used public-key cryptosystem in the world today.

Of particular importance, RSA is one of the public-key cryptosystems used in the Transport Layer Security (TLS) protocol and its predecessor, the Secure Sockets Layer (SSL) protocol, which are used to provide secure communications on the Internet. Essentially, RSA is used to transmit a session key for a symmetric cryptosystem which will be used for the remainder of the communication. In fact, VeriSign, the world's largest SSL certificate dealer, uses RSA for the public-key component in all of the certificates that they have ever issued[1].

Even though RSA is the most used cryptosystem in the world today, when compared to other public-key cryptosystems such as elliptic curve cryptography (ECC), RSA requires longer keylengths and is computationally more expensive. As computing devices become increasingly smaller (*e.g.*, hand-held devices and smartcards) and keylengths become increasingly larger (to offer an adequate level of security), these shortcomings of RSA become increasingly problematic. Eventually, the computational performance that RSA can achieve will no longer be adequate for constrained devices and another public-key cryptosystem will have to replace RSA as

[1]As of April 2007, VeriSign has issued more than $500,000$ certificates. Currently, there are about $61,000$ sites secured with active VeriSign certificates and it is estimated that the VeriSign Secured Seal is viewed on average 80-100 million times daily.

the standard[2]. Of course, the scale of this change is very large and will take years to accomplish. In the meantime, it is possible to obtain modest improvements in computational performance by simply replacing RSA with one of the many suggested variants of RSA (which are designed to be more efficient than RSA in some manner). Most of these variants can inter-operate with systems that were designed to use the original RSA and so migrating to these is a relatively simple task.

The idea of modifying the original form of RSA is not new. In fact, in the patent for RSA [103] (filed in 1977), Rivest, Shamir and Adleman comment that

> "In alternative embodiments, the present invention may use a modulus n which is a product of three or more primes (not necessarily distinct). Decoding may be performed modulo each of the prime factors of n and the results combined using "Chinese remaindering" or any equivalent method to obtain the result modulo n."

While RSA is a very well-studied public-key cryptosystem, the security of most of the variants of RSA are not well studied. It is the aim of this thesis to investigate the security of some of these variants of RSA. In particular, we will analyze the security of several variants of RSA with respect to algebraic attacks.

1.1 Overview of Thesis

We give a brief overview of the remainder of this work.

In the remainder of this chapter, we review the definition of public-key cryptosystems, discuss the RSA cryptosystem and introduce some notation and assumptions that will be used throughout the thesis.

In Chapter 2, we provide all the mathematical background needed for the attacks in the remainder of the thesis. In particular, some results from the theory of continued fractions and lattice basis reduction are presented.

In Chapter 3, we consider instances of RSA with private exponents that are close to a fractional multiple of $\lambda(N) = \mathrm{lcm}(p-1, q-1)$, where $N = pq$ is the RSA modulus and p and q are primes. RSA with small private exponent is a special case of this scenario.

[2]This change is inevitable since it will be economically driven. In Canada alone, Internet sales have exceeded \$100 billion since 2000 [110].

In Chapter 4, we present a lattice-based extension of Wiener's small private exponent attack. The attack applies to multiple instances of RSA each having a common small private exponent.

In Chapter 5, we analyze the security of Multi-prime RSA, a variant of RSA in which the modulus contains more than two distinct primes.

In Chapter 6, we introduce and investigate the security of a variant of RSA called Common Prime RSA. This is work that we presented at the RSA conference in 2006.

In Chapter 7, we present some attacks on a new variant of RSA called Dual RSA. This variant, essentially, consists of two unique instances of RSA with common public and private exponents (but different moduli).

Finally, in Chapter 8, we give some concluding remarks and suggest some possible directions for future work.

1.2 Public-Key Cryptography

The notion of public-key cryptography was first publicly developed and introduced by Diffie and Hellman [42] and Merkle [85] in the mid 1970's. We refer the reader to Diffie [41] for a history of the beginnings of public-key cryptography. We will use the following definition for a public-key cryptosystem.

A *public-key cryptosystem* is a five-tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where the following conditions are satisfied:

1. $\mathcal{P}$ is a finite set of possible *plaintexts*.
2. $\mathcal{C}$ is a finite set of possible *ciphertexts*.
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*.
4. For each key $K \in \mathcal{K}$, there is an *encryption rule* $enc_K \in \mathcal{E}$ and a corresponding *decryption rule* $dec_K \in \mathcal{D}$. Each $enc_K : \mathcal{P} \to \mathcal{C}$ and $dec_K : \mathcal{C} \to \mathcal{P}$ are functions such that $dec_K(enc_K(m)) = m$ for every plaintext element $m \in \mathcal{P}$.
5. for each key $K \in \mathcal{K}$ and each plaintext $m \in \mathcal{P}$, both $enc_K(m)$ and $dec_K(enc_K(m))$ are easy to compute.
6. for almost every key $K \in \mathcal{K}$, each easily computable algorithm equivalent to dec_K is computationally infeasible to derive from enc_K.

7. The encryption rule enc_K is made public and the decryption rule dec_K is kept private.

Alternatively, we can think of a public-key cryptosystem as consisting of three efficiently computable algorithms: a key generation algorithm, an encryption algorithm, and a decryption algorithm. Here, the key generation algorithm (either implicitly or explicitly) defines the keyspace $\mathcal{K}$ and the encryption and decryption algorithms define the plaintext and ciphertext spaces $\mathcal{P}$ and $\mathcal{C}$.

1.3 The RSA Cryptosystem

The RSA cryptosystem was the first publicly known public-key cryptosystem. Introduced in 1977 in a Scientific American article written by Gardner [49], the full research paper was published a year later by its inventors Rivest, Shamir and Adleman [102]. The cryptosystem was originally called the MIT public-key cryptosystem, in reference to the authors' affiliation at the time.

Using the definition of a public-key cryptosystem given above, we can specify the original version of RSA, often called textbook RSA, as follows.

For *textbook RSA*, let $N = pq$, where p and q are primes, let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N$ (the integers modulo N) and define the keyspace as

$$\mathcal{K} = \{(N, p, q, e, d) : ed \equiv 1 \pmod{\phi(N)}\},$$

where $\phi(N) = (p-1)(q-1)$ is Euler's totient function. For each key $K = (N, p, q, e, d)$, the encryption rule $enc_K : \mathbb{Z}_N \rightarrow \mathbb{Z}_N$ is defined by

$$enc_K(x) = x^e \bmod N,$$

and the decryption rule $dec_K : \mathbb{Z}_N \rightarrow \mathbb{Z}_N$ is defined by

$$dec_K(y) = y^d \bmod N,$$

for $x, y \in \mathbb{Z}_N$. The pair (e, N) is the RSA public key and the triple (d, p, q) is the RSA private key.

As the names suggest, the public key (e, N) is public knowledge while the private key (d, p, q) is kept private. Here, N is called the RSA modulus or simply the modulus, p and q are called the RSA primes, e is called the public or encrypting exponent and d is called the private or decrypting exponent.

The fact that the decryption rule recovers the original plaintext follows from Euler's theorem, which states that

$$a^{\phi(N)} \equiv 1 \pmod{N},$$

for any integer a that is relatively prime to N. Given a public key (e, N) and a plaintext message $m \in \mathbb{Z}_N$ that is relatively prime to N (*i.e.*, $m \in \mathbb{Z}_N^*$), the encryption rule computes the ciphertext

$$c = m^e \bmod N.$$

Using the decryption rule, and the key equation $ed = 1 + k\phi(N)$, we recover the plaintext since

$$\begin{aligned} c^d \bmod N &\equiv (m^e)^d \pmod{N} \\ &\equiv m^{ed} \pmod{N} \\ &\equiv m^{1+k\phi(N)} \pmod{N} \\ &\equiv m(m^{\phi(N)})^k \pmod{N} \\ &\equiv m(1^k) \pmod{N} \\ &\equiv m \pmod{N} \\ &= m. \end{aligned}$$

This argument, using Euler's theorem, only applies to plaintext messages $m \in \mathbb{Z}_N^*$ instead of all messages in the plaintext space $\mathcal{P} = \mathbb{Z}_N$. However, it is easily shown using the Chinese Remainder Theorem, that it holds for all plaintext messages $m \in \mathbb{Z}_N$ (of course, each plaintext message $m \in \mathbb{Z}_N \setminus \mathbb{Z}_N^*$ allows the modulus to be easily factored since $\gcd(c, N)$ will reveal a multiple of one of the primes p or q).

Defining the public and private exponents as inverses modulo $\phi(N)$ gives a sufficient (but not necessary) condition for the encryption rule to recover the plaintext from any ciphertext. The necessary condition is that the public and private exponents be inverses of each other modulo Carmichael's function $\lambda(N) = \text{lcm}(p-1, q-1)$. This follows since Carmichael's function $\lambda(N)$ is, by definition, the smallest number m such

$$a^m \equiv 1 \pmod{N},$$

for any integer a that is relatively prime to N. Thus, it is sufficient to define the public and private exponents as inverses modulo any multiple of $\lambda(N)$.

Of course, $\phi(N)$ is a multiple of $\lambda(N)$ since

$$\begin{aligned}\phi(N) &= (p-1)(q-1)\\ &= \gcd(p-1,q-1)\,\mathrm{lcm}(p-1,q-1)\\ &= \gcd(p-1,q-1)\,\lambda(N),\end{aligned}$$

which allows us to use $\phi(N)$ in the key generation algorithm. However, in current standards and implementations of RSA (*e.g.*, see PKCS #1 [104]) the public and private exponents are defined as inverses modulo $\lambda(N)$.

RSA has the multiplicative property that the encryption of the product of two plaintext messages is the same as the product of the encryptions of the two plaintext messages. That is, for plaintext messages m_1 and m_2, we have

$$enc_K(m_1 m_2) = enc_K(m_1) enc_K(m_2).$$

This property, often called the homomorphic property of RSA, follows from the basic properties of modular multiplication. Exploiting this homomorphic property of RSA, Davida [38] showed that textbook RSA is insecure against a chosen ciphertext attack[3]. A simplification of the attack by Judy Moore[4] is as follows. Suppose an adversary is given a ciphertext $c = m^e \bmod N$ and wants to compute m. Selecting a random $x \in \mathbb{Z}_N$, the adversary asks for the plaintext of the ciphertext $c_0 = cx^e \bmod N$. Since

$$\begin{aligned}m_0 &= {c_0}^d \bmod N\\ &= (cx^e)^d \bmod N\\ &= c^d x^{ed} \bmod N\\ &= mx \bmod N,\end{aligned}$$

the adversary can simply compute $m = m_0 x^{-1} \bmod N$ to recover the desired plaintext.

Another, more practical attack, that uses the homomorphic property of RSA is by Boneh, Joux and Nguyễn [18]. Their attack uses the fact that, in practice, RSA is mostly used to encrypt short messages (generally a session key for a symmetric key cryptosystem). Essentially, their attack is a meet-in-the-middle attack that assumes that the desired ℓ-bit plaintext m can be

[3]Given a ciphertext c, a chosen ciphertext attack allows the adversary to choose a number of ciphertexts, not including c, and obtain their corresponding plaintexts in order to assist in decrypting c.

[4]Denning credits, without reference, the simplification to Moore in [40].

factored into two $\ell/2$-bit factors m_1 and m_2 (*i.e.*, $m = m_1 m_2$). The attack begins by constructing a table containing each $\ell/2$-bit number m_1' and its encryption $(m_1')^e \bmod N$. Then, for each possible $\ell/2$-bit number m_2', the value $c/m_2' \bmod N$ is checked against each encryption in the table. When $m_2' = m_2$, the value of $c/m_2' \bmod N$, given by ${m_1}^e \bmod N$, will be in the table. When this match is found, the factorization of m (and hence m itself) is found. The attack requires as much time as it takes to compute $2^{\ell/2+1}$ modular exponentiations (modulo N), requires $2^{\ell/2}\ell$ bits of memory and succeeds with probability 18% (over the choice of the plaintext m).

These attacks are easily avoided by imposing some structure on the plaintexts. In particular, a proper padding scheme, such OAEP [4], is sufficient. In the remainder of this work, we implicitly assume that a proper padding scheme is used in all variants considered and do not discuss this further.

1.3.1 CRT Decryption

The main drawback to textbook RSA (besides being insecure without a padding scheme) are the long key lengths and the expensive computational costs of key generation, encryption and decryption. In order to reduce the effects of these drawbacks, many variants of RSA have been proposed. One such variation is based on the observation that decryption can be accomplished by first decrypting a ciphertext modulo each of the primes p and q and then combining the results using the Chinese Remainder Theorem (CRT). While the idea of performing decryption using CRT was mentioned as early as 1977 in the RSA patent [103] (when the modulus consists of more than two primes) and in 1979 by Rabin [100], it was not fully appreciated for use in RSA until Quisquater and Couvreur's work in 1982 (see [99]). Essentially, the decryption algorithm is replaced with the following method: Given a ciphertext $c = m^e \bmod N$, first compute

$$\begin{aligned} c_p &= c^{d_p} \bmod p \\ c_q &= c^{d_q} \bmod q, \end{aligned}$$

where $d_p = d \bmod p-1$ and $d_q = d \bmod q-1$, then compute the plaintext message m using Garner's algorithm (*e.g.*, see [84, §14.5])

$$m = c_q + (q^{-1} \bmod p)(c_p - c_q)q.$$

The CRT-exponents, d_p and d_q, and the value $(q^{-1} \bmod p)$ can be precomputed and included in the private key so that they do not need to be computed for each decryption. Assuming quadratic complexity for multiplication, and performing modular exponentiation using the square-and-multiply method (see [84, Chapter 14] for more information), decryption costs can be reduced by a factor of four if the computations are done sequentially, and by a factor of eight if done in parallel and the RSA primes are roughly the same size.

1.3.2 Security of RSA

The security of RSA is based on the difficulty of solving the so-called RSA problem. Given an RSA public key (e, N) and a ciphertext $c = m^e \bmod N$, the *RSA problem* is to compute the plaintext m. Essentially, it is the problem of computing e^{th} roots modulo N. Clearly, if the RSA problem can be solved efficiently then RSA is completely insecure. While there hasn't been a significant amount of research into the difficulty of this problem, it is believed to be difficult to solve. In fact, the security of RSA is based on the *RSA assumption*: the RSA problem is hard to solve when the modulus N is sufficiently large and randomly generated and the plaintext $m \in \mathbb{Z}_N^*$ is randomly chosen.

Since RSA has been publicly disclosed, there has been no evidence (known to the public) to suggest that the RSA assumption is untrue. For more information about the RSA problem, see Rivest and Kaliski [101].

Another problem that is commonly associated with the security of RSA is the well-known integer factorization problem. It is clear that the RSA problem is not harder to solve than the integer factorization problem, because factoring the RSA modulus N allows one to compute the private exponent d, which allows one to efficiently solve the RSA problem. It is not clear, however, if the converse is true. That is, it is unknown if solving the RSA problem allows one to efficiently solve the integer factorization problem. In 1998, Boneh and Venkatesan [20] provided evidence that suggests that the RSA problem may be easier to solve than the integer factorization problem when the public exponent is small (or the product of small factors). In 2005, Brown [21] showed that solving the RSA problem with a straight line program is almost as difficult as the integer factorization problem provided that the public exponent is small or has a small factor.

Even though the RSA problem may be easier to solve than the integer factorization problem, in practice, the security of RSA is always analyzed based on the integer factorization problem. Perhaps one reason for this is that the integer factorization problem is a well-known and well-studied problem. We give a brief overview of the current state of the art in factoring methods below (see Crandall and Pomerance [37] for more information).

The best generic factoring method is Pollard's general number field sieve (NFS). Following Lenstra [72], we will use

$$L[n] = e^{1.923(\log n)^{1/3}(\log\log n)^{2/3}}, \tag{1.1}$$

as the heuristic expected runtime of the NFS to factor a composite number n. Notice that the expected runtime of the NFS is a function of only the size of the number being factored. The largest integer factored using the NFS, as of April 2007 [126], is RSA200, a 200-digit number (665 bits) which was factored in May 2005.

H. Lenstra's elliptic curve method (ECM) for factoring can be substantially faster than the NFS if one of the prime factors of n is significantly smaller than $\sqrt{n}$. Again, following Lenstra [72], we use

$$E[n,p] = (\log_2 n)^2 e^{\sqrt{2}(\log p)^{1/2}(\log\log p)^{1/2}}, \tag{1.2}$$

as the heuristic expected runtime of the ECM to find a factor p of n. Notice that the expected runtime of the ECM is a function of both the smallest factor and the number being factored, with the size of the smallest factor dominating the complexity. The largest factor found with the ECM, as of April 2007 [126], is 66 digits in length (220 bits) which was found in April 2005.

Since the assumed difficulty of the integer factorization problem is the *de facto* security measure for RSA, the level of security for an instance of RSA is determined by the size of the modulus. Currently, 1024-bit moduli are recommended for non-critical encryption. The expected complexity required to factor a 1024-bit RSA modulus is roughly 2^{80} operations, which is currently infeasible.

At present, the best attack on a random instance of RSA is factoring the modulus with the NFS (if the primes are relatively balanced) or the ECM (if the primes are very unbalanced). There are, however, many other attacks that exploit certain key choices (small private exponent, for example), additional information (such as partial key exposure attacks), or implementation details (such as side channel attacks). For more information, see Boneh's survey of attacks on RSA [12], or the more recent attacks by Blömer and May [7, 8, 9], May [80], Ernst, Jochemsz, May and de Weger [46], Jochemsz and May [69, 70], and Acıiçmez, Koç and Seifert [1].

1.4 An Alternate History

There are at least two histories of the advent of public-key cryptography. The usual history is that public-key cryptography was developed by Diffie, Hellman and Merkle in the mid 1970's and that the first public-key cryptosystem was invented by Rivest, Shamir and Adleman in 1977. See Diffie [41] for more details of this history.

A second history, however, was born in 1997, when the British Government Communications Headquarters (GCHQ) unclassified several documents which claimed that the ideas of public-key cryptography had been secretly developed by some of its members in the early 1970's. In particular, it is claimed that the notion of non-secret digital encryption (essentially public-key cryptography without the concept of digital signatures) was developed by Ellis [44] in 1970, the equivalent of the RSA cryptosystem (for encryption only, as digital signatures were not part of non-secret digital encryption) was developed by Cocks [26] in 1973 and what it known as the Diffie-Hellman-Merkle key exchange method was developed by Williamson [124, 125] in 1974. While none of the original documents exist (or have been made public), it is generally acknowledged that this secret development of public-key cryptography did indeed occur. See Ellis [45] for a brief summary of this history.

It has also been claimed that the American National Security Agency (NSA) had developed public-key cryptography in the 1960's. These claims are without substantiation, but make for an interesting read (see Bellovin [5] for more information).

1.5 Assumptions and Notation

In the rest of this thesis, we use the following notation and assumptions unless otherwise stated.

We only consider instances of RSA (and their variants) with balanced primes. By balanced primes, we mean that the primes in the modulus are roughly the same size. In particular, for an RSA modulus $N = pq$, we assume that

$$4 < \frac{1}{2}N^{1/2} < p < N^{1/2} < q < 2N^{1/2}. \tag{1.3}$$

A consequence of this is that Euler's totient function $\phi(N) = (p-1)(q-1)$

satisfies

$$\begin{aligned}
N - \phi(N) &= N - (p-1)(q-1) \\
&= N - (N - p - 1 + 1) \\
&= p + q - 1 \\
&< 3N^{1/2}.
\end{aligned} \tag{1.4}$$

For notational convenience, we define $\Lambda = N - \phi(N) = p + q - 1$. Thus, we have that $|\Lambda| < 3N^{1/2}$.

Even though textbook RSA defines the public and private exponent as inverses modulo Euler's totient function $\phi(N) = (p-1)(q-1)$, as mentioned earlier, the current standards and implementations (*e.g.*, see PKCS #1 [104]) defines the exponents as inverses $\lambda(N) = \operatorname{lcm}(p-1, q-1)$. In the variants of RSA that we consider in this work, the public and private exponents will be defined modulo $\phi(N)$ or modulo $\lambda(N)$. The choice will usually depend on the context in which the variant was introduced.

When referring to RSA, textbook RSA or standard RSA, we mean the following: let $N = pq$, where p and q are balanced primes. Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_N$, and define the keyspace as

$$\mathcal{K} = \{(N, p, q, e, d) : ed \equiv 1 \pmod{\phi(N)}\},$$

or

$$\mathcal{K} = \{(N, p, q, e, d) : ed \equiv 1 \pmod{\lambda(N)}\}.$$

For a key $K = (N, p, q, e, d)$, the encryption rule $enc_K : \mathbb{Z}_N \to \mathbb{Z}_N$ is defined by

$$enc_K(x) = x^e \bmod N,$$

and the decryption rule $dec_K : \mathbb{Z}_N \to \mathbb{Z}_N$ is defined by

$$dec_K(y) = y^d \bmod N,$$

where $x, y \in \mathbb{Z}_N$. If CRT-decryption is used, an alternate decryption rule $dec'_K : \mathbb{Z}_N \to \mathbb{Z}_N$ defined by

$$dec'_K(y) = (y^{d_q} \bmod q) + (q^{-1} \bmod p)(y^{d_p} \bmod p - y^{d_q} \bmod q)q,$$

where $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$ can be used. The RSA public key is the pair (e, N) and the RSA private key is the triple (d, p, q).

Most of the variants of RSA that we consider in this thesis involve variations to the key generation algorithm (*i.e.*, the keyspace is a proper subset of the RSA keyspace). Instead of giving a (semi) formal definition of each variant, we will simply explain what the variation is and possibly give an explicit algorithm if needed.

In several of the documents cited in this thesis we have included a uniform resource locator (URL) for an electronic version of the document (or for a webpage containing the document). Some of these documents only exist online while others are very difficult to obtain otherwise (*e.g.*, non-electronic technical reports from the 1980's). The URL's included have all been tested and are valid as April 2007. Also, since this is an electronic thesis, each of these references are hyperlinked to the source document or webpage.

Chapter 2

Mathematical Preliminaries

In this chapter we provide all the mathematical background needed for the attacks presented in this thesis.

2.1 Some Notation

Here we collect some notation that is needed for the mathematical background reviewed in this chapter. The notation will also be used in the remainder of the thesis unless otherwise noted.

Sets

We use $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$ to denote the set of integers, rational numbers and real numbers, respectively. For any positive integer N, we use $\mathbb{Z}_N$ to denote the ring of integers modulo N.

Vectors

For any vector $x \in \mathbb{R}^n$, we use $\|x\|$ to denote the Euclidean norm and $\|x\|_\infty$ to denote the infinity norm. That is, for any vector $x = (x_1, \ldots, x_n)$, we have

$$\|x\| = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}$$

$$\|x\|_\infty = \max_{1 \le i \le n} (|x_i|).$$

For any two vectors $x, y \in \mathbb{R}^n$, we use $\langle x, y \rangle$ to denote the inner (or dot) product of the two vectors. That is, if $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$

then

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i.$$

Polynomials

For any polynomial $h(x_1, \ldots, x_n)$, we use $\|h(x_1, \ldots, x_n)\|$ to denote the Euclidean norm of the coefficient vector of $h(x_1, \ldots, x_n)$ and $\|h(x_1, \ldots, x_n)\|_\infty$ to denote the infinity norm of the coefficient vector of $h(x_1, \ldots, x_n)$. For any polynomial

$$h(x_1, \ldots, x_n) = \sum_{i_1, \ldots, i_n} a_{i_1, \ldots, i_n} x_1^{i_1} \cdots x_n^{i_n},$$

with m monomials, the coefficient vector of $h(x_1, \ldots, x_n)$ is simply the m-dimensional vector consisting of the m coefficients of $h(x_1, \ldots, x_n)$. The Euclidean and infinity norms of $h(x_1, \ldots, x_n)$ are thus given by

$$\|h(x_1, \ldots, x_n)\| = \left(\sum_{i_1, \ldots, i_n} |a_{i_1, \ldots, i_n}|^2 \right)^{1/2}$$
$$\|h(x_1, \ldots, x_n)\|_\infty = \max_{i_1, \ldots, i_n} \left(|a_{i_1, \ldots, i_n}| \right).$$

The infinity norm of a polynomial is also called the height of a polynomial.

When working with the coefficient vectors of polynomials, we will assume that there is an underlying ordering to the vector components. This ordering will depend on each particular situation, but will be constant throughout that situation. For example, if we are considering the polynomials

$$h_1(x, y) = 1 + 2x + 3xy$$
$$h_2(x, y) = 7 + 2xy + 4x^3,$$

we might order the components of their corresponding coefficient vectors by increasing total degree of each monomial. Thus, the coefficient vectors for h_1 and h_2 would be given by

$$h_1(x, y) \to (1, 2, 3, 0)$$
$$h_2(x, y) \to (7, 0, 2, 4).$$

Asymptotic Notation

When describing the asymptotic nature of a function we use both little-oh and big-oh notation.

A function $f(n)$ is said to be *little-oh* of a function $g(n)$, denoted by $f(n) \in o(g(n))$, if for any constant $c > 0$ there exists a constant $n_0 > 0$ such that $0 \leq f(n) < cg(n)$ for all $n \geq n_0$. Informally, if $f(n) \in o(g(n))$, then $f(n)$ is negligible compared to $g(n)$ for all sufficiently large values of n.

A function $f(n)$ is said to be *big-oh* of a function $g(n)$, denoted $f(n) \in O(g(n))$, if there exist constants $c > 0$ and $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$. Informally, if $f(n) \in O(g(n))$, then $f(n)$ is no larger than some constant multiple of $g(n)$ for all sufficiently large values of n. When using big-oh notation, we will always assume that the function $g(n)$ is the smallest simple function such that $f(n) \in O(g(n))$. Thus, $f(n)$ and $g(n)$ are comparable, up to a constant, for all sufficiently large values of n.

2.2 Continued Fractions

The first mathematical tool that we review is continued fractions. For more information on continued fractions, see Olds [95].

An expression of the form

$$a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{\ddots + \cfrac{1}{a_n}}}},$$

where a_1 is any integer and $a_2, \ldots, a_n$ are positive integers is called a *finite simple continued fraction*, which we will call a continued fraction. As a shorthand for this cumbersome type of expression, we often express a continued fraction as the tuple

$$[a_1, \ldots, a_n].$$

Any rational number p/q can be expanded (or expressed) as a continued fraction

$$\frac{p}{q} = [a_1, \ldots, a_n].$$

Here, the a_i are simply the quotients obtained by computing the greatest common divisor (gcd) of p and q using the Euclidean algorithm. In particular, applying the Euclidean algorithm to p and q we obtain

$$\begin{aligned} p &= a_1 q + r_1 \\ q &= a_2 r_1 + r_2 \\ r_1 &= a_3 r_2 + r_3 \\ &\vdots \\ r_{n-2} &= a_n r_{n-1} + 0. \end{aligned}$$

For $0 \le i \le n$, we define the i^{th} *convergent* of the continued fraction $[a_1, \ldots, a_n]$ to be

$$c_i = [a_1, \ldots, a_i].$$

Each convergent c_i can be expressed as a rational number p_i/q_i.

The main result from the theory of continued fractions that we use in this thesis is the following theorem (see for example [55]).

Theorem 2.1. *Let $\alpha \in \mathbb{Q}$ and $c, d \in \mathbb{Z}$ satisfy*

$$\left|\alpha - \frac{c}{d}\right| < \frac{1}{2d^2}.$$

Then c/d, in lowest terms, is one of the convergents in the continued fraction expansion of α.

This result is applied to situations in which we can derive an equation

$$\alpha = \frac{c}{d} + \beta,$$

where α is the only known quantity and β is small, with the goal of computing c and d (or c/d). From this equation, we have

$$\left|\alpha - \frac{c}{d}\right| = |\beta|,$$

so that whenever $\beta < 1/(2d^2)$, we know from Theorem 2.1 that c/d, in lowest terms, will be one of the convergents in the continued fraction expansion of the known quantity α. This technique is, of course, only useful if there is a way to test for the correct convergent.

2.3 Lattice Background

The majority of attacks that we discuss in this thesis are based on techniques that use lattice basis reduction. In this section we give some basic background information about lattices and then outline the two main lattice-based techniques that we employ. For more information about lattices and cryptography, see the survey by Nguyễn and Stern [94]. For more information about the geometry of numbers, see [22, 54, 108]. For more information about lattices and lattice basis reduction, see [27, 53, 77, 79].

2.3.1 Definitions and Basic Facts

A *lattice* is a discrete (additive) subgroup of $\mathbb{R}^n$. Equivalently, given $m \leq n$ linearly independent vectors $b_1, \ldots, b_m \in \mathbb{R}^n$, the set

$$\begin{aligned} \mathcal{L} &= \mathcal{L}(b_1, \ldots, b_m) \\ &= \left\{ \sum_{i=1}^{m} \alpha_i b_i \ : \ \alpha_i \in \mathbb{Z} \right\}, \end{aligned} \tag{2.1}$$

is a lattice. The b_i are called *basis vectors* of $\mathcal{L}$ and $\mathcal{B} = \{b_1, \ldots, b_m\}$ is called a *lattice basis* for $\mathcal{L}$. Thus, the lattice generated by a basis $\mathcal{B}$ is the set of all integer linear combinations of the basis vectors in $\mathcal{B}$.

The *dimension* (or *rank*) of the a lattice, denoted $\dim(\mathcal{L})$, is equal to the number of vectors making up the basis. The dimension of a lattice is equal to the dimension of the vector subspace spanned by $\mathcal{B}$. A lattice is said to be *full dimensional* (or *full rank*) when $\dim(\mathcal{L}) = n$.

It is often useful to represent a lattice $\mathcal{L}$ by a *basis matrix*. Given a basis $\mathcal{B}$, a basis matrix $\mathcal{M}$ for the lattice generated by $\mathcal{B}$ is simply the $m \times n$ matrix whose ℓ^{th} row is b_ℓ. The lattice can then be described as

$$\mathcal{L} = \{v : v = y\mathcal{M}, y \in \mathbb{Z}^n\}.$$

Similarly, a lattice can be generated by the columns of an $n \times m$ basis matrix whose ℓ^{th} column is b_ℓ (*i.e.*, $\mathcal{L} = \{v : v = \mathcal{M}^T y, y^T \in \mathbb{Z}^m\}$).

Lattices with dimension greater than one (*i.e.*, $m \geq 2$) have infinitely many bases (each with the same cardinality). Given a lattice $\mathcal{L}$ with basis matrix $\mathcal{M}$ and any $m \times m$ unimodular matrix $\mathcal{U}$ (*i.e.*, $\mathcal{U}$ is an integral matrix with $\det(\mathcal{U}) = \pm 1$) then $\mathcal{M}' = \mathcal{U}\mathcal{M}$ is also a basis matrix for $\mathcal{L}$. Any two bases for a lattice $\mathcal{L}$ can be related in this way.

The *volume* (or *determinant*) of a lattice, denoted by $\mathrm{vol}(\mathcal{L})$ is, by definition, the square root of the Gramian determinant[1], which is independent of the particular choice of basis. This definition corresponds to the actual m-dimensional volume of the parallelepiped spanned by the b_i's and leads to the following result, called *Hadamard's inequality*, which relates the volume of a lattice to any of its bases:

$$\mathrm{vol}(\mathcal{L}) \leq \prod_{i=1}^{m} \|b_i\|, \tag{2.2}$$

where equality holds if and only if the basis vectors are mutually orthogonal. When a lattice is full dimensional its volume is given by

$$\mathrm{vol}(\mathcal{L}) = |\det(\mathcal{M})|. \tag{2.3}$$

In this case, it is clear that the volume is independent of the choice of basis since any two basis matrices are related by a unimodular matrix.

If $b_i \in \mathbb{Q}^n$ for all $1 \leq i \leq m$ (*i.e.*, $\mathcal{L}$ is a subgroup of $\mathbb{Q}^n$) then the lattice $\mathcal{L}$ is called a *rational lattice*. If $b_i \in \mathbb{Z}^n$ for all $i \leq i \leq m$ (*i.e.*, $\mathcal{L}$ is a subgroup of $\mathbb{Z}^n$) then the lattice $\mathcal{L}$ is called an *integer lattice*. The volume of a full dimensional integer lattice (with dimension n) is also equal to the index $[\mathbb{Z}^n : \mathcal{L}]$ of $\mathcal{L}$ in $\mathbb{Z}^n$.

Given any lattice of dimension m, for $1 \leq i \leq m$, the i^{th} *successive minima* of $\mathcal{L}$, denoted by $\lambda_i(L)$ are defined to be the radii of the smallest balls centred about the origin (of $\mathbb{R}^n$) such that there exist i linearly independent lattice vectors contained in this ball. That is,

$$\lambda_i(L) = \min_{\substack{v_1,\ldots,v_i \in L \\ \text{lin. ind.}}} \max_{1 \leq j \leq i} \|v_j\|. \tag{2.4}$$

When the lattice is understood, we will sometimes use λ_i to denote the successive minima, in order to simplify the notation. Essentially, λ_i is the i^{th} smallest Euclidean length of any non-zero vector in $\mathcal{L}$. Of special importance is λ_1, which is the Euclidean length of a smallest non-zero vector in $\mathcal{L}$.

Minkowski has shown the following result relating the successive minima to the volume of the a lattice.

Theorem 2.2 (Minkowski's Second Theorem). *For any m-dimensional lattice $\mathcal{L}$, and for all $r \leq m$,*

$$\lambda_1 \lambda_2 \cdots \lambda_r \leq \sqrt{\gamma_m^r}\, \mathrm{vol}(\mathcal{L})^{r/m}, \tag{2.5}$$

where γ_m is Hermite's constant of dimension m.

[1]The Gramian determinant is the determinant of the symmetric Gram matrix $\mathcal{G}$, whose components are defined by $\mathcal{G}(b_1, \ldots, b_m)[i, j] = \langle b_i, b_j \rangle$.

Hermite's constant of dimension m is the supremum of $\lambda_1(\mathcal{L})^2/\mathrm{vol}(\mathcal{L})^{2/m}$ taken over all m dimensional lattices $\mathcal{L}$. That is,

$$\gamma_m = \sup_{\mathcal{L}, \dim(\mathcal{L})=m} \left(\frac{\lambda_1(\mathcal{L})}{\mathrm{vol}(\mathcal{L})^{1/m}} \right)^2 . \tag{2.6}$$

There are only nine values of m such that Hermite's constant γ_m is known. These values are given in the following table:

m	1	2	3	4	5	6	7	8	24
$(\gamma_m)^m$	1	4/3	2	4	8	64/3	64	2^8	4^{24}

For more information, see Gruber and Lekkerkerker [54] for the first eight constants and Cohn and Kumar [28] for $m = 24$. The best known asymptotic bounds for Hermite's constant (see Milnor & Husemoller [89] for the lower bound and Conway & Sloane [31] for the upper bound) are given by

$$\frac{m}{2\pi e} + \frac{\log(\pi m)}{2\pi e} + o(1) \le \gamma_m \le \frac{1.744\, m}{2\pi e}(1 + o(1)), \tag{2.7}$$

where $e \approx 2.71828$ is the base of the natural logarithm (often called Euler's number). In fact, it is always true that $\gamma_m \le m$. Using this simple bound for γ_m in Theorem 2.2, with $m = 1$, yields the following result about the smallest non-zero vector in a lattice.

Corollary 2.3 (Minkowski). *Let $\mathcal{L}$ be an m-dimensional lattice. Then there exists a vector $v \in \mathcal{L}$ such that*

$$\|v\| \le \sqrt{m}\, \mathrm{vol}(\mathcal{L})^{1/m}.$$

The number of lattice points in a full dimensional lattice in nice sets[2] of $\mathbb{R}^n$ is often estimated (up to a small additive error) to be the volume of the set divided by the volume of the lattice. This estimate, which dates back to Gauss, can be proven when the lattice dimension m is fixed and the nice set is the ball centred at the origin with radius growing to infinity. Using this estimate, the first minimum of a lattice is often approximated by

$$\lambda_1 \approx \sqrt{\tfrac{m}{2\pi e}}\, \mathrm{vol}(\mathcal{L})^{1/m}, \tag{2.8}$$

which is close to the limit given in Corollary 2.3. Of course, for some specific lattices, this approximation for λ_1 can be a significant over-estimate. For example, the lattice $\mathcal{L} = \mathbb{Z}^m$ has $\lambda_1 = 1$, which is much smaller than the approximation (2.8) for large enough dimension m.

[2]See Nguyễn and Stern [94] for an example of a nice set.

2.3.2 Reduced Bases

For a given lattice $\mathcal{L}$ with dimension $m \geq 2$ some bases are "better" than others. Here, "better" depends on the actual application. Usually, we are interested in so-called reduced bases of a lattice. There are several notions of a *reduced basis*, but in essence, a reduced basis is simply a basis made up of short vectors. *Lattice basis reduction*, or simply basis reduction, is a process by which a reduced basis is found from a given basis.

The first basis reduction algorithm, due to Gauss, is for 2-dimensional lattices. Let $\mathcal{L}$ be a lattice with dimension $m = 2$. Gauss's basis reduction algorithm transforms any basis of $\mathcal{L}$ into a basis (b_1, b_2) such that b_1 is a shortest vector in the lattice and the component of b_2 parallel to b_1 has length at most $1/2$. The new basis (b_1, b_2) is said to be *Gaussian-reduced.* The Gaussian algorithm, which has runtime quadratic in the input size, is given in Algorithm 2.1. This algorithm has been generalized by Nguyễn and Stehlé [92] to lattices of any dimension. However, the generalized algorithm, called the greedy algorithm, is only optimal for lattices of dimension $m \leq 4$. By optimal, we mean that each reduced basis vector satisfies $\|b_i\| = \lambda_i(\mathcal{L})$ for $i = 1, \ldots, m$. The complexity of the greedy algorithm is also quadratic in the size of the input.

Algorithm 2.1 Gaussian Reduction

```
Input: $(b_1, b_2)$
1: repeat
2:     if $\|b_1\| > \|b_2\|$ then
3:         swap $b_1$ and $b_2$
4:     end if
5:     $\mu \leftarrow \frac{\langle b_1, b_2 \rangle}{\|b_1\|^2}$
6:     $b_2 \leftarrow b_2 - \lceil \mu \rfloor b_1$     $(\lceil \alpha \rfloor = \lfloor \alpha + 0.5 \rfloor)$
7: until $\|b_1\| < \|b_2\|$
Output: $(b_1, b_2)$ which is Gaussian reduced
```

Before describing the next important class of reduced bases, we recall the *Gram-Schmidt Orthogonalization* process. Given m linearly independent vectors $b_1, \ldots, b_m \in \mathbb{R}^n$, define the vectors $b_1^*, \ldots, b_m^* \in \mathbb{R}^n$ by the recurrence

$$\begin{aligned} b_1^* &= b_1, \\ b_i^* &= b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^* \quad \text{for} \quad 2 \leq i \leq m, \end{aligned} \tag{2.9}$$

where $\mu_{i,j} = \langle b_i, b_j^* \rangle / \|b_j^*\|^2$ are called the *Gram-Schmidt coefficients*. We will call $b_1^*, \dots, b_m^*$ the Gram-Schmidt orthogonalization of $b_1, \dots, b_m$. The Gram-Schmidt orthogonalization process creates an orthogonal basis for the span of the $b_1, \dots, b_m$. Unfortunately, since the $\mu_{i,j}$ are usually not integers, the lattice $\mathcal{L}(b_1^*, \dots, b_m^*)$ is not, in general, the same lattice as $\mathcal{L}(b_1, \dots, b_m)$. However, letting $\mathcal{L} = \mathcal{L}(b_1, \dots, b_m)$, the Gram-Schmidt orthogonalization of $b_1, \dots, b_m$ satisfies the following:

$$\begin{aligned} \mathrm{vol}(\mathcal{L}) &= \prod_{i=1}^{m} \|b_1^*\| \\ \lambda_1(\mathcal{L}) &\geq \min_{1 \leq i \leq m} \{ \|b_1^*\|, \dots, \|b_m^*\| \}. \end{aligned} \tag{2.10}$$

We can now proceed to the next class of reduced bases. A basis $b_1, \dots, b_m$ of a lattice $\mathcal{L}$ is said to be *Lovász-reduced* or *LLL-reduced* if

$$|\mu_{i,j}| \leq \frac{1}{2} \quad \text{for } 1 \leq j < i \leq n, \tag{2.11}$$

and

$$\|b_i^* + \mu_{i,i-1} b_{i-1}^*\|^2 \geq \frac{3}{4} \|b_{i-1}^*\|^2 \quad \text{for } 1 < i \leq n, \tag{2.12}$$

or equivalently

$$\|b_i^*\|^2 \geq \left(\frac{3}{4} - \mu_{i,i-1}^2 \right) \|b_{i-1}^*\|^2 \quad \text{for } 1 < i \leq n, \tag{2.13}$$

where the b_i^* and $\mu_{i,j}$ are defined by the Gram-Schmidt orthogonalization process acting on the b_i. Notice that the vectors $b_i^* + \mu_{i,i-1} b_{i-1}^*$ and b_{i-1}^* are the projections of b_i and b_{i-1}, respectively, on the orthogonal complement of the span of $\{b_1, \dots, b_{i-2}\}$. Some useful properties of LLL-reduced bases are given in the following theorem (see Cohen [27]).

Theorem 2.4. *Let $b_1, \dots, b_m$ be an LLL-reduced basis of a rational lattice $\mathcal{L} \in \mathbb{Q}^n$ and let $b_1^*, \dots, b_m^*$ be its Gram-Schmidt orthogonalization. Then*

1. *The volume of $\mathcal{L}$ satisfies*

$$\mathrm{vol}(\mathcal{L}) \leq \prod_{i=1}^{m} \|b_i\| \leq 2^{m(m-1)/4} \mathrm{vol}(\mathcal{L}). \tag{2.14}$$

2. *The reduced basis vectors satisfy*

$$\|b_j\| \leq 2^{(i-1)/2} \|b_i^*\| \quad \textit{for } 1 \leq j \leq i \leq m. \tag{2.15}$$

3. *The smallest basis vector b_1 satisfies*

$$\|b_1\| \leq 2^{(m-1)/4}\mathrm{vol}(\mathcal{L})^{1/m}. \tag{2.16}$$

4. *For every $x \in \mathcal{L}$ with $x \neq 0$ we have*

$$\|b_1\| \leq 2^{(m-1)/2}\|x\|. \tag{2.17}$$

5. *For any $t \leq m$ linearly independent vectors $x_1, \ldots, x_t \in \mathcal{L}$ we have*

$$\|b_j\| \leq 2^{(m-1)/2} \max(\|x_1\|, \ldots, \|x_t\|) \quad \text{for } 1 \leq j \leq t. \tag{2.18}$$

The results of Theorem 2.4 lead directly to the following bounds on each of the LLL-reduced basis vectors.

Corollary 2.5. *Let $b_1, \ldots, b_m$ be an LLL-reduced basis of an integral lattice $\mathcal{L} \in \mathbb{Z}^n$. Then*

1. *For $1 \leq \ell \leq m$ (Blömer and May [8]),*

$$\|b_\ell\| \leq 2^{\frac{m(m-1)}{4(m-\ell+1)}}\mathrm{vol}(\mathcal{L})^{\frac{1}{m-\ell+1}}. \tag{2.19}$$

2. *For $\ell = 1$ or $1 < \ell \leq m$ and $\|b_1\| \geq 2^{(\ell-2)/2}$ (Proos [97]),*

$$\|b_\ell\| \leq 2^{\frac{m+\ell-2}{4}}\mathrm{vol}(\mathcal{L})^{\frac{1}{m-\ell+1}}. \tag{2.20}$$

LLL-reduced bases are an important class of reduced bases because there exists a polynomial time algorithm to compute them. The first such algorithm, due to Lovász, is called the *Lovász reduction algorithm*, or more commonly, the *LLL-algorithm* [74]. For an m-dimensional lattice $\mathcal{L} \in \mathbb{Q}^n$, the LLL-algorithm has runtime $O(nm^5B^3)$ where B is a bound on the bitlength of the components of the basis vectors (*i.e.*, $B = \max_i(\log\|b_i\|_\infty)$).

Some other notions of reduced bases include Minkowski-reduced and (Korkine-Zolotareff) KZ-reduced. They are defined as follows. Let $\mathcal{B} = (b_1, \ldots, b_m)$ be a basis for the lattice $\mathcal{L}$. The basis $\mathcal{B}$ is said to be *Minkowski-reduced* if b_1 is a shortest vector of $\mathcal{L}$ and, for each $i = 2, \ldots, m$, the vector b_i is a shortest vector independent from $b_1, \ldots, b_{i-1}$ such that $b_1, \ldots, b_i$ can be extended to a basis of $\mathcal{L}$. The basis $\mathcal{B}$ is said to be *KZ-reduced* if b_1 is a shortest vector of $\mathcal{L}$ and, for each $i = 2, \ldots, m$, the vector b_i is a shortest vector of the lattice $\mathcal{L}_i$ which is the projection of $\mathcal{L}$ onto the subspace of $\mathbb{R}^n$ perpendicular to $b_1, \ldots, b_{i-1}$. For 2-dimensional lattices, KZ-reduction and

Gaussian-reduction are equivalent. Also, for each $i = 1, \ldots, m$, the vectors b_i of a KZ-reduced basis satisfy

$$\sqrt{\frac{4}{i+3}}\,\lambda_i(\mathcal{L}) \leq \|b_i\| \leq \sqrt{\frac{i+3}{4}}\,\lambda_i(\mathcal{L}).$$

Thus, the size of each b_i is at most a factor of $\sqrt{n}$ away from $\lambda_i(\mathcal{L})$.

In [105], Schnorr introduced a hierarchy of polynomial-time lattice basis reduction algorithms. These algorithms, called *blockwise Korkine-Zolotareff reductions* or *BKZ-reductions*, can compute a reduced basis ranging from LLL-reduced to KZ-reduced depending on a parameter called the blocksize. These algorithms are super-exponential in the blocksize (requiring an exhaustive search on sets defined by the blocksize).

2.3.3 Algorithmic Problems

There are three main algorithmic problems dealing with lattice basis reduction: the shortest vector problem, the closest vector problem, and the smallest basis problem. We are only concerned with the shortest vector problem in this thesis. In the rest of this section, we will assume that all lattices are rational lattices ($\mathcal{L} \subseteq \mathbb{Q}^n$) with dimension $m \leq n$, unless otherwise stated.

Given a basis for a lattice $\mathcal{L}$, the *shortest vector problem (SVP)* is to find $v \in \mathcal{L}$ such that $\|v\| = \lambda_1(\mathcal{L})$. The approximate shortest vector problem is to find a vector $v \in \mathcal{L}$ such that $\|v\| = f(m)\,\lambda_1(\mathcal{L})$ for some approximation factor $f(m)$.

It has been shown by Ajtai [2] that SVP is NP-hard under randomized reductions. Micciancio [86] has further shown that approximating SVP to within a factor less than $\sqrt{2}$ is also NP-hard under randomized reductions. The NP-hardness of SVP under deterministic reductions remains an open problem. The best algorithm known for exact SVP, by Ajtai, Kumar and Sivakumar [3], requires randomized $2^{O(m)}$-time.

There is no known algorithm to approximate SVP to within a polynomial factor of the dimension of the lattice. There are, however, some polynomial time algorithms that can approximate it to within a slightly exponential factor. From (2.17), we see that the LLL-algorithm approximates SVP to within a factor of $2^{(m-1)/2}$. This was improved to $2^{O(m(\log\log m)^2/\log m)}$ by Schnorr [105] and in randomized polynomial time it was further lowered to $2^{O(m\log\log m/\log m)}$ by Ajtai, Kumar and Sivakumar [3].

In cryptography, the most important algorithmic problem is the shortest vector problem (or the approximate version) and its natural generalization of finding more than one short vector in a lattice. In addition, most lattices considered are integer lattices.

For many years, the algorithm of choice for these problems was the LLL-algorithm. For a lattice $\mathcal{L} \subseteq \mathbb{Z}^n$ with dimension m, the LLL-algorithm has complexity $O(nm^5B^3)$, where B is an upper bound on the bitlength of the input basis vectors. The complexity is cubic in B due to exact (long) integer arithmetic. Floating-point variants have often been implemented to reduce the complexity by a factor of B, but these variants, unfortunately, have been unstable in the worst case (*i.e.*, the algorithms are not guaranteed to terminate and sometimes the output may not be an LLL-reduced basis). In 2005, Nguyễn and Stehlé [93] presented a new algorithm, called the L^2-algorithm, which is a natural floating-point variant of the LLL-algorithm. The algorithm has complexity $O(nm^4(m + B)B)$ and it outputs an LLL-reduced basis.

2.4 Linear Lattice Problems

The first lattice-based method we consider uses the following heuristic.

Heuristic 2.6 (Small Vectors in a Lattice). *Let $\mathcal{L}$ be an integer lattice with dimension m. If $v \in \mathcal{L}$ satisfies Minkowski's bound (Corollary 2.3) for a smallest vector in a lattice*

$$\|v\| \leq \sqrt{m}\,\mathrm{vol}(\mathcal{L})^{1/m},$$

then it is likely that v is a smallest vector in $\mathcal{L}$.

For a given class of lattices, experimental evidence suggests that this heuristic either holds extremely well or does not hold at all. There does not seem to be a middle ground except when the vector just satisfies Minkowski's bound (*i.e.*, $v \approx \sqrt{m}\,\mathrm{vol}(\mathcal{L})^{1/m}$). For example, if the heuristic holds for a random instance of a certain problem, then it will very likely hold for almost all instances of that problem. And, when the heuristic fails for a random instance of a certain problem, then it will very likely fail for all instances of that problem.

Using this heuristic, we try to find small solutions of linear multivariate equations. Basically, the method consists of constructing a lattice basis with the known coefficients of the linear equation such that the lattice is known to contain a small target vector. This target vector will contain some information that we wish to recover (*e.g.*, part of the solution). If the target vector satisfies Minkowski's bound for the lattice, we apply Heuristic 2.6

and try to recover this vector using lattice basis reduction. We illustrate this method with a simple example equation in three variables. Suppose that for some $a, b \in \mathbb{Z}$ we know that

$$ax + by = z,$$

where x, y and z are unknown and x and z are small. We would like to compute the unknown values x, y and z. To simplify this illustration, suppose that we also know a real $\alpha > 1$ such that $cz < \alpha x < z$, for some c that is not too small. Consider the system of equations given by $ax + by = z$ and the trivial equation $\alpha x = \alpha x$. We can write this system as the vector matrix equation

$$(x, y) \begin{bmatrix} \alpha & a \\ & b \end{bmatrix} = (\alpha x, z),$$

where we omit all zero entries in the matrix. Notice that the target vector $v = (\alpha x, z)$ satisfies $\|v\| \leq \sqrt{2}\,|z|$. Also, since $x, y \in \mathbb{Z}$, we know that v is a vector in the lattice $\mathcal{L}$ generated by the basis vectors (α, a), $(0, b)$. Since the basis matrix $\mathcal{M} = \left[\begin{smallmatrix} \alpha & a \\ & b \end{smallmatrix}\right]$ is triangular, we can easily compute the volume of the lattice, given by $\mathrm{vol}(\mathcal{L}) = |\alpha b|$. Now, from Corollary 2.3 (Minkowski), we know that a shortest vector v' in a 2-dimensional lattice must satisfy

$$\|v'\| \leq \sqrt{2}\,\mathrm{vol}(\mathcal{L})^{1/2}.$$

If the size of the target vector v satisfies this bound and Heuristic 2.6 holds, then v will likely be a smallest vector in $\mathcal{L}$. Using an algorithm such as LLL or L^2, we compute an LLL-reduced basis for $\mathcal{L}$. If v is a smallest vector in $\mathcal{L}$ and if $\pm v$ are the only smallest vectors, then the first reduced basis vector will be $\pm(\alpha x, z)$, which immediately reveals z. And, since we know that $(x, y) \left[\begin{smallmatrix} \alpha & a \\ & b \end{smallmatrix}\right] = (\alpha x, z)$, we can easily solve for x and y. Of course, this can only happen if the target vector v satisfies Minkowski's bound for a smallest vector in a lattice. Therefore, this can happen only when $\|v\| \leq \mathrm{vol}(\mathcal{L})^{1/2}$, or more simply when

$$|z| \leq \sqrt{\alpha b},$$

which becomes the enabling equation for the heuristic method. Thus, if this enabling equation holds, we expect that the target vector v will be a short vector in the lattice. If we are lucky and Heuristic 2.6 holds, v will be a shortest vector in the lattice and we can recover it by reducing the lattice basis.

In general, the method consists of constructing a lattice, with basis vectors consisting of the coefficients of the linear equation, such that the lattice contains a target vector satisfying Minkowski's bound for a smallest vector. We then compute a reduced basis for the lattice and hope that the known vector is recovered as one of the reduced vectors. When the target vector is found, we possibly solve for the vector that constructs the target vector (*i.e.*, the vector that multiplies the basis matrix) and then we extract the desired information. Bounds for the method are obtained by forcing the size of the target vector to satisfy Minkowski's bound for a smallest vector (Corollary 2.3).

In practice, the target vector v is often chosen so that all of its components are roughly the same size. This is done to improve the likelihood that Heuristic 2.6 holds. For example, it often happens that the heuristic will fail for a given problem with a target vector with some small components but will hold when the problem is slightly modified so that the target vector's components are equally sized. Thus, in the general method outlined above, the problem is often first stated and then modified so that the target vector has components that are roughly the same size.

2.5 Coppersmith's Methods

The second type of lattice-based methods that we utilize are Coppersmith's methods and their heuristic extensions. These methods allow us to efficiently compute sufficiently small roots of polynomials (*i.e*, solutions to polynomial equations), where the solutions are over the integers $\mathbb{Z}$ or over a ring $\mathbb{Z}_N$.

To motivate the techniques in this chapter let's consider the univariate modular case. Let N be an integer with unknown factorization and let

$$f_N(x) = x^d + a_{d-1}\,x^{d-1} + \cdots + a_2\,x^2 + a_1\,x + a_0 \in \mathbb{Z}[x].$$

That is, $f_N(x)$ is a monic polynomial[3] of degree d. The goal is to efficiently find all $|x_0| < X$ satisfying

$$f_N(x_0) \equiv 0 \pmod{N},$$

for as large a bound X as possible.

[3] If the polynomial is not monic, we can simply multiply $f_N(x)$ by the inverse of the leading coefficient modulo N. If the inverse does not exist, this yields the (partial) factorization of N which in most applications in cryptography yields more information than the desired small solution x_0. Thus, we assume without loss of generality that $f_N(x)$ is monic.

In some instances, we can efficiently compute small solutions of the modular equation by simply solving the integer equation $f_N(x) = 0$. Let X be a bound on the size of the solutions that can be found this way. For example, when $f_N(x) = x^d - a_0$ we can efficiently find all solutions up the bound $X = N^{1/d}$. This follows since $|x_0| < X = N^{1/d}$ can be found by simply computing the d^{th} roots of a_0 over the integers. More generally, if each coefficient of $f_N(x)$ satisfies

$$|a_i| < \frac{N^{(1-i/d)}}{(d+1)},$$

then all solutions $|x_0| < X = N^{1/d}$ can be found by solving $f_N(x) = 0$ over the integers, since N divides $p(x_0)$ and

$$\begin{aligned} |f_N(x_0)| &\leq \sum_{i=0}^{d} |a_i|\,|x_0^i| \\ &< \sum_{i=0}^{d} \frac{N^{(1-i/d)}}{d+1} N^{i/d} \\ &= N. \end{aligned}$$

A more useful sufficient condition for solutions of $f_N(x) \equiv 0 \pmod N$ to be solutions of $f_N(x) = 0$ is the following result attributed to Howgrave-Graham [66].

Theorem 2.7. *Let $h(x) \in \mathbb{Z}[x]$ be the sum of at most ω monomials. For some positive bound X, suppose that x_0 satisfies*

1. $h(x_0) \equiv 0 \pmod N$,
2. $\|h(xX)\| < \frac{1}{\sqrt{\omega}} N$, *and*
3. $|x_0| < X$.

Then x_0 is a root of $h(x)$ over $\mathbb{Z}$. That is, $h(x_0) = 0$.

Of course the coefficients of $f_N(x)$ will not, in general, be small enough to satisfy the conditions in Theorem 2.7 (or the conditions in the previous discussion).

In order to make use of this theorem, we use lattice basis reduction to construct a polynomial with sufficiently small coefficients. Recalling that x_0 satisfies $f_N(x_0) \equiv 0 \pmod N$, we construct a lattice with basis vectors that correspond to the coefficient vectors of polynomials that all have a solution x_0 modulo N. Using the LLL- or L^2-algorithm, we compute an LLL-reduced basis for the lattice. The smallest reduced basis vector then corresponds to a polynomial $f(x)$ with small coefficients (also having the solution x_0 modulo N). If this polynomial satisfies Theorem 2.7, we can then compute x_0 by solving $f(x) = 0$ over the integers, which can be done efficiently (*e.g.*, see the methods discussed in [121, Chapter 15]).

Essentially, this is the framework for all the lattice-based methods for finding small solutions of polynomials.

2.5.1 Small Solutions of Modular Polynomials

We again begin by discussing univariate modular polynomials. Let N be a positive integer with unknown factorization and let $f_N(x) \in \mathbb{Z}[x]$ be a monic polynomial with degree d. The goal is to efficiently find all $|x_0| < X$ satisfying

$$f_N(x_0) \equiv 0 \pmod{N},$$

for as large a bound X as possible.

Early work by Håstad [56, 57] and Vallée, Girault and Toffin [118, 51] (in the mid to late 1980's), showed that lattice-based methods can efficiently find solutions $|x_0| < X$ for a bound as large as

$$X = N^{\frac{2}{d(d+1)} - \epsilon},$$

where $\epsilon > 0$ is a function of the degree d. Essentially, the method uses lattice basis reduction to find a polynomial $h(x)$, with small coefficients, that is a constant multiple of $f_N(x)$ modulo N. The basis vectors used to construct the lattice simply consisted of the coefficient vectors of the $d+1$ polynomials

$$f_i(xX) = \begin{cases} N(xX)^i & \text{for } 0 \le i \le d-1, \\ f_N(xX) & \text{for } i = d. \end{cases}$$

Notice that in order to use Theorem 2.7, the polynomials are each evaluated at xX. For a degree d polynomial

$$f_N(x) = a_0 + a_1 x + \ldots a_{d-1} x^{d-1} + x^d,$$

we use the basis matrix

$$\mathcal{M} = \begin{bmatrix} N & & & & & \\ & NX & & & & \\ & & NX^2 & & & \\ & & & \ddots & & \\ & & & & NX^{d-1} & \\ a_0 & a_1X & a_2X^2 & \cdots & a_{d-1}X^{d-1} & X^d \end{bmatrix}.$$

Notice that any element in the lattice $\mathcal{L}$ (generated by $\mathcal{M}$) can be written as

$$\Big((ca_0 - c_0N), (ca_1 - c_1N)X, \ldots, (ca_{d-1} - c_{d-1}N)X^{d-1}, cX^d\Big),$$

which corresponds to the coefficient vector of some polynomial $h(x)$ given by

$$h(x) = (ca_0 - c_0N) + (ca_1 - c_1N)x + \cdots + (ca_{d-1} - c_{d-1}N)x^{d-1} + cx^d,$$

when evaluated at xX. In particular, notice that

$$h(x) \equiv cf_N(x) \pmod{N}.$$

Thus, all roots of $f_N(x)$ modulo N are also roots of $h(x)$ modulo N.

The main improvement over these early results came in 1996, when Coppersmith [33, 34] increased the bound from $N^{2/d(d+1)-\epsilon}$ to $N^{1/d-\epsilon}$. This improvement is the result of considering polynomial combinations of $f_N(x)$ modulo N^u for some integer u (as compared to a constant multiple of $f_N(x)$ modulo N used earlier). In the original presentation [33, 34], Coppersmith was working in an "unnatural" space. The presentation was difficult to follow and was not easily transferred to practical implementations. However, shortly after, in 1997, Howgrave-Graham [66] gave an alternate presentation that was more natural and more easily implemented. In fact, all current uses of Coppersmith's univariate modular method use Howgrave-Graham's approach. We outline the method below.

Given a polynomial $f_N(x)$ with root x_0 modulo N, we fix some integer $m \geq 1$ and consider a set of polynomials

$$f_{i,j}(x) = x^i f_N(x)^j N^{m-j},$$

for various values of $i, j \geq 0$. Since x_0 satisfies $f_N(x_0) \equiv 0 \pmod{N}$, notice that x_0 also satisfies $f_{i,j}(x_0) \equiv 0 \pmod{N^m}$ for any $i, j \geq 0$ and $m \geq 1$.

With care, we select ω of these polynomials (*i.e.*, ω pairs (i,j)) and use the coefficient vectors of $f_{i,j}(xX)$ for each as a basis vector for a lattice $\mathcal{L}$. Notice again that we use the coefficient vectors of the polynomials evaluated at xX so that Theorem 2.7 can be used. With a careful choice of (i,j) pairs, the basis matrix is a triangular $\omega \times \omega$ matrix. Thus, the lattice $\mathcal{L}$ is an ω-dimensional full rank integer lattice.

Since the lattice is full rank and the basis matrix is triangular, the product of the diagonal elements is equal to the volume of the lattice. Since $f_N(x)$ is monic, we find that

$$\mathrm{vol}(\mathcal{L}) = X^{C_X} N^{C_N},$$

for some $C_X, C_N \geq 0$.

Applying the LLL- or L^2-algorithm, we compute an LLL-reduced basis for the lattice. From the properties of LLL-reduced bases (Theorem 2.4), we know that the smallest vector in the reduced basis corresponds to a polynomial $h(x)$, having x_0 as a root modulo N^m, and satisfying

$$\|h(xX)\| \leq 2^{\frac{\omega-1}{4}} \mathrm{vol}(\mathcal{L})^{\frac{1}{\omega}}. \tag{2.21}$$

Also, a simple extension of Theorem 2.7, from solutions modulo N to solutions modulo N^m, shows that if $|x_0| < X$ and

$$\|h(xX)\| < \frac{N^m}{\sqrt{\omega}}, \tag{2.22}$$

then $h(x_0) = 0$. Using both (2.21) and (2.22), we see that a sufficient condition for x_0 to be a solution of $h(x) = 0$ is given by

$$2^{\frac{\omega-1}{4}} \mathrm{vol}(\mathcal{L})^{\frac{1}{\omega}} < \frac{N^m}{\sqrt{\omega}}.$$

Substituting $\mathrm{vol}(\mathcal{L}) = X^{C_X} N^{C_N}$ into this inequality, we find that an equivalent sufficient condition is given by

$$\begin{aligned} X^{C_X} &< \frac{N^{m-C_N}}{2^{\frac{\omega-1}{4}}\sqrt{\omega}} \\ &= N^{m-C_N-\epsilon}. \end{aligned}$$

The inequality $X^{C_N} < N^{m-C_N-\epsilon}$ becomes what is often referred to as the *enabling equation* for the attack. All solutions $|x_0| < X$, where X satisfies the enabling equation, are solutions of $h(x) = 0$ and they can be found with standard root finding techniques. Often, the bound X is expressed as $X = N^\gamma$ so that the enabling equation can be described in terms of the exponents of N. In this case, the enabling equation would be

$$\gamma C_N < m - C_N - \epsilon.$$

The main result of Coppersmith's technique for univariate polynomials is given in the following theorem. The result is a slight generalization of Coppersmith's, which was first used by Boneh, Durfee and Howgrave-Graham in their lattice-based factoring method [17]. The general result, as given below, is from May [82] (for a proof, see Hinek [58]).

Theorem 2.8 (Coppersmith). *For every $\epsilon > 0$ there exists an N_0 such that the following holds: Let $N > N_0$ be an integer with unknown factorization which has a divisor $b \geq N^\beta$. Let $g_b(x)$ be a monic univariate polynomial of degree d and let c_N be a function that is upper-bounded by a polynomial in $\log(N)$. All solutions x_0 of the relation $g_b(x_0) \equiv 0 \pmod{b}$, such that*

$$|x_0| \leq c_N N^{\beta^2/d - \epsilon},$$

can be found in time polynomial in $\log_2(N)$.

Coppersmith's method for finding small solutions of univariate modular polynomials is easily extended to multivariate modular polynomials. The first extension was used in Boneh and Durfee's lattice-based attack on small private exponent RSA [13], in which they consider a bivariate modular equation. Unfortunately, all extensions to the multivariate case lead to heuristic attacks instead of provable results. We briefly outline the general strategy below.

Given a multivariate polynomial $f_N(x_1, \ldots, x_n)$ with solution $(y_1, \ldots, y_n)$ modulo N, we fix some integer $m \geq 1$ and consider the set of polynomials

$$f_{i_1,\ldots,i_n,j}(x_1, \ldots, x_n) = {x_1}^{i_1} {x_2}^{i_2} \cdots {x_n}^{i_n} f_N(x_1, \ldots, x_n)^j N^{m-j},$$

for carefully chosen $i_1, \ldots, i_n, j \geq 0$. In this case, the difficulty in choosing the best $i_1, \ldots, i_n, j$ increases with the number of variables and the complexity of the polynomial f_N. Just as in the univariate case, we construct a full rank lattice using the coefficient vectors of the $f_{i_1,\ldots,i_n,j}(x_1, \ldots, x_n)$, evaluated at $(x_1 X_1, \ldots, x_n X_n)$, and compute an LLL-reduced basis for the lattice. The reduced basis allows us to construct n linearly independent polynomials $h_1, \ldots, h_n$, each having the root $(y_1, \ldots, y_n)$ modulo N^m. If the size of all of these polynomials also satisfies the conditions in the multivariate generalization of Theorem 2.7, given in the theorem below, then $(y_1, \ldots, y_n)$ is a root of each of the polynomials over the integers.

Theorem 2.9 (Howgrave-Graham). *Let $h(x_1,\dots,x_n) \in \mathbb{Z}[x_1,\dots,x_n]$ be the sum of at most ω monomials and let u be a positive integer. For any $(y_1,\dots,y_n) \in \mathbb{Z}^n$ satisfying $|y_i| < X_i$ for $1 \le i \le n$, if*

1. $h(y_1,\dots,y_n) \equiv 0 \pmod{N^u}$
2. $\|h(x_1X_1,\dots,x_nX_n)\| < \frac{1}{\sqrt{\omega}}N^u$,

then $(y_1,\dots,y_n)$ is a root of $h(x_1,\dots,x_n)$ over $\mathbb{Z}$. That is, $h(y_1,\dots,y_n) = 0$.

Up to this point, everything we have presented is a natural generalization of the univariate case. The main difference now is that, given the polynomials $h_1,\dots,h_n$, we might not be able to solve for the desired common root. This is because the polynomials obtained from the LLL-reduced basis, while guaranteed to be linearly independent, are not guaranteed to be algebraically independent[4].

If the polynomials are algebraically independent, then we can solve for $(y_1,\dots,y_n)$ using repeated resultant computations to remove variables until we reach a univariate polynomial which we can solve and then start backtracking until the entire root $(y_1,\dots,y_n)$ is recovered (see [58, §3.1.3] for more details). When the polynomials have four or more variables, the resultant computations become too expensive and Gröbner basis techniques can be used (see Jochemsz and May [70]).

If the polynomials are not algebraically independent, sometimes information can still be obtained from them. For example, Blömer and May [7] show a bivariate case in which the polynomials are always algebraically dependent, but information contained in the smallest polynomial can be used to compute as much information as the desired solution. In other instances, even if the first n polynomials are algebraically dependent there might be n algebraically independent polynomials found within the first $n' > n$ reduced basis vectors.

Known Results for Modular Polynomials

We finish this section on modular polynomials by collecting all known results for extensions of Coppersmith's univariate method to multivariate methods. First we give some definitions to describe certain classes of polynomials.

Let $f(x_1,\dots,x_n)$ be a polynomial where the degree of x_i is $\lambda_i D$, for some fixed D, for $1 \le i \le n$. We say that $f(x_1,\dots,x_n)$ is a *generalized rectangle* with parameters D and λ_i, for $1 \le i \le n$.

[4]Two polynomials h_1 and h_2 are algebraically independent if and only if their only common factors are constants (*i.e.*, $\gcd(h_1,h_2) = \text{constant}$).

Let $f(x_1,\dots,x_n)$ be a polynomial with monomials $x_1^{k_1}x_2^{k_2}\cdots x_n^{k_n}$ where

$$\begin{aligned}
&0 \le k_1 \le \lambda_1 D\\
&0 \le k_2 \le \lambda_2 D - \frac{\lambda_2}{\lambda_1}k_1\\
&\qquad\vdots\\
&0 \le k_n \le \lambda_n D - \sum_{i=1}^{n-1}\frac{\lambda_n}{\lambda_i}k_i,
\end{aligned}$$

for some fixed D, for $1 \le i \le n$. We say that $f(x_1,\dots,x_n)$ is a *generalized lower triangle* with parameters D, λ_i and k_i, for $1 \le i \le n$.

With these definitions, we now list all the known extensions of Coppersmith's univariate modular methods in the following theorem.

Theorem 2.10. *For every $\epsilon > 0$ there exists an N_0 such that the following holds: Let $N > N_0$ be an integer with unknown factorization.*

Let $f_N(x_1,\dots,x_n)$ be one of the polynomials listed below and let $X_1,\dots,X_n$ be positive integers.

Let $(y_1,\dots,y_n)$ satisfy $f_N(y_1,\dots,y_n) \equiv 0 \pmod N$ and suppose $|y_i| < X_i$ for $1 \le i \le n$.

For each $(y_1,\dots,y_n)$, if the enabling equation accompanying the given polynomial is satisfied (for any $\tau > 0$, if applicable), then we can construct a set of n linearly independent polynomials $g_1,\dots,g_n \in \mathbb{Z}[x_1,\dots,x_n]$, in time polynomial in $\log_2(N)$, such that $g_i(y_1,\dots,y_n) = 0$ for each $1 \le i \le n$.

Further, if these polynomials are also algebraically independent then we can compute each solution $(y_1,\dots,y_n)$ in time polynomial in $\log_2(N)$.

1. *[Boneh and Durfee [13]]: $f_N(x_1,x_2) = a_0 + a_1x_2 + a_2x_2x_2$, with enabling equation*

$$X_1^{2+3\tau}X_2^{1+3\tau+3\tau^2} < N^{1+3\tau-\epsilon}. \tag{2.23}$$

2. *[Blömer and May [8]]: $f_N(x_1,x_2,x_3) = a_0 + a_1x_1 + a_2x_2 + a_3x_2x_3$, with enabling equation*

$$X_1^{1+4\tau}X_2^{2+4\tau}X_3^{1+4\tau+6\tau^2} < N^{1+4\tau-\epsilon}. \tag{2.24}$$

3. *[Jochemsz and May [69]]: $f_N(x_1,\dots,x_n)$ is a generalized rectangle with parameters D and λ_i, for $1 \le i \le n$, with enabling equation*

$$\prod_{i=1}^{n} X_i^{\lambda_i} < N^{\frac{2}{(n+1)D}-\epsilon}. \tag{2.25}$$

4. *[Jochemsz and May [69]]: $f_N(x_1, \ldots, x_n)$ is a generalized lower triangle with parameters D, λ_i and k_i, for $1 \le i \le n$, with enabling equation*

$$\prod_{i=1}^{n} X_i^{\lambda_i} < N^{\frac{1}{D}-\epsilon}. \tag{2.26}$$

2.5.2 Small Solutions of Integer Polynomials

The natural starting point for finding small solutions of polynomials is with bivariate polynomials. The first result in this area was also by Coppersmith [32, 34], in 1996. As with the univariate modular case, Coppersmith's presentation took place in an "unnatural" space[5]. In 2004, Coron [35] presented a simplification of the method, much like Howgrave-Graham simplified the univariate modular case, that is slightly weaker than Coppersmith's original description but much more natural. The bounds on the solution are the same, but the runtime is exponential (rather than polynomial) in the degree of the polynomial. The main result, as given by Coron, is the following theorem.

Theorem 2.11 (Coppersmith). *Let $f(x, y) = \sum_{i,j} a_{i,j} x^i y^j$ be an irreducible polynomial in two variables over $\mathbb{Z}$. Let X and Y be upper bounds on the desired integer solution (x_0, y_0), and let $W = \|f(xX, yX)\|_\infty$.*

1. *If $f(x, y)$ has maximum degree d in each variable separately and*

$$XY < W^{2/(3d)-\epsilon},$$

for some $\epsilon > 0$, then in time polynomial in $(\log W, 2^d)$, one can find all integer pairs (x_0, y_0) such that $|x_0| < X$, $|y_0| < Y$, and $f(x_0, y_0) = 0$.

2. *If $f(x, y)$ has total degree d and*

$$XY < W^{1/d-\epsilon},$$

for some $\epsilon > 0$, then in time polynomial in $(\log W, 2^d)$, one can find all integer pairs (x_0, y_0) such that $|x_0| < X$, $|y_0| < Y$, and $f(x_0, y_0) = 0$.

The method is very similar to the methods for finding small solutions of modular polynomials. We briefly outline Coron's simplification of Coppersmith's method below.

[5]This unnatural space, however, turns out to be more efficient when implementing the method. See Blömer and May [10] or Jochemsz and May [69] for more information.

Let $f(x,y) \in \mathbb{Z}[x,y]$ be an irreducible polynomial with root (x_0, y_0) over the integers, satisfying $|x_0| < X$ and $|y_0| < Y$ for some bounds X and Y. The problem is first converted to a bivariate modular equation. Without loss of generality, we will assume that $f(0,0) \neq 0$ and $\gcd(f(0,0), XY) = 1$. For some positive integer k, we define the modulus $N = u(XY)^k$, where

$$u = W + ((1 - W) \bmod |f(0,0)|),$$

and we compute the polynomial

$$\begin{aligned} g(x,y) &= a_{0,0}^{-1} f(x,y) \bmod n \\ &= 1 + \sum_{(i,j) \neq (0,0)} b_{i,j} x^i y^j. \end{aligned}$$

We now look for small solutions of $g(x,y)$ modulo N using the methods of the previous section. In this case, we only need one of the reduced basis vectors. Let $h(x,y)$ be the polynomial corresponding to the smallest reduced basis vector. Using an extension of a result by Mignotte [87, Theorem 2], Coron shows that $h(x,y)$ and the original polynomial $f(x,y)$ are algebraically independent. Thus, if $h(x,y)$ satisfies Howgrave-Graham's condition (*i.e.*, if $\|h(xX, yY)\|$ is small enough so that $h(x_0, y_0) = 0$), we can solve the system of two equations in two unknowns

$$\begin{aligned} f(x,y) &= 0 \\ h(x,y) &= 0, \end{aligned}$$

for the root (x_0, y_0). Unlike the pure modular bivariate case, this result is provable.

Coron's simplification of Coppersmith's bivariate polynomial method is easily extended to multivariate polynomials in general (for example, see [46], [69] and [70]). Like the extensions of the univariate modular case, however, these extensions are only a heuristic.

Given a multivariate polynomial $f(x_1, \ldots, x_n) \in \mathbb{Z}[x_1, \ldots, n_n]$ with solution $(y_1, \ldots, y_n)$ over the integers, we convert the problem to a modular multivariate polynomial problem which is solved using the techniques of the previous section. Using the original polynomial f and the smallest $n-1$ polynomials obtained from the reduced basis $g_1, \ldots, g_{n-1}$ we try to solve the system of n equations in n unknowns. If all n polynomials are algebraically independent, we can use resultant computations or Gröbner basis methods to compute the solution $(y_1, \ldots, y_n)$. Just as Coron showed that the original polynomial f is algebraically independent of the polynomial corresponding to the smallest reduced basis vector, we can show that f is algebraically independent of each of $g_1, \ldots, g_{n-1}$. This was shown by Hinek and Stinson [65]. Unfortunately, nothing is known about the algebraic independence of the polynomials corresponding to the reduced basis vectors $g_1, \ldots, g_{n-1}$.

Known Results for Integer Polynomials

We finish this section on integer polynomials by collecting all known results for extensions of Coppersmith's bivariate integer polynomial method to multivariate polynomials in general. As with the modular case, we first give some definitions to describe certain classes of polynomials.

Let $f(x_1, x_2)$ be a polynomial with monomials $x_1^{k_1} x_2^{k_2}$ for $k_1 = 0, \ldots, D$ and $k_2 = 0, \ldots, \lambda k_1$. We say that $f(x_1, x_2)$ is an *upper triangle* with parameters D and λ.

Let $f(x_1, x_2)$ be a polynomial with monomials $x_1^{k_1} x_2^{k_2}$ for $k_2 = 0, \ldots, D$ and $k_1 = 0, \ldots, \gamma D + \lambda(D - k_1)$. We say that $f(x_1, x_2)$ is an *extended rectangle* with parameters D, γ and λ.

With these definitions, we now list all the known results for extensions to Coppersmith's bivariate integer polynomial methods in the following theorem.

Theorem 2.12. *For every $\epsilon > 0$ there exists an N_0 such that the following holds: Let $N > N_0$ be an integer with unknown factorization.*

Let $f(x_1, \ldots, x_n)$ be one of the polynomials listed below, let $X_1, \ldots, X_n$ be positive integers and let $W = \|f(x_1 X_1, \ldots, x_n X_n)\|_\infty$.

Let $(y_1, \ldots, y_n)$ satisfy $f(y_1, \ldots, y_n) = 0$, where $|y_i| < X_i$ for $1 \le i \le n$.

For each $(y_1, \ldots, y_n)$, if the enabling equation accompanying the given polynomial is satisfied (for any $\tau > 0$, if applicable), then we can construct $n - 1$ linearly independent polynomials $g_1, \ldots, g_{n-1} \in \mathbb{Z}[x_1, \ldots, x_n]$, in time polynomial in $\log_2(N)$, such that $g_i(y_1, \ldots, y_n) = 0$ for each $1 \le i \le n - 1$.

Further, if these polynomials are also algebraically independent then we can compute each solution $(y_1, \ldots, y_n)$ in time polynomial in $\log_2(N)$.

1. *[Blömer and May [10]]: $f(x_1, x_2)$ is an upper triangle with parameters D and λ, with enabling equation*

$$X_1^{(\lambda+\tau)^2} X_2^{2(\lambda+\tau)} < W^{\frac{(\lambda+2\tau)}{D} - \epsilon}. \tag{2.27}$$

2. *[Blömer and May [10]]: $f(x_1, x_2)$ is an extended rectangle with parameters D, γ and λ, with enabling equation*

$$X_1^{\lambda^2+3\gamma\lambda+2\tau\lambda+4\tau\gamma+\tau^2+3\gamma^2} X_2^{\lambda+3\gamma+2\tau} < W^{\frac{(\lambda+2\gamma+2\tau)}{D} - \epsilon}. \tag{2.28}$$

3. *[Ernst, Jochemsz, May and de Weger [46]]:*
   $f(x_1, x_2, x_3) = a_0 + a_1x_1 + a_2x_2 + a_3x_2x_3$*, with enabling equation*

$$X_1^{1+3\tau} X_2^{2+3\tau} X_3^{1+3\tau+3\tau^2} < W^{1+3\tau-\epsilon}. \tag{2.29}$$

4. *[Ernst, Jochemsz, May and de Weger [46]]:*
   $f(x_1, x_2, x_3) = a_0 + a_1x_1 + a_2x_2 + a_3x_3 + a_4x_2x_3$*, with enabling equation*

$$X_1^{2+3\tau} X_2^{3+3\tau} X_3^{3+6\tau+3\tau^2} < W^{2+3\tau-\epsilon}. \tag{2.30}$$

5. *[Jochemsz and May [70]]:*
   $f(x_1, x_2, x_3) = a_0 + a_1x_1 + a_2x_1^2 + a_3x_2 + a_4x_3 + a_5x_1x_2 + a_6x_1x_3 + a_7x_2x_3$*, with enabling equation*

$$X_1^{7+9\tau+3\tau^2} X_2^{5+\frac{9}{2}\tau} X_3^{5+\frac{9}{2}\tau} < W^{3+3\tau-\epsilon}. \tag{2.31}$$

6. *[Jochemsz and May [69]]:* $f(x_1, \ldots, x_n)$ *is a generalized rectangle with parameters* D *and* λ_i *for* $0 \le i \le n$*, with enabling equation*

$$\prod_{i=1}^{n} X_i^{\lambda_i} < W^{\frac{2}{(n+1)D}-\epsilon}. \tag{2.32}$$

7. *[Jochemsz and May [69]]:* $f(x_1, \ldots, x_n)$ *is a generalized lower triangle with parameters* D*,* λ_i *and* k_i *for* $0 \le i \le n$*, with enabling equation*

$$\prod_{i=1}^{n} X_i^{\lambda_i} < N^{\frac{1}{D}-\epsilon}. \tag{2.33}$$

2.5.3 Optimizing the Lattice Construction

Choosing which polynomials to include as basis vectors to obtain the optimal enabling equation (and hence optimal bounds) is a tedious and non-trivial task. Until recently, there has been no general strategy for constructing good lattices (*i.e.*, lattices that lead to good bounds). In 2005, Blömer and May [10] presented a method of computing optimal bounds for finding small integer roots of any bivariate polynomial. In 2006, Jochemsz and May [68] presented a heuristic strategy that applies to all multivariate polynomials having either modular or integer roots. Their strategy can be used to compute a bound (which is not necessarily optimal) and it can be extended in some cases to compute better bounds.

In Theorems 2.10 and 2.12, notice that some of the enabling equations contain a parameter τ. The parameter arises because of an optimization to the lattice construction. Instead of just using one parameter (m) to generate the lattice, another parameter $t = \tau m$ is also used. The parameter t is used to include more polynomials of a certain form. When only one parameter is used, the enabling equation can generally be written as a polynomial in the parameter m. For large enough N, only the coefficient of the largest power of m is used in the final version of the enabling equation. When m and t are both used in the construction the enabling equation is generally a complicated bivariate polynomial in m and t. In this case, to simplify the optimization process, we let $t = \tau m$, so that we are reduced to the single parameter problem. The resulting enabling equation has τ as a variable and this enabling equation then needs to be optimized with respect to the choice for τ.

2.5.4 The Algebraic Independence Issue

The results of Theorems 2.8 (univariate modular polynomial) and 2.11 (bivariate integer polynomial) provide provable methods of computing small roots of polynomials. The results of 2.10 and 2.12, on the other hand, only provide a method of constructing ℓ linearly independent polynomials (in ℓ variables) each with a common root over the integers (for some $\ell > 1$). If these ℓ polynomials are also algebraically independent we can compute the common root. For most problems arising from cryptographic application, it is assumed that these polynomials are algebraically independent. In the remainder of this thesis, we will use the following well accepted assumption.

Assumption 2.13. *The polynomials found by lattice basis reduction in Theorems 2.10 and 2.12 are algebraically independent.*

While this assumption is usually true for applications in cryptography, there have been some instances where it is false. For example, Blömer and May [7] show a class of lattices that whose LLL-reduced basis vectors are always algebraically dependent. Also, Hinek [59, 61] has observed that the likelihood that the smallest two polynomials obtained are algebraically dependent increases as the actual parameters approach the experimental limits for the attack for some lattices.

For the attacks presented in this thesis, we have found that Assumption 2.13 holds in our experimental investigations.

2.6 A Note on Proofs

In the remainder of this thesis, we present many results on the security of several variants of RSA. A result is labelled a *theorem* if it can be rigorously proven. For example, results based on Theorem 2.1 (continued fractions) are generally theorems. Any results that cannot be proven are labelled as *attacks.* Thus, any result that uses a multivariate extension of Coppersmith's methods are attacks since they rely on Assumption 2.13. Similarly, results that are based on Heuristic 2.6 are also attacks since they cannot be proven.

Chapter 3

Cryptanalysis of RSA with Certain Private Exponents

In this chapter we consider instances of RSA with private exponent close to a rational multiple of $\lambda(N) = \text{lcm}(p-1, q-1)$. In particular, we consider instances in which $|d - \frac{a}{b}\lambda(N)|$ is small for some integers $a \geq 0$ and $b \geq 1$. The attacks presented in this chapter illustrate the dangers of using such private exponents.

The use of small private exponents (*i.e.*, $a = 0$) was an early suggestion to reduce the computational costs for decryption (consisting of a single modular exponentiation). Using a small private exponent reduces the total number of modular multiplications needed in the modular exponentiation and hence reduces the overall decryption costs. Choosing a private exponent to be too small, of course, is completely insecure (as Wiener has shown and which we outline below).

Unlike the use of a small private exponent, however, there are no known practical reasons for choosing a private exponent that is close to $\frac{a}{b}\lambda(N)$, for positive integers a and b.

3.1 Background

When $a = 0$ and the private exponent is positive, $d > 0$, we have the familiar small private exponent RSA, which is well studied. In 1990, Wiener [123] showed that any instance of RSA with parameters satisfying

$$kdg < \frac{pq}{\frac{3}{2}(p+q)},$$

is completely insecure, where k is the constant in the key equation

$$ed = 1 + k\lambda(N),$$

and $g = \gcd(p-1, q-1)$. Using information obtained from the continued fraction expansion of e/N, Wiener showed how to efficiently factor the modulus N for these instances. When the primes are balanced and g is small (*e.g.*, 2) this simplifies to the bound $d < cN^{0.25}$, where c is a small constant that is independent of N. In 1999, Boneh and Durfee [13] extended Coppersmith's lattice-based technique for finding small solutions of modular univariate polynomials [33] to modular bivariate polynomials, in order to increase the bound on insecure private exponents to $d < N^{0.292}$. This bound on d is found in the limiting case of large N and infinite computing power, and it also relies on a heuristic about lattice basis reduction (see Assumption 2.13). However, it works well in practice, and has been shown experimentally to work up to $N^{0.280}$ for a 1000-bit modulus (see [13, §8]). In 2001, Blömer and May [7] presented a refined lattice-based attack which simplifies the analysis, but it did not improve Boneh and Durfee's bound.

When $a = 0$ and $d < 0$ we have small negative private exponents. Negative exponents correspond to large exponents when considered modulo $\lambda(N)$ (or $\phi(N)$). In 2004, Hinek [60] showed that all of the attacks on small private exponent RSA also work for small negative private exponents, yielding the same bounds. That is, if $d < N^{\delta}$ is vulnerable to one of the small private exponent attacks, then so is any $|d| < N^{\delta}$. Thus, private exponents satisfying $|d| < N^{0.292}$ should be considered unsafe.

The problem of large private exponents ($a > 1$) was first considered in 1996 by Chen, Chang and Yang [23], who applied Wiener's continued fraction attack to private exponents close to rational multiples of $\lambda(N)$. It was shown that instances of RSA with private exponent d satisfying $|d - \frac{a}{b}\lambda(N)| < cN^{0.25}$, where c is some small number depending only on b, are insecure when the public exponent is smaller than the modulus and b is small.

In 2005, Chen, Ku and Yen [24] presented a new lattice-based attack on the special case of private exponents close to $\lambda(N)$. It was shown that instances of RSA with private exponent d satisfying $|d - \lambda(N)| < N^{0.25}$ are insecure, provided that the modulus is large enough.

3.2 Assumptions and Notation

For the remainder of this chapter, we assume that $N = pq$ is an n-bit balanced RSA modulus, $d = N^{\delta}$ is a valid private exponent close to some rational multiple of $\lambda(N) = \text{lcm}(p-1, q-1)$ and $e = N^{\alpha}$ is its corresponding public exponent defined modulo $\lambda(N)$. We also assume that the public exponent is computed after the private exponent is chosen. Thus, the public exponent will satisfy $0 \leq e < \lambda(N)$ and, with high probability, also satisfy $e \approx \lambda(N)$. Since the public and private exponents are defined modulo $\lambda(N)$, we know there exists a positive integer k satisfying

$$ed = 1 + k\lambda(N). \tag{3.1}$$

Finally, since $e < \lambda(N)$, we know that $k < d$.

3.3 Continued Fraction Attack

In this section we show how Wiener's continued fraction attack on small private exponent RSA can be applied to instances of RSA with private exponent close, but not too close, to a rational multiple of $\lambda(N)$.

In [23, Section 3.3], Chen, Chang and Yang analyzed this problem and showed that RSA is insecure when the private exponent satisfies

$$\frac{N}{e} < \left| \frac{a}{b}\lambda(N) - d \right| < \frac{N^{1/4}}{b^2}, \tag{3.2}$$

for a small positive integer b and non-negative integer $a < b$. Their attack is a generalization of Wiener's continued fraction attack on small private exponent RSA [123].

Upon first inspection, this attack seems to be very strong since for any private exponent d it is possible to choose integers $a < b$ to make $|\frac{a}{b}\lambda(N) - d|$ arbitrarily small. A careful look at the upper and lower bounds, however, illustrate the limits of the attack. The choice of b cannot be too large otherwise the upper bound becomes too restrictive (since the upper bound depends on the inverse of b^2). The choice of b must also ensure that the difference between $\frac{a}{b}\lambda(N)$ and d is not too small.

We revisit this problem and refine the analysis of Chen *et al.*, to obtain an improved bound for unsafe private exponents. The main result is contained in the following theorem.

Theorem 3.1. *Let $N = pq$ be an RSA modulus with balanced primes, let (e, N) be a valid public key and (d, p, q) be its corresponding private key,* *where* $ed \equiv 1 \pmod{\lambda(N)}$. *Given the public key, if* $g = \gcd(p-1, q-1)$ *and the private exponent* d *satisfies*

$$\frac{\phi(N)}{e} < \left|\frac{a}{b}\lambda(N) - d\right| < \frac{N^{1/4}}{2\sqrt{g}\,b}, \tag{3.3}$$

for some integers $a \geq 0$ *and* $b \geq 1$*, then the modulus* N *can be factored in time polynomial in* $\log_2(N)$.

Proof: We begin by letting $D = \frac{a}{b}\lambda(N) - d$, and assuming that the condition on the private exponent, (3.3), holds. Thus, we assume that

$$\frac{\phi(N)}{e} < |D| < \frac{N^{1/4}}{2\sqrt{g}\,b}. \tag{3.4}$$

Using the key equation $ed = 1 + k\lambda(N)$, notice that multiplying D by the public exponent e yields

$$\begin{aligned} eD &= e\left(\frac{a}{b}\lambda(N) - d\right) \\ &= \frac{ea}{b}\lambda(N) - \big(1 + k\lambda(N)\big) \\ &= \frac{(ea - kb)}{b}\lambda(N) - 1 \\ &= \frac{k'}{b}\lambda(N) - 1, \end{aligned} \tag{3.5}$$

where $k' = ea - kb$. Since $e < \lambda(N)$, it follows from (3.5) that $|k'/b| < |D|$, or

$$|k'| < |Db|. \tag{3.6}$$

Let $g = \gcd(p-1, q-1)$. Since $\lambda(N)$ can be written as $\phi(N)/g = (N - \Lambda)/g$, we can write (3.5) as

$$\begin{aligned} eD &= \frac{k'}{b}\lambda(N) - 1 \\ &= \frac{k'}{b}\frac{\phi(N)}{g} - 1 \\ &= \frac{k'}{b}\frac{(N - \Lambda)}{g} - 1 \\ &= \frac{k'_0 N}{bg_0} - \frac{k'_0 \Lambda}{bg_0} - 1, \end{aligned} \tag{3.7}$$

where $k_0' = k'/\gcd(k', g)$ and $g_0 = g/\gcd(k', g)$. Dividing this equation by DN then gives

$$\frac{e}{N} = \frac{k_0'}{Dbg_0} - \frac{k_0'\Lambda}{Dbg_0N} - \frac{1}{DN}. \tag{3.8}$$

Since $|k_0'| \leq |k'| < |Db|$, where the latter inequality comes from (3.6), we can rearrange (3.8) to obtain

$$\begin{aligned} \left|\frac{e}{N} - \frac{k_0'}{Dbg_0}\right| &= \left|-\frac{k_0'\Lambda}{Dbg_0N} - \frac{1}{DN}\right| \\ &\leq \left|\frac{k_0'\Lambda}{Dbg_0N}\right| + \left|\frac{1}{DN}\right| \\ &< \left|\frac{\Lambda}{g_0N}\right| + \left|\frac{1}{DN}\right|. \end{aligned} \tag{3.9}$$

Let's consider the two terms in the right-hand side of the last inequality. Since $\Lambda = p + q - 1 > p$ and $g_0 \leq g = \gcd(p-1, q-1) < p$, it follows that $|\Lambda/g_0| > 1$ and so the first term is greater than $1/N$. From the original assumption in the proof, (3.4), we know $e|D| > \phi(N)$, which implies that $|D| > 1$, since $\phi(N) > e$. Thus, the second term is less than $1/N$ and the sum of both terms must be less than twice the first. Combining this with $\Lambda < 3N^{1/2}$, it follows that

$$\begin{aligned} \left|\frac{e}{N} - \frac{k_0'}{Dbg_0}\right| &< \left|\frac{\Lambda}{g_0N}\right| + \left|\frac{1}{DN}\right| \\ &< \frac{2\Lambda}{g_0N} \\ &< \frac{6}{g_0N^{1/2}}. \end{aligned} \tag{3.10}$$

Since $g_0 \leq g$, notice that the upper bound $|D|$ in the original assumption, (3.4), satisfies

$$\begin{aligned} |D| &< \frac{N^{1/4}}{\sqrt{12g}\,b} \\ &\leq \frac{N^{1/4}}{\sqrt{12g_0}\,b}, \end{aligned} \tag{3.11}$$

which is sufficient to ensure that

$$\frac{6}{g_0N^{1/2}} \leq \frac{1}{2(Dbg_0)^2}. \tag{3.12}$$

Combining this inequality with the inequality in (3.10), then yields

$$\left|\frac{e}{N} - \frac{k_0'}{Dbg_0}\right| < \frac{1}{2(Dbg_0)^2}. \tag{3.13}$$

Now, even though D is not necessarily an integer, notice that

$$Db = a\lambda - kb,$$

is an integer. Therefore, from Theorem 2.1 (continued fractions), we know that $k_0'/(Dbg_0)$, in lowest terms, is one of the convergents in the continued fraction expansion of e/N.

Letting μ_i/η_i denote the i^{th} convergent of e/N, we will assume that the m^{th} convergent yields $k_0/(Dbg_0)$. That is, we assume $\mu_m/\eta_m = k_0/(Dbg_0)$. From the derivation of (3.5), recall that

$$eD = \frac{k'}{bg}\phi(N) - 1, \tag{3.14}$$

which can be rewritten as

$$\begin{aligned}\phi(N) &= \frac{eDbg}{k'} + \frac{bg}{k'}\\ &= \frac{eDbg_0}{k_0'} + \frac{bg_0}{k_0'}\\ &= \frac{e\eta_m}{\mu_m} + \frac{bg_0}{k_0'}.\end{aligned} \tag{3.15}$$

Since $|D| > \phi(N)/e$, from (3.4), it follows from (3.14) that $|k'/bg| > 1$. Thus, the last term in (3.15) satisfies

$$\begin{aligned}\left|\frac{bg_0}{k_0'}\right| &= \left|\frac{bg}{k'}\right|\\ &< 1.\end{aligned} \tag{3.16}$$

Therefore, given μ_m and η_m we can compute $\phi(N)$ as

$$\phi(N) = \begin{cases} \left\lceil \frac{e\eta_m}{\mu_m} \right\rceil & \text{if } k_0' > 0,\\ \left\lfloor \frac{e\eta_m}{\mu_m} \right\rfloor & \text{if } k_0' < 0.\end{cases} \tag{3.17}$$

Once $\phi(N)$ is known, the modulus N is easily factored by solving the system of equations $\{\phi(N) = (p-1)(q-1), N = pq\}$ for p and q. Thus, the correct convergent yields the factorization of N.

To find the correct convergent, we simply compute and test each convergent, in order, until we are able to factor the modulus N. In particular, for each convergent η_i/μ_i, we compute candidates for $\phi(N)$ given by $X = \lfloor e\eta_i/\mu_i \rfloor$ and $X = \lceil e\eta_i/\mu_i \rceil$. We then try to factor N by trying to solve the system of equations $\{X = (p-1)(q-1), N = pq\}$ for p and q. As shown above, the correct convergent will yield $\phi(N)$, which then yields the factorization of N. Since the total number of convergents in the continued fraction expansion of e/N is polynomial in $\log_2(N)$ and all operations are polynomial in $\log_2(N)$, the result follows. ❑

The result of Attack 3.1 is an improvement to that presented by Chen, Chang and Yang, given in (3.2), in two ways. First, the upper bound on $|\frac{a}{b}\lambda(N) - d|$ is increased by a factor b. Second, the effects of g are explicitly included in the bound.

3.3.1 Toy Example

We demonstrate Attack 3.1 with the following toy example. Consider an instance of RSA with public key

$$(e, N) = (243608017, 2613354137).$$

Given the public key we compute the continued fraction expansion of e/N,

$$[0, 10, 1, 2, 1, 2, 18, 1, 9, 4, 3, 2, 1, 3, 54, 13],$$

and the first few convergents

$$0, \frac{1}{10}, \frac{1}{11}, \frac{3}{32}, \frac{4}{43}, \frac{11}{118}, \frac{202}{2167}, \frac{213}{2285}, \ldots.$$

The first five convergents do not yield the factorization of N, so let's consider the 6th convergent, $\mu_6/\eta_6 = 11/118$. Let

$$\begin{aligned} X &= \left\lfloor \frac{e\eta_6}{\mu_6} \right\rfloor \\ &= \left\lfloor \frac{243608017 \times 118}{11} \right\rfloor \\ &= 2613249636. \end{aligned}$$

Solving the system

$$\begin{aligned} X &= (p-1)(q-1) \\ N &= pq, \end{aligned}$$

for integer solutions yields $p = 41443$ and $q = 63059$, both being prime. Thus, $N = 41443 \times 63059$, is the factorization of the modulus N. Using this information, we can compute $g = 2$, $\lambda(N) = 1306624818$, and the private exponent $d = 522649939$.

In this example, the private key was chosen to be close to $2\lambda(N)/5$. Notice that, when substituting the appropriate values, the sufficient condition for success of Attack 3.1, given by (3.3), becomes

$$10.73 < |D| < 15.99,$$

which is satisfied in this instance since

$$\begin{aligned} |D| &= \left|\frac{a}{b}\lambda - d\right| \\ &= \left|\frac{2}{5}1306624818 - 522649939\right| \\ &= 11.8. \end{aligned}$$

3.3.2 Very Small $|D|$

It is interesting to note that while Attack 3.1 (and the attack by Chen, Chang and Yang) works quite well when the private exponent is close to a rational multiple of λ, it does not necessarily work well when it is *extremely* close. In particular, when $e|D| < \phi(N)$, the attack no longer terminates in polynomial time. To see this recall that given the correct convergent μ_m/η_m, the formula for $\phi(N)$, from (3.15), is given by

$$\phi(N) = \frac{e\eta_m}{\mu_m} + \frac{bg_0}{k_0'}. \tag{3.18}$$

As shown above, when $e|D| > \phi(N)$, the absolute value of the last term in (3.18) is less than 1. This ensures that the number of candidates for $\phi(N)$ that we need to test for each convergent is only two (*i.e.*, the floor or ceiling of $e\eta_i/\mu_i$). When $e|D| < \phi(N)$, however, this is no longer the case.

Let's assume that $e|D| < \phi(N)$ and let ℓ be the smallest integer such that $e2^\ell > \lambda(N)$. Since $e < \lambda(N)$ we know that $\ell \geq 1$. With high probability the actual value of ℓ will be rather small, since it is expected that e will be large (since e is computed after d is chosen). Since $a, b, \lambda(N), d \in \mathbb{Z}$, it follows from

$$D = \frac{a}{b}\lambda(N) - d,$$

that $|D| \geq 1/b$. To see this, notice that $|D| < 1/b$ implies that $|D| = 0$, which in turn implies that either $\gcd(d, \lambda(N)) > 1$ or $d = a = 0$, and each of these conditions contradict the requirement that d be invertible in $\mathbb{Z}_{\lambda(N)}$ (as specified in the key generation algorithm). Thus, $|D| \geq 1/b$, and with high probability we will have $eD > 1$. From (3.5), $eDb = k'\lambda(N) + b$, it then follows that k' satisfies

$$\begin{aligned} |k'| &= \left| \frac{eDb + b}{\lambda} \right| \\ &> \left| \frac{eDb}{\lambda} \right| \\ &> \left| \frac{Db}{2^\ell} \right|. \end{aligned} \tag{3.19}$$

Using this bound for $|k'|$ and $1/|D| \leq b$, as shown above, notice that the last term in (3.18) satisfies

$$\begin{aligned} \left| \frac{bg_0}{k'_0} \right| &= \left| \frac{bg}{k'} \right| \\ &< \left| \frac{bg2^\ell}{Db} \right| \\ &= \left| \frac{g2^\ell}{D} \right| \\ &\leq \left| bg2^\ell \right|. \end{aligned} \tag{3.20}$$

Therefore, for each convergent in the continued fraction expansion of e/N, we need to compute $2bg2^\ell$ candidates for $\phi(N)$. For small b, g and ℓ this may be feasible. However, when $b = N^\beta$ for some

$$\frac{\log_2 \log_2(N)}{\log_2(N)} \ll \beta < \frac{1}{4},$$

this corresponds to a search space that is exponential in the bitlength of N. For example, a 1024-bit modulus with $b > N^{0.078125}$, corresponds to a search space greater than 2^{80}, which is currently believed to be infeasible.

3.4 Lattice-Based Attacks

In this section we consider lattice-based attacks on RSA with a private exponent close to a rational multiple of $\lambda(N)$. Rather than focusing on special cases, as was done in [60] and [24], we consider the general case where $|\frac{a}{b}\lambda(N) - d|$ is small for some integers $a \geq 0$ and $b \geq 1$. The main result is contained in the following theorem.

Attack 3.2. *For every $\epsilon > 0$ there exists N_0 such that for every $N > N_0$ the following holds: Let N be an RSA modulus with balanced primes, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $ed \equiv 1 \pmod{\lambda(N)}$ and the private exponent satisfies $|\frac{a}{b}\lambda(N) - d| < N^{\delta}$ for some integers $a \geq 0$ and $b \geq 1$. Given the public key $(e = N^{\alpha}, N)$, if the private exponent d, $g = \gcd(p-1, q-1) = N^{\gamma}$ and $b = N^{\beta}$ satisfy*

$$(i) \quad \gamma + \beta < \alpha + 1/4 \quad (or\ gb < eN^{1/4}) \quad and$$

$$(ii) \quad \delta < \frac{1}{4} + \alpha - \gamma - 2\beta - \frac{1}{4}\sqrt{12\alpha - 12\gamma - 12\beta + 3} - \epsilon,$$

then the modulus can be factored in time polynomial in $\log_2(N)$, provided that Assumption 2.13 holds.

The method of solution in our proof is an extension of Boneh and Durfee's method for solving the small inverse problem [14].

Justification: Let $D = \frac{a}{b}\lambda(N) - d$. Since D is not necessarily an integer we consider the equation $Db = a\lambda(N) - db$, where $Db \in \mathbb{Z}$. Recalling the key equation $ed = 1 + k\lambda(N)$, notice that multiplying Db by the public exponent e yields

$$\begin{aligned} eDb &= e(a\lambda(N) - db) \\ &= ea\lambda(N) - edb \\ &= ea\lambda(N) - (1 + k\lambda(N))b \\ &= (ea - kb)\lambda(N) - b \\ &= k'\lambda(N) - b, \end{aligned} \tag{3.21}$$

where $k' = ea - kb$. If follows from (3.21) and $e < \lambda(N)$ that k' satisfies

$$\begin{aligned} |k'| &< (|D| + 1)|b| \\ &< 2N^{\delta+\beta}. \end{aligned} \tag{3.22}$$

Replacing $\lambda(N)$ with $\phi(N)/g = (N - \Lambda)/g$ and multiplying through by g, (3.21) becomes

$$eDbg = k'(N - \Lambda) - bg. \tag{3.23}$$

This suggests we look for small solutions, modulo the public exponent e, of the polynomial

$$f_e(x, y, z) = x + Ny - yz, \tag{3.24}$$

since $(x_0, y_0, z_0) = (-gb, k', \Lambda)$ satisfies $f_e(x_0, y_0, z_0) \equiv 0 \pmod{e}$. We define the bounds

$$\begin{aligned} X &= N^{\beta+\gamma} \\ Y &= 2N^{\delta+\beta} \\ Z &= 3N^{1/2}. \end{aligned} \tag{3.25}$$

Recalling that $g = N^\gamma$, $b = N^\beta$, $|k'| < 2N^{\delta+\beta}$ and $\Lambda < 3N^{1/2}$, notice that $x_0 = -gb$, $y_0 = k'$ and $z_0 = \Lambda$ satisfy $|x_0| < X$, $|y_0| < Y$ and $|z_0| < Z$. From Theorem 2.10, we know that we can recover (x_0, y_0, z_0) provided that N is sufficiently large, Assumption 2.13 holds and

$$X^{1+4\tau} Y^{2+4\tau} Z^{1+4\tau+6\tau^2} < e^{1+4\tau}, \tag{3.26}$$

for some $\tau > 0$. Substituting the values for the bounds X, Y, Z, the modulus $e = N^\alpha$, and neglecting all factors that do not depend on N, inequality (3.26) is satisfied when

$$(\beta+\gamma)(1+4\tau) + (\delta+\beta)(2+4\tau) + \tfrac{1}{2}(1+4\tau+6\tau^2) - \alpha(1+4\tau) < 0,$$

or, collecting in terms of τ, when

$$3\,\tau^2 - 2\,(2\alpha - 2\gamma - 2\delta - 4\beta - 1)\,\tau + 3\,\beta + \gamma + 2\,\delta - \alpha + \tfrac{1}{2} < 0.$$

For any fixed non-negative α, γ, δ and β, the left-hand side of this inequality is minimized when $\tau = \frac{1}{3}(2\alpha - 2\gamma - 2\delta - 4\beta - 1)$. Substituting this value of τ into the last inequality and solving for δ we see that (3.26) is satisfied when

$$\delta < \frac{1}{4} + \alpha - \gamma - 2\beta - \frac{1}{4}\sqrt{12\alpha - 12\gamma - 12\beta + 3} - \epsilon,$$

where the ϵ term is added to correct for all the negligible and low order terms ignored in the methods implicit in Theorem 2.10. Thus, we have derived condition (ii) of the attack.

In order for the square root in this enabling equation to be real, notice that we require $12\alpha - 12\gamma - 12\beta + 3 > 0$. This simplifies to $\gamma + \beta < \alpha + 1/4$ (or $gb < eN^{1/4}$), which motivates condition (i) in the attack statement.

Therefore, for sufficiently large N, we can recover $(x_0, y_0, z_0) = (-bg, k', \Lambda)$ provided that conditions (i) and (ii) and Assumption 2.13 hold. Once $z_0 = \Lambda$ is known we can compute $\phi(N) = N - \Lambda$, which lets us compute the factorization of N. Since all computations are polynomial in $\log_2(N)$, the result follows. ❏

3.4.1 Attack with known g and b

In this section we consider a lattice-based attack on RSA with private exponent close to $\frac{a}{b}\lambda(N)$, for some integer $a \geq 0$ and $b > 0$, when the values of b and $g = \gcd(p-1, q-1)$ are both known. In this scenario, we show that the bound on δ in Attack 3.2 can be significantly increased.

The relevance of this attack, of course, depends on the availability of b and g. At this time, we can only suggest that b and g be determined by an exhaustive search if they are both small enough. For each guess, we mount the following attack until it succeeds in factoring the modulus.

Attack 3.3. *For every $\epsilon > 0$ there exists N_0 such that for every $N > N_0$ the following holds: Let N be an RSA modulus with balanced primes, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $ed \equiv 1 \pmod{\lambda(N)}$ and $|\frac{a}{b}\lambda(N) - d| < N^{\delta}$ for some integers $a \geq 0$ and $b \geq 1$. Given the public key $(e = N^{\alpha}, N)$, $g = \gcd(p-1, q-1) = N^{\gamma}$, and $b = N^{\beta}$, if the private exponent d satisfies*

$$\delta < \frac{1}{6} + \alpha + \gamma - \frac{1}{3}\sqrt{1 + 6\alpha + 6\beta + 6\gamma} - \epsilon,$$

then the modulus can be factored in time polynomial in $\log_2(N)$, provided Assumption 2.13 holds.

Justification: Let $D = \frac{a}{b}\lambda(N) - d$, and again consider the equation

$$Db = a\lambda(N) - db,$$

where $Db \in \mathbb{Z}$. Following in the same manner as in the proof of Theorem 3.2, we have

$$eDbg = k'(N - \Lambda) - bg,$$

where $k' = ea - kb$ satisfies $|k'| < 2N^{\delta+\beta}$. As b and g are now known, this this suggests that we look for small solutions, modulo ebg, of the polynomial

$$f_{ebg}(x, y) = Nx - xy - gb,$$

since $(x_0, y_0) = (k', \Lambda)$ satisfies $f_{ebg}(x_0, y_0) \equiv 0 \pmod{ebg}$. Define the bounds

$$\begin{aligned} X &= 2N^{\delta+\beta} \\ Y &= 3N^{1/2}, \end{aligned} \tag{3.27}$$

notice that $x_0 = k'$ and $y_0 = \Lambda$ satisfy $|x_0| < X$ and $|y_0| < Y$. Therefore, from Theorem 2.10, we know that for sufficiently large N we can recover (x_0, y_0) provided that

$$X^{2+3\tau} Y^{1+3\tau+3\tau^2} < (ebg)^{1+3\tau}, \tag{3.28}$$

for some $\tau > 0$ and Assumption 2.13 holds. Substituting the values for the bounds X, Y and the modulus $ebg = N^{\alpha+\beta+\gamma}$, inequality (3.28) is satisfied when (collected in terms of τ)

$$\frac{3}{2}\tau^2 + (\frac{3}{2} + 3\delta - 3\alpha - 3\gamma)\tau + 2\delta + \beta - \alpha - \gamma + \frac{1}{2} < 0,$$

where we have neglected all terms that do not depend on N. For any fixed non-negative α, β, γ and δ, the left-hand side of this inequality is minimized when $\tau = \alpha + \gamma - \delta - \frac{1}{2}$. Substituting this value for τ into the last inequality and solving for δ we see that (3.28) is satisfied when

$$\delta < \frac{1}{6} + \alpha + \gamma - \frac{1}{3}\sqrt{1 + 6\alpha + 6\beta + 6\gamma} - \epsilon,$$

where the ϵ term is added to correct for all the negligible and low order terms ignored in the methods implicit in Theorem 2.10. Thus, for sufficiently large N, we can compute $(x_0, y_0) = (k', \Lambda)$ provided that Assumption 2.13 holds. Like the previous attack, once Λ is known we can compute $\phi(N) = N - \Lambda$ and factor the modulus N. Since all computations are polynomial in $\log_2(N)$, the result follows. ❏

3.4.2 Toy Example

To demonstrate the lattice-based attacks presented in this chapter, we mount the method in Attack 3.3 (known b and g) on the same toy example from Section 3.3.1. The basic methodology is the same for both attacks. The only significant difference is that the first attack looks for small solutions of a trivariate polynomial while the second looks for small solutions of a bivariate polynomial. We chose to illustrate the second attack to reduce the space needed for the presentation.

Consider, once again, an instance of RSA with public key

$$(e, N) = (243608017, 2613354137).$$

Suppose also that we know $g = 2$ and $b = 5$. Letting $M = egb$, we will try to find small solutions, modulo M, of

$$f_M(x, y) = Nx - xy - gb,$$

since $(x_0, y_0) = (k', \Lambda)$ satisfies $f_M(x_0, y_0) \equiv 0 \pmod{M}$, and (k', Λ) is small in some sense. For some positive integer m (lattice parameter), we consider polynomials of the form

$$f_{i,j,k}(x, y) = x^i y^k f_M(x, y)^j M^{m-j},$$

where $i, k \geq 0$ and $0 \leq j \leq m$. For simplicity, we let $k = 0$ (*i.e*, f_M is not multiplied by any monomial that is a multiple of y). Notice that each polynomial has the root (x_0, y_0) modulo M^m. For $m = 2$, we choose the six polynomials corresponding to

$$(i, j) = \{(0, 2), (1, 1), (0, 1), (2, 0), (1, 0), (0, 0)\}, \tag{3.29}$$

with $k = 0$ in for each. Using the bounds X and Y (for x_0 and y_0) given by (3.27), we have

$$\begin{aligned} X &= 20 \\ Y &= 153363. \end{aligned} \tag{3.30}$$

With these bounds we consider the coefficient vectors $f_{i,j,k}(xX, yY)$ for each $(i, j, 0)$ pair in (3.29). The six coefficient vectors are each used as a lattice basis vector for a lattice $\mathcal{L}$, whose every element corresponds to a polynomial with root (x_0, y_0) modulo M^2. The basis matrix for $\mathcal{L}$, where each row is a basis vector, is given by

$$\mathcal{M} = \begin{bmatrix} X^2Y^2 & -2NX^2Y & 2gbXY & N^2X^2 & -2NgbX & g^2b^2 \\ & MX^2Y & & -NMX^2 & gbMX & \\ & & MXY & & -NMX & gbM \\ & & & M^2X^2 & & \\ & & & & M^2X & \\ & & & & & M^2 \end{bmatrix},$$

where the columns correspond to the coefficients of the monomials

$$x^2y^2, x^2y, xy, x^2, x, 1,$$

in that order. Using the LLL algorithm, we find that the two smallest vectors in the new basis correspond to the polynomials

$$\begin{aligned} p_1 = 3025x^2y^2 - 632225550x^2y + 60500xy \\ + 33033813725025x^2 - 6322255500x + 302500 \end{aligned}$$

$$\begin{aligned} p_2 = {} & 8305x^2y^2 + 700333660x^2y + 166100xy \\ & - 163877765181765x^2 + 7003336600x + 830500. \end{aligned}$$

If these polynomials were algebraically independent, we would simply solve the system $\{p_1(x,y) = 0, p_2(x,y) = 0\}$ for all integer solutions and test each solution. These polynomials are, in fact, not algebraically independent though. They have the common factor

$$\begin{aligned} p_{1,2}(x,y) &= \gcd(p_1(x,y), p_2(x,y)) \\ &= 550 + 55xy - 5747505x. \end{aligned}$$

It turns out that all polynomials corresponding to the vectors in the new lattice basis that are small enough to have (x_0, y_0) as a root over the integers have this common factor. Thus, Assumption 2.13 fails with this particular lattice[1]. While it seems that the attack has failed, there is still the common factor $p_{1,2}(x,y)$ to consider. Since this polynomial has such a simple structure (linear in each variable) we try to solve $p_{1,2}(x,y) = 0$ for any integer solutions. We find two integer solutions

$$(x,y) = \{(-11, 104501), (110, 104500)\}.$$

Notice that the first solution, $(y_0, x_0) = (-11, 104501)$, yields the correct value for $y_0 = \Lambda$ (and $x_0 = k'$). That is,

$$\begin{aligned} N - y_0 &= 2613354137 - 104501 \\ &= 2613249636 \\ &= \phi(N). \end{aligned}$$

We then solve the system $\{\phi(N) = (p-1)(q-1), N = pq\}$, for p and q, which yields the desired solution $p = 41443$ and $q = 63059$.

3.5 Typical Instances

In a typical instance of RSA with a specially chosen private exponent it is very likely that the public exponent will be roughly the same size as the modulus. Thus, the approximation $\alpha \approx 1$ (*i.e.*, $e \approx N$) will often be valid. In addition, if the balanced primes p and q are chosen randomly we expect

[1]This is actually expected. For the particular structure of the polynomial $f_M(x,y)$, Blömer and May [7, §5] showed that all reduced basis vectors are algebraically dependent when none of the basis vectors correspond to polynomials multiplied by a multiple of y.

that $g = \gcd(p-1, q-1)$ will be a very small even integer such as 2, 4 or 6. If p and q were simply random odd integers, we could easily compute the expected distribution of g and prove that it is very likely that g is a small even integer. Being odd primes, though, p and q have additional structure. For example, the distribution of primes modulo fixed integers is not uniform (*e.g.*, see [52]). Computing the expected distribution of g when p and q are odd primes seems to be a difficult problem and is beyond the scope of this work. We have, however, carried out some experiments to approximate this distribution when p and q are random primes with the same bitlength. In our experiments, we find that $g = 2$ with probability almost $1/2$ and that $g \leq 6$ with probability roughly $3/4$. Further, these probabilities were found to be independent of the bitlength of the primes (when the primes ranged from 128 to 1024 bits). For more information about these observations, see Appendix A.

Based on this experimental evidence we use the approximation $\gamma \approx 0$ (*i.e.*, we assume that g is very small). Using the approximations $\alpha \approx 1$ and $\gamma \approx 0$ in the result of the continued fraction attack in Section 3.3, condition (*ii*) of Attack 3.1 simplifies to $|D| < cN^{1/4-\beta}$, where c is a small positive constant that is independent of N. Thus, the sufficient condition for the continued fraction attack is given by

$$\delta < 1/4 - \beta - \epsilon, \tag{3.31}$$

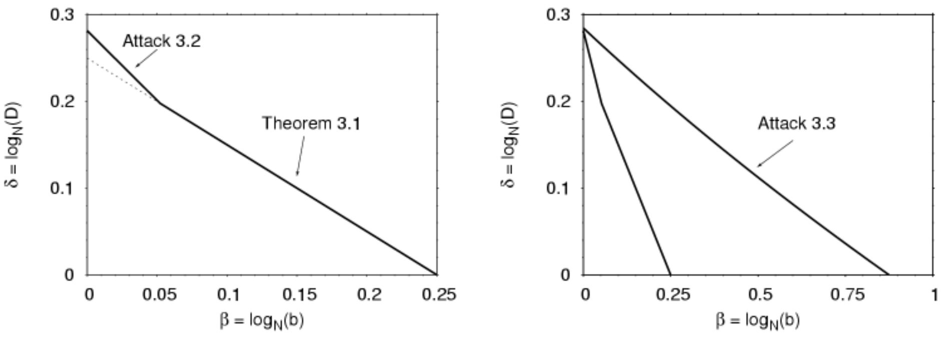
where ϵ can be made arbitrarily small by considering sufficiently large N. For the first lattice-based attack in Section 3.4, condition (*ii*) of Attack 3.2 simplifies to

$$\delta < \frac{5}{4} - 2\beta - \frac{1}{4}\sqrt{15 - 12\beta} - \epsilon, \tag{3.32}$$

and the result of the second lattice-based attack (with known g and b) reduces to

$$\delta < \frac{7}{6} - \frac{1}{3}\sqrt{7 + 6\beta} - \epsilon. \tag{3.33}$$

We illustrate these results in the plots in Figure 3.1, where we show the bound on the size of the private exponent δ as a function of the size of b (given by β). Figure 3.1(a) compares the bound for the first two attacks (without knowledge of b and g). Notice that the lattice-based attack is superior to the continued fraction attack for small values of b while it is



(a) Unknown b and g (b) Known b and g

Figure 3.1: Bounds for δ when $\alpha \approx 1$ and $\gamma \approx 0$. Plot (a) shows bounds from Theorem 3.1 (continued fractions) and Attack 3.2 (lattice-based). Plot (b) shows the bound from Attack 3.3 (lattice-based) with known b and g along with the bounds from plot (a).

inferior for larger values of b, with a crossover point at

$$\begin{aligned}\beta &= \frac{5}{8} - \frac{\sqrt{21}}{8} \\ &\approx 0.052.\end{aligned}$$

In Figure 3.1(b), we show the bound on the private exponent for Attack 3.3 when b and g are known. Also included in the plot are the bounds for the attacks with unknown b and g to illustrate the strength of knowing b and g. It is quite clear that the second lattice-based attack (known b and g) is significantly superior to the other attacks for large values of b. It should be pointed out though, that while the second lattice attack is much superior for larger values of b, it is not known how knowledge of such a large b can be obtained.

3.6 Discussion

In this chapter, we improved Chen, Chang and Yang's bound for private exponents close to a rational multiple of $\lambda(N)$ using Wiener's continued fraction method (*cf.* [23]). We also presented new, stronger, bounds using lattice-based techniques based on Boneh and Durfee's small private exponent attack.

Consider the two sets, C_1 and C_2, as defined below.

1. For any $\epsilon > 0$, let $C_1(\epsilon)$ be the set of RSA public keys (e, N) such that there exist integers $d, a, b = N^\beta$ satisfying

$$ed \equiv 1 \pmod{\lambda(N)}$$

$$\left|\frac{a}{b}\lambda - d\right| < N^{\delta_1 - \epsilon}$$

$$\delta_1 = \frac{1}{4} + \alpha - \gamma - 2\beta - \frac{1}{4}\sqrt{12\alpha - 12\gamma - 12\beta + 3},$$

where $a \geq 0$, $b \geq 1$, $e = N^\alpha$ and $g = \gcd(p-1, q-1) = N^\gamma$.

2. For any $\epsilon > 0$, let $C_2(\epsilon)$ be the set of RSA public keys (e, N) such that there exist integers $d, a, b = N^\beta$ satisfying

$$ed \equiv 1 \pmod{\lambda(N)}$$

$$\left|\frac{a}{b}\lambda - d\right| < N^{\delta_2 - \epsilon}$$

$$\delta_2 < \frac{1}{6} + \alpha + \gamma - \frac{1}{3}\sqrt{1 + 6\alpha + 6\beta + 6\gamma}$$

$$g = N^\gamma < \log_2^c(N)$$

$$b = N^\beta < \log_2^c(N),$$

where $a \geq 0$, $b \geq 1$, $e = N^\alpha$, $g = \gcd(p-1, q-1) = N^\gamma$ and $c > 0$ is a small constant (the size of g and b are each polynomial in the bitlength of N).

Using the notion of *weak keys*, as defined by Blömer and May [9], the results of this chapter show that these sets of RSA public keys are classes of weak keys. The size of these classes (for sufficiently small ϵ) are each greater than $N^{1/4-\varepsilon}$, since the subclass defined by private exponents vulnerable to Wiener's original attack (letting $a = 0$, $d > 0$) already has size $N^{1/4-\varepsilon}$. Here, $\varepsilon > 0$ is a small real number that accounts for the fact that private exponents must be relatively prime to $\lambda(N)$. The actual sizes of C_1 and C_2 are currently unknown.

In [9], Blömer and May present the class of weak keys defined by the set $C_{BM}(\epsilon)$ defined by the following: for any $\epsilon > 0$, let $C_{BM}(\epsilon)$ be the set of RSA public keys (e, N) such that there exist integers x, y, k satisfying

$$ex + y = k\phi(N)$$

$$x < \tfrac{1}{3}N^{1/4-\epsilon}$$

$$|y| < exN^{-3/4}.$$

This class of weak keys has size $N^{3/4-\varepsilon}$ when there is no restriction on the RSA primes (*i.e.*, they are not restricted to be balanced). One practical difficulty with the definition of this class of weak keys, though, is that it is difficult to determine whether or not a public key is in the class or not, even when we know the value of each of the parameters in the key equation $ed = 1 + k\phi(N)$.

In contrast, notice that determining membership in C_i is very straightforward. Letting $d = \frac{a}{b}\lambda(N)$, we simply solve the key equation for $\frac{a}{b}$, giving

$$\frac{a}{b} = \frac{1 + k\lambda(N)}{e\lambda(N)}.$$

Since all of the quantities on the right-hand side are known in the key generation algorithm, we can simply test a and b for membership in C_i (*i.e.*, test if $|\frac{a}{b}\lambda(N) - d| < N^{\delta_i}$). Thus, it is easier for a modified key generation algorithm to avoid producing a weak key defined by C_1 or C_2 than it is to produce one that avoids a weak key in C_{BM}. Some of the weak keys in C_{BM} are easy, however, to detect. This follows since $C_1 \subset C_{BM}$.

Open Problems/Future Work: While it is clear that $C_1 \subset C_{BM}$, it is not known if there is any relationship, between C_2 and C_{BM}. More work on the relationship and differences between the attacks in this Chapter and Blömer and May's may lead to a better characterization of known weak keys. The sizes of the weak key classes C_1 and C_2 are currently unknown. Computing lower bounds on these sizes would an interesting exercise.

Chapter 4

Common Small Private Exponents

In this chapter we show that Wiener's small private exponent attack, when viewed as a lattice-based attack, is easily extended to many instances of RSA with the same small private exponent. The new attack, while only a heuristic, works well in practice and can recover private exponents much larger than previously possible when at least two instances of RSA have the same private exponent. In the limiting case of many RSA instances with large moduli and common private exponent, the attack can recover private exponents approaching $N^{1/2}$.

4.1 Background

The problem of multiple instances of RSA with a common (small) private exponent has, to our knowledge, only been considered in the special case of Dual RSA (see [114] or Chapter 7) where two instances of RSA share a common private and public exponent. In Chapter 7, we show a heuristic lattice-based attack that can recover the common private exponent provided that it is smaller than $N^{1/3}$. The attacks in this section are a direct generalization of this attack to multiple instances of RSA (without the restriction of having a common public exponent).

There are also some other problems consisting of multiple instances of RSA with some common property that have been considered. In particular, the instances may share a common (small) public exponent or a common modulus. We give a brief overview of the research done for each of these below.

Common Public Exponent

There are no known attacks on multiple instances of RSA with the same small public exponent that lead to a total break of any of the instances. There are, however, some known protocol failures.

The first known protocol failure occurs when the same plaintext m is encrypted with several public keys (e, N_i), each having the same public exponent e and a different modulus N_i. The attack on this protocol failure was published in 1985 by Håstad [56], who mentions, without reference, that the attack was already known to at least Blum, Lieberherr and Williams. In fact, the attack was mentioned as early as 1983 by Blum [11]. Essentially, when the number of ciphertexts exceeds the common public exponent, the attack combines the ciphertexts together with the Chinese Remainder Theorem to obtain m^e, which yields m since e is known.

The next protocol failure occurs when ℓ related plaintexts $m_1, \ldots, m_\ell$, are each encrypted with the same public exponent e, but different modulus N_i. That is, the plaintext m_i is encrypted with the public key (e, N_i), where each modulus N_i, is unique. In this context, the plaintexts m_i are related if $m_i = f_i(m)$ for known polynomials $f_i(\cdot)$, for some unknown m. Here, m is the actual secret part of the plaintext. In 1985, Håstad [56, 57] presented a lattice-based method that recovers m provided that $\ell > \frac{1}{2}e(e+1)$ ciphertexts are known. This bound on the number of ciphertexts needed can be reduced to $\ell > e$, when Coppersmith's method for finding small solutions of univariate modular equations [33], is used instead of Håstad's method for solving simultaneous modular equation (*e.g.*, see [12]).

Collectively, the attacks on these protocol failures are often referred to as the *Håstad broadcast attack*.

Common Modulus

The common modulus protocol was an early, recurrent[1], proposal in which a central key authority (*i.e.*, a trusted third party) would generate an RSA modulus and distribute valid public/private key pairs, all with the same modulus, to users within the system. Only the central key authority would have knowledge of the factorization of the common modulus.

In 1983, Simmons [109] showed that a protocol failure existed when the same plaintext was encrypted with two different public keys, having the same modulus and relatively prime public exponents. Given the two

[1]It is mentioned, without reference, in [56], [39], and [90], that this type of protocol had been reinvented several times.

ciphertexts and the two public keys, he showed that the plaintext can be easily computed. This attack can be mounted by anyone that has access to the ciphertext and public keys.

In 1984, DeLaurentis [39] showed that the protocol was completely insecure. Given one public/private key pair $(e_1, N)/(d_1, N)$, and another public key (e_2, N), he showed that an integer d_2' satisfying $e_2 d_2' \equiv 1 \pmod{\phi(N)}$, can be deterministically computed in time polynomial in $\log_2(N)$. The integer d_2' can then be used as a decryption exponent for the encryption exponent e_2. Therefore, any user with a valid public/private key can compute a valid decryption exponent for any other user given their public key, without requiring knowledge of the factorization of the modulus N. In addition, using an idea attributed to Simmons, DeLaurentis showed that given a valid public/private exponent pair, the modulus can be factored with a probabilistic polynomial time Las Vegas algorithm using the results of Miller [88]. A deterministic polynomial time algorithm, using Coppersmith's method for finding small roots of bivariate integer equations, was presented in 2004 by May [81], and later refined by Coron and May [36]. All of these attacks can be mounted by any user in the group (*i.e.*, anyone possessing a valid public/private key pair with the common modulus).

When a single user generates many instances of RSA with the same modulus, the attacks of DeLaurentis and May are no longer relevant. There is, however, a small private exponent attack for this scenario. In 1999, Howgrave-Graham and Seifert [67] extended Wiener's small private exponent attack to many instances of RSA, each having the same modulus N. When all of the private keys are sufficiently small, their heuristic lattice-based attack can factor the modulus in time polynomial in $\log_2(N)$. In the limiting case of infinitely many public keys, an infinite modulus and infinite computing power the method can factor the modulus provided that each private exponent is smaller than $N^{1-\epsilon}$, where ϵ is some small positive quantity.

4.2 Notation and Assumptions

In this chapter we assume that a single user has generated r instances of RSA, each with the same small private exponent d. Thus, we have r key equations

$$\begin{aligned} e_1 d &= 1 + k_1 \phi(N_1) \\ &\vdots \\ e_r d &= 1 + k_r \phi(N_r), \end{aligned} \tag{4.1}$$

where $\phi(N_i) = N_i - \Lambda_i$ for each $i = 1, \dots, r$. Since the (common) private exponent is small, we also have $k_i < d$ for each $i = 1, \dots, r$. All of the moduli are assumed to be the same size and we arbitrarily order them so that $N_1 < N_2 < \cdots < N_r < 2N_1$. Further, we assume that each modulus N_i is balanced so that $|\Lambda_i| < 3N_i^{1/2}$ for each $i = 1, \dots, r$.

4.3 Wiener's Attack and Lattices

We briefly outline the relationship between Wiener's continued fraction attack and computing reduced lattice bases for a certain 2-dimensional lattice. For more detail, see [7, Theorem 5], where Blömer and May show the equivalence of the two methods, using a slightly different lattice basis.

Since computing continued fractions and lattice basis reduction in two dimensions are equivalent, we can reformulate Wiener's attack as a lattice-based problem. We begin by writing the key equation, $ed = 1 + k(N - \Lambda)$, along with the trivial equation $dN^{1/2} = dN^{1/2}$, as the following vector-matrix equation

$$(d, k) \begin{bmatrix} N^{1/2} & e \\ & -N \end{bmatrix} = (dN^{1/2}, 1 - k\Lambda), \tag{4.2}$$

where empty entries in a matrix denote zeroes. Since $(d, k) \in \mathbb{Z}^2$, we know that $(dN^{1/2}, 1 - k\Lambda)$ is a vector in the lattice $\mathcal{L}$, generated by the rows of the matrix

$$\mathcal{B} = \begin{bmatrix} N^{1/2} & e \\ & -N \end{bmatrix}.$$

Using the LLL algorithm, with $\mathcal{B}$ as input, we find that the smallest reduced basis vector is $(dN^{1/2}, 1 - k\Lambda)$ whenever d satisfies the bound given in Wiener's continued fraction attack ($d < cN^{1/4}$, where c is a small constant that does not depend on N). Recovering the vector $(dN^{1/2}, 1 - k\Lambda)$ yields the private exponent d (and allows the modulus to be factored easily).

4.4 Heuristic Attack

When two or more instances of RSA share a common small private exponent, the lattice-based version of Wiener's small private exponent attack is easily extended. The main result is the following attack.

Attack 4.1. *For any integer $r \geq 1$, let $N_1, N_2, \ldots, N_r$ be balanced n-bit RSA moduli satisfying $N_1 < N_2 < \cdots < N_r < 2N_1$. Let $(e_1, N_1), \ldots, (e_r, N_r)$ be valid RSA public keys each with the same private exponent $d < N_r^\delta$. If*

$$\delta < \frac{1}{2} - \frac{1}{2(r+1)} - \log_{N_r}(6), \tag{4.3}$$

then there exists a heuristic lattice-based algorithm that reveals d in time polynomial in $\log_2(N_r)$ and r.

Justification: Given the r public keys $(e_1, N_1), \ldots, (e_r, N_r)$, we begin by considering the r key equations, $e_i d = 1 + k_i(N_i - \Lambda_i)$, along with the trivial equation $d = d$, written as

$$\begin{array}{llllll} d & & & & = & d \\ e_1 d & -N_1 k_1 & & & = & 1 - k_1\Lambda_1 \\ e_2 d & & -N_2 k_2 & & = & 1 - k_2\Lambda_2 \\ \vdots & & \ddots & & \vdots & \vdots \\ e_r d & & & -N_r k_r & = & 1 - k_r\Lambda_r. \end{array} \tag{4.4}$$

Notice that this system of $r+1$ equations can be written as the vector-matrix equation

$$(d, k_1, \ldots, k_r)\mathcal{B}_0 = (d, 1 - k_1\Lambda_1, \ldots, 1 - k_r\Lambda_r)$$

where

$$\mathcal{B}_0 = \begin{bmatrix} 1 & e_1 & e_2 & \cdots & e_r \\ & -N_1 & & & \\ & & -N_2 & & \\ & & & \ddots & \\ & & & & -N_r \end{bmatrix}. \tag{4.5}$$

Letting $x = (d, k_1, \ldots, k_r)$ and $v_0 = (d, 1 - k_1\Lambda_1, \ldots, 1 - k_r\Lambda_r)$, we can further simplify this to $x\mathcal{B}_0 = v_0$. Notice that the components of v_0 are not equally sized. In particular, the first coordinate is much smaller than each of the rest. However, multiplying the first column of $\mathcal{B}_0$ by $M = \lfloor N_r^{1/2} \rfloor$, we can construct a new matrix

$$\mathcal{B} = \begin{bmatrix} M & e_1 & e_2 & \cdots & e_r \\ & -N_1 & & & \\ & & -N_2 & & \\ & & & \ddots & \\ & & & & -N_r \end{bmatrix}, \tag{4.6}$$

so that $x\mathcal{B}$ yields a vector with components that are, roughly, the same size. In particular,

$$(d, k_1, \ldots, k_r)\mathcal{B} = (dM, 1 - k_1\Lambda_1, \ldots, 1 - k_r\Lambda_r),$$

where each component of the vector on the right-hand side of the equation has size, in absolute value, roughly equal to $N_r^{\delta+1/2}$. Letting

$$v = (dM, 1 - k_1\Lambda_1, \ldots, 1 - k_r\Lambda_r),$$

we see that

$$\begin{aligned} \|v\|_2^2 &= (dM)^2 + (1 - k_1\Lambda_1)^2 + \cdots + (1 - k_r\Lambda_r)^2 \\ &\leq \left(N_r^{\delta+1/2}\right)^2 + r\left(3N_r^{\delta+1/2}\right)^2 \\ &= (9r+1)\left(N_r^{\delta+1/2}\right)^2, \end{aligned}$$

so that the size of v is bounded by

$$\|v\|_2 \leq \sqrt{9r+1}\, N_r^{\delta+1/2}. \tag{4.7}$$

Since $x\mathcal{B} = v$ and $x \in \mathbb{Z}^{r+1}$, we know that v is an integer linear combination of the rows of $\mathcal{B}$. Letting $\mathcal{L}$ be the lattice generated by the rows of $\mathcal{B}$ (*i.e.*, $\mathcal{B}$ is a basis matrix), we then have $v \in \mathcal{L}$. When $\delta < 1/2$, notice that the size of v is small compared to the size of each of the basis vectors, whose absolute size is roughly N_r. This suggests that we look for small vectors in $\mathcal{L}$ with the hope that v is found. From Minkowski's bound on a smallest vector in a lattice, Theorem 2.3, we know that a smallest vector in $\mathcal{L}$ is smaller than

$$\sqrt{r+1}\,\mathrm{vol}(\mathcal{L})^{1/(r+1)},$$

with respect to the Euclidean norm. From (4.6), it is easy to see that the determinant of $\mathcal{L}$ satisfies

$$\begin{aligned}\mathrm{vol}(\mathcal{L}) &= \left| M \prod_{i=1}^{r} (-N_i) \right| \\ &= \left\lfloor N_r^{1/2} \right\rfloor \prod_{i=1}^{r} N_i \\ &> N_1^{1/2} N_1^r,\end{aligned}$$

since $\lfloor N_r^{1/2} \rfloor \geq N_1^{1/2}$ and $N_i > N_1$, for all $i = 2, \ldots, r$. Combining this with $N_r < 2N_1$, we then have

$$\mathrm{vol}(\mathcal{L}) > \left(\frac{N_r}{2} \right)^{r+1/2}. \tag{4.8}$$

From the bounds on v and $\mathrm{vol}(\mathcal{L})$, given by (4.7) and (4.8), it follows that a sufficient condition for v to satisfy Minkowski's bound for a smallest vector is given by

$$\sqrt{9r+1}\, N_r^{\delta+1/2} \leq \sqrt{r+1} \left(\frac{N_r}{2} \right)^{\frac{r+1/2}{r+1}}.$$

Looking at the exponents of N_r, this is equivalent to

$$\delta + \frac{1}{2} + \log_{N_r}\left(\sqrt{9r+1}\right) \leq \frac{r+\frac{1}{2}}{r+1} + \log_{N_r}\left(\frac{\sqrt{r+1}}{2^{\frac{r+1/2}{r+1}}} \right),$$

or simply

$$\begin{aligned}\delta &< \frac{1}{2} - \frac{1}{2(r+1)} + \log_{N_r}\left(\sqrt{\frac{r+1}{9r+1}} \frac{1}{2^{\frac{r+1/2}{r+1}}} \right) \\ &= \frac{1}{2} - \frac{1}{2(r+1)} - \log_{N_r}\left(\sqrt{\frac{9r+1}{r+1}}\, 2^{\frac{r+1/2}{r+1}} \right).\end{aligned} \tag{4.9}$$

Now, since

$$\begin{aligned}
\log_{N_r}\left(\sqrt{\frac{9r+1}{r+1}}2^{\frac{r+1/2}{r+1}}\right) &= \log_{N_r}\left(\sqrt{\frac{9r+1}{r+1}}\right) + \log_{N_r}\left(2^{\frac{r+1/2}{r+1}}\right) \\
&= \log_{N_r}\left(\sqrt{\frac{9r+1}{r+1}}\right) + \left(\frac{r+1/2}{r+1}\right)\log_{N_r}(2) \\
&< \log_{N_r}(\sqrt{9}) + \log_{N_r}(2) \\
&= \log_{N_r}(6),
\end{aligned}$$

for all $r \geq 1$, it follows that

$$\delta < \frac{1}{2} - \frac{1}{2(r+1)} - \log_{N_r}(6),$$

is a sufficient condition to ensure that v satisfies Minkowski's bound for a smallest vector in $\mathcal{L}$. From Heuristic 2.6, it is then likely that v is a smallest vector in $\mathcal{L}$ (we know that v cannot be the smallest vector since $-v \in \mathcal{L}$). Using the LLL algorithm, with $\mathcal{B}$ as input, we compute a reduced basis. Let b be the smallest reduced basis vector returned by LLL. If v and $-v$ are the smallest vectors in $\mathcal{L}$ and if the next smallest vector is significantly larger than v, then we expect LLL to compute v (or $-v$) as the smallest basis vector. When this is the case, we simply divide the absolute value of the first component of b by M to reveal the common private exponent d. Since the dimension of $\mathcal{L}$ is linear in r and all components of the basis vectors in $\mathcal{L}$ are polynomial in $\log_2(N_r)$, the result follows. ❏

4.5 Toy Example

To illustrate the lattice-based method of Attack 4.1, we mount the attack on the three RSA public keys

$$\begin{aligned}
(e_1, N_1) &= (587438623, 2915050561) \\
(e_2, N_2) &= (2382816879, 3863354647) \\
(e_3, N_3) &= (2401927159, 3943138939).
\end{aligned}$$

Letting $M = \lfloor N_3^{1/2} \rfloor = 62794$, we construct the basis matrix

$$\begin{aligned}\mathcal{B} &= \begin{bmatrix} M & e_1 & e_2 & e_3 \\ 0 & -N_1 & 0 & 0 \\ 0 & 0 & -N_2 & 0 \\ 0 & 0 & 0 & -N_3 \end{bmatrix} \\ &= \begin{bmatrix} 62794 & 587438623 & 2382816879 & 2401927159 \\ 0 & -2915050561 & 0 & 0 \\ 0 & 0 & -3863354647 & 0 \\ 0 & 0 & 0 & -3943138939 \end{bmatrix}.\end{aligned}$$

Applying the LLL algorithm, with $\mathcal{B}$ as input, we obtain the reduced basis

$$\begin{bmatrix} -41130070 & 14375987 & 50221643 & 50147516 \\ -164834250 & 35361394 & -123133882 & 20371086 \\ 56702982 & -82125533 & -204896642 & 213808127 \\ -172055560 & -473917348 & 151104970 & -181526469 \end{bmatrix},$$

whose smallest basis vector is

$$b = (-41130070, 14375987, 50221643, 50147516)\,.$$

Dividing the first component of b by M and taking the absolute value yields

$$\left| \frac{-41130070}{62794} \right| = 655,$$

which is the common private exponent d. Here, $\delta = \log_{N_3}(d) \approx 0.2935$.

For this example, notice that the sufficient condition on the size of δ for the attack to succeed, (4.3), is given by

$$\begin{aligned}\delta &< \frac{1}{2} - \frac{1}{2(r+1)} - \log_{N_3}(6) \\ &= \frac{1}{2} - \frac{1}{8} - \log_{3943138939}(6) \\ &\approx 0.3189434953,\end{aligned}$$

which is satisfied in this example since $\delta \approx 0.2935$.

4.6 Practical Effectiveness

Since Attack 4.1 is only a heuristic, its true value lies in its effectiveness in practice. In Figures 4.1 and 4.2, we show the success rate of mounting the attack on random instances of RSA with 1024-bit moduli when a common small private is shared among several moduli. For several values of r ranging from 2 to 35, we show the success rate as a function of $\Delta = \delta - \delta_0$, where δ is the size of the private exponent and

$$\delta_0 = \frac{1}{2} - \frac{1}{2(r+1)} - \log_{N_r}(6),$$

is the maximum δ allowed by Attack 4.1, which ensures success. Each data point in each plot represents the average taken over several repeated random instances. The number of experiments for each data point ranged from 1,500 when $r = 2$ (here the time for lattice reduction was very small) to 150 when $r = 35$ (here the time for lattice reduction was much larger).

If Heuristic 2.6 holds, we expect a success rate of 100% for each $\Delta < 0$ and a sucess rate near 100% when $\Delta \approx 0$ (when the target vectors are close to Minkowski's bound). Any success rate greater than 0% when $\Delta > 0$ is an added bonus since Attack 4.1 does not apply to this region.

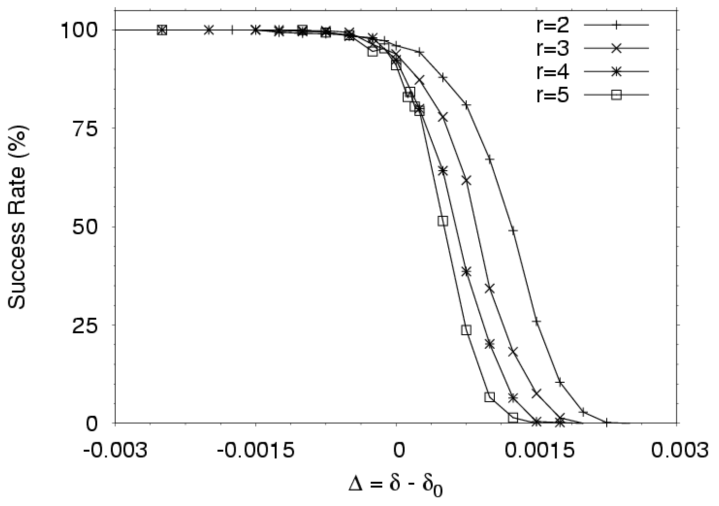


Figure 4.1: Effectiveness of Attack 4.1 for 1024-bit moduli and $2 \leq r \leq 5$.

As can be seen in Figures 4.1 and 4.2, the attack works extremely well until Δ is approximately equal to zero, at which point the success rate rapidly descends to 0%. As the number of instances increases (*i.e.*, r increases), the effectiveness of the attack seems to diminish more rapidly when Δ is close to and greater than zero. Independent of the number of instances, we have observed that every experiment resulted in a successful attack when

$\Delta < -0.0025$. Thus, we conclude that Heuristic 2.6 holds for the lattices used in Attack 4.1. Further, we conclude that Attack 4.1 is effective in practice for multiple instances of RSA with 1024-bit moduli.

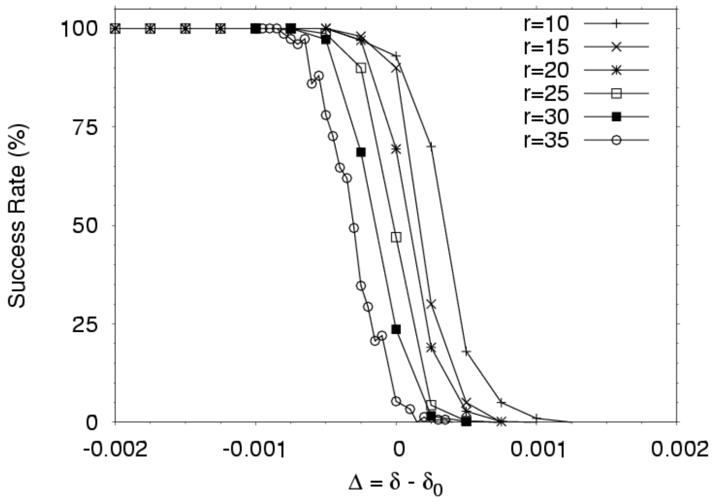


Figure 4.2: Effectiveness of Attack 4.1 for 1024-bit moduli and $10 \leq r \leq 35$ (in multiples of 5).

In Figure 4.3, we illustrate the effectiveness of the attack for different modulus sizes when three instances of RSA share a common small private exponent (*i.e.*, $r = 3$). As can be seen in the plot, the attack remains very effective for each modulus size that we considered. As the size of the moduli increase, the sharpness of the cut-off point between a successful attack and an unsuccessful attack becomes much pronounced (seemingly tending towards a step function). From this experimental evidence, we conclude that Attack 4.1 is effective in practice for three instances of RSA whose moduli range from 512- to 4096-bits.

Based on all the experimental evidence that we acquired, we conclude that Attack 4.1 is extremely effective in practice for any reasonable number of instances and size of moduli.

4.7 Discussion

In this chapter, we extended Wiener's small private exponent attack, when viewed as a lattice-based attack, to sets of RSA instances having a common

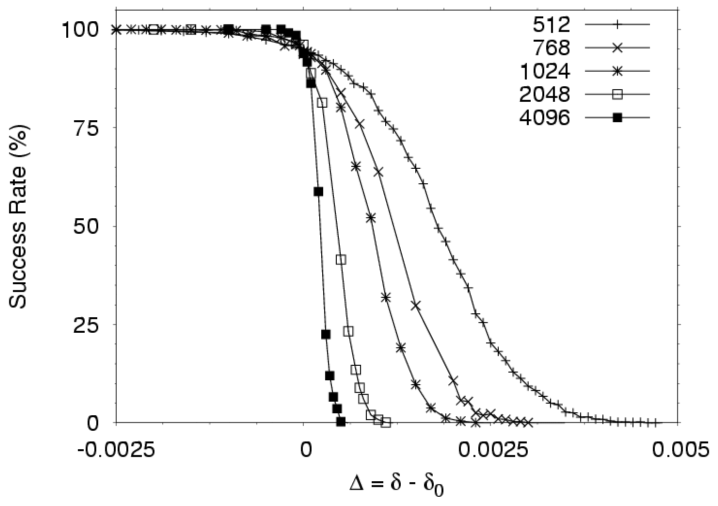


Figure 4.3: Effectiveness of Attack 4.1 for $r = 3$ with different modulus sizes $(512, 768, 1024, 2048, 4096)$.

small private exponent. The attack relies on an assumption about small vectors in lattices and so it is only a heuristic. However, it turns out that the assumption holds quite well in practice for the particular lattices considered and the attack works extremely well. In the limiting case of many RSA instances with large moduli sharing a common private exponent, the attack is expected to recover private exponents approaching 1/2 the bitlength of the moduli.

Open Problems/Future Work: The next step for research in this particular problem is to prove that $(dM, 1 - k_1\Lambda_1, \ldots, 1 - k_r\Lambda_r)$ is a smallest vector in the lattices considered here or to construct a different provable attack.

Chapter 5

Multi-prime RSA

In this chapter we investigate the security of multi-prime RSA, a variant of RSA in which the modulus has three or more distinct prime factors.

5.1 Background

Multi-prime RSA is, essentially, as old as RSA itself. The idea of using more than two primes in the modulus appears in the patent for RSA [103], which was filed in 1977. Even though the idea of multi-prime RSA was already contained in the patent for RSA, a patent for multi-prime RSA was granted in 1998 to Collins, Hopkins, Langford and Sabin [29]. Multi-prime RSA gained some credibility as a public key cryptosystem in 2000, when RSA Security Inc.[1] entered into an agreement with Compaq Computer Corporation to use Compaq's patented *MultiPrime* technology [30] in their RSA BSAFE product line. The patent is actually that of Collins *et al.*

The appeal of multi-prime RSA is that decryption costs can be reduced compared to RSA. Like RSA, decryption computations can be done modulo each prime and then combined with the Chinese Remainder Theorem. The overall cost for decryption decreases as the number of primes in the modulus increases.

While multi-prime RSA had been known for as long as RSA and was being used in commercial software, there had been very little research into the security of it until fairly recently.

In 2002, Boneh and Shacham [19] considered the security of multi-prime RSA with respect to factoring the modulus with the NFS and ECM.

[1]Now RSA, The Security Division of EMC.

In 2002, Hinek, Low and Teske [64] extended some small private exponent and partial key exposure attacks on RSA to multi-prime RSA.

In 2002, Ciet, Koeunne, Laguillaumie and Quisquater [25] extended the main small private exponent attacks on RSA to several variants of RSA including multi-prime RSA.

In 2004, Hinek [59] extended some partial key exposure attacks on RSA, by Blömer and May [8], to multi-prime RSA.

In 2005, Hinek [61] presented some new small private exponent partial key exposure attacks on multi-prime RSA. In 2006, we improved these attacks in [63].

5.2 Multi-prime RSA

Multi-prime RSA is a simple extension of RSA in which the modulus has more than two primes. Throughout this chapter, though, we will assume that RSA is a special case of multi-prime RSA in which there are only two primes. We will, also, often refer to multi-prime RSA having r primes in the modulus as "r-prime RSA". Most of the notation and assumptions that we use in this chapter are direct extensions of those used for RSA.

For multi-prime RSA with r primes, the modulus, $N = \prod_{i=1}^{r} p_i$, is simply the product of r distinct primes. As with RSA, we only consider multi-prime RSA with balanced primes. That is, if we label the primes so that $p_i < p_{i+1}$ for $i = 1, \ldots, r-1$, then we assume that

$$4 < \frac{1}{2}N^{1/r} < p_1 < N^{1/r} < p_r < 2N^{1/r}. \tag{5.1}$$

The key generation algorithm for multi-prime RSA is essentially the same as for RSA, except that the modulus requires r random distinct balanced primes instead of two. We will assume that the public and private exponents are defined modulo $\phi(N) = \prod_{i=1}^{r}(p_i - 1)$. Thus, e and d must satisfy

$$ed \equiv 1 \pmod{\phi(N)}, \tag{5.2}$$

which we call the key relation. From this equivalence, we have the key equation

$$ed = 1 + k\phi(N), \tag{5.3}$$

where k is some positive integer. As with RSA, we use Λ to denote the difference between the modulus N and Euler's totient function $\phi(N)$. That is, $N = \phi(N) - \Lambda$. Expanding $\phi(N)$, we see that Λ can be written as

$$\begin{aligned}\Lambda &= N - \phi(N)\\ &= N - \prod_{i=1}^{r}(p_i - 1)\\ &= \sum_{i=1}^{r}\frac{N}{p_i} - \sum_{\substack{i,j=1\\ i<j}}^{r}\frac{N}{p_ip_j} + \sum_{\substack{i,j,k=1\\ i<j<k}}^{r}\frac{N}{p_ip_jp_k} + \cdots + (-1)^r. \qquad (5.4)\end{aligned}$$

As is shown in [64], a simple computation using this expression for Λ and 5.1 (condition for balanced primes) shows that Λ satisfies

$$|\Lambda| < (2r-1)N^{1-1/r}.$$

Thus, $\phi(N)$ and N have roughly an $(r-1)/r$ fraction of their most significant bits in common.

The encryption algorithm for multi-prime RSA is identical to that of RSA. The public (encrypting) exponent will usually be denoted by $e = N^{\alpha}$.

Just as with RSA, there are two kinds of decryption algorithms. Textbook decryption for multi-prime RSA is identical that of RSA. In this case, the private exponent is denoted by $d = N^{\beta}$ or $d = N^{\delta}$, depending on the context. For partial key exposure attacks we let $d = N^{\beta}$ and use δ as an estimate of the unknown part of d. For example, if $\widehat{d}$ is a known approximation to d, we let the unknown part satisfy $|d - \widehat{d}| \leq N^{\delta}$. In all other scenarios, we let $d = N^{\delta}$. We will refer to this case as textbook multi-prime RSA, or simply multi-prime RSA.

When decryption uses the Chinese Remainder Theorem, the decryption algorithm for multi-prime RSA is the obvious extension to the decryption algorithm for RSA when using the Chinese Remainder Theorem (we simply compute r partial decryptions before the combining stage instead of two). In this case, the private exponent is denoted by $d = N^{\beta}$ and the CRT-exponents, $d_i = d \bmod (p_i - 1)$, each satisfy $d_i < N^{\delta}$. We will refer to this case as CRT multi-prime RSA, multi-prime RSA with CRT decryption.

The public key is simply (e, N) and the private key is $(d, p_1, \ldots, p_r)$[2].

Efficiency of Multi-prime RSA

The efficiency of multi-prime RSA compared to RSA varies greatly depending on the decryption method used.

[2]Other possibilities for the private key exist which contain information that can be used to speed up decryption computations (see PKC #1 [104]).

When textbook decryption is used, the encryption and the decryption algorithms for multi-prime RSA are identical to RSA. Thus, the computational costs of encryption and decryption are the same. The key generation algorithm, however, is different. For multi-prime RSA the key generation algorithm needs to generate r random primes each of size $N^{1/r}$. For RSA, the algorithm needs to generate two primes each size $N^{1/2}$. Using the Miller-Rabin primality test with trial division, we can generate an n-bit random (probable) prime with expected runtime $O(n^4/\log(n) + tn^3)$ (see Shoup [107]). Here, the method mistakenly outputs a composite number instead of a prime number with probability at most 4^{-t}. Since this complexity of finding primes is not linear[3] in the bitlength of the desired prime, multi-prime RSA key generation will be more efficient than the RSA key generation algorithm (when generating a modulus of the same size). While this may be desirable for some (constrained) applications that require the generation of many key pairs, this is not, in general, the reason for the interest in multi-prime RSA.

When the Chinese Remainder Theorem is used for decryption, not only is the key generation algorithm for multi-prime RSA more efficient than that for RSA, but so is the decryption algorithm. Using basic quadratic complexity for multiplication, the ratio of the worst case cost for multi-prime decryption to the worst case cost for RSA decryption is $4/r^2$ when computed sequentially and $8/r^3$ when computed in parallel. (We measure the cost by the number of bit operations and ignore the combining stage of the Chinese Remainder Theorem.) Therefore, the cost for decryption decreases with each additional prime in the modulus. Of course, as we shall see in the next section, the number of primes cannot be too large. As the number of primes increases (for a fixed modulus size) the size of each prime decreases, making the modulus easier to factor with the ECM method for factoring.

Multi-prime RSA is always implemented using the Chinese Remainder Theorem for decryption in practice. As we shall see below, all of the known attacks on multi-prime RSA (except simply factoring the modulus) apply to textbook multi-prime RSA (which uses the normal decryption method).

[3]This complexity estimation is using the classical quadratic-time algorithms. Faster algorithms are known, but the complexity of generating a random (probable) prime is still super-linear with these algorithms.

5.3 Factoring the Modulus

The security of multi-prime RSA, just as with RSA, is based on the difficulty of factoring the modulus. For a given modulus size, the estimated security of r-prime RSA should be no less than the estimated security of RSA with the same modulus size. Thus, the number of primes in the modulus must be small enough so that the expected complexity of the ECM is not less than the expected complexity of the NFS. Any value of r that satisfies this is considered a "safe" value for that given modulus size.

In Table 5.1, we list the estimated maximum number of balanced primes that are considered safe for various popular modulus sizes. The data in the table is taken from [30], and was determined by the crossover point of the expected runtimes of the NFS and ECM (see equations (1.1) and (1.2)).

Table 5.1: Estimated maximum number of safe primes allowed for multi-prime RSA various modulus sizes.

Modulus size (bitlength)	1024	2048	4096	8192
Maximum number of primes (r')	3	3	4	5

For a given modulus size, if $r \leq r'$, then the minimum expected complexity of factoring an r-prime RSA modulus (using either NFS or ECM) is no less than factoring an RSA modulus with the same size. Once $r > r'$, the expected complexity of factoring an r-prime RSA modulus with the ECM is less than factoring an RSA modulus of the same size. And since the sizes of the primes decrease with increasing number of primes, the expected complexity of factoring an r-prime RSA modulus with $r > r'$ decreases with increasing number of primes. Thus, the strength of the factoring attack (*i.e.*, ECM) increases with each addition prime (when $r > r'$).

As will be illustrated in the remainder of this chapter, this correspondence of security with number of primes is unique. In all other known attacks on multi-prime RSA, for $r > 2$, the attacks decrease in strength with each additional prime in the modulus.

5.4 Small Private Exponent Attacks

All of the small exponent attacks on textbook RSA have been extended to multi-prime RSA. In this section we list the main results.

Wiener's continued fraction attack was extended to multi-prime RSA by Hinek, Low and Teske [64] and also by Ciet *et al.* [25]. The result of the attack on multi-prime RSA, [64, Theorem 2], is given below.

Theorem 5.1. *For every integer $r \geq 2$ the following holds: Let N be an r-prime RSA modulus with balanced primes. Given a valid public key (e, N) with corresponding private key $(d, p_1, \ldots, p_r)$, if*

$$d < \frac{N^{1/(2r)}}{\sqrt{2(2r-1)}},$$

then the private exponent can be computed in time polynomial in $\log_2(N)$.

Letting $r = 2$ recovers the condition $d < N^{1/4}/\sqrt{6}$, obtained by Boneh [12], which is the same as Wiener's original result [123] up to a multiplicative constant. A proof of Theorem 5.1 can be found in [64, §4.1].

Blömer and May's lattice-based attack was extended to multi-prime RSA (as well as to arbitrary public exponent) by Hinek, Low and Teske [64, equation 18]. The result is as follows[4].

Attack 5.2. *For every $\epsilon > 0$ and integer $r \geq 2$ there exists an N_0 such that, for every $N > N_0$, the following holds: Let N be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public key and let $(d, p_1, \ldots, p_r)$ be its corresponding private key, where $e = N^\alpha$ and $d = N^\delta$. Given the public key, if the private exponent satisfies*

$$\delta \leq \frac{6}{5r} - \frac{1}{5} - \frac{3\alpha}{5} + \frac{2}{5r}\sqrt{\alpha^2 r^2 - \alpha r(r-1) + 4(r-2)^2} - \epsilon,$$

then d can be recovered in time polynomial in $\log_2(N)$*, provided that Assumption 2.13 holds.*

Letting $r = 2$ and $\alpha = 1$ recovers the bound $\delta < (\sqrt{6} - 1)/5 \approx 0.290$, originally obtained by Blömer and May [7, §4]. A proof of the result in Attack 5.2 can be found in [64, §4.2].

The strongest small private exponent attack on RSA, Boneh and Durfee's lattice-based attack with geometrically progressive matrices, was extended to multi-prime RSA by Ciet *et al.* [25]. The result of this extension is as follows.

[4]The notation used in [64] is slightly different as they use the variable $a_r = (1-r)/r$ to simplify their presentation.

Attack 5.3. *For every $\epsilon > 0$ and integer $r \geq 2$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public key and let $(d, p_1, \ldots, p_r)$ be its corresponding private key, $d = N^{\delta}$. Given the public key, if $e \approx N$ and the private exponent satisfies*

$$\delta \leq 1 - \sqrt{1 - \frac{1}{r}} - \epsilon,$$

then the private exponent d can be recovered in time polynomial in $\log_2(N)$, provided that Assumption 2.13 holds.

Letting $r = 2$ recovers the bound $\delta \leq 1 - \sqrt{1/2} \approx 0.292$, originally obtained by Boneh and Durfee [13, §5]. A proof of the result in Attack 5.3 can be found in [25, §4.2.1].

5.5 Partial Key Exposure Attacks: Known MSB

In this and the next two subsections, we consider partial key exposure attacks. These attacks assume that the adversary has knowledge of some of the bits of the private key. Typically, it is assumed that some of the most or least significant bits of the private exponent are known. While these attacks may seem, at first, quite contrived and unrealistic, they can be quite relevant in certain practical settings in which side-channel attacks can be used to extract exactly this information. For more information about side-channel attacks, see [122, 96, 91, 1].

In this section we assume that the adversary has an approximation to the high order bits of the private exponent $d = N^{\beta}$. That is, for a given public key (e, N), the adversary knows $\widehat{d}$ such that

$$|d - \widehat{d}| < N^{\delta},$$

for some $0 \leq \delta \leq \beta$.

The two main lattice-based results for known partial key exposure attacks on RSA when either the public or private exponent is small are by Ernst, Jochemsz, May and de Weger [46]. We extend their results here to obtain new attacks on multi-prime RSA. The first attack uses the key equation with the approximation $\widehat{d}$ to compute d. We have the following new attack.

Attack 5.4. *For every $\epsilon > 0$ and integer $r \geq 2$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public key and let $(d, p_1, \ldots, p_r)$ be its corresponding private key, where $e = N^\alpha$ and $d = N^\beta$. Given the public key and $\widehat{d}$ satisfying $|d - \widehat{d}| \leq N^\delta$, if*

$$\delta \leq \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)(3\alpha r + 3\beta r - 2r - 1)} - \epsilon, \tag{5.5}$$

then the private exponent d can be recovered in time polynomial in $\log_2(N)$, provided that Assumption 2.13 holds.

Justification: Starting with the key equation $ed = 1 + k\phi(N)$, we replace d with $\widehat{d} + d_0$ and $\phi(N)$ with $N - \Lambda$, to obtain

$$e(\widehat{d} + d_0) = 1 + k(N - \Lambda).$$

Here d_0, k and Λ are the only unknowns. This suggests that we look for small integer solutions of the polynomial

$$f(x, y, z) = ex - Ny + yz + (e\widehat{d} - 1) \in \mathbb{Z}[x, y, z], \tag{5.6}$$

since $(x_0, y_0, z_0) = (d_0, k, \Lambda)$ is a root of $f(x, y, z)$ over the integers. Since

$$\begin{aligned} |d_0| &= |d - \widehat{d}| < N^\delta \\ |k| &= \left|\frac{ed - 1}{\phi(N)}\right| < 2N^{\alpha+\beta-1} \\ |\Lambda| &= |N - \phi(N)| < (2r - 1)N^{1-1/r}, \end{aligned} \tag{5.7}$$

we define the bounds

$$\begin{aligned} X &= N^\delta \\ Y &= 2N^{\alpha+\beta-1} \\ Z &= (2r - 1)N^{1-1/r}, \end{aligned} \tag{5.8}$$

so that $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$ and

$$\begin{aligned} W &= \|f(xX, yY, zX)\|_\infty \\ &= \max(eX, NY, YZ, e\widehat{d} - 1) \\ &= NY \\ &= 2N^{\alpha+\beta}. \end{aligned} \tag{5.9}$$

From Theorem 2.12, we then know that for sufficiently large N we can recover (x_0, y_0, z_0) provided that

$$X^{1+3\tau} Y^{2+3\tau} Z^{1+3\tau+3\tau^2} \leq W^{1+3\tau}, \tag{5.10}$$

and that Assumption 2.13 holds. Substituting the values for X, Y, Z and W, this inequality reduces to

$$(r-1)\,\tau^2 + (\delta\, r - 1)\,\tau + \frac{1}{3}\,(\delta\, r + \alpha\, r + \beta\, r - r - 1) \leq 0, \tag{5.11}$$

where we have ignored all factors that are independent of N. We can minimize the left-hand side of this inequality for any choice of α, δ and $r \geq 2$ by letting $\tau = -\frac{1}{2}(\delta r - 1)/(r-1)$. Using this value for τ and solving for δ, we find a sufficient condition for inequality 5.10 to hold is given by

$$\delta \leq \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)(3\alpha r + 3\beta r - 2r - 1)} - \epsilon, \tag{5.12}$$

where we have added the ϵ term to correct for all lower order terms neglected in the methods implicit in Theorem 2.12. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

The second attack uses the partial knowledge of the private exponent to compute an approximation of the constant k in the key equation. In some instances (*i.e.*, for particular sizes of public and private exponents), using the approximations of both d and k improve the result of the previous attack. We have the following new attack.

Attack 5.5. *For every $\epsilon > 0$ and integer $r \geq 2$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public key and let $(d, p_1, \ldots, r_r)$ be its corresponding private key, where $e = N^\alpha$ and $d = N^\beta$. Given the public key and $\widehat{d}$ satisfying $|d - \widehat{d}| \leq N^\delta$, if*

1. *$\alpha > 1 - \delta$, $\delta \leq \beta - 1/r$ and*

$$\delta \leq \frac{3r^2 + 6\alpha r + 3 - r^2\alpha^2 - 6r - 2\alpha r^2}{4\alpha r^2} - \epsilon, \ \textit{or} \tag{5.13}$$

2. *$\alpha > 1 + 1/r - \beta$, $\delta \geq \beta - 1/r$ and*

$$\delta \leq \frac{\alpha + \beta - 1}{3} + \frac{2}{3r} - \frac{2}{3r}\sqrt{(\alpha r + \beta r - r - 1)(\alpha r + \beta r + 2r - 4)} - \epsilon, \tag{5.14}$$

then the private exponent d can be recovered in time polynomial in $\log_2(N)$, *provided that Assumption 2.13 holds.*

Justification: First we use N, e and $\widehat{d}$ to compute an approximation $\widehat{k}$ of the constant k in the key equation $ed = 1 + k\phi(N)$. Letting

$$\widehat{k} = \left\lfloor \frac{e\widehat{d} - 1}{N} \right\rfloor, \tag{5.15}$$

notice that

$$\begin{aligned}
|k - \widehat{k}| &= \left| \frac{ed-1}{\phi(N)} - \frac{e\widehat{d}-1}{N} + \nu \right| \\
&= \left| \frac{(ed-1)N - (e\widehat{d}-1)\phi(N)}{\phi(N)N} + \nu \right| \\
&= \left| \frac{(ed-1)N - (e\widehat{d}-1)(N-\Lambda)}{\phi(N)N} + \nu \right| \\
&\le \left| \frac{e(d-\widehat{d})}{\phi(N)} \right| + \left| \frac{(e\widehat{d}-1)\Lambda}{\phi(N)N} \right| + 1 \\
&\le \frac{e}{\phi(N)} \left| (d - \widehat{d}) \right| + \frac{e}{\phi(N)} \frac{\widehat{d}\Lambda}{N} + 1 \\
&= \frac{e}{\phi(N)} \left(\left| (d-\widehat{d}) \right| + \frac{\widehat{d}\Lambda}{N} \right) + 1 \\
&\le \frac{e}{\phi(N)} \left(N^{\delta} + (2r-1)\widehat{d}N^{-1/r} \right) + 1,
\end{aligned} \tag{5.16}$$

where $|\nu| < 1$ is the discrepancy introduced by the floor operation in the $\widehat{k}$. Letting $\gamma = \max(\alpha + \delta - 1, \alpha + \beta - 1 - \frac{1}{r})$, we then have that

$$|k \pm 1 - \widehat{k}| < 2rN^{\gamma}. \tag{5.17}$$

Using this approximation for k (or perhaps $k \pm 1$) we can then write the key equation as

$$e(\widehat{d} + d_0) = 1 + (\widehat{k} + k_0)(N - \Lambda), \tag{5.18}$$

where $d_0 = d - \widehat{d}$, $k_0 = k - \widehat{k}$ and Λ are the only unknowns. This suggests that we look for small integer solutions of the polynomial

$$g(x,y,z) = ex - yN + \widehat{k}z + yz + e\widehat{d} - 1 - \widehat{k}N \in \mathbb{Z}[x,y,x], \tag{5.19}$$

since $(x_0, y_0, z_0) = (d_0, k_0, \Lambda)$ is a root of $g(x, y, z)$ over the integers. Since

$$\begin{aligned} |d_0| &= |d - \widehat{d}| < N^{\delta} \\ |k_0| &= |k - \widehat{k}| < 2rN^{\gamma} \\ |\Lambda| &= |N - \phi(N)| < (2r-1)N^{1-1/r}, \end{aligned} \tag{5.20}$$

we define the bounds

$$\begin{aligned} X &= N^{\delta} \\ Y &= 2rN^{\gamma} \\ Z &= (2r-1)N^{1-1/r}, \end{aligned} \tag{5.21}$$

so that $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$ and

$$\begin{aligned} W &= \|g(xX, yY, zX)\|_{\infty} \\ &= \max(eX, NY, \widehat{k}Z, YZ, e\widehat{d} - 1 - \widehat{k}N) \\ &= NY \\ &= 2rN^{\gamma+1}. \end{aligned} \tag{5.22}$$

From Theorem 2.12, we then know that, for sufficiently large N, we can recover (x_0, y_0, z_0) provided that

$$X^{2+3\tau}Y^{3+6\tau+3\tau^2}Z^{3+3\tau} \leq W^{2+3\tau}, \tag{5.23}$$

and that Assumption 2.13 holds. Substituting the values for X, Y, Z and W, this inequality reduces to

$$(\gamma r)\tau^2 + (\gamma r + \delta r - 1)\tau + \tfrac{1}{3}\left(2\delta r + r - 3 + \gamma r\right) < 0, \tag{5.24}$$

where we have ignored all factors that are independent of N. When $\gamma > 0$, we can minimize the left-hand side of this inequality for any choice of α, δ and $r \geq 2$ by letting $\tau = -\frac{1}{2}(\gamma r + \delta r - 1)/(\gamma r)$. Using this value for τ and solving for δ, we find that a sufficient condition for inequality 5.23 to hold is given by

$$\delta \leq \frac{\gamma}{3} + \frac{1}{r} - \frac{2}{3r}\sqrt{\gamma^2 r^2 - 3\gamma r + 3\gamma r^2} - \epsilon, \tag{5.25}$$

where the ϵ term is added to correct for all the lower order terms neglected in the methods implicit in Theorem 2.12.

We now consider the two cases for γ separately. That is, we consider $\gamma = \alpha + \delta - 1$ and $\gamma = \alpha + \beta - 1 - 1/r$. The cases can be distinguished by the values of δ and $\beta - 1/r$ (the cases overlap when $\delta = \beta - 1/r$). For each case, we must ensure that $\gamma > 0$ so that the optimization of τ given above holds.

1. When $\delta \leq \beta - 1/r$ and hence $\gamma = \alpha + \delta - 1$, inequality (5.25) becomes
$$\delta \leq \frac{3r^2 + 6\alpha r + 3 - r^2\alpha^2 - 6r - 2\alpha r^2}{4\alpha r^2} - \epsilon. \tag{5.26}$$
For this case, in order to ensure $\gamma > 0$, we must have $\alpha > 1 - \delta$.

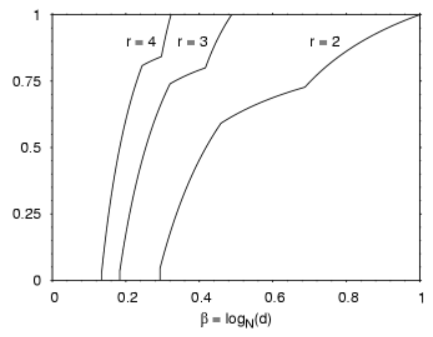
2. When $\delta \geq \beta - 1/r$ and hence $\gamma = \alpha + \beta - 1 - 1/r$, inequality (5.25) becomes
$$\delta \leq \frac{\alpha + \beta - 1}{3} + \frac{2}{3r} - \frac{2}{3r}\sqrt{(\alpha r + \beta r - r - 1)(\alpha r + \beta r + 2r - 4)} - \epsilon. \tag{5.27}$$
In order to ensure that $\gamma = \alpha + \beta - 1 - 1/r > 0$, we must have that $\alpha > 1 + 1/r - \beta$ in this case.

Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

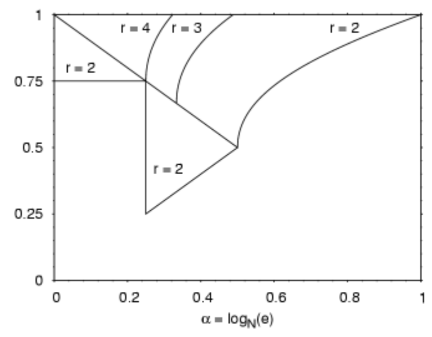
While it is possible to have public and private exponents both significantly smaller than $\phi(N)$ (see Sun and Yang [115] for example), it is much more common that only one of the exponents is small. We consider the strongest attacks for these typical instances (when only one exponent is small) below. The attacks are summarized in Figure 5.1 for the first few values of r. Notice that the effectiveness of each attack decreases with each additional prime in the modulus.

5.5.1 Small Private Exponent

In typical RSA, when the private exponent is chosen to be small, the public exponent is roughly the same order of magnitude as the modulus N. Thus, in this scenario we can approximate the public exponent by $e \approx N$ (*i.e.*, $\alpha \approx 1$). Using this approximation in Attack 5.4, we have that for sufficiently large N the private exponent can be computed if
$$\delta \leq \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)(r+3\beta r - 1)} - \epsilon, \tag{5.28}$$
Similarly, in Attack 5.5, we have that, for sufficiently large N, the private exponent can be computed if
$$\delta \leq \frac{3}{4r^2} - \epsilon, \tag{5.29}$$



(a) Small private exponent ($\alpha \approx 1$).



(b) Small public exponent ($\beta \approx 1$).

Figure 5.1: Partial key exposure attacks with known MSB and small private exponent. Plot (a) shows fraction $(\beta - \delta)/\beta$ of MSBs of d required. Plot (b) shows fraction $(1 - \delta)$ of MSBs of d required.

when $\beta \leq (3 + 4r)/(4r^2)$, or if

$$\delta \leq \frac{\beta}{3} + \frac{2}{3r} - \frac{2}{3}\sqrt{(r\beta - 1)(r\beta - 4 + 3r)} - \epsilon, \tag{5.30}$$

when $\beta \geq (3 + 4r)/(4r^2)$. Since $\alpha \approx 1$, we have $\gamma > 0$ for both cases. Letting $r = 2$ in the preceding three equations recovers the original results for RSA by Ernst *et al.* [46, Theorem 1]. In particular, when $r = 2$, the private exponent can be recovered if:

1. $\delta \leq \frac{5}{6} - \frac{1}{3}\sqrt{1 + 6\beta} - \epsilon$, or

2. $\beta \leq \frac{11}{16}$ and $\delta \leq \frac{3}{16} - \epsilon$, or

3. $\beta \geq \frac{11}{16}$ and $\delta \leq \frac{1}{3} + \frac{1}{3}\beta - \frac{1}{3}\sqrt{4\beta^2 + 2\beta - 2} - \epsilon$.

In addition to Attacks 5.4 and 5.5, both of the small private exponent attacks (Wiener [123] and Boneh and Durfee [13]) can be considered as partial key exposure attacks with known MSB and small private exponent. In this case, zero bits of the private key are required to successfully mount the attack (*i.e.*, $\delta = 0$).

We illustrate the partial key exposure attacks with small private exponent for the first few values of r when $\alpha \approx 1$ in Figure 5.1(a). For each value of β, the fraction of bits of the private exponent required for the attack to succeed is given (*i.e.*, $(\beta - \delta)/\beta$). For each value of r, Attack 5.3 is strongest for small β, Attack 5.4 is strongest for intermediate values of β and Attack 5.5 is strongest for larger values of β.

5.5.2 Small Public Exponent

When the public exponent is chosen to be small, the private exponent is, with high probability, roughly the same order of magnitude as the modulus N. Thus, in this scenario we can approximate the private exponent by $d \approx N$ (*i.e.*, $\beta \approx 1$). Using this approximation in Attack 5.5, we have that for sufficiently large N the private exponent can be computed when $e > N^{1/r}$ ($\alpha > 1/r$ to ensure $\gamma > 0$) and

$$\delta \le \frac{\alpha}{3} + \frac{2}{3r} - \frac{2}{3r}\sqrt{(\alpha r - 1)(\alpha r + 3r - 4)} - \epsilon. \tag{5.31}$$

Letting $r = 2$ recovers the original RSA result obtained by Ernst *et al.* [46, Theorem 2]. Namely, $\delta \le \frac{1}{3} + \frac{1}{3}\alpha - \frac{1}{3}\sqrt{4\alpha^2 + 2\alpha - 2} - \epsilon$.

When the public exponent is smaller than $N^{1/r}$, an attack that is not based on lattice basis reduction can be used. The original attack on RSA is by Boneh, Durfee and Frankel [16] and was extended to the multi-prime case by Hinek, Low and Teske [64]. The main result of the attack follows.

Attack 5.6. *For every integer $r \ge 2$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public exponent and let $(d, p_1, \ldots, r_r)$ be its corresponding private key, where $e \le N^{1/r}$ and $k = (ed - 1)/\phi(N) > \eta e$ for some $0 < \eta < 1$. Given (e, N) and an approximation $\widehat{d}$ of d satisfying $|d - \widehat{d}| < e$, the private exponent can be computed in time polynomial in r, $\log_2(N)$ and $1/\eta$.*

For a justification of Attack 5.6, see [64, §5.2].

In Figure 5.1(b), we illustrate the partial key exposure attacks with small public exponent for the first few values of r when $\beta \approx 1$. For each value of α, the fraction of bits of the private exponent required for the attack to succeed is given (*i.e.*, $1 - \delta$). In addition to Attacks 5.5 and 5.6, we have included two additional partial key exposure attacks by Boneh, Durfee and Frankel [16] that only apply to RSA. We include these attacks to give a complete picture of the best partial key exposure attacks on multi-prime RSA (including RSA). The results of these attacks are given below. For arguments that these attacks cannot be extended to the multi-prime case, see Hinek, Low and Teske [64, §5.2].

The first attack applies to instances of RSA with public exponent in the range $1/4 \leq \alpha \leq 1/2$. It is the strongest known partial key exposure attack with known MSB against RSA, occurring when $e \approx N^{1/4}$. The main result of the attack follows (see [16, Theorem 4.3] for more details, including a proof of the result).

Attack 5.7. *For every $\epsilon > 0$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an RSA modulus with balanced primes, let (e, N) be a valid public key with prime public exponent and let $(d, p_1, \ldots, r_r)$ be its corresponding private key, where $e = N^{\alpha}$. Given (e, N) where $\frac{1}{4} \leq \alpha \leq \frac{1}{2}$ and $\widehat{d}$ satisfying $|d - \widehat{d}| < N^{\alpha-\epsilon}$, the private exponent can be computed in time polynomial in $\log_2(N)$ and $1/\epsilon$.*

This attack requires that the public exponent be prime. If e is not prime, the attack can still be mounted provided that the factorization of e is known. If e has t distinct prime factors then the runtime of the modified attack is polynomial in $\log_2(N)$, $1/\epsilon$ and 2^t. For details, see [16, Corollary 4.4]. For a 1024-bit modulus and randomly chosen $e < N^{1/2}$, it should be possible to completely factor e using the ECM.

The second attack, observed by Blömer and May [8, §1], is not explicitly mentioned by Boneh, Durfee and Frankel but follows from two of their results [16, Theorems 3.3 and 4.1]. The main result of the attack, which applies to instances of RSA with public exponent smaller than $N^{1/4}$, is as follows.

Attack 5.8. *For every $\epsilon > 0$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an RSA modulus with balanced primes, let (e, N) be a valid public key and let $(d, p_1, \ldots, r_r)$ be its corresponding private key. Given (e, N) where $0 \leq \alpha \leq \frac{1}{2}$ and $\widehat{d}$ satisfying $|d - \widehat{d}| < N^{1/4-\epsilon}$, the private exponent can be computed in time polynomial in $\log_2(N)$ and $1/\epsilon$.*

5.6 Partial Key Exposure Attacks: Known LSB

In this section we consider attacks in which some of the least significant bits of the private exponent are known. In particular, we assume that the adversary knows $\widehat{d}$ for some (known) M such that $d \equiv \widehat{d} \pmod{M}$. Thus, we can write the private exponent as $d = d_0 M + \widehat{d}$ where d_0 is the only unknown. We will let $M = N^{\beta-\delta}$ so that the size of the unknown part of the private exponent satisfies $|d_0| < N^{\delta}$.

The most general partial key exposure attack on RSA with known LSB is due to Ernst *et al.* [46]. We extend their result to multi-prime RSA here to obtain the following new attack.

Attack 5.9. *For every $\epsilon > 0$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public exponent and let $(d, p_1, \ldots, r_r)$ be its corresponding private key, where $e = N^{\alpha}$ and $d = N^{\beta}$. Given (e, N), $\widehat{d}$ and M where $d \equiv \widehat{d} \bmod M$ and $M = N^{\beta-\delta}$, if*

$$\delta \leq \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)(3r\beta + 3r\alpha - 2r - 1)} - \epsilon, \tag{5.32}$$

then the private exponent can be recovered in time polynomial in $\log_2(N)$, provided that Assumption 2.13 holds.

Justification: Beginning with the key equation, $ed = 1 + k\phi(N)$, we substitute $d = d_0 M + \widehat{d}$ and $\phi(N) = N - \Lambda$ to obtain

$$eMd_0 + e\widehat{d} = 1 + kN - k\Lambda,$$

where d_0, k and Λ are the only unknowns. This suggests that we look for small integer solutions of the polynomial

$$f(x, y, z) = (eM)x - Ny + yz + e\widehat{d} - 1 \in \mathbb{Z}[x, y, z], \tag{5.33}$$

since $(x_0, y_0, z_0) = (d_0, k, \Lambda)$ is a root of $f(x, y, z)$ over the integers.

Since

$$\begin{aligned} |d_0| &= \left|\frac{d - \widehat{d}}{M}\right| < N^{\delta} \\ |k| &= \left|\frac{ed - 1}{\phi(N)}\right| < 2N^{\alpha+\beta-1} \\ |\Lambda| &= |N - \phi(N)| < (2r-1)N^{1-1/r}, \end{aligned} \tag{5.34}$$

we define the bounds

$$\begin{aligned} X &= N^{\delta} \\ Y &= 2N^{\alpha+\beta-1} \\ Z &= (2r-1)N^{1-1/r}, \end{aligned} \tag{5.35}$$

so that $|x_0| < X$, $y_0| < Y$, $|z_0| < Z$ and

$$\begin{aligned} W &= \|f(xX, yY, zX)\|_\infty \\ &= \max(eMX, NY, YZ, e\widehat{d} - 1) \\ &= NY \\ &= 2N^{\alpha+\beta}. \end{aligned} \tag{5.36}$$

Notice that the polynomial $f(x, y, z)$ has the same set of monomials as the polynomial in the proof of Attack 5.4 and that the bounds X, Y, Z and $W = \max(eMX, NY, YZ, ed_0 - 1) = NY = 2N^{\alpha+\beta}$, are also the same. From the proof of Attack 5.4, we can then conclude that the root (x_0, y_0, z_0) can be computed for sufficiently large N when

$$\delta \le \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)(3\alpha r + 3\beta r - 2r - 1)} - \epsilon,$$

provided Assumption 2.13 holds. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

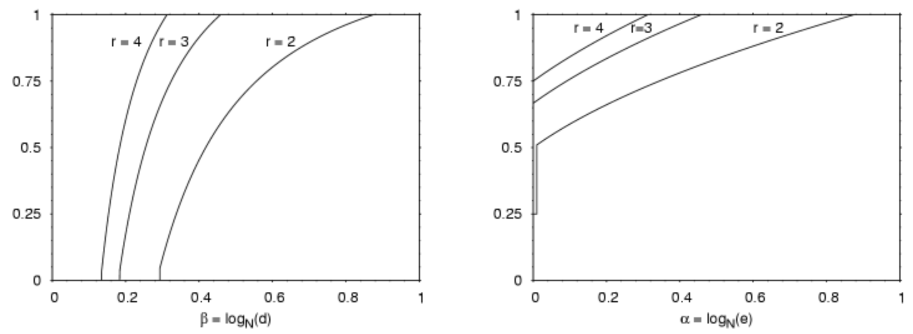
As with the known MSB attacks, we consider the best partial key exposure attacks with known LSB on multi-prime RSA when only one of the private and public exponent is small. In Figure 5.2, we summarize these attacks for the first few values of r. As with the attacks with known MSB, we see that the effectiveness of each attack decreases with each additional prime in the modulus.

5.6.1 Small Private Exponent

When the private exponent is small and the keys have been chosen in the standard way then the public exponent can be approximated, with high probability, by $e \approx N$ (*i.e.*, $\alpha \approx 1$). Using this approximation in Attack 5.9, we see that, for sufficiently large N, the private exponent can be computed if

$$\delta \le \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)\,(3\beta r + r - 1)} - \epsilon. \tag{5.37}$$

Letting $r = 2$ recovers the original result $\delta \le \frac{5}{6} - \frac{1}{3}\sqrt{1+6\beta}$, obtained by Ernst *et al.* [46, Theorem 3]. Combining this attack with Boneh and Durfee's small private exponent attack gives the strongest attacks with known least significant bits. Again, we can consider Boneh and Durfee's small private exponent attack as a partial key exposure attack in which zero of the LSB of the private exponent are needed ($\delta = 0$). We illustrate these attacks for the first few values of r in Figure 5.2(a).



(a) Small private exponent ($\alpha \approx 1$). (b) Small public exponent ($\beta \approx 1$).

Figure 5.2: Partial key exposure attacks with known LSB. Plot (a) shows fraction $(\beta - \delta)/\beta$ of LSBs of d required. Plot (b) shows fraction $(1 - \delta)$ of LSBs of d required.

5.6.2 Small Public Exponent

When the public exponent is small and the keys have been chosen in the standard way, then with high probability, the private exponent can be approximated by $d \approx N$ (*i.e.*, $\beta \approx 1$). Using this approximation in Attack 5.9, we see that, for sufficiently large N, the private exponent can be computed if

$$\delta \leq \frac{2}{3} + \frac{1}{3r} - \frac{2}{3r}\sqrt{(r-1)(3\alpha r + r - 1)} - \epsilon. \tag{5.38}$$

Letting $r = 2$ recovers the original result $\delta \leq \frac{5}{6} - \frac{1}{3}\sqrt{1 + 6\alpha}$, obtained by Blömer and May [8, Theorem 11], and again by Ernst *et al.* [46, §4.3]. In Figure 5.2(b), we illustrate the best known partial key exposure attacks with known LSB for instances of multi-prime RSA with small public exponent and large private exponent ($d \approx 1$) for $r \in \{2, 3, 4\}$. In addition to Attack 5.9, we have included Boneh, Durfee and Frankel's small public exponent partial key exposure attack on RSA [16, Theorem 3.1], as given below.

Attack 5.10. *Let N be an n-bit RSA modulus with balanced primes such that $N = 3 \bmod 4$. Let (e, N) be a valid public key and (d, p, q) be its corresponding private key defined modulo $\phi(N)$. If the public exponent satisfies $e < 2^{(n/4)-3}$ then there is an algorithm that given (e, N) and the $n/4$ least significant bits of d computes all of d in time polynomial in n and linear in $e \log_2(e)$.*

When the public exponent is very small, this attack on RSA gives the strongest partial key exposure attack with known LSB. It was argued by Hinek, Low and Teske [64, §5.1], that this attack most likely cannot be extended to multi-prime RSA.

5.7 Partial Key Exposure Attacks: Known Partial Factorization

In this section, we extend the notion of partial key exposure attacks to include attacks in which any part of the private key, which is meant to be secret, is known. We allow the adversary to know some of the least or most significant bits of one or more of the primes in the modulus in addition to zero or more of the most or least significant bits of the private exponent. In particular, for balanced r-prime RSA we assume that the adversary knows v of the r primes in N, for some $1 \leq v \leq r-2$, in addition to zero or more of the most or least significant bits of the remaining unknown primes and the private exponent.

We begin this section with some factoring results and then present two attacks on multi-prime RSA that recover the private exponent. The factoring attacks are simple applications of the lattice-based factoring results of Coppersmith [32] and Boneh, Durfee and Howgrave-Graham [17]. The attacks apply to any composite integer having the same form as a balanced r-prime RSA modulus. Of the attacks that recover the private exponent, the first attack recovers sufficiently small private exponents using only the public key and the partial factorization. This attack can be seen as an extension of Wiener's continued fraction attack. The second attack uses lattice basis reduction techniques based on Coppersmith's methods to improve the results of the first. This attack, like all of the other lattice-based attacks on multi-prime RSA, rely on Assumption 2.13 and so it is only heuristic.

5.7.1 Factoring r-prime RSA Moduli

Boneh, Durfee and Howgrave-Graham presented a lattice-based factoring method for composite integers of the form $N = p^r q$ in [17, Theorem 3.1]. Based on their method, we present a new method to factor composite integers with the same form as multi-prime RSA moduli with balanced primes. The main result is as follows.

Theorem 5.11. *There exists an N_0 such that for every $N > N_0$ the following holds: Let N be a balanced r-prime RSA modulus. For any $s \in [2, r]$, given $r-s$ of the primes in the factorization of N, $(s-1)/s$ of the most or* *least significant bits of one of the unknown primes,* $(s-2)/(s-1)$ *of the most or least significant bits of another one of the unknown primes,* $\ldots$*,* $2/3$ *of the most or least significant bits of another one of the unknown primes and* $1/2$ *of the most or least significant bits of one of the two remaining unknown primes, then* N *can be factored in time linear in* r *and polynomial* $\log_2(N)$*.*

Proof: First we relabel the primes so that $P = p_1 p_2 \cdots p_s$ is the product of the s unknown primes. And, without loss of generality, we assume that we know the $(s-i)/(s-i+1)$ most or least significant bits of p_i for $i = 1, \ldots, s-1$. For each unknown prime p_i we compute $\widehat{p_i}$ (and possibly ℓ) from the known most or least significant bits of p_i so that

$$p_i = \begin{cases} \widehat{p_i} + p_{i,0} & \text{for known MSB} \\ p_{i,0}2^{\ell} + \widehat{p_i} & \text{for known LSB,} \end{cases}$$

where $p_{i,0}$ is unknown and satisfies

$$|p_{i,0}| < {p_i}^{1-(s-i)/(s-i+1)} < ((2N)^{1/r})^{1-(s-i)/(s-i+1)}. \tag{5.39}$$

We now compute the unknown primes one at a time; always computing the prime with the most known information first and then redefining P accordingly. Thus, we first compute p_1, then p_2, $\ldots$, until we compute p_{s-1} which then yields p_s. In particular, for $i = 1, \ldots, s-1$, let $p_0 = 1$ and define

$$\begin{aligned} P_i &= P \times \prod_{j=1}^{i-1} {p_j}^{-1} & (5.40) \\ &= \frac{p_0 p_1 \cdots p_s}{p_0 \cdots p_{i-1}} & (5.41) \\ &= p_i p_{i+1} \cdots p_s. & (5.42) \end{aligned}$$

For each i, we consider the polynomial

$$f_i(x) = x + c, \tag{5.43}$$

where c is a constant given by

$$c = \begin{cases} \widehat{p_i} & \text{if the MSB of } p_i \text{ are known,} \\ \widehat{p_i}(2^{-\ell} \bmod P_i) & \text{if the LSB of } p_i \text{ are known.} \end{cases} \tag{5.44}$$

Since P_i is the product of odd primes, the inverse of 2^{ℓ} is guaranteed to exist. We consider this polynomial since $x_0 = p_{i,0}$ is a root of $f_i(x)$ modulo

p_i. Since P_i is a multiple of p_i, we can use the results of Theorem 2.10, to find all small solutions of $f_i(x)$ modulo p_i, even though we do not know p_i. Finding all such small solutions will yield $p_{i,0}$, which allows us to simply compute the entire prime p_i.

Now, for each i, notice that the size of the prime p_i is roughly $P_i^{\frac{1}{s-i+1}}$. From Theorem 2.10, since $f_i(x)$ is linear, we know that we can compute all x_0 such that $f_i(x_0) \equiv 0 \bmod p_i$ provided that

$$\begin{aligned} |x_0| &< P_i^{(\frac{1}{s-i+1})^2} \\ &= (N^{\frac{s-i+1}{r}})^{(\frac{1}{s-i+1})^2} \\ &= (N^{\frac{1}{r}})^{\frac{1}{s-i+1}} \\ &= p_i^{\frac{1}{s-i+1}}. \end{aligned} \tag{5.45}$$

Therefore, we need to know the

$$1 - \frac{1}{s-i+1} = \frac{s-i}{s-i+1}, \tag{5.46}$$

fraction of bits of p_i in order to be able to compute it using this method. Since all computations can be done in time polynomial in $\log_2(N)$ and there are fewer than r functions $f_i(x)$ that need to be considered, the result follows. ❏

Notice that Theorem 5.11 implies an upper bound on the minimum fraction of bits needed to factor a balanced r-prime RSA modulus. In particular, sufficiently large balanced r-prime RSA moduli N can be factored given $r-s$ of the primes in N and an additional fraction of bits, relative to the size of N, given by[5]

$$\frac{1}{r} \times \left(\frac{s-1}{s} + \frac{s-2}{s-1} + \cdots + \frac{1}{2} \right). \tag{5.47}$$

The actual number of bits is then roughly $\log_2 N$ times this bound. We illustrate this bound for the first few values of r and each possible choice of s in the Table 5.2. The table shows the fraction of bits which is the sum of the contributions of the known primes and the partially known primes. The fraction is relative to the size of the modulus. Of course, it should be pointed out that these values are only an estimate and are not a proven sufficient bound. The bound obtained here is in the limiting case of large

[5]This sum can also be written as $\frac{1}{r}(s - \Psi(s+1) - \gamma)$, where $\Psi(\cdot)$ is the digamma function and γ is Euler's constant.

moduli and is optimistic for fixed size moduli. However, the asymptotic bounds obtained in most applications of Coppersmith's methods seem to be good estimates for the actual bounds obtained in practice.

Table 5.2: Fraction of bits required to factor N. Fractions are sum of the contributions of known primes and partially known primes rounded to three decimal places.

Number of known primes $(r-s)$	0	1	2	3
Fraction of bits for $r=2$	0.250			
Fraction of bits for $r=3$	0.389	0.500		
Fraction of bits for $r=4$	0.479	0.542	0.625	
Fraction of bits for $r=5$	0.543	0.583	0.633	0.700

Setting $r = 2$ and $s = r$ in (5.47) recovers the well known result of Coppersmith [32]; that an RSA modulus can be factored with knowledge of 1/4 of the bits of the factorization of N. In particular, 1/2 of the most or least significant bits of one of the primes is needed. The same result can also be obtained using Boneh, Durfee and Howgrave-Graham's factoring method. This result for RSA can be used to attack certain proposals that fix the most significant bits of the modulus in order to reduce the memory requirements for the public keys (*e.g.*, see [119] and [71]).

5.7.2 Small Private Exponent Attacks

Here we present two small private exponent partial key exposure attacks on multi-prime RSA. We presented the first attack in [61, Lemma 1]. The main result of the attack and a proof are given below.

Theorem 5.12. *Let N be a balanced r-prime RSA modulus, let (e, N) be a valid r-prime public key and let $(d, p_1, \dots, p_r)$ be its corresponding private key where $d = N^\delta$. Given the public key and any $1 \le v \le r-2$ of the primes in the factorization of N, if*

$$\delta < \frac{v}{r} - \frac{v+1}{\log_2 N},$$

then d can be computed in time polynomial in $\log_2(N)$.

Proof: Let P be the product of the v known primes and define $Q = N/P$. Since the primes in N are pairwise distinct we can write Euler's totient function of N as $\phi(N) = \phi(Q)\phi(P)$. This allows the key equation, $ed = 1 + k\phi(N)$, to be written as

$$ed = 1 + k\phi(Q)\phi(P), \tag{5.48}$$

where e and $\phi(P)$ are known quantities. Now, reducing this equation modulo $\phi(P)$ yields

$$d \equiv e^{-1} \bmod \phi(P), \tag{5.49}$$

and so whenever $d < \phi(P)$ it follows that $d = e^{-1} \bmod \phi(P)$. Since

$$\phi(P) > \tfrac{1}{2}P > \tfrac{1}{2}\times\tfrac{1}{2^v}N^{v/r}, \tag{5.50}$$

a sufficient condition that $d < \phi(P)$ is given by $d = N^\delta < \frac{1}{2}\times\frac{1}{2^v}N^{v/r}$, or simply

$$\delta < \frac{v}{r} - \frac{v+1}{\log_2 N}. \tag{5.51}$$

The result follows since computing the inverse of e modulo $\phi(P)$ where $e, \phi(P) < N$ can be done in time polynomial in $\log_2 N$. ❑

The second attack that we consider is new and has yet to be published elsewhere. It uses zero or more of the most significant bits of d in addition to the known factors of N to fully recover d. This attack is a generalization of Attack 5.4 and is an improvement of the attack found in [61, Theorem 2]. The details of the attack are as follows.

Attack 5.13. *For every $\epsilon > 0$ there exists an N_0 such that for every $N > N_0$ the following holds: Let N be a balanced r-prime RSA modulus, let (e, N) be a valid public exponent and let $(d, p_1, \ldots, r_r)$ be its corresponding private key, where $e = N^\alpha$ and $d = N^\beta$. Given (e, N), $1 \leq v \leq r-2$ of the primes in the factorization of N, and $\widehat{d}$ such that $|d - \widehat{d}| \leq N^\delta$ for some $0 \leq \delta \leq \beta$, if*

$$\delta \leq \frac{2}{3} + \frac{v+1}{3r} - \frac{2}{3r}\sqrt{(r-v-1)(3r\alpha + 3r\beta - 2r - v - 1)} - \epsilon, \tag{5.52}$$

then the private exponent can be recovered in time polynomial in $\log_2(N)$. *provided Assumption 2.13 holds.*

Justification: Let P be the product of the known v primes and define $Q = N/P$. As in the proof of the last attack, we write the key equation as $ed = 1 + k\phi(P)\phi(Q)$, since all the primes are distinct. Next, we replace the private key d with $d_2 + d_1\phi(P) + d_0$ where $d_0 = e^{-1} \bmod \phi(P)$ and $d_2 = \widehat{d} - \big(\widehat{d} \bmod \phi(P)\big)$. This ensures that $d \equiv d_0 \pmod{\phi(P)}$. We can also replace $\varphi(Q)$ with $Q - \Lambda_Q$, where

$$\Lambda_Q < (r-v)2^{r-v-1}N^{1-\frac{v}{r}-\frac{1}{r}},$$

since the primes are balanced. Thus, we have

$$e(d_2 + d_1\phi(P) + d_0) = 1 + k\phi(P)(Q - \Lambda_Q),$$

where d_1, k and Λ_Q are the only unknowns. This suggests that we look for small integer roots of the polynomial

$$f(x,y,z) = ex - \phi(P)Q\,y + \phi(P)\,yz + d_2\phi(P) + d_0 - 1 \in \mathbb{Z}[x,y,z], \quad (5.53)$$

since $(x_0, y_0, z_0) = (d_1, k, \Lambda_Q)$ is a root of $f(x,y,z)$ over the integers. Since

$$\begin{aligned} |d_1| &= \left|\frac{d - d_2 - d_0}{\phi(P)}\right| < 2^{v+1}N^{\delta - v/r} \\ |k| &= \left|\frac{ed-1}{\phi(N)}\right| < 2N^{\alpha+\beta-1} \\ |\Lambda_Q| &= |Q - \phi(Q)| < (r-v)2^{r-v-1}N^{1-\frac{v}{r}-\frac{1}{r}}, \end{aligned} \quad (5.54)$$

we define the bounds

$$\begin{aligned} X &= 2^{v+1}N^{\delta - v/r} \\ Y &= 2N^{\alpha+\beta-1} \\ Z &= (r-v)2^{r-v-1}N^{1-\frac{v}{r}-\frac{1}{r}}, \end{aligned} \quad (5.55)$$

so that $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$ and

$$\begin{aligned} W &= \|f(xX, yY, zX)\|_\infty \\ &= \max(e\phi(P)X, (ed_2 + ed_0 - 1), \phi(P)QY, \phi(P)YZ) \\ &= \phi(P)QY \\ &= 2N^{\alpha+\beta}. \end{aligned} \quad (5.56)$$

From Theorem 2.12, we then know that for sufficiently large N we can recover (x_0, y_0, z_0) provided that

$$X^{1+3\tau}Y^{2+3\tau}Z^{1+3\tau+3\tau^2} \leq W^{1+3\tau}, \quad (5.57)$$

and provided that Assumption 2.13 holds. Substituting the values for X, Y, Z and W, this inequality reduces to

$$(r-v-1)\,\tau^2 + (\delta r - 2v - 1)\,\tau + \frac{1}{3}\,(\delta r + r\beta - 2v + r\alpha - r - 1) \leq 0, \tag{5.58}$$

where we have ignored all factors that are independent of N. We can minimize the left-hand side of this inequality for any choice of α, δ, r and $v \leq r-2$ by letting $\tau = -\frac{1}{2}(\delta r - 2v - 1)/(r-v-1)$. Using this value for τ and solving for δ, we find that a sufficient condition for inequality 5.57 to hold is given by

$$\delta \leq \frac{2}{3} + \frac{v+1}{3r} - \frac{2}{3r}\sqrt{(r-v-1)\,(3r\alpha + 3r\beta - 2r - v - 1)} - \epsilon, \tag{5.59}$$

where the ϵ term is added to correct for all the lower order terms neglected. Since all the computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

5.7.3 Small Private Exponent

When a small private exponent is chosen we can expect that $\alpha \approx 1$ with high probability. Using this approximation in Attack 5.13, we see that for sufficiently large N, the private exponent can be recovered when

$$\delta \leq \frac{2}{3} + \frac{v+1}{3r} - \frac{2}{3r}\sqrt{(r-v-1)(3\beta r + r - v - 1)} - \epsilon. \tag{5.60}$$

We illustrate this bound for all valid values of v for the few values of r in Figure 5.3. Also included in the figure is Attack 5.12 which gives a superior bound to Attack 5.13 for small enough private exponents. Notice that for fixes v (number of known primes) that the attacks are less effective with each additional prime in the modulus.

5.7.4 Small Public Exponent

When a small public exponent is used, we can approximate the private exponent with $\beta \approx 1$ with high probability. Using this approximation in Attack 5.13, we see that for sufficiently large N, the private exponent can be recovered when

$$\delta \leq \frac{2}{3} + \frac{v+1}{3r} - \frac{2}{3r}\sqrt{(r-v-1)(3\alpha r + r - v - 1)} - \epsilon. \tag{5.61}$$

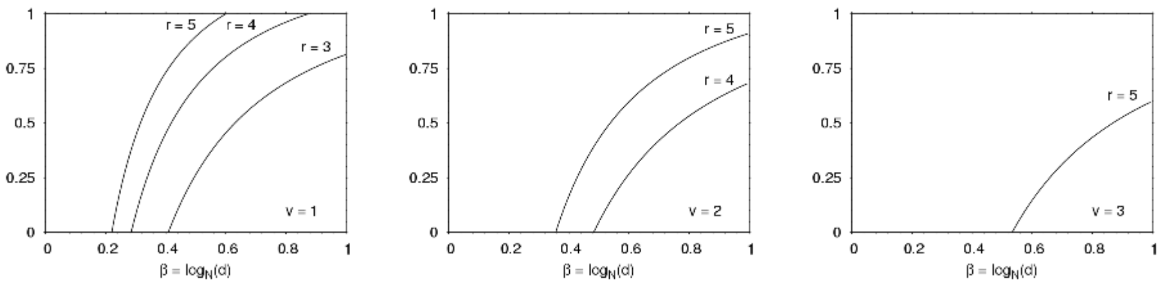


Figure 5.3: Fraction of MSB, $(\beta - \delta)/\beta$, of the private exponent needed for Attack 5.13 with small private exponents. The plots, from left to right, are for $v = 1, 2, 3$.

We illustrate this bound for all valid values of v for the first few values of r in Figure 5.4. Notice that for fixed values of v (number of known primes) the feasible region of the attack decreases with increasing number of primes in the modulus.

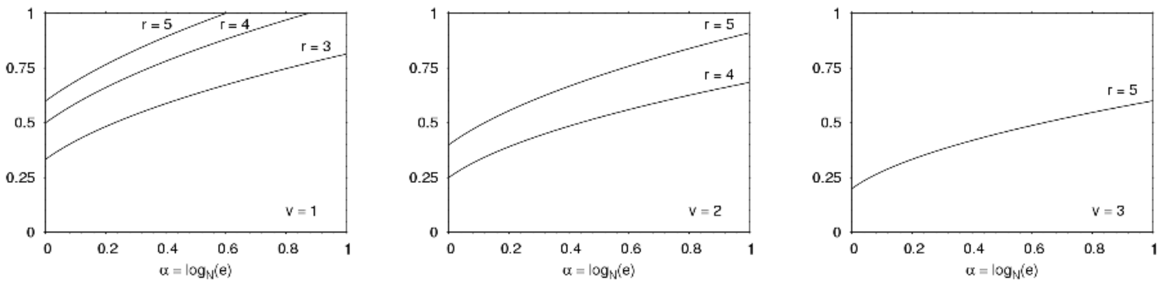


Figure 5.4: Fraction of MSB, $1 - \delta$, of the private exponent needed for Attack 5.13 with small public exponents. The plots, from left to right, are for $v = 1, 2, 3$.

Similar to Attack 5.5 with small public exponents, one can compute an approximation of k and try to find small solutions of a polynomial with monomials $\{x, y, z, yz\}$. In particular, one can compute

$$\widehat{k} = \left\lceil \frac{e(d_2\phi(P) + d_0) - 1}{\phi(P)Q} \right\rceil,$$

so that $k = \widehat{k} + k_0$ where $|k_0| \leq \max(\alpha + \delta - 1, \alpha + \beta - 1 - \frac{1}{r})$. It turns out that using this approximation for k does not lead to an improvement over the results of Attack 5.13.

It is interesting to notice that when $\beta \approx 1$ and the public exponent approaches 3 ($\alpha \approx 0$), the total number of bits required to mount the attack, including both the MSB of d and the known factors of N is $(1 - \frac{1}{r}) \log_2(N)$. This is the same number of bits of the factorization of the modulus that knowing all but one prime in the factorization would give.

5.8 Chinese Remainder Theorem Attack

There is currently only one known attack on multi-prime RSA with CRT decryption, other than simply factoring the modulus. In [12], Boneh gives the result of an attack on RSA with small CRT-exponents (with unknown origin). This attack uses Fast Fourier Transforms to evaluate a large degree polynomial that will reveal a prime factor of N. The attack was extended to multi-prime RSA by Hinek, Low and Teske [64, §4.3]. The main result is given in the following theorem.

Theorem 5.14. *Let $N = p_1 \cdots p_r$ be an r-prime RSA modulus with balanced primes, let (e, N) be a valid public key and let $(d, p_1, \ldots, p_r)$ be its corresponding private exponent key. For $i = 1, \ldots, r$, let $d_i = d \bmod (p_i - 1)$ be the CRT exponents and let d_{SL} be the second largest CRT exponent with bitlength m. If $d_i \not\equiv d_j \pmod{2^{m/2}}$ for all $i \neq j$ then N can be factored in time $O(r\sqrt{d_{SL}} \log^2 N)$.*

The size of the CRT-exponents, therefore, should be chosen large enough so that the complexity of Attack 5.14 matches the complexity of factoring the modulus using the methods in Section 5.3.

5.9 Experimental Results: Partial Key Exposure Attacks

Here we present some experimental results to illustrate the effectiveness of the attacks from Sections 5.5 and 5.6 in practice. All computations were done with Maple [78], except for the lattice basis reduction which was done with Shoup's NTL [106]. The experiments were carried out on either a Sun Fire V100 server with one UltraSPARC IIe processor with 2GB of memory running at 550 MHz, or on a Sun Fire V440 server with four UltraSPARC IIIi processors with 8 GB of memory each running at 1.062 GHz.

For each data point obtained, we used a binary search approach to find the smallest fraction of bits needed for a successful attack for each size of the private or public exponent considered. Each individual experiment (in the binary search) used a randomly chosen modulus with private or public exponent chosen to be a specific size. Also, unless otherwise stated, all experimental results for r-prime RSA with $r = 2, 3$ used 1024-bit moduli and all results with $r = 4$ used 2048-bit moduli. The choice of modulus size was based on Table 5.1 when $r = 2, 3$. While Table 5.1 suggests using a 4096-bit modulus when $r = 4$, we instead used a 2048-bit modulus to decrease the time for the lattice basis reduction. It is expected that general trend will be the same for a 4096-bit modulus.

5.9.1 Known MSB

Here we show experimental results for the attacks that exploit knowledge of some of the most significant bits in the private exponent, as given in Section 5.5. For the first few values of r, the effectiveness of the attacks are illustrated in Figures 5.5 and 5.6 for r-prime RSA with small private exponent and small public exponent, respectively. Notice that the scale in Figure 5.6 has been modified.

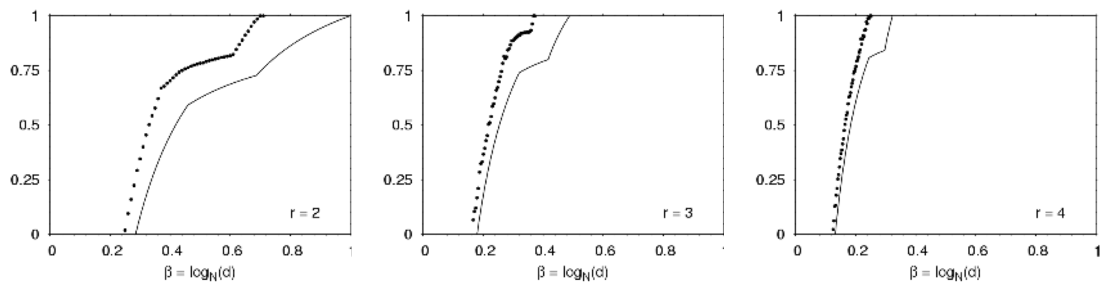


Figure 5.5: Experimental fraction of MSB, $(\beta - \delta)/\beta$, of the private exponent needed to attack small private exponent multi-prime RSA with limited resources. Lower line corresponds to theoretical bound when $N \to \infty$.

All of the attacks in Figure 5.5 are lattice-based. In each of the experiments the lattice used has dimension 20, which is one of the smallest lattice sizes allowed by the attacks. Already with this small lattice dimension, that attack is fairly successful compared to the theoretical bound (lower line in the plots). The lattice basis reduction took between 1–3 minutes for the experiments with $r = 2, 3$ (1024-bit moduli) and about 5–6 minutes for $r = 4$ (2048-bit moduli).

There are both latticed-based and non-lattice-based attacks illustrated

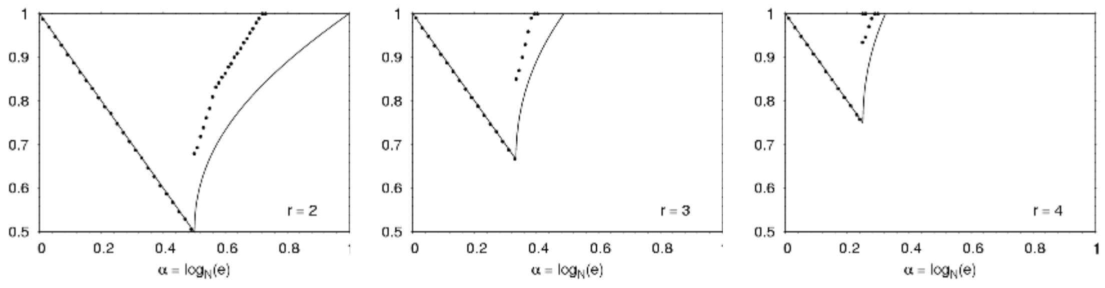


Figure 5.6: Experimental fraction of MSB, $(1 - \delta)$, of the private exponent needed to attack small public exponent multi-prime RSA with limited resources. Lines correspond to theoretical bound when $N \to \infty$.

in Figure 5.6. The non-lattice-based attacks that apply to multi-prime RSA with public exponents smaller than $N^{1/r}$ are in practice just as effective as the theory predicts. In addition, these attacks are very efficient, requiring a few seconds of work with Maple. For the lattice-based attacks, each experiment for $r = 2, 3$ (1024-bit moduli) used a lattice with dimension 20, with the lattice basis reduction taking roughly 1–3 minutes. Like the known MSB experiments, the attack is fairly successful compared to the theoretical bounds. When $r = 4$, however, the lattice-based attack with such a small dimension was not very successful at all. The results shown in Figure 5.6 for $r = 4$ are for experiments using lattices with dimension 32 with 256-bit moduli. The lattice basis reduction for these experiments required about 10 minutes. Based on other experiments, we have found that the effectiveness of the attacks is almost independent of the modulus size, so the results for $r = 4$ should be indicative of the results with same lattice dimension and 2048-bit moduli.

We did not implement the two attacks by Boneh, Durfee and Frankel that apply only to RSA (*i.e.*, Attacks 5.7 and 5.8).

5.9.2 Known LSB

Here we show experimental results for the attacks which exploit knowledge of some of the least significant bits in the private exponent, as given in Section 5.6. For the first few values of r, the effectiveness of the attacks is illustrated in Figures 5.7 and 5.8 for r-prime RSA with small private exponent and small public exponent, respectively.

In Figure 5.7, we demonstrate the effectiveness of Attack 5.9 when the

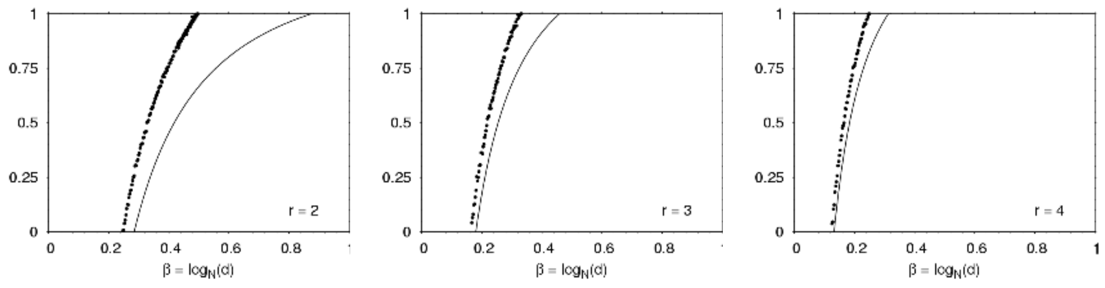


Figure 5.7: Experimental fraction of LSB, $(\beta - \delta)/\beta$, of the private exponent needed to attack small private exponent multi-prime RSA with limited resources. Lower line corresponds to theoretical bound when $N \to \infty$.

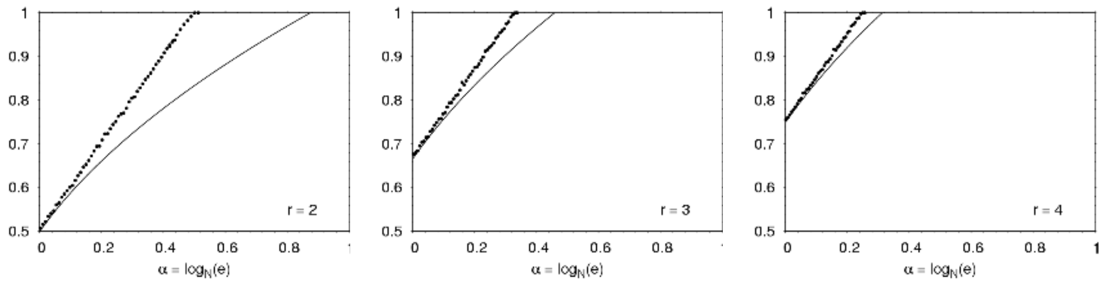


Figure 5.8: Experimental fraction of LSB, $(1 - \delta)$, of the private exponent needed to attack small public exponent multi-prime RSA with limited resources. Lower line corresponds to theoretical bound when $N \to \infty$.

private exponent is small. In each of the experiments, a lattice with dimension 16 was used. The experimental bounds with such a small lattice dimension are fairly close to the theoretical bounds. The lattice basis reduction took between 1–3 minutes for the experiments with $r = 2, 3$ (1024-bit moduli) and between 5–8 minutes when $r = 4$ (2048-bit moduli). We did not investigate the effectiveness of the extension of Boneh and Durfee's small private exponent attack, as it only applies to a very small range of private exponents not covered by the lattice-based attack in Section 5.6.1 and the extension of Wiener's attack (which is much more efficient that the lattice-based attacks).

In Figure 5.8, we demonstrate the effectiveness of Attack 5.9 when the public exponent is small. In each of the experiments, a lattice with dimension

16 was used. The experimental bounds with such a small lattice dimension are fairly close to the theoretical bounds. Even when $r = 4$ (2048-bit moduli), such a small lattice yields a good experimental bound unlike the case for known MSB. The lattice basis reduction took between 1–3 minutes for the experiments with $r = 2, 3$ (1024-bit moduli) and between 3–8 minutes when $r = 4$ (2048-bit moduli).

We did not implement Boneh, Durfee and Frankel's small public exponent attack on RSA (*i.e.*, Attack 5.10).

5.9.3 Known Partial Factorization

We did not perform any experiments for the attacks in Section 5.7. The direct computation method, Attack 5.12, simply involves computing a modular inverse. And, since the lattice-based attack, Attack 5.13, uses the same lattice methods as the known LSB partial key exposure attacks from Section 5.6, the effectiveness is expected to be same as the results shown Figures 5.7 and 5.8.

5.10 Discussion

In this chapter, we analyzed the security of multi-prime RSA with respect to algebraic attacks. There are essentially three types of algebraic attacks on multi-prime RSA: The factoring attacks of Section 5.3; attacks on textbook multi-prime RSA; and attacks on multi-prime RSA using CRT decryption.

The factoring attacks from Section 5.3 compute the prime factorization of N given only N. These are the only attacks that become stronger with increasing number of primes in the modulus. Because of this, the number of primes in the modulus cannot be chosen to be too large. When the number of primes is chosen properly, the security of multi-prime RSA is therefore no less than the security of RSA, with respect to the NFS and ECM factoring methods.

All of the attacks on textbook multi-prime RSA, Sections 5.4–5.7, exploit the key equation $ed = 1 + k\phi(N)$. In each of these attacks, the effectiveness of the attack decreases with increasing number of primes for a fixed modulus size. Thus, for each of these attacks, multi-prime RSA is more secure than RSA.

The only known attack on multi-prime RSA with CRT decryption shows that the CRT-exponents should not be chosen too small. The complexity of the attack is linear in the number of primes in the modulus, so multi-prime RSA is more secure than RSA with respect to this attack. Further, there have recently been some new attacks on RSA by Jochemsz and May [70], that can factor the modulus in polynomial time if the CRT-exponents are smaller than $N^{0.073}$. If this attack can be extended to multi-prime RSA, the bound is expected to be much smaller. (So much smaller in fact, that an exhaustive search for larger CRT-exponents will be feasible.)

Considering all of the known attacks on multi-prime RSA (with or without CRT decryption), the evidence suggests that multi-prime RSA with a safe number of primes is no less secure than RSA. Because of the reduced decryption costs when using the Chinese Remainder Theorem, using multi-prime RSA with $r > 2$ seems a viable alternative to RSA.

Open Problems/Future Work: While the attacks collected in this work represent the current state of the art in attacks against multi-prime RSA, more research needs to be done if it to be used a replacement for RSA. There are many unanswered questions about the security of multi-prime RSA. In particular, it is unknown if any partial key exposure attacks for full sized exponents exists for $r > 2$ (*cf.* [46] for $r = 2$), if there exist any attacks, other than factoring, that become more effective with increasing number of primes for a fixed modulus size and, most importantly, if any other attacks on multi-prime RSA CRT decryption exist.

Chapter 6

Common Prime RSA

In this chapter we consider a variant of RSA whose private exponent can be chosen smaller than $N^{1/4}$ and resists all known small private exponent attacks such as Wiener's continued fraction attack and Boneh and Durfee's lattice-based attack.

In this variant, called Common Prime RSA, the public and private exponents are inverses of each other modulo $\lambda(N) = \text{lcm}(p-1, q-1)$, where the primes p and q have the added structure that $p-1$ and $q-1$ must have a common large prime factor (*i.e*, $\gcd(p-1, q-1)$ has a large prime factor).

6.1 Background

The idea of using RSA primes with $\gcd(p-1, q-1)$ having a large prime factor is not new. Since 1990, it has been suggested several times with different purposes.

In 1990, it was suggested by Wiener [123] as a defence to his continued fraction attack on small private exponent RSA. Also in 1990, Girault [50] proposed an identity-based identification scheme using such primes where the common prime factor is made public.

In 1995, it was used by Lim & Lee [75] to improve server-aided RSA computations. This proposal, however, was later shown to be insecure by McKee & Pinch [83].

In 2006, Hinek [62] introduced Common Prime RSA, showing that instances of RSA with private exponents smaller than $N^{1/4}$ are secure against all known attacks provided that the size of the common prime factor is chosen properly. Later in 2006, Jochemsz [68] showed that the region of safe prime factors (admitting private exponents $d < N^{1/4}$) is smaller than proposed by Hinek, due to a new lattice-based attack. This new attack was subsequently presented by Jochemsz and May [69].

6.2 Common Prime RSA

Common Prime RSA was introduced by Hinek in [62], as an investigation into Wiener's suggestion that using a large common factor can be used to reduce his attack. In particular, the motivation was to determine whether instances of RSA with private exponents smaller than $N^{1/4}$ can be secure.

Common Prime RSA is a variant of RSA in which the primes have a very special structure. For some large prime g let $p = 2ga+1$ and $q = 2gb+1$ be balanced primes with the restrictions that $\gcd(a,b) = 1$ and $h = 2gab+a+b$ is prime. The first restriction ensures that $\gcd(p-1, q-1) = 2g$ while the second ensures that $(pq-1)/2 = gh$ is a semiprime[1] roughly the same size as $N = pq$. We will call primes p and q satisfying the above properties *common primes*, since $p-1$ and $q-1$ share a large common prime factor. From the structure of the common primes, we can write the modulus $N = pq$ as

$$\begin{aligned} N &= pq \\ &= (2ga+1)(2gb+1) \\ &= 2g(2gab+a+b)+1, \end{aligned}$$

and $N-1$ as

$$\begin{aligned} N-1 &= 2g(2gab+a+b) \\ &= 2gh. \end{aligned}$$

We define *Common Prime RSA* to be any instance of RSA that uses balanced common primes along with public and private exponents that are inverses of each other modulo $\lambda(N) = \operatorname{lcm}(p-1, q-1) = 2gab$. For notational convenience, we define $\gamma \in \mathbb{R}$ so that $g = N^{\gamma}$. Since we only consider instances of RSA with balanced primes we know that $g < N^{1/2}$ and so $0 < \gamma < 1/2$.

To generate common primes we can use Algorithm 6.1. Given n and γ, the algorithm generates balanced common primes p and q with a common factor $g = N^{\gamma}$ and such that $\gcd(a,b) = 1$.

One drawback of using Common Prime RSA, however, is that the computational cost of generating the primes is significantly larger than with typical RSA, as can be seen in Table 6.1. Here the average time needed to

[1]A semiprime is a natural number that is the product of two (not necessarily distinct) primes.

Algorithm 6.1 Common Prime Generation

```
Input: $(n, \gamma)$ such that $0 < \gamma < 1/2$.
1: $g \leftarrow$ random$\lceil \gamma n \rceil$-bit prime
2: $a, b \leftarrow$ random $\lceil (\frac{1}{2} - \gamma)n - 1 \rceil$-bit positive integers
3: $p \leftarrow 2ga + 1, \quad q \leftarrow 2gb + 1, \quad h \leftarrow 2gab + a + b$
4: If $p, q, h$ are not all prime OR $\gcd(a, b) \neq 1$
   then go back to step 2.
Output: Return the primes $p$ and $q$
```

generate common primes for various modulus sizes are computed. For each modulus size, the average is computed by generating 100 common prime pairs for each value of the 8 values of g given by

$$\gamma \in \{0.3, 0.325, 0.35, 0.375, 0.4, 0.425, 0.45, 0.475\}.$$

We also show the ratio of the average time needed to generate common primes pairs to the average time needed to generate two random primes of the same size (as would be used by RSA). As can be seen, the time needed to generate common primes is significantly longer than that needed to generate typical RSA primes. The computations were done on a Sun Fire V440 server with four UltraSPARC IIIi processors with 8 GB of memory each running at 1.062 GHz.

When generating instances of Common Prime RSA we are interested in instances with small private exponent. Thus, we first generate a random private exponent of desired size that is relatively prime to $\lambda(N) = 2gab$. When computing the private exponent's associated public exponent, we can simply use $e = d^{-1} \bmod \lambda(N)$, or we can add multiples of $\lambda(N)$ to this value to generate larger public exponents. When the public exponent is simply the inverse of d modulo $\lambda(N)$, the size of the public exponent e will be, with high probability, roughly the same size as $\lambda(N)$. Letting $e = N^{\alpha}$, we then have that $\alpha \approx 1 - \gamma$ with high probability. When adding multiples of $\lambda(N)$

Table 6.1: Average time needed to generate Common Prime RSA moduli for several modulus sizes. Also included is the ratio of common prime generation to random prime generation used in RSA.

Modulus Size (bits)	256	512	768	1024
Time (sec)	1.3 ± 1.2	16.7 ± 17.5	85.2 ± 84.0	272.6 ± 271.9
Common Prime / RSA	50.0	164.3	568.7	349.0

to generate larger public exponents, we only consider public exponents that have size roughly the same as the modulus N (*i.e.*, $\alpha \approx 1$). In this situation, the public exponent is the same size as a typical instance of RSA with small private exponent.

The analysis in [62] focused on small private exponent Common Prime RSA with public exponent satisfying $\alpha \approx 1-\gamma$. In this chapter, we improve this analysis and also consider instances when $\alpha \approx 1$.

6.3 Attacks on Common Prime Moduli

In this section we consider attacks that exploit the special structure of the common primes to help factor common prime moduli. Each of the attacks make use of either the equation for the modulus

$$N = 2g(2gab + a + b) + 1,$$

or the related equation

$$N - 1 = 2gh,$$

where g and h are primes.

6.3.1 Factoring N

It has been shown by McKee and Pinch [83], that the special structure of the Common Prime RSA primes p and q lead to an efficient factoring method for $N = pq$ if g is large enough. We restate their result in the following attack.

Attack 6.1 (McKee & Pinch). *Let N be a valid Common Prime RSA modulus with $g = N^{\gamma}$. There exists a method that factors N with $O(N^{1/4-\gamma/2})$ expected operations, each requiring time polynomial in $\log_2(N)$.*

Their method is a modification of Pollard's rho method (the result is an attack since the rho method has not yet been rigorously proved). In particular, the usual map $x \mapsto x^2 + 1 \bmod N$ is replaced with $x \mapsto x^{N-1} + 3 \bmod N$. Since $N-1 = 2gh$ and $p-1 = ga$ there can be at most a values of $x^{N-1} \bmod p$. Thus the expected number of steps is $O(\sqrt{a}) = O(N^{1/4-\gamma/2})$. To obtain an expected complexity of at least 2^{ℓ}, it follows that g should be chosen so that

$$\gamma < \frac{1}{2} - \frac{2\ell}{\log_2(N)}.$$

Since the common primes are balanced, we know that $\gamma < 1/2$. Thus, this attack shows that γ should not be chosen too close to its upper limit.

6.3.2 Factoring N with known a and b

Here we show that knowledge of both a and b leads to a very efficient method of factoring a common prime modulus N. The main result is given in the following theorem.

Theorem 6.2. *Let N be a valid Common Prime RSA modulus. Given the modulus N, a and b, we can factor N in time polynomial in* $\log_2(N)$.

Proof: Given a, b and N, notice that g is the only unknown in the equation for the modulus $N = 2g(2gab + a + b) + 1$. Rearranging this equation we obtain the quadratic equation

$$4abg^2 + 2(a+b)g - N + 1 = 0,$$

which has solutions

$$g = \frac{-2(a+b) \pm \sqrt{4(a+b)^2 - 4(4ab)(-N+1)}}{2(4ab)}.$$

Since g is positive, we conclude that (after some simplification)

$$g = \frac{-(a+b) + \sqrt{a^2 + (4N-2)\,ab + b^2}}{4ab}.$$

Once g has been computed, we easily compute the factorization of N since $p = 2ga + 1$ and $q = 2gb + 1$. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

While this is a very strong result, it is unclear how a and b might be obtained from N (and possibly from a public exponent e) other than by simply guessing. For an exhaustive search on a and b to have an expected complexity of at least 2^ℓ, it follows that g should be chosen so that

$$\gamma < \frac{1}{2} - \frac{\ell}{2\log_2(N)},$$

since a and b each has bitlength $(1/2 - \gamma)\log_2(N)$.

6.3.3 Factoring N with known g

Here we show that knowledge of g leads to methods for factoring a common prime modulus. Unlike in the previous section (known a and b), however, the efficiency of the methods depend on size of the additional information.

There are two scenarios that we must consider. The first, when $g \geq a+b$, always leads to an efficient method for factoring N. This scenario is equivalent to $g \geq N^{1/4}$. The main result is contained in the following theorem.

Theorem 6.3. *Let N be a valid Common Prime RSA modulus with $g \geq a+b$. Given the modulus N and g, we can factor N in time polynomial in $\log_2(N)$.*

Proof: We consider two cases: $g > a+b$ and $g = a+b$.

First we assume that $g > a+b$. Given N and g, let $m = (N-1)/(2g)$ and $c = a+b$ so that the equation for the modulus

$$N = 2g(2gab + a + b) + 1,$$

can be written as

$$m = 2gab + c.$$

Since $c = a+b < g$, by assumption, reducing this equation modulo g gives

$$c = m \bmod g.$$

Thus, we know the numerical value of c. Substituting $b = c - a$ back into the equation for the modulus $N = 2g(2gab + a + b) + 1$ yields, after some rearrangement, the quadratic equation

$$2ga^2 - 2gca + (N-1)/(2g) - c = 0,$$

which has solutions

$$\frac{2gc \pm \sqrt{4g^2c^2 - 4(2g)\left(\frac{N-1}{2g} - c\right)}}{2(2g)},$$

or more simply

$$\frac{gc \pm \sqrt{2g^2c^2 - (N-1) + 2gc}}{2g}.$$

These two solutions correspond to a and b. Computing $2ga+1$ and $2gb+1$ thus yields the factorization of N.

In the next case, we assume that $g = a+b$. Again, starting with the equation for the modulus

$$N = 2g(2gab + a + b) + 1,$$

we replace $a + b$ with g to obtain, after some manipulation,

$$\frac{N-1}{4g^2} = ab + \frac{1}{2}.$$

We next replace b with $g - a$ to obtain, after some rearrangement, the quadratic equation

$$a^2 - ga + \frac{N-1}{4g^2} - \frac{1}{2},$$

which has solutions

$$\frac{g \pm \sqrt{g^2 - 4\left(\frac{N-1}{4g^2} - \frac{1}{2}\right)}}{2}.$$

These solutions correspond to a and b. Computing $2ga+1$ and $2gb+1$ thus yields the factorization of N.

Since all computations, for each case considered, can be done in time polynomial in $\log_2(N)$, the result follows. ❏

The second scenario that we must examine is when $g < a + b$. Since a and b are the same size this is equivalent to the scenario that $g < N^{1/4}$, which has already been considered by McKee and Pinch [83]. We restate their result in the following theorem.

Theorem 6.4 (McKee & Pinch). *Let N be a valid Common Prime RSA modulus with $g < a + b$. Given the modulus N and g, there exists a method that factors N with $O(N^{1/4-\gamma})$ expected operations, where each operation requires time polynomial in* $\log_2(N)$.

The method uses Shanks' baby-step giant-step methodology (see [83] for more details). As can be seen by the result, the efficiency of the factoring method depends on the size of g. For the method to run in time polynomial in $\log_2(N)$, the size of g should satisfy $\gamma \approx \frac{1}{4} - c\log_2(\log_2(N))$ for some constant c (*i.e.*, γ should be very close to $\frac{1}{4}$). To obtain an expected complexity of at least 2^ℓ, it follows that g should be chosen so that

$$\gamma < \frac{1}{4} - \frac{\ell}{\log_2(N)}. \tag{6.1}$$

As with the attack with known a and b, it remains to show how g might be obtained. For an exhaustive search on g to have an expected complexity of at least 2^ℓ, it follows that g should be chosen so that

$$\gamma > \frac{\ell}{\log_2(N)}. \tag{6.2}$$

In the next subsection we show that the special structure of the common primes p and q leads to another method of obtaining g that is more efficient than exhaustive search, but is still infeasible if g is chosen large enough.

6.3.4 Factoring $N-1$ (or Computing g)

Starting with the equation for the common prime modulus

$$N = 2g(2gab + a + b) + 1, \tag{6.3}$$

and recalling that $h = 2gab + a + b$, we have

$$N - 1 = 2gh. \tag{6.4}$$

Therefore, we can obtain g (and h) by simply factoring $(N-1)/2$.

Since $(N-1)/2$ is essentially the same size as N, it is expected that the NFS will factor $(N-1)/2$ in about the same time as factoring N. Factoring $(N-1)/2$ with the ECM might be more fruitful, however, depending on how unbalanced g and h are. Clearly, if g is chosen too small, it will easily be recovered by the ECM.

As a rough estimate of evaluating the minimum size of g for a given modulus size, we equate the heuristic runtime of the ECM, given in equation (1.2), with that of factoring the modulus with the NFS, given by equation (1.1). For a given modulus size, let γ_{ecm} denote the smallest value for γ such that for any $\gamma < \gamma_{ecm}$, the expected runtime of the ECM is less than the expected runtime of the NFS for that size (of modulus). Some values for γ_{ecm} for some common modulus sizes are given in the following table.

$\log_2(N)$	$\log_2(\gamma_{ecm})$	γ_{ecm}
1024	292	0.284
2048	529	0.258
4096	940	0.229
8192	1640	0.200

It should be pointed out that these bounds for γ_{ecm} are just estimates. While the formulas for the expected runtimes for the ECM and NFS give a good indication of the asymptotic runtime of the methods, the current factoring records should also be considered. For example, as of April 2005, the largest factor obtained by the ECM is a 220-bit number. While this record will be increased as experiments are continued, the bound for g for 1024-bit moduli will offer some security or at least a few years to come. For the larger modulus sizes shown in the table, factoring $(N-1)/2$ with a minimum g as shown will be infeasible with the ECM for many years to come.

6.4 Small Private Exponent Attacks

In this section we consider attacks on Common Prime RSA having small private exponents. In addition to exploiting the special structure of the common primes the attacks in this section will also use the key equation

$$ed = 1 + k(2gab),$$

along with the fact that the private exponent may be small.

6.4.1 Attacks with known g

When the private exponent d is smaller than the common factor g, we can very efficiently compute the private exponent. We state this result in the following theorem.

Theorem 6.5. *Let N be a valid Common Prime RSA modulus with $g = N^{\gamma}$. Let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $e = N^{\alpha}$ and $d = N^{\delta}$. Given the public key and g, if the private exponent satisfies*

$$\delta < \gamma, \tag{6.5}$$

then d can be computed in time polynomial in $\log_2(N)$ and N can be factored in probabilistic time polynomial in $\log_2(N)$.

Proof: Let N, e and g be known. We begin by reducing the key equation

$$ed = 1 + k(2gab),$$

modulo g to obtain

$$ed \equiv 1 \pmod{g},$$

which gives us

$$d \equiv e^{-1} \pmod{g}.$$

If $\delta < \gamma$, which is equivalent to $d < g$, this last relation becomes

$$d = e^{-1} \bmod g.$$

Thus, when $\delta < \gamma$, we can simply compute the private exponent using this last equation. Since we can compute this modular inverse in time polynomial in $\log_2(N)$, the first part of the result follows.

Once d has been computed, notice that multiplying $(ed-1)$ by $2g$ (all known quantities now) gives

$$\begin{aligned} 2g(ed-1) &= 2g(k2gab) \\ &= k(4g^2ab) \\ &= k\phi(N). \end{aligned}$$

Therefore, we can easily compute a multiple of $\phi(N)$ which allows us to factor N with using Miller's result [88]. Since all computations can be done in time polynomial in $\log_2(N)$, the second part of the result follows. ❑

The next attack allows us to factor the modulus N when the private exponent d is larger than the common factor g, provided that $g < N^{1/4}$. This size restriction on g is not a drawback to the attack though, since we have already seen in Section 6.3.3 that knowledge of g when $g > N^{1/4}$ leads to an efficient factoring method. The result is given in the following attack.

Attack 6.6. *For every $\epsilon > 0$ there exists N_0 such that for every $N > N_0$ the following holds: Let N be a valid Common Prime Modulus with common factor $g = N^\gamma < N^{1/4}$, let (e, N) be a valid public key, and let (d, p, q) be its corresponding private key, where $e = N^\alpha$ and $d = N^\delta$. Given g, if the private exponent satisfies*

$$\delta < \frac{7}{6} - \frac{5\gamma}{3} - \frac{1}{3}\sqrt{40\gamma^2 - 38\gamma + 7} - \epsilon, \tag{6.6}$$

when $\alpha \approx 1-\gamma$, or

$$\delta < \frac{7}{6} - \frac{5\gamma}{3} - \frac{1}{3}\sqrt{16\gamma^2 - 32\gamma + 7} - \epsilon, \tag{6.7}$$

when $\alpha \approx 1$, then the modulus N can be factored in time polynomial in $\log_2(N)$*, provided Assumption 2.13 holds.*

Justification: Starting with the equation for the modulus, namely,

$$N = 2g(2gab + a + b) + 1,$$

we divide both sides by $4g^2$, and rearrange, to obtain

$$\frac{N-1}{4g^2} = ab + \frac{a+b}{2g}.$$

Letting $M = \lceil (N-1)/(4g^2) \rceil$, we see that M is a good (over) approximation of ab. In particular, notice that

$$\begin{aligned} |M - ab| &= \frac{a+b}{2g} \\ &< N^{1/2-2\gamma}. \end{aligned} \tag{6.8}$$

Letting m denote the difference between M and ab we then have $m = M - ab$, where $|m| < N^{1/2-2\gamma}$. Substituting $M - m$ for ab in the key equation

$$ed = 1 + k(2gab), \tag{6.9}$$

we obtain, after some rearrangement,

$$2gkM - 2gkm + 1 = ed, \tag{6.10}$$

where only k, m and d are unknowns. This suggests that we look for small solutions, modulo the public exponent e, of the polynomial

$$f_e(x, y) = 2gMx - 2gxy + 1,$$

since $(x_0, y_0) = (k, m)$ satisfies $f_e(x_0, y_0) \equiv 0 \bmod e$. For bounds X and Y such that $|x_0| < X$ and $|y_0| < Y$, we know from Theorem 2.10, that if

$$X^{2+3\tau} Y^{1+3\tau+3\tau^2} \leq e^{1+3\tau}, \tag{6.11}$$

for some $\tau > 0$, then x_0 and y_0 can be recovered for sufficiently large e. We consider the cases $\alpha \approx 1 - \gamma$ and $\alpha \approx 1$ separately.

Let's first assume that $\alpha \approx 1 - \gamma$. From the key equation

$$ed = 1 + k(2gab),$$

notice that $k < N^{\delta}$. Also, we have already shown above that $|m| < N^{1/2-2\gamma}$. Thus, we define the bounds

$$\begin{aligned} X &= N^{\delta} \\ Y &= N^{1/2-2\gamma}, \end{aligned}$$

so that $x_0 = k$ and $y_0 = m$ both satisfy $|x_0| < X$ and $|y_0| < Y$. Substituting these bounds and $e = N^{1-\gamma}$, and neglecting all factors that do not depend on N, inequality (6.11) is satisfied when

$$\left(\frac{3}{2} - 6\gamma\right)\tau^2 + \left(3\delta - \frac{3}{2} - 3\gamma\right)\tau + 2\delta - \frac{1}{2} - \gamma < 0.$$

For all $\gamma < \frac{1}{4}$, when all variables but τ are fixed, the left-hand side of this inequality is minimized when

$$\tau = \frac{2\gamma + 1 - 2\delta}{2(4\gamma - 1)}.$$

Substituting this value for τ into the last inequality and solving for δ, we see that (6.11) is satisfied when

$$\delta < \frac{7}{6} - \frac{5\gamma}{3} - \frac{1}{3}\sqrt{40\gamma^2 - 38\gamma + 7} - \epsilon,$$

where the ϵ term is added to correct for all the negligible and low order terms ignored here and in the methods implicit in Theorem 2.10.

Next we assume that $\alpha \approx 1$. In this case, from the key equation

$$ed = 1 + k(2gab),$$

notice that $k < N^{\alpha+\delta-1+\gamma} = N^{\delta+\gamma}$. Again, we have $|m| < N^{1/2-2\gamma}$. Thus, we define the bounds

$$X = N^{\delta+\gamma}$$
$$Y = N^{1/2-2\gamma},$$

so that $x_0 = k$ and $y_0 = m$ both satisfy $|x_0| < X$ and $|y_0| < Y$. Substituting these bounds and $e = N$, and neglecting all factors that do not depend on N, inequality (6.11) is satisfied when

$$\left(\frac{3}{2} - 6\gamma\right)\tau^2 + \left(3\delta - \frac{3}{2} - 3\gamma\right)\tau + 2\delta - \frac{1}{2} < 0.$$

For all $\gamma < \frac{1}{4}$, when all variables but τ are fixed, the left-hand side of this inequality is minimized when

$$\tau = \frac{2\gamma + 1 - 2\delta}{2(4\gamma - 1)}.$$

Substituting this value for τ into the last inequality and solving for δ, we see that (6.11) is satisfied when

$$\delta < \frac{7}{6} - \frac{5\gamma}{3} - \frac{1}{3}\sqrt{16\gamma^2 - 32\gamma + 7},$$

where the ϵ term is added to correct for all the negligible and low order terms ignored here and in the methods implicit in Theorem 2.10.

In both cases, if Assumption 2.13 holds, we can compute $(x_0, y_0) = (k, m)$. With this, we can compute ab since

$$ab = M - m.$$

Furthermore, from $N - 1 = 4g^2ab + a + b$ we can compute $a + b$ since

$$\begin{aligned} a + b &= N - 1 - 4g^2ab \\ &= N - 1 - 4g^2(M - m). \end{aligned}$$

Thus, we have two equations with two unknowns (a and b). Combining the two equations, we can generate a quadratic equation (in a or b) which can be easily solved. The roots of either quadratic will correspond to a and b. Once a and b are computed, we simply compute the factorization of N since $p = 2ga+1$ and $q = 2gb+1$. Since all computations require time polynomial in $\log_2(N)$, the result follows. ❏

6.4.2 Continued Fraction Attack

In this section we reanalyze Wiener's continued fraction attack when applied to Common Prime RSA. The main result is the following theorem.

Theorem 6.7. *Let N be a valid Common Prime RSA modulus with common factor $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $e = N^\alpha$ and $d = N^\delta$. If the private key satisfies*

$$\delta < \frac{3}{4} - \frac{\alpha}{2} - \gamma - \log_N(\sqrt{24}), \tag{6.12}$$

then we can compute the private exponent d in time polynomial in $\log_2(N)$.

In particular, when $\alpha \approx 1 - \gamma$, this sufficient condition becomes

$$\delta < \frac{1}{4} - \frac{\gamma}{2} - \log_N(\sqrt{24}),$$

and when $\alpha \approx 1$, it becomes

$$\delta < \frac{1}{4} - \gamma - \log_N(\sqrt{24}).$$

Proof: First let's assume that $\delta < \frac{3}{4} - \frac{\alpha}{2} - \gamma - \log_N(\sqrt{24})$. From this, notice that

$$\begin{aligned}
& \delta < \frac{3}{4} - \frac{\alpha}{2} - \gamma - \log_N(\sqrt{24}) \\
\iff \quad & 2\delta < \frac{3}{2} - \alpha - 2\gamma - \log_N(24) \\
\iff \quad & 2\delta + \alpha - 1 + \gamma < \frac{1}{2} - \gamma - \log_N(24) \\
\iff \quad & N^{2\delta+\alpha-1+\gamma} < \frac{1}{24} N^{\frac{1}{2}-\gamma} \\
\iff \quad & N^{\delta+\alpha-1+\gamma} N^{\delta} < \frac{1}{24} N N^{-\gamma} N^{-\frac{1}{2}} \\
\iff \quad & kd < \frac{N}{8g} \times \frac{1}{3N^{\frac{1}{2}}} \\
\implies \quad & kd < \frac{N}{8g} \times \frac{1}{\Lambda} \\
\iff \quad & \frac{k\Lambda}{gdN} < \frac{1}{2(2gd)^2}.
\end{aligned}$$

Therefore, we have

$$\delta < \frac{3}{4} - \frac{\alpha}{2} - \gamma - \log_N(\sqrt{24}) \quad \implies \quad \frac{k\Lambda}{gdN} < \frac{1}{2(2gd)^2}. \tag{6.13}$$

Next, we consider the key equation

$$ed = 1 + k(2gab).$$

Multiplying the last term by $(2g)/(2g)$ yields

$$\begin{aligned}
ed &= 1 + \frac{k}{2g}(4g^2ab) \\
&= 1 + \frac{k}{2g}\phi(N) \\
&= 1 + \frac{k}{2g}(N - \Lambda),
\end{aligned}$$

where we have replaced $\phi(N)$ with $N - \Lambda$. Dividing both sides of this last equation by N and d, yields, after some rearrangement,

$$\frac{e}{N} - \frac{k}{2gd} = -\frac{k\Lambda}{2gdN} + \frac{1}{dN}.$$

Since $\Lambda > N^{1/2}$ and $g < N^{1/2}$, we have that

$$\begin{aligned}\left|-\frac{k\Lambda}{2gdN}+\frac{1}{dN}\right| &< \left|-\frac{k\Lambda}{2gdN}\right|+\left|\frac{1}{dN}\right| \\ &< 2\left|\frac{k\Lambda}{2gdN}\right| \\ &= \frac{k\Lambda}{gdN},\end{aligned}$$

and so

$$\left|\frac{e}{N}-\frac{k}{2gd}\right| < \frac{k\Lambda}{gdN}.$$

Combining this inequality with (6.13) then gives

$$\left|\frac{e}{N}-\frac{k}{2gd}\right| < \frac{1}{2(2gd)^2}, \tag{6.14}$$

and so from Theorem 2.1 (continued fractions), we know that $k/(2gd)$, in lowest terms, will be one of the convergents in the continued fraction expansion of e/N.

Next, we test each convergent until we find the one that yields the private exponent d. Let u_i/v_i denote the i^{th} convergent in the continued fraction expansion of e/N. Notice that the correct convergent, u_j/v_j say, satisfies

$$\frac{u_j}{v_j}=\frac{k}{2gd}.$$

It follows from the key equation, $ed = 1 + k\lambda(N)$, that $\gcd(k,d) = 1$. Therefore, we know that $v_j = c_j d$, where $c_j = (2g)/\gcd(k,2g)$. Therefore, to extract c_j from v_j, we simply compute

$$\begin{aligned}\gcd(v_j, N-1) &= \gcd(c_j d, 2gh) \\ &= c_j.\end{aligned}$$

This follows since $\gcd(d,h) = 1$ (as $d < N^{1/4} < h$ and h is prime), and $\gcd(d, \gcd(k,2g)) = 1$ (as $\gcd(d,k) = 1$). Therefore, we compute

$$d_i = \frac{v_i}{\gcd(v_i, N-1)},$$

until we have determined the private exponent d (when $i = j$). We test for the correct d by encrypting and decrypting a few random plaintexts. Alternatively, notice that multiplying the right-hand side of the key equation

$$ed = 1 + k(2gab),$$

by $(2g)/(2g)$ yields

$$\begin{aligned} ed &= 1 + \frac{k}{2g} 4g^2 ab \\ &= 1 + \frac{k}{2g} \phi(N). \end{aligned}$$

Since $\frac{k}{2g} = \frac{u_j}{v_j} d_j$, we can compute $\phi(N)$ using

$$\frac{ed_j - 1}{\frac{u_j}{v_j} d_j} = 4g^2 ab.$$

The correct convergent will then give us $\phi(N)$, which we can use to factor N. Since the number of convergents is polynomial in $\log_2(N)$ and all computations require time polynomial in $\log_2(N)$, the result follows. ❑

Applying the techniques of Verheul and van Tilborg [120] and Dujella [43], we can extend the bound on the private exponent at the expense of an exhaustive search on each convergent in the continued fraction expansion of e/N. For a search space of 2^ℓ, the bound on the private exponent can be increased by $2^{\ell/2}$. The result is given in the following corollary.

Corollary 6.8. *Let N be a valid Common Prime RSA modulus with common factor $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $e = N^\alpha$ and $d = N^\delta$. Let m be given by*

$$m = 2\left(\delta - \left(\frac{3}{4} - \frac{\alpha}{2} - \gamma - \log_N(\sqrt{24})\right)\right)\log_2(N).$$

There exists a method to compute a list that contains the value $k/(2gd)$, in lowest terms, in time linear in 2^m and polynomial in $\log_2(N)$. Further, the size of the list is linear in 2^m and the size of the elements in the list are polynomial in $\log_2(N)$.

Thus, to ensure that the complexity of computing this list is at least 2^ℓ, it follows that the private exponent should satisfy

$$\delta > \frac{3}{4} - \frac{\alpha}{2} - \gamma - \log_N(\sqrt{24}) + \frac{\ell}{2\log_2(N)}. \tag{6.15}$$

6.4.3 Some Lattice-Based Attacks

In this section we present some attacks based on Coppersmith's method for finding small solutions to univariate modular equations. In particular, we make use of Theorem 2.10 and Assumption 2.13. As with most lattice-based attacks on RSA, we begin with the key equation.

The first attack is motivated by a technique used by May (see [82, §3]). The result, which computes sufficiently small private exponents, is contained in the following theorem.

Theorem 6.9. *For every $\epsilon > 0$ there exists N_0 such that for every $N > N_0$ the following holds: Let N be a valid Common Prime RSA modulus with $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key with $d = N^\delta$. If the private exponent satisfies*

$$\delta < \gamma^2 - \epsilon, \tag{6.16}$$

then d can be computed in time polynomial in $\log_2(N)$.

Proof: We begin by computing the inverse of the public exponent e modulo gh, which we denote by $\widehat{e}$. Thus, we have

$$\widehat{e} = e^{-1} \bmod gh,$$

where $hg = (N-1)/2$. If the inverse does not exist, we have factored $(N-1)/2 = gh$ and can apply Theorem 6.5 (with known g) to compute the private exponent if $\delta < \gamma$. Therefore, we assume without loss of generality that the inverse exists and so there must also exist an integer K such that

$$e\widehat{e} = 1 + Kgh.$$

Multiplying the key equation,

$$ed = 1 + k(2gab),$$

by $\widehat{e}$ then gives

$$\widehat{e}ed = \widehat{e} + \widehat{e}(k2gab).$$

Replacing $\widehat{e}e$ with $1 + Kgh$, this equation, after some rearrangement, can be written as

$$d - \widehat{e} = (2kab - Khd)g.$$

This equation suggests that we look for small solutions, modulo g, of the polynomial

$$f_g(x) = x - \widehat{e},$$

since $(x_0) = d$ satisfies $f_g(x_0) \equiv 0 \bmod g$. Even though we do not know g, we do know a multiple of it (*i.e.*, $(N-1)/2 = gh$). Thus we can use Theorem 2.8 to find all small solutions of $f_g(x)$. In this case, we have the parameters $\sigma = 1$ (degree of the polynomial) and $\beta = \gamma$ (since $g = N^\gamma > \left(\frac{N-1}{2}\right)^\gamma$). Therefore, applying the result of Theorem 2.8, we can compute $x_0 = d$ for sufficiently large N provided that

$$\begin{aligned} d &< N^{\frac{\beta^2}{\sigma}-\epsilon} \\ &= N^{\gamma^2-\epsilon}. \end{aligned}$$

Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❑

If in addition to the private exponent satisfying $\delta < \gamma^2 - \epsilon$, the public exponent is also sufficiently small then we can factor the modulus N. We formalize this result in the following corollary.

Corollary 6.10. *For every $\epsilon > 0$ there exists N_0 such that for every $N > N_0$ the following holds: Let N be a valid Common Prime RSA modulus with $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key with $d = N^\delta$. If the private and public exponents satisfy*

$$\begin{aligned} &\delta < \gamma^2 - \epsilon \\ &\alpha < 2 - \delta - 2\gamma, \end{aligned} \tag{6.17}$$

then the modulus N can be factored in probabilistic time polynomial in $\log_2(N)$.

Proof: We first observe that the condition on the public exponent implies that $k < h$. To see this, notice that

$$\begin{aligned} &\alpha < 2 - \delta - 2\gamma \\ \iff\quad &\alpha + \delta - 1 + \gamma < 1 - \gamma \\ \iff\quad &k < h. \end{aligned}$$

Thus, when $\alpha < 2 - \delta - 2\gamma$ we know that $k < h$. Also, since $h > N^{1/2}$ we know that $a, b < h$.

When the private exponent satisfies $\delta < \gamma^2 - \epsilon$, we know from Theorem 6.9 that we can compute the private exponent d. With e, d and N known we can then compute g since

$$\begin{aligned}\gcd(ed-1, N-1) &= \gcd(k(2gab), 2gh) \\ &= 2g.\end{aligned}$$

This gcd computation must yield $2g$ since $a, b, k < h$ and h is a prime. Once $2g$ is known, we compute

$$\begin{aligned}2g(ed-1) &= k(4g^2ab) \\ &= k\phi(N),\end{aligned}$$

which is a multiple of $\phi(N)$. Using Miller's result [88], we can use $k\phi(N)$ and N to factor N in probabilistic time polynomial in $\log_2(N)$. ❏

The attack in Theorem 6.9 also leads to a simple partial key-exposure attack. If the private exponent is written as

$$d = d_2 2^{\ell_2} + d_1 2^{\ell_1} + d_0,$$

where everything is known except d_1, then we can compute d_1 (and hence compute the private exponent) provided that d_1 is small enough. The result is given in the following corollary.

Corollary 6.11. *For every $\epsilon > 0$ there exists N_0 such that for every $N > N_0$ the following holds: Let N be a valid Common Prime RSA modulus with $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key. Let the private exponent be written as $d = d_2 2^{\ell_2} + d_1 2^{\ell_1} + d_0$, where everything is known except $d_1 = N^{\delta_1}$. If the unknown part of the private exponent, d_1, satisfies*

$$\delta_1 < \gamma^2 - \epsilon, \tag{6.18}$$

then the private exponent d can be computed in time polynomial in $\log_2(N)$.

Proof: Let $d = d_2 2^{\ell_2} + d_1 2^{\ell_1} + d_0$, where everything is known except d_1, so that the equation can be written

$$e(d_2 2^{\ell_2} + d_1 2^{\ell_1} + d_0) = 1 + k2gab,$$

or, more simply as

$$e_1 d_1 = c_1 + k2gab,$$

where $e_1 = e2^{\ell_1}$ and $c_1 = 1 - (d_2 2^{\ell_2} + d_0)$. Using this key equation we proceed identically as in the proof of Theorem 6.9. ❏

The second lattice-based attack we consider was suggested to us by an anonymous referee for what eventually resulted in [62]. The main result is given in the following attack.

Attack 6.12. *For every $\epsilon > 0$, there exists N_0 such that for every $N > N_0$, the following holds: Let N be a valid Common Prime RSA modulus with common factor $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $e = N^\alpha$ and $d = N^\delta$. If $\alpha \approx 1 - \gamma$ and the private exponent satisfies*

$$\delta < \frac{2\gamma}{5} - \epsilon, \tag{6.19}$$

then d can be computed in time polynomial in $\log_2(N)$, provided that Assumption 2.13 holds.

Justification: We begin with the observation that $\operatorname{lcm}(p-1, q-1) = 2gab$ can be written can be as $(p-1)b$ and also as $(q-1)a$, since $p - 1 = 2ga$ and $q - 1 = 1gb$. Thus, we can write the key equation as both

$$\begin{aligned} ed &= 1 + k(p-1)b \\ ed &= 1 + k(q-1)a. \end{aligned} \tag{6.20}$$

Writing these equations as

$$\begin{aligned} ed - 1 + kb &= kpb \\ ed - 1 + ka &= kqa, \end{aligned} \tag{6.21}$$

we can multiply them together to obtain

$$e^2d^2 + e(ka + kb - 2)d - k^2ab(N-1) - k(a+b) - 1 = 0. \tag{6.22}$$

This suggests that we look for small integer solutions of the polynomial

$$f(x, y, z, u) = e^2x + ey - (N-1)z - u - 1, \tag{6.23}$$

since $(x_0, y_0, z_0, u_0) = (d^2, (ka+kb-2)d, k^2ab, k(a+b))$ is a root of $f(x, y, z, u)$ over the integers. Recalling that

$$\begin{aligned} e &= N^\alpha \\ d &= N^\delta \\ k &= N^{\alpha+\delta-1+\gamma} \\ a &= N^{1/2-\gamma} \\ b &= N^{1/2-\gamma}, \end{aligned} \tag{6.24}$$

we define the bounds

$$
\begin{aligned}
X &= N^{2\delta} \\
Y &= N^{\alpha+2\delta-1/2} \\
Z &= N^{2\alpha+2\delta-1} \\
U &= N^{\alpha+\delta-1/2},
\end{aligned}
\tag{6.25}
$$

so that $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$, $|u_0| < U$ and

$$
\begin{aligned}
W &= \|f(xX, yY, zZ, uU)\|_\infty \\
&= \max(e^2X, eY, (N-1)Z, U, 1) \\
&= e^2X \\
&= N^{2\alpha+2\delta}.
\end{aligned}
\tag{6.26}
$$

From Theorem 2.12, we then know that for sufficiently large N we can recover (x_0, y_0, z_0, u_0) provided that

$$XYZU < W, \tag{6.27}$$

and Assumption 2.13 holds. Substituting the values for X, Y, Z, U and W, this is inequality is satisfied when

$$4\alpha + 7\delta - 2 < 2\alpha + 2\delta.$$

Letting $\alpha \approx 1 - \gamma$, this simplifies to

$$\delta < \frac{2\gamma}{5} - \epsilon,$$

where we have added the ϵ term to correct for all lower order terms neglected in the methods implicit in Theorem 2.12. Thus, for sufficiently large N, if Assumption 2.13 holds, we can compute $x_0 = d^2$ which immediately gives the private exponent d. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

When the public exponent e is roughly the same size as the modulus (*i.e.* $\alpha \approx 1$), this attack is no longer meaningful. Working through the details, one finds that the sufficient condition for the attack to succeed when $\alpha \approx 1$ is given by $\delta < 0$.

6.4.4 Jochemsz-May Attack

In 2006, Jochemsz [68] and Jochemsz and May [69] presented a lattice-based attack on small private exponent Common Prime RSA when the public exponent satisfies $\alpha \approx 1-\gamma$. This attack is stronger than the lattice-based attacks of the previous section. We restate their result in the following attack.

Attack 6.13 (Jochemsz-May). *For every $\epsilon > 0$, there exists N_0 such that for every $N > N_0$, the following holds: Let N be a valid Common Prime RSA modulus with common factor $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $e = N^{1-\gamma}$ and $d = N^\delta$. If the private exponent satisfies*

$$\delta < \frac{1}{4}\left(4 + 4\gamma - \sqrt{13 + 20\gamma + 4\gamma^2}\right) - \epsilon, \tag{6.28}$$

then d can be computed in time polynomial in $\log_2(N)$, provided that Assumption 2.13 holds.

The attack is actually stronger since we can also factor the modulus N if the attack is successful. This follows since the lattice-based method computes the root $(x_0, y_0, z_0) = (d, ka, kb)$. Since the key generation algorithm ensures that $\gcd(a, b) = 1$, we can compute $\gcd(ka, kb) = k$ which immediately reveals both a and b. With a and b known, we can simply compute the factorization of N using Theorem 6.3 (factoring with known a and b). Thus, we have the following corollary to Attack 6.13.

Corollary 6.14. *In addition to computing the private exponent d, if Attack 6.13 is successful, then the modulus N can be factored in time polynomial in $\log_2(N)$.*

When the public exponent is roughly the same size as the modulus (*i.e.* $\alpha \approx 1$), then the effectiveness of Attack 6.13 is significantly reduced. We give the details in the following attack.

Attack 6.15. *For every $\epsilon > 0$, there exists N_0 such that for every $N > N_0$, the following holds: Let N be a valid Common Prime RSA modulus with common factor $g = N^\gamma$, let (e, N) be a valid public key and let (d, p, q) be its corresponding private key, where $e \approx N$ and $d = N^\delta$. If the private exponent satisfies*

$$\delta < 1 - \frac{\sqrt{13}}{4} - \epsilon \tag{6.29}$$

$$\approx 0.0986 - \epsilon, \tag{6.30}$$

then N can be factored in time polynomial in $\log_2(N)$*, provided that Assumption 2.13 holds.*

Justification: Beginning with (6.22), from the proof in Attack 6.12, we have that

$$e^2d^2 + e(ka + kb - 2)d - k^2ab(N-1) - k(a+b) - 1 = 0.$$

Instead of linearizing each of the unknown terms, (*e.g.*, d^2, k^2ab, *etc.*), Attack 6.13 considers each variable d, k, a and b individually. Thus, we look for small integer solutions to the polynomial

$$f(x,y,z) = e^2x^2 + ex(y+z-2) - (y+z+1) - (N-1)yz, \tag{6.31}$$

which has the integer root $(x_0, y_0, z_0) = (d, ka, kb)$. When the public exponent satisfies $\alpha \approx 1$ we have $d = N^\delta$, $k < N^\delta$, and $a, b < N^{1/2-\gamma}$. Defining the bounds

$$\begin{aligned} X &= N^\delta \\ Y &= N^{\delta+1/2-\gamma} \\ Z &= N^{\delta+1/2-\gamma}, \end{aligned} \tag{6.32}$$

we then have $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$ and

$$\begin{aligned} W &= \|f(xX, yY, zZ)\|_\infty \\ &= e^2X^2 \\ &= N^{2+2\delta}. \end{aligned}$$

From Theorem 2.12, we then know that for sufficiently large N we can recover (x_0, y_0, z_0, u_0) provided that

$$X^{7+9\tau}(YZ)^{5+\frac{9}{2}\tau} < W^{3+3\tau},$$

for some $\tau > 0$ and Assumption 2.13 holds. Substituting the values for X, Y, Z and W, we see that this inequality is satisfied when

$$3\delta\tau^2 + \left(12\delta + \frac{3}{2}\right)\tau + 11\delta - 1 < 0.$$

For all fixed $\delta > 0$, the left-hand side of this inequality is minimized when

$$\tau = \frac{1-8\delta}{4\delta}.$$

Substituting this value for τ and solving for δ, we find a sufficient condition to recover (x_0, y_0, z_0) is given by

$$\begin{aligned} \delta &< 1 - \frac{1}{4}\sqrt{13} - \epsilon \\ &\approx 0.0986 - \epsilon, \end{aligned}$$

where we have added the ϵ term to correct for all lower order terms ignored in the methods implicit in Theorem 2.12. Thus, for sufficiently large N, if Assumption 2.13 holds, we can compute $(x_0, y_0, z_0) = (d, ka, kb)$ and compute the factorization of N as in Corollary 6.14. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

6.5 Summary of Security

Here we summarize all of the conditions on the Common Prime RSA parameters (α, δ and γ) to avoid all known attacks.

In order to avoid the factoring attacks from Section 6.3, the common factor g should be chosen so that

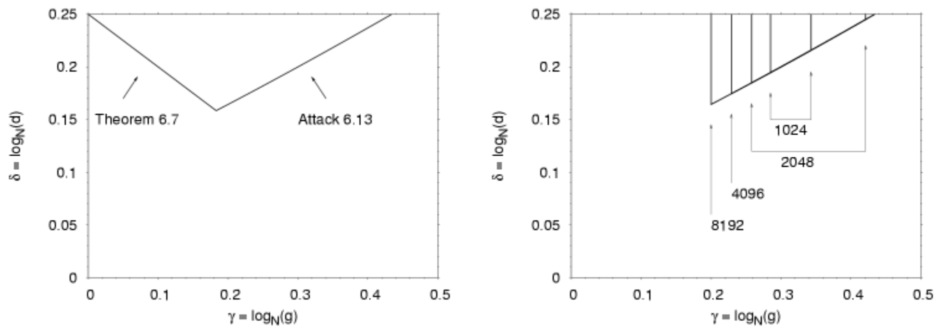
$$\gamma_{ecm} < \gamma < \frac{1}{2} - \frac{2\ell}{\log_2(N)}, \tag{6.33}$$

where the first inequality prevents factoring of $(N-1)/2$ with the ECM and the second prevents McKee and Pinch's factoring method from running with expected time less than 2^ℓ.

For the attacks on small private exponent, we will consider the two sizes of public exponents separately. When the public exponent satisfies $\alpha \approx 1-\gamma$ (*i.e.*, when the public exponent is simply computed as the inverse of the private exponent modulo $\lambda(N)$), the private exponent should satisfy

$$\delta > \begin{cases} \frac{1}{4} - \frac{\gamma}{2} - \log_N(\sqrt{24}) & \text{Theorem 6.7 (continued fractions)} \\ \gamma^2 - \epsilon & \text{Theorem 6.9 (lattice-based)} \\ \frac{2}{5}\gamma - \epsilon & \text{Attack 6.12 (lattice-based)} \\ 1 + \gamma - \frac{1}{4}\sqrt{4\gamma^2 + 20\gamma + 13} - \epsilon & \text{Attack 6.13 (lattice-based),} \end{cases}$$

to avoid the various attacks presented in this chapter. In Figure 6.1(a), we illustrate these bounds in the limit of large N. The region below any of the curves (shaded regions) are insecure. As can be seen, there is a significant region in which no known attacks exist and the private exponent is significantly below $N^{1/4}$. Of course, for a finite modulus size, the factoring attacks must be considered. In Figure 6.1(b), we include the bounds imposed by the factoring attacks for some common modulus sizes (1024, 2048, 4096 and 8192). Notice that the region of safe parameters[2] is significantly reduced as the size of the modulus decreases.



(a) $N \to \infty$ (b) Finite N

Figure 6.1: Unsafe private exponent choices when $\alpha \approx 1 - \gamma$. Plot (a) corresponds to the limit $N \to \infty$. Plot (b) corresponds to common finite modulus sizes 1024, 2048, 4096 and 8192.

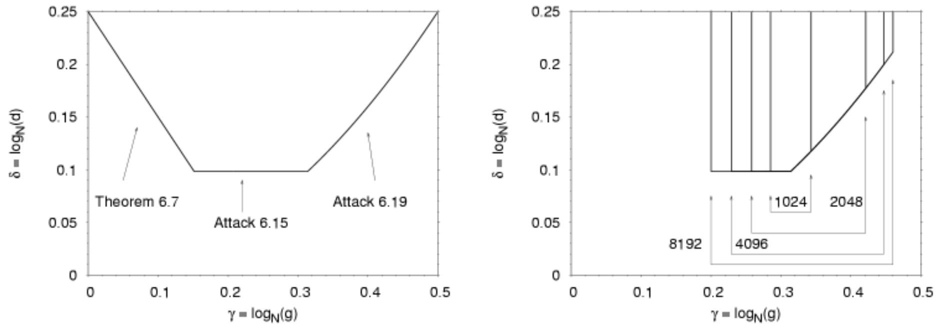
When the public exponent satisfies $\alpha \approx 1$, the private exponent should satisfy

$$\delta > \begin{cases} \frac{1}{4} - \gamma - \log_N(\sqrt{24}) & \text{Theorem 6.7 (continued fractions)} \\ \gamma^2 & \text{Theorem 6.9 (lattice-based)} \\ 1 - \frac{\sqrt{13}}{4} & \text{Attack 6.15 (Jochemsz and May),} \end{cases}$$

in order avoid the various attacks presented in this chapter. In Figure 6.2(a), we illustrate these bounds in the limit of large N. Again, the region below any of the curves (shaded regions) are insecure. As can be seen, when e is roughly the same size as the modulus N, the region in which there are no known attacks is significantly larger than when e is roughly the same size as $\lambda(N)$. In Figure 6.2(b), we show the bounds imposed by the factoring attacks for the same four common modulus sizes. While the region of safe parameters does decrease with decreasing modulus size (as in the previous case), notice that is still an appreciable region of parameters with no known attacks having private exponent smaller than $N^{1/4}$. In fact, for

[2]By safe parameters, we mean that no known attack exists.

2048-bit (or greater) modulus sizes, there is still a large region in which private exponents as small as about $N^{0.1}$ resist all known attacks. Recently, however, Jochemsz and May [70] developed a lattice-based attack that can recover small CRT-exponents (and hence small private exponents) which are smaller than $N^{0.073}$. While this bound is smaller than $N^{0.1}$, they have observed that the bound is pessimistic in practice when the public exponent is smaller than the modulus (the experimental bounds increase with decreasing public exponent sizes).



(a) $N \to \infty$ (b) Finite N

Figure 6.2: Unsafe private exponent choices when $\alpha \approx 1$. Plot (a) corresponds to the limit $N \to \infty$, plot (b) corresponds to common finite modulus sizes 1024, 2048, 4096 and 8192.

6.6 Discussion

In this chapter, we analyzed the security of Common Prime RSA with two different public exponent sizes. For instances generated from the key generation method in Algorithm 6.1, the public exponent is roughly the same size as $\lambda(N)$ since $e = d^{-1} \bmod \lambda(N)$ (thus, $\alpha \approx 1 - \gamma$). For these instances, private exponents that are somewhat smaller than $N^{1/4}$ can be used that resist all known attacks. For instances with public exponents $\alpha \approx 1$ (which corresponds to typical RSA with a small private exponent), private exponents significantly smaller than $N^{1/4}$ can be used that resist all known attacks. In fact, private exponents as small as roughly $N^{0.1}$ can be used.

Open Problems/Future Work: As suggested by an anonymous referee, it might be possible to improve the efficiency of the key generation algorithm using sieving techniques (rather then simply generating numbers with the hope that all the conditions for p, q, g and h are satisfied).

Another way to improve the efficiency of the key generation algorithm would be to relax the restriction that $h = 2gab + a + b$ be a prime. Since $h \approx N^{1/2+\gamma}$, if we assume that h is a random number then h will very likely have a large prime factor. If this prime factor is the same size as g (or larger) then the factoring attacks on $N - 1$ will be more efficient. Thus, we can significantly reduce the key generation time without decreasing the (known) security.

The analysis here (and by Jochemsz and May [69]) shows that instances of RSA with private exponents smaller than $N^{1/4}$ exist that resist all known attacks. It remains to be shown if any new attacks can be found on private exponents smaller than $N^{1/4}$, independent of the choice of parameters.

Chapter 7

Cryptanalysis of Dual RSA

In this chapter we analyze the security of Dual RSA, a variant of RSA designed to reduce the key storage requirements when two instances of RSA are required.

7.1 Background

There has been some work done in trying to reduce the storage requirements for RSA. In 1995, Vanstone and Zuccherato [119] proposed some methods to reduce the storage requirements of RSA moduli. One of these methods, using the high order bits of N to encode the user's name, was shown to be insecure by Coppersmith [32].

In 1998, Lenstra [71] presented some methods for generating n-bit RSA moduli that can be stored with only $n/2$ bits. The main idea has, apparently, been reinvented many times since at least 1984. We refer the reader to [71] for more detail.

In 2005, Lenstra and de Weger [73] introduced Twin RSA, which consists of pairs of RSA moduli differing by a fixed small even number such as ± 2. This allows the storage of only one RSA modulus for each pair that might be required.

In 2006, Sun, Wu, Ting and Hinek [114] introduced another variant of RSA called Dual RSA. Dual RSA is essentially two distinct instances of RSA that share the same public and private exponents. When two instances of RSA are required, Dual RSA allows for reduced key storage requirements since only one public and private exponent need to be stored.

7.2 Dual RSA

Dual RSA is essentially two distinct instances of RSA that share the same public and private exponents. Combining these two instances one obtains a single Dual RSA instance with public key (e, N_1, N_2) and private key (d, p_1, q_1, p_2, q_2), where e and d satisfy both

$$\begin{aligned} ed &\equiv 1 \pmod{\phi(N_1)} \\ ed &\equiv 1 \pmod{\phi(N_2)}. \end{aligned}$$

From these two relations, it follows that there exists positive integers k_1 and k_2 such that

$$\begin{aligned} ed &= 1 + k_1\varphi(N_1) = 1 + k_1(N_1 - \Lambda_1) \\ ed &= 1 + k_2\varphi(N_2) = 1 + k_2(N_2 - \Lambda_2). \end{aligned} \tag{7.1}$$

These are called the Dual RSA key equations or simply the key equations.

When small CRT-exponents are used, such as in Rebalanced- and Generalized Rebalanced-RSA (see Appendix B.1 and B.2), the Dual version has public key (e, N_1, N_2) and private key $(d_p, d_q, p_1, q_1, p_2, q_2)$, where e, d_p and d_q satisfy $ed_p \equiv 1 \pmod{(p_i - 1)}$ and $ed_q \equiv 1 \pmod{(q_i - 1)}$ for $i = 1, 2$. From these four relations, it follows that there exists four positive constants k_{p_1},k_{q_1}, k_{p_2} and k_{q_2} such that

$$\underbrace{\begin{aligned} ed_p &= 1 + k_{p_1}(p_1 - 1) \\ ed_q &= 1 + k_{q_1}(q_1 - 1) \end{aligned}}_{\text{for } N_1} \quad , \quad \underbrace{\begin{aligned} ed_p &= 1 + k_{p_2}(p_2 - 1) \\ ed_q &= 1 + k_{q_2}(q_2 - 1) \end{aligned}}_{\text{for } N_2} \quad , \tag{7.2}$$

which are called the Dual RSA-CRT equations, or simply the CRT equations.

To simplify notation, we will simply use

$$\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\},$$

to denote a valid public/private key pair for schemes I and II and

$$\{(e, N_1, N_2), (d_p, d_q, p_1, q_1, p_2, q_2)\},$$

to denote a valid public/private key pair for scheme III. In addition, we assume that both moduli, N_1 and N_2, are comprised of balanced primes.

There are three schemes (or variants) of Dual RSA presented in [114], which correspond to different variants of RSA. Below, we describe the schemes and then present some attacks for each.

Throughout the remainder of the chapter, we will assume that N_1, N_2 and $N = (N_1 + N_2)/2$ are all n-bit integers, unless otherwise stated. Also, we use real $\alpha \leq 0$ to represent the size of the public exponent ($e = N^{\alpha}$) and $\delta \geq 0$ to represent the size of the private exponent ($d = N^{\delta}$) or CRT-exponents ($d_p, d_q < N^{\delta}$).

7.2.1 Small Public Exponent Dual RSA (Scheme I)

The first scheme is Dual RSA with small public exponent, called Dual RSA-Small-e or simply Scheme I. The key generation algorithm takes (n_e, n) as input, with $n_e < n/2$, and outputs a valid public/private key pair with an n_e-bit public exponent $e = n^{\alpha}$ and two n-bit moduli. The private exponent d will, with very high probability, have the same bitlength as the moduli. The key generation method is restated in Algorithm 7.1.

Algorithm 7.1 Key Generation : Small Public Exponent Dual RSA.

```
Input: $(n_e, n)$ such that $n_e < n/2$.
1: Randomly select an $n_e$-bit integer $x_1$ and an $(n/2 - n_e)$-bit integer $x_2$
   such that $p_1 = x_1x_2 + 1$ is prime.
2: Randomly select an $(n/2 - n_e)$-bit integer $y_2$ such that $p_2 = x_2y_2 + 1$ is
   prime.
3: Randomly select an $n_e$-bit integer $y_1$ such that $q_1 = y_1y_2 + 1$ is prime.
4: Randomly select an $n_e$-bit integer $e$ such that $\gcd(x_1x_2y_1y_2, e) = 1$.
   Compute $d$ and $k_1$ satisfying $ed = 1 + k_1(p_1 - 1)(q_1 - 1)$.
5: If $q_2 = k_1x_2 + 1$ is not prime then go back to step 4.
6: Let $N_1 = p_1q_1$, $N_2 = p_2q_2$, and $k_2 = y_1$.
Output: The public key $(e, N_1, N_2)$ and the private key $(d, p_1, q_1, p_2, q_2)$.
```

From the key generation algorithm, we see that $N_1 = p_1q_1$ and $N_2 = p_2q_2$, where $p_1 = x_1x_2 + 1$, $q_1 = y_1y_2 + 1$, $p_2 = x_1y_2 + 1$, $q_2 = k_1x_2 + 1$ and $y_1 = k_2$. This allows us to write $N_1 - 1$ and $N_2 - 1$ as the following system of equations with five unknowns:

$$\begin{aligned} N_1 - 1 &= x_1x_2k_2y_2 + x_1x_2 + k_2y_2 \\ N_2 - 1 &= x_1x_2k_1y_2 + k_1x_2 + x_1y_2, \end{aligned}$$

where x_1, k_1 and k_2 are n_e-bit numbers and x_2 and y_2 are $(n/2 - n_e)$-bit numbers. In particular, we have $|x_1|, |k_1|, |k_2| \leq N^{\alpha}$ and $|x_2|, |y_2| \leq N^{1/2-\alpha}$.

Instances of RSA with public exponents as small as $e = 3$ are considered safe when used properly. In Section 7.3, we show that this is not the case for Dual RSA. In particular, Dual RSA is insecure when public exponents smaller than (roughly) $N^{1/4}$ are used.

7.2.2 Small Private Exponent Dual RSA(Scheme II)

The second scheme is Dual RSA with small private exponent, called Dual RSA-Small-d or simply Scheme II. The key generation algorithm for this scheme is in fact identical to Algorithm 7.1, except that the roles of the public and private exponent are interchanged. Thus, the key generation algorithm takes (n_d, n) as input, with $n_d < n/2$, and outputs a valid public/private key pair with an n_d-bit private exponent $d = N^\delta$ and two n-bit moduli. The public exponent e will, with very high probability, have the same bitlength as the moduli.

Instances of RSA with private exponents smaller than $N^{0.292}$ are considered unsafe (*cf.* Boneh and Durfee's small private exponent attack [13]). In Section 7.4, we show that this bound is increased to $N^{0.333}$ for Dual RSA.

7.2.3 Dual Generalized Rebalanced-RSA (Scheme III)

The third, and last, scheme is an extension of Generalized Rebalanced-RSA to the Dual RSA setting, called Dual Generalized Rebalanced-RSA or simply Scheme III. The key generation algorithm takes (n_e, n_d, n_k, n) as input, with $n_e < n/2$ and $n_e + n_d = n/2 + n_k$, and outputs a valid public/private key pair with an n_e-bit public exponent ($e = N^\alpha$), two n_d-bit CRT-exponents ($d_p, d_q < N^\delta$) and two n-bit moduli (N_1 and N_2). The value n_k is a security parameter which is the bitlength of the constants k_{p_i} and k_{q_i} from (7.2). Thus, we have $|n_k| < N^{\alpha+\delta-1/2}$. The key generation algorithm is restated in Algorithm 7.2 (page 137).

From the key generation algorithm, notice that the Dual RSA moduli N_1 and N_2 satisfy

$$\begin{aligned} N_1 &= p_1 q_1 & N_2 &= p_2 q_2 \\ p_1 &= p_a p_b + 1 & p_2 &= p_{a'} p_b + 1 \\ q_1 &= q_a q_b + 1 & q_2 &= q_{a'} q_b + 1 \\ p_a &= p_{a_1} p_{a_2} \cdots p_{a_{(k-1)}} p_{a_k} & p_{a'} &= p_a k_{p_1} / p_{a_{i'}} \\ q_a &= q_{a_1} q_{a_2} \cdots q_{a_{(k-1)}} q_{a_k} & q_{a'} &= q_a k_{q_1} / q_{a_{j'}}, \end{aligned} \tag{7.3}$$

for some i' and j', where $k = \lceil (n/2 - n_e)/n_k \rceil$; p_{a_i}, q_{a_i}, k_{p_1}, and k_{q_1} are all n_k-bit numbers for all i; and p_b and q_b are n_e-bit numbers. In particular, we have

$$|p_{a_i}|, |q_{a_i}|, |k_{q_1}|, |k_{p_1}| < N^{\alpha+\delta-1/2},$$

for all i, and

$$|p_b|, |q_b| < N^\alpha.$$

In Section 7.5, we present some attacks on Dual Generalized Rebalanced-RSA. It is shown that the number of insecure parameters (public and CRT-exponents) is increased when using Dual Generalized Rebalanced-RSA, as compared to using two independent instances of Generalized Rebalanced-RSA.

Algorithm 7.2 Key Generation : Dual Generalized Rebalanced-RSA

```
Input: $(n_e, n_d, n_k, n)$ such that $n_e < n/2$ and $n_e + n_d = n/2 + n_k$.
1: Randomly select an $n_e$-bit integer $e$ and set $k$ to be the smallest integer larger than $(n/2 - ne)/n_k$ (i.e., $k = \lceil (n/2 - ne)/n_k \rceil$).
2: Randomly select $k-1$ $n_k$-bit integers $p_{a_1}, \ldots, p_{a_{(k-1)}}$ and an even integer $p_{a_k}$ such that $p_a = p_{a_1} \cdots p_{a_{(k-1)}} p_{a_k}$ has bitlength $(n/2 - ne)$ and $\gcd(e, p_a) = 1$.
3: Randomly select an $n_k$-bit integer $k_{p_1}$ such that $\gcd(e, k_{p_1}) = 1$.
4: Compute $d_p$ and $p_b$ such that $ed_p = (k_{p_1} p_a) p_b + 1$, where $e < p_b < 2e$ and $k_{p_1} p_a < d_p < 2k_{p_1} p_a$. If $p_1 = p_a p_b + 1$ is not prime then go back to step 3.
5: If $(k_{p_1} p_a p_b / p_{a_{i'}}) + 1$ is prime for some $1 \le i' \le k-1$ then let $p_2 = (k_{p_1} p_a p_b / p_{a_{i'}}) + 1$. Otherwise, go back to step 3.
6: Randomly select $k-1$ $n_k$-bit integers $q_{a_1}, \ldots, q_{a_{(k-1)}}$ and an even integer $q_{a_k}$ such that $q_a = q_{a_1} \cdots q_{a_{(k-1)}} q_{a_k}$ has bitlength $(n/2 - n_e)$ and $\gcd(e, q_a) = 1$.
7: Randomly select an $n_k$-bit integer $k_{q_1}$ such that $\gcd(e, k_{q_1}) = 1$.
8: Compute $d_q$ and $q_B$ such that $ed_q = (k_{q_1} q_a) q_b + 1$, where $e < q_b < 2e$ and $k_{q_1} q_a < d_q < 2k_{q_1} q_a$ If $q_1 = q_a q_b + 1$ is not prime then go back to Step 7.
9: If $(k_{q_1} q_a q_b / q_{a_{j'}}) + 1$ is prime for some $1 \le j' \le k-1$ then let $q_2 = (k_{q_1} q_a q_b / q_{a'_j}) + 1$. Otherwise, go back to step 7.
10: Let $N_1 = p_1 q_1$, $N_2 = p_2 q_2$, $k_{p_2} = p_{a_{i'}}$ and $k_{q_2} = q_{a_{j'}}$.
Output: The public key $(e, N_1, N_2)$ and the private key $(d_p, d_q, p_1, q_1, p_2, q_2)$.
```

7.3 Cryptanalysis of Small Public Exponent Dual RSA

In this section we consider the security of Dual RSA when the public key e is small ($e < N^{1/2}$) and the private key d is large (likely the size of the moduli). That is, we consider the security of Scheme I.

Many of the attacks on Scheme I exploit the special structure of the Dual RSA moduli N_1 and N_2. In particular, recall that

$$\begin{array}{rcl} N_1 - 1 & = & x_1 x_2 k_2 y_2 + x_1 x_2 + k_2 y_2 \\ N_2 - 1 & = & x_1 x_2 k_1 y_2 + k_1 x_2 + x_1 y_2, \end{array} \tag{7.4}$$

where x_1, k_1 and k_2 are n_e-bit numbers and x_2 and y_2 are $(n/2 - n_e)$-bit numbers. More specifically, $|x_1|, |k_1|, |k_2| \leq N^{\alpha}$ and $|x_2|, |y_2| \leq N^{1/2-\alpha}$.

7.3.1 Very Small Public Exponent Attack

When the Dual RSA public exponent is very small we can factor the moduli with a simple brute force attack. The main result is in the following attack.

Theorem 7.1. *For any integer $\ell > 0$, let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with public exponent $e = N^{\alpha}$. If the public exponent satisfies*

$$\alpha < \frac{\ell}{3 \log_2(N)}, \tag{7.5}$$

then the both N_1 and N_2 can be factored in time polynomial in $\log_2(N)$ and linear in 2^{ℓ}.

Proof: We begin by noticing that three of the unknown values in (7.4), the equations for $N_1 - 1$ and $N_2 - 1$, are n_e-bit numbers. In particular, we have $|x_1|, |k_1|, |k_2| \leq N^{\alpha}$. We simply perform an exhaustive search on the values of x_1, k_1 and k_2. For each guess, the triple $(x_1', {k_1}', {k_2}')$, is tested by substituting the values into (7.4) and solving the remaining system for x_2' and y_2'. If x_2' and y_2' are integers and $p_1' = x_1' x_2' + 1$, $q_1' = y_1' y_2' + 1$, $p_2' = x_1' y_2' + 1$ and $q_2' = k_1' x_2' + 1$ are all primes then $(x_1', x_2', y_2', {k_1}', {k_2}') = (x_1, x_2, y_2, k_1, k_2)$ and we have successfully factored N_1 and N_2. Since $\alpha < \ell/(3\log_2(N))$, we only need to test at most $N^{3\alpha} = 2^{\ell}$ triples. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❑

Using 1024-bit Dual RSA moduli and a search space of 2^{80}, the bound in (7.5) becomes $\alpha < 5/192 \approx 0.026$, which corresponds to public exponents

$e < 2^{27}$. Thus, for 1024-bit moduli, we expect the brute force method of Theorem 7.1 to be able to factor both moduli faster than simply factoring one of the moduli using the best known factoring techniques (NFS) when using a public exponent $e < 2^{27}$.

The commonly used small public exponents for RSA, such as $e = 3$ and $e = 2^{16}+1$, are completely insecure in the Dual RSA setting for any modulus size. When $e = 2^{16}+1$, the size of the search space is only 2^{48}. When $e = 3$, the size of the search space is less than 10.

7.3.2 Finding ${k_1}'$ and ${k_2}'$

Recall that k_1 and k_2 are the constants in the key equations $ed = 1+k_1\varphi(N_1)$ and $ed = 1 + k_2\varphi(N_2)$. In this section we present two attacks that can recover ${k_1}' = k_1/\gcd(k_1, k_2)$ and ${k_2}' = k+2/\gcd(k_1, k_2)$. These attacks are important, since knowledge of ${k_1}'$ and ${k_2}'$ lead to other attacks, shown in Sections 7.3.3 and 7.3.4, that can factor the Dual RSA moduli.

Continued Fraction Method

The first method of obtaining ${k_1}'$ and ${k_2}'$ uses continued fractions. The main result is in the following theorem.

Theorem 7.2. *Let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with n-bit moduli and public exponent $e = N^\alpha$ and let k_1 and k_2 be the constants in the key equations. If $n > 14$ and*

$$\alpha < \frac{1}{4} - \log_N(19), \tag{7.6}$$

then we can generate a list that contains k_2/k_1, in lowest terms, in time polynomial in $\log_2(N)$. The size of the list is also polynomial in $\log_2(N)$.

Proof: Let the bitlength of the moduli be greater than 14 (*i.e.*, $n > 14$). Starting with (7.4), the equations for $N_1 - 1$ and $N_2 - 1$, observe that

$$\frac{N_1 - 1}{x_1x_2k_1y_2} = \frac{k_2}{k_1} + \frac{x_1x_2 + k_2y_2}{x_1x_2k_1y_2}, \tag{7.7}$$

and

$$\frac{N_2 - 1}{x_1x_2k_1y_2} = 1 + \frac{k_1x_2 + x_1y_2}{x_1x_2k_1y_2}. \tag{7.8}$$

Letting $A = (x_1x_2 + k_2y_2)/(x_1x_2k_1y_2)$ and $B = (k_1x_2 + x_1y_2)/(x_1x_2k_1y_2)$, we then have

$$\frac{N_1 - 1}{x_1x_2k_1y_2} = \frac{k_2}{k_1} + A, \tag{7.9}$$

$$\frac{N_2 - 1}{x_1x_2k_1y_2} = 1 + B. \tag{7.10}$$

Since x_1, k_1 and k_2 are n_e-bit integers and x_2 and y_2 are $(n/2 - n_e)$-bit numbers, it follows that A satisfies

$$\begin{aligned} |A| &= \left|\frac{x_1x_2 + k_2y_2}{x_1x_2k_1y_2}\right| \\ &< \frac{(2^{n/2})(2^{n/2-n_e}) + (2^{n/2})(2^{n/2-n_e})}{(\frac{1}{2}2^{n/2-n_e})(\frac{1}{2}2^{n_e})(\frac{1}{2}2^{n/2-n_e})(\frac{1}{2}2^{n_e})} \\ &= \frac{2\times2^{n/2}}{(\frac{1}{2})^4\times2^{n/2}\times2^{n/2}} \\ &= \frac{2^5}{2^{n/2}}, \end{aligned} \tag{7.11}$$

and similarly B satisfies

$$\begin{aligned} |B| &= \left|\frac{k_1x_2 + x_1y_2}{x_1x_2k_1y_2}\right| \\ &< \frac{(2^{n/2})(2^{n/2-n_e}) + (2^{n/2})(2^{n/2-n_e})}{(\frac{1}{2}2^{n/2-n_e})(\frac{1}{2}2^{n_e})(\frac{1}{2}2^{n/2-n_e})(\frac{1}{2}2^{n_e})} \\ &= \frac{2^5}{2^{n/2}}. \end{aligned} \tag{7.12}$$

Since $n > 14$, by assumption, we have that $|A|, |B| < 1$, or simply $A, B < 1$ as both are positive numbers. Using this bound for B, notice that (7.9) divided by (7.10) can be written as

$$\begin{aligned} \frac{N_1 - 1}{N_2 - 1} &= \frac{\frac{k_2}{k_1} + A}{1 + B} \\ &= \left(\frac{k_2}{k_1} + A\right)\left(1 - B + B^2 - B^3 + \cdots\right) \\ &= \frac{k_2}{k_1} + A - \left(\frac{k_2}{k_1} + A\right)\left(B - B^2 + B^3 - \cdots\right), \end{aligned}$$

so that

$$\frac{N_1 - 1}{N_2 - 1} - \frac{k_2}{k_1} = A - \left(\frac{k_2}{k_1} + A\right)\left(B - B^2 + B^3 - \cdots\right). \tag{7.13}$$

Since N_1 and N_2 have the same bitlength, we know that $1/2 < N_1/N_2 < 2$, which implies that $1/2 < k_2/k_1 < 2$. To see this, recall that in Dual RSA, $ed - 1 = k_1(N_1 - \Lambda_1) = k_2(N_2 - \Lambda_2)$. Using this bound for k_2/k_1, and $|A| < 1$, we see that

$$\begin{aligned}\left|\frac{k_2}{k_1} + A\right| &\le \left|\frac{k_2}{k_1}\right| + |A| \\ &< 3. \end{aligned} \tag{7.14}$$

Taking the absolute value of both sides of (7.13), repeatedly using the triangle inequality, and using the bound shown above gives

$$\begin{aligned}\left|\frac{N_1 - 1}{N_2 - 1} - \frac{k_2}{k_1}\right| &= \left|A - \left(\frac{k_2}{k_1} + A\right)\left(B - B^2 + B^3 - \cdots\right)\right| \\ &\le |A| + 3\left|B - B^2 + B^3 - \cdots\right| \\ &\le |A| + 3|B| + 3\left(\left|B^2\right| + 3\left|B^4\right| + \cdots\right). \end{aligned} \tag{7.15}$$

To simplify the right-hand side of this inequality, notice that since $n > 14$ we have

$$\begin{aligned}|B^2| + |B^3| + |B^4| + \cdots &= \sum_{i=2}^{\infty} |B|^i \\ &< \sum_{i=2}^{\infty} \left|\frac{2^5}{2^{n/2}}\right|^i \\ &= \frac{2^{n/2+10}}{2^n\left(2^{n/2} - 2^5\right)} \\ &< \frac{1}{3} \times \frac{2^5}{2^{n/2}}, \end{aligned} \tag{7.16}$$

so that (7.15) simplifies to

$$\begin{aligned}\left|\frac{N_1-1}{N_2-1}-\frac{k_2}{k_1}\right| &\leq |A|+3\,|B|+3\left(\left|B^2\right|+3\left|B^4\right|+\cdots\right)\\ &< \frac{2^5}{2^{n/2}}+3\times\frac{2^5}{2^{n/2}}+\frac{2^5}{2^{n/2}}\\ &< 5\times\frac{2^5}{2^{n/2}}\\ &< \frac{160}{\sqrt{N}}. \qquad (7.17)\end{aligned}$$

Letting $\alpha < 1/4 - \log_N(19)$, we have $k_1 < N^{1/4}/19$ since $|k_1| < N^{\alpha}$ and k_1 is positive. Using this bound for k_1, we see that

$$\begin{aligned}\left|\frac{N_1-1}{N_2-1}-\frac{k_2}{k_1}\right| &< \frac{160}{\sqrt{N}}\\ &< \frac{1}{2{k_1}^2}. \qquad (7.18)\end{aligned}$$

Therefore, from Theorem 2.1 (Continued Fractions), we know that k_2/k_1, in lowest terms, will be one of the convergents in the continued fraction expansion of $(N_1-1)/(N_2-1)$. Simply computing each convergent generates a list of size polynomial in $\log_2(N)$ that will contain k_2/k_1 in lowest terms. Since all computations are polynomial in $\log_2(N)$, the result follows. ❑

Applying Verheul and van Tilborg's extension of Wiener's attack (see [120]) or Dujella's refinement [43], adding an exhaustive search of size 2^{ℓ} to each convergent allows us to increase the bound in Theorem 7.2 by $\ell/2$. This also increases both the size of the list and the time needed to compute the list. The result is given in the following corollary.

Corollary 7.3. *Let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with public exponent $e = N^{\alpha}$ and let k_1 and k_2 be the constants in the key equations. If the bitlength of the moduli is greater than* 14 *and*

$$\alpha < \frac{1}{4}+\frac{\ell}{2}-\log_N(19), \qquad (7.19)$$

then we can generate a list that constraints k_2/k_1, in lowest terms, in time polynomial in $\log_2(N)$ *and linear in* 2^{ℓ}. *The size of the list is also polynomial in the* $\log_2(N)$ *and linear in* 2^{ℓ}.

Using 1024-bit moduli and search space of 2^{80}, a list containing k_2/k_1, in lowest terms, can be generated, provided that the public exponent has bitlength less than 332.

Lattice-Based Method

We now show that ${k_1}'$ and ${k_2}'$ can be obtained from a lattice-based method. The main result is the following attack.

Attack 7.4. *For every $\epsilon > 0$ there exists N_0 such that for every $N_1, N_2 > N_0$ the following holds: Let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with public exponent $e = N^\alpha$ and private exponent $d = N^\delta$. Let k_1 and k_2 be the constants in the key equations and let ${k_1}' = k_1/\gcd(k_1, k_2)$ and ${k_2}' = k_2/\gcd(k_1, k_2)$. If the public and private exponents satisfy*

$$\alpha + \delta < \frac{5}{4} + \frac{\ell}{2\log_2(N)} - \epsilon, \tag{7.20}$$

then ${k_1}'$ and ${k_2}'$ can be computed in time polynomial in $\log_2(N)$ and linear in 2^ℓ, provided that Assumption 2.13 holds.

Justification: In the following, let $m = \log_2(N)$. Notice that dividing the difference between the key equations, $ed = 1 + k_1(N_1 - \Lambda_1)$ and $ed = 1 + k_2(N_2 - \Lambda_2)$, by $\gcd(k_1, k_2)$ gives

$${k_1}'(N_1 - \Lambda_1) = {k_2}'(N_2 - \Lambda_2). \tag{7.21}$$

Let k_2 be a known approximation of ${k_2}'$ such that ${k_2}' = k_2 + {k_2}''$, where $|{k_2}''| < N^{\alpha+\delta-1-\ell/m}$. Thus, the ℓ most significant bits of k_2 and ${k_2}'$ are the same. Substituting this representation for k_2 in (7.21), and rearranging, we obtain

$${k_1}'N_1 - {k_2}''N_2 + (k_2\Lambda_2 + {k_2}''\Lambda_2 - {k_1}'\Lambda_1) - k_2N_2 = 0, \tag{7.22}$$

which suggests we look for small solutions of the polynomial

$$f(x, y, z) = N_1x - N_2y + z - k_2N_2, \tag{7.23}$$

since $(x_0, y_0, z_0) = ({k_1}', {k_2}'', k_2\Lambda_2 + {k_2}''\Lambda_2 - {k_1}'\Lambda_1)$ is a root of $f(x, y, z)$. Defining the bounds

$$\begin{aligned} X &= N^{\alpha+\delta-1}, \\ Y &= N^{\alpha+\delta-1-\ell/m}, \\ Z &= N^{\alpha+\delta-1/2}, \end{aligned} \tag{7.24}$$

we have that $|x_0| \leq X$, $|y_0| \leq Y$ and $|z_0| \leq Z$. From Theorem 2.12, noting that

$$\begin{aligned} W &= \|f(xX, yY, zZ)\|_\infty \\ &= \max\left(|N_1 X|, |N_2 Y|, |Z|, |k_2 N_2|\right) \\ &= |N_1 X| \\ &= N^{\alpha+\delta}, \end{aligned} \tag{7.25}$$

we can recover (x_0, y_0, z_0) whenever $XYZ < W$, provided that N is sufficiently large and Assumption 2.13 holds. The bound $XYZ < W$ is satisfied when $3\alpha + 3\delta - 5/2 - \ell/m < \alpha + \delta$, or simply $\alpha + \delta < 5/4 + \ell/(2m)$. The ϵ term is added to correct for all lower order terms ignored in the methods implicit in Theorem 2.12. Once x_0 and y_0 are known, we know $k_1{}'$ and $k_2{}'$ since $k_1{}' = x_0$ and $k_2{}' = k_2 + y_0$.

The arguments above relied on the knowledge of k_2. Since k_2 is not known, we carry out an exhaustive search performing the lattice-based method above for each guess. Since we require that the ℓ most significant bits of k_2 and $k_2{}'$ must agree, we must try at most 2^ℓ guesses. Since all computations, including lattice-basis reduction and integer root finding, require time that is polynomial in m, the result follows. ❏

The size of the private exponent in this scheme is, with very high probability, the same size as the moduli (*i.e.*, $\delta \approx 1$). Using this approximation, this lattice based method is expected to work whenever

$$\alpha < \frac{1}{4} + \frac{\ell}{2\log_2(N)}, \tag{7.26}$$

which is very similar to the bound obtained using continued fractions with an exhaustive search for each convergent.

Using 1024-bit moduli and search space of 2^{80}, it is expected that $k_1{}'$ and $k_2{}'$ can be computed for public exponents with bitlengths approaching 296.

7.3.3 Small Public Exponent Attack with k_1 and k_2

Here we show that knowledge of k_1 and k_2 leads to a simple attack if the bitlength of the public exponent is very small or very close to $n/2$. The main result is the following theorem.

Theorem 7.5. *Let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with public exponent $e = N^\alpha < N^{1/2}$ and let k_1 and k_2 be the constants in the key equations. If k_1 and k_2 are known then N_1 and N_2 can be factored in time polynomial in $\log_2(N)$ and linear in $\min\{e, N^{1/2}/e\}$.*

Proof: Let k_1 and k_2 be known. Notice that the system of equations for $N_1 - 1$ and $N_2 - 1$ given by (7.4) is now reduced to only three unknown variables: x_1, x_2, and y_2. If n_e is smaller than $n/2 - n_e$, we perform an exhaustive search for x_1 and solve the resulting system of two unknowns. Since $|x_1| < N^\alpha = e$, we need at most e values for x_1. Similarly, if $n/2 - n_e$ is smaller than n_e, we perform a brute force search on x_2 (or y_2) and solve the resulting system of two unknowns. Since both x_2 and y_2 are bounded in size by $N^{1/2-\alpha}$, we need at most $\sqrt{N}/e$ values for either value. In both scenarios, for each guess (of x_1, x_2 or y_2) we solve the resulting system for the remaining unknowns and test if the solution yields the desired factorization of N_1 and N_2. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

Using 1024-bit moduli and allowing an exhaustive search of size 2^{80}, the moduli can be factored if the bitlength of the public exponent is less than 80 or greater than 432.

This attack assumes that k_1 and k_2 are known. The attacks in the previous two subsections, however, only recover ${k_1}'$ and ${k_2}'$. If we make the assumption that k_1 and k_2 behave like random numbers, then with high probability $k = \gcd(k_1, k_2)$ will be very small. Given ${k_1}'$ and ${k_2}'$, we simply perform an exhaustive search for k. For each guess k', we compute candidates $k'{k_1}'$ and $k'{k_2}'$ for k_1 and k_2 and try to factor N_1 and N_2 with the attack in Attack 7.5.

7.3.4 Lattice-Based Attack with ${k_1}'$ and ${k_2}'$

The simple brute force attack of the previous section required knowledge of both k_1 and k_2. In this section, we present a lattice-based attack that can be used to factor the Dual RSA moduli given ${k_1}'$ and ${k_2}'$, which is the information gained from Theorem 7.2 and Attack 7.4. The main result is the following attack.

Attack 7.6. *For every $\epsilon > 0$ there exists N_0 such that for every $N_1, N_2 > N_0$ the following holds: Let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with public exponent $e = N^\alpha < N^{1/2}$ and private exponent $d =$*

N^{δ}. Let k_1 and k_2 be the constants in the key equations and let ${k_1}' = k_1/k$ and ${k_2}' = k_2/k$, where $k = \gcd(k_1, k_2) = N^{\gamma}$. Given ${k_1}'$ and ${k_2}'$, if

$$\alpha + \delta > 1 + \gamma - \epsilon, \tag{7.27}$$

then N_1 and N_2 can be factored in time polynomial in $\log_2(N)$ provided that N is sufficiently large and Assumption 2.13 holds.

Justification: Let ${k_1}'$ and ${k_2}'$ be known. Recall that dividing the difference between the key equations, $ed = 1 + k_1(N_1 - \Lambda_1)$ and $ed = 1 + k_2(N_2 - \Lambda_2)$, by $\gcd(k_1, k_2)$ yields

$$k_1'(N_1 - \Lambda_1) = k_2'(N_2 - \Lambda_2), \tag{7.28}$$

where Λ_1 and Λ_2 are the only unknowns in this case. Since $\gcd({k_1}', {k_2}') = 1$, we can reduce this equation modulo ${k_2}'$ to obtain $\Lambda_1 \equiv N_1 \pmod{{k_2}'}$. Letting $\sigma_1 = N_1 \bmod {k_2}'$, we can write $\Lambda_1 = \sigma_1 + \tau_1 {k_2}'$, where τ_1 is the only unknown part. Substituting Λ_1 in (7.28) we obtain

$$k_1'(N_1 - \sigma_1 - \tau_1 k_2') = k_2'(N_2 - \Lambda_2), \tag{7.29}$$

which suggests that we look for small solutions, modulo N_1, of the polynomial

$$f_{N_1}(x, y) = k_1' k_2' x - k_2' y + k_2' N_2 - k_1' \sigma_1,$$

since $(x_0, y_0) = (\tau_1, \Lambda_2)$ is a root of $f_{N_1}(x, y)$ modulo N_1. Since $|\Lambda_1| = N^{1/2}$ and $|{k_1}'| = N^{\alpha+\delta-1-\gamma}$, we have $|\tau_1| = N^{1/2-(\alpha+\delta-1-\gamma)}$. Defining the bounds

$$\begin{aligned} X &= N^{3/2-\alpha-\delta+\gamma} \\ Y &= 3N^{1/2}, \end{aligned} \tag{7.30}$$

notice that $|x_0| \le X$ and $|y_0| \le Y$. From Theorem 2.10, for sufficiently large N_1, we can compute $(x_0, y_0) = (\tau_1, \Lambda_2)$ provided that $XY < N_1$ and Assumption 2.13 holds. Ignoring all terms that do not depend on N, this inequality is satisfied whenever $(3/2 - \alpha - \delta + \gamma) + 1/2 < 1$. Thus, we obtain the sufficient condition

$$\alpha + \delta > 1 + \gamma - \epsilon,$$

where ϵ is added to correct for all lower order terms ignored in the methods implicit in Theorem 2.10 and for the factor of 3 in Y. Once τ_1 and Λ_2 are known we can easily compute $\varphi(N_1)$ and $\varphi(N_2)$ which then allow us to factor N_1 and N_2. Since all computations can be done in time polynomial in

$\log_2(N)$, the result follows. ❑

Since the size of the public exponent is smaller than $N^{1/2}$ we expect, with high probability, that the private exponent will be roughly the same size as the moduli. Thus, with this assumption, the condition in Attack 7.6 simplifies to

$$\alpha > \gamma - \epsilon.$$

Therefore, when the public exponent is greater than $\gcd(k_1, k_2)$, it is expected that the moduli can be factored given k_1' and k_2' when the moduli are sufficiently large.

7.4 Cryptanalysis of Small Private Exponent Dual RSA

In this section we consider the security of Dual RSA when the private key d is small ($d < N^{1/2}$) and public key e is large (likely the size of the moduli).

All of the known small private exponent attacks on RSA also apply to Dual RSA (scheme II). These include Wiener's continued fraction attack [123], Boneh and Durfee's lattice-based attack [14] and Blömer and May's lattice-based attack [7]. We summarize the strongest results, by Boneh and Durfee, here. From [14, Section 6], we see that, in general, RSA is considered unsafe when the size of the private exponent satisfies

$$\delta < \frac{7}{6} - \frac{1}{3}\sqrt{1+6\alpha} - \epsilon. \tag{7.31}$$

When the public exponent is the same size as the modulus a stronger result is known, see Boneh and Durfee [14, Section 5], which shows that RSA is unsafe when the size the of private exponent satisfies

$$\begin{aligned} \delta &< 1 - \frac{1}{\sqrt{2}} - \epsilon \\ &\approx 0.292 - \epsilon. \end{aligned} \tag{7.32}$$

In addition to these lattice-based attacks, notice that each of the attacks on Scheme I from Section 7.3 can also be mounted against Scheme II. Thus, Dual RSA with small private exponent should be considered insecure when the size of the private (and public) exponent satisfies

$$\delta > \gamma, \tag{7.33}$$

and

$$\delta + \alpha < \frac{5}{4} + \frac{\ell}{2m}, \tag{7.34}$$

where $\gamma = \log_N(\gcd(k_1, k_2))$ and $m = \log_2(N)$. These last conditions correspond to mounting the attack in Attack 7.4 to obtain $k_1{}'$ and $k_2{}'$ for use in the attack in Attack 7.6. As the size of the public exponent decreases, this attack is stronger than the attack behind (7.31). Below, we present a new heuristic attack on Scheme II that is stronger than all of these attacks.

7.4.1 Lattice-Based Attack

The attack on Dual RSA with small private exponent that we present is only a heuristic. Since it is only a heuristic, we provide an argument for the validity of the attack instead of a proof. The main result is given in the following attack.

Attack 7.7. *Let $\{(e, N_1, N_2), (d, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual RSA with small private exponent $d = N^\delta < N^{1/2}$ and large public exponent $e = N^\alpha$. If the private and public exponents satisfy*

$$\delta + \frac{2}{3}\alpha < 1, \tag{7.35}$$

then there is a heuristic lattice-based algorithm that recovers the private exponent d in time polynomial in $\log_2(N)$.

Justification: The justification is similar to that for Attack 4.1. Essentially, we construct a lattice with a known small vector that reveals the private exponent d. If that small vector is also a smallest vector in the lattice then we hope that it can be recovered with the LLL algorithm.

Given the Dual RSA public key (e, N_1, N_2), consider the following three equations:

$$\begin{array}{lllcl} Ad & & & = & Ad \\ ed & -k_1N_1 & & = & 1 + k_1\Lambda_1 \\ ed & & -k_2N_2 & = & 1 + k_2\Lambda_2. \end{array} \tag{7.36}$$

This is simply the key equations along with the trivial equation $Ad = Ad$. Letting

$$\mathcal{B} = \begin{bmatrix} A & e & e \\ 0 & -N_1 & 0 \\ 0 & 0 & -N_2 \end{bmatrix}, \tag{7.37}$$

we can write the system of equations (7.36) as the vector-matrix equation

$$(d, k_1, k_2)\mathcal{B} = (Ad, 1 - k_1\Lambda_1, 1 - k_2\Lambda_2). \tag{7.38}$$

Letting $v = (Ad, 1 - k_1\Lambda_1, 1 - k_2\Lambda_2)$ we can force each component of v to be of similar size by letting $A = eN^{-1/2} = N^{\alpha-1/2}$. Thus, each component of v is now bound by $N^{\alpha+\delta-1/2}$, and so $\|v\|_2 < \sqrt{3}\,N^{\delta+\alpha-1/2}$. The hope is that v will be the smallest vector in the lattice $\mathcal{L}$ generated by the rows in $\mathcal{B}$. If this is true then we can easily compute the private exponent d by simply finding the smallest vector in $\mathcal{L}$ and dividing the first component by $A = eN^{-1/2}$.

From Theorem 2.3 (Minkowski), we know that a smallest vector in $\mathcal{L}$ is bounded in size by $\sqrt{3}\det(\mathcal{L})^{1/3} = \sqrt{3}\,N^{(\alpha+3/2)/3}$, since $\det(\mathcal{L}) = eN^{-1/2}N_1N_2$. Therefore, a necessary condition for v to be a smallest vector in $\mathcal{L}$ is that

$$\sqrt{3}\,N^{\delta+\alpha-1/2} < \sqrt{3}\,N^{(\alpha+3/2)/3},$$

which simplifies to

$$\delta + \frac{2}{3}\alpha < 1. \tag{7.39}$$

When the size of the private and public exponents satisfies this bound, we run the LLL algorithm with input $\mathcal{B}$. If $\pm v$ is the smallest reduced basis vector obtained, we can simply compute the private exponent by dividing the first component of v by $A = eN^{-1/2}$. Since all computations are polynomial in $\log_2(N)$, the result follows. ❑

When the public exponent is the same size as the moduli, which is expected with high probability, we can use the approximation $\alpha \approx 1$ to further simplify this condition to

$$\delta < \frac{1}{3}. \tag{7.40}$$

Thus, Dual RSA with a private exponent $d < N^{1/3}$ should be considered unsafe, provided that Attack 7.7 works in practice.

Since we cannot prove that $\pm v$ is a smallest vector in $\mathcal{L}$, we rely on experiments to demonstrate the effectiveness of the attack. In Table 7.1 we show some experimental results of this attack using 1024-bit Dual RSA moduli and 1000 trials for each private exponent size. As can be seen, the attack is quite successful until the bitlength of the private exponent satisfies $n_d > 339$ (approximately $\delta > 0.331$) at which point it quickly becomes very ineffective. Since $1024/3 = 341 + 1/3$, we see that the attack (for 1024-bit moduli) is successful for private exponents slightly smaller than the heuristic bound $\delta < 1/3$ in practice.

The key generation algorithm for Scheme II, Algorithm 7.1 with e and d interchanged, computes the public key after choosing the private key. Since

Table 7.1: Experimental results for Attack 7.7 on Dual RSA with 1024-bit moduli for various private exponent bitlengths.

n_d	336	337	338	339	340	341	342	343
success rate (%)	100.0	100.0	98.5	83.6	17.4	0.2	0.0	0.0

the public exponent is computed as the inverse of the private exponent modulo $\varphi(N_1)$, it is expected that the size of the public exponent will be roughly the same size as the moduli. Because of this, it is difficult to generate instances of Dual RSA-Small-d with public exponents smaller than N, and is the reason for only testing the attack on instances where the public exponent is roughly the same size as the moduli. Based on the success of the attack in this case, we believe that the attack will work when e is smaller than N.

7.5 Cryptanalysis of Dual Generalized Rebalanced-RSA

All of the attacks on Generalized Rebalanced-RSA also apply to Dual Generalized Rebalanced-RSA. Thus, as given in Appendix B.2, instances of Generalized Rebalanced-RSA with parameters satisfying

$$4\alpha + 14\delta < 5 - \epsilon, \tag{7.41}$$

can be broken in time polynomial $\log_2(N)$ with an attack by Jochemsz and May [70]. It should be noted that this attack has been experimentally shown to be more effective than the bounds suggest. Also, if the CRT-exponents satisfy

$$\delta < \frac{2\ell}{\log_2(N)}, \tag{7.42}$$

then the Dual RSA moduli can be factored in time linear in 2^ℓ.

In the rest of this section we present two new attacks on Dual Generalized Rebalanced-RSA. The first attack is similar to the heuristic lattice-based attack on Dual RSA with small private exponent (Section 7.4.1) while the second attack uses continued fractions in the same way as one of the attacks on Dual RSA with small public exponent (Section 7.3.2). Both new attacks are stronger than those given by 7.41 for public exponents smaller than $N^{3/8}$ (*i.e.*, $\alpha < 3/8$).

7.5.1 Small Exponent Attack

The first attack on Scheme III consists of two parts. The first part, which recovers d_pd_q, $k_{p_1}k_{q_1}$ and $k_{p_2}k_{q_2}$, is due to Mu-En Wu. Like Attack 7.7, this first part is a heuristic. The second part of the attack uses the information gained from the first part to factor N_1 and N_2. The main result is given below.

Attack 7.8. *For every $\epsilon > 0$ there exists N_0 such for every $N_1, N_2 > N_0$ the following holds: Let $\{(e, N_1, N_2), (d_p, d_q, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual Generalized Rebalanced-RSA with public exponent $e = N^\alpha$ and CRT-exponents $d_p, d_q < N^\delta$.*

If the public and private exponents satisfy

$$4\alpha + 6\delta < 3, \tag{7.43}$$

then there exists a heuristic lattice-based computation that computes d_pd_q, $k_{p_1}k_{q_1}$ and $k_{p_2}k_{q_2}$ in time polynomial in $\log_2(N)$.

With these quantities known, if

$$\delta < \frac{1}{2} - \epsilon, \tag{7.44}$$

then N_1 and N_2 can be factored in time polynomial in $\log_2(N)$ provided Assumption 2.13 holds.

Justification: The first part of the attack begins with the two RSA-CRT equations (7.2):

$$\underbrace{\begin{array}{l} ed_p = 1 + k_{p_1}(p_1 - 1) \\ ed_q = 1 + k_{q_1}(q_1 - 1) \end{array}}_{\text{for } N_1} \quad , \quad \underbrace{\begin{array}{l} ed_p = 1 + k_{p_2}(p_2 - 1) \\ ed_q = 1 + k_{q_2}(q_2 - 1) \end{array}}_{\text{for } N_2} \quad .$$

Multiplying together the two RSA-CRT equations for each moduli we obtain, after some rearrangement,

$$\begin{aligned} e^2d_pd_q - N_1k_{p_1}k_{q_1} &= e(d_p + d_q) - 1 - k_{p_1}k_{q_1}\Lambda_1 \\ e^2d_pd_q - N_2k_{p_2}k_{q_2} &= e(d_p + d_q) - 1 - k_{p_2}k_{q_2}\Lambda_2. \end{aligned} \tag{7.45}$$

Combining these equations with the trivial equation $d_pd_q = d_pd_q$, we have the following system of three equations:

$$\begin{array}{llcl} d_pd_q & & = & d_pd_q \\ e^2d_pd_q & -N_1k_{p_1}k_{q_1} & = & e(d_p + d_q) - k_{p_1}k_{q_1}\Lambda_1 - 1 \\ e^2d_pd_q & \qquad -N_2k_{p_2}k_{q_2} & = & e(d_p + d_q) - k_{p_2}k_{q_2}\Lambda_2 - 1, \end{array} \tag{7.46}$$

which can be written as the vector-matrix equation $x\mathcal{B} = v_0$, where

$$x = (d_p d_q, k_{p_1} k_{q_1}, k_{p_2} k_{q_2}),$$

$$\mathcal{B}_0 = \begin{bmatrix} 1 & e^2 & e^2 \\ 0 & -N_1 & 0 \\ 0 & 0 & -N_2 \end{bmatrix},$$

and

$$v_0 = \big(d_p d_q, e(d_p + d_q) - \Lambda_1 k_{p_1} k_{q_1} - 1, e(d_p + d_q) - \Lambda_2 k_{p_2} k_{q_2} - 1\big).$$

As in Section 7.4.1, we would like each component of v_0 to be roughly the same size. Multiplying the first column $\mathcal{B}_0$ by e^2 and the last two columns by $N^{1/2}$ gives a new matrix

$$\mathcal{B} = \begin{bmatrix} e^2 & e^2 N^{1/2} & e^2 N^{1/2} \\ 0 & -N_1 N^{1/2} & 0 \\ 0 & 0 & -N_2 N^{1/2} \end{bmatrix}, \tag{7.47}$$

such that $x\mathcal{B} = v$, where

$$v = \big(d_p d_q e^2,\ (e(d_p+d_q) - \Lambda_1 k_{p_1} k_{q_1} - 1) N^{1/2},\ (e(d_p+d_q) - \Lambda_2 k_{p_2} k_{q_2} - 1) N^{1/2}\big).$$

Notice that each component of v has size at most $N^{2\alpha+2\delta}$ and that the size of v is bounded by $\|v\|_2 \leq \sqrt{3}\, N^{2\alpha+2\delta}$. From Theorem 2.3 (Minkowski), we know that a smallest vector in the lattice $\mathcal{L}$ spanned by the columns of $\mathcal{B}$ is bounded by $\sqrt{3}\,\det(\mathcal{L})^{1/3} = \sqrt{3}\, N^{(2\alpha+3)/3}$, since $\det(\mathcal{L}) = e^2 N N_1 N_2$. Thus, a necessary condition for v to be a smallest vector in $\mathcal{L}$ is given by $2\alpha + 2\delta < (2\alpha + 3)/3$, which simplifies to

$$4\alpha + 6\delta < 3. \tag{7.48}$$

Assuming that v is the smallest vector and that LLL finds it, knowledge of v only provides knowledge of $d_p d_q$. Of course, since $x\mathcal{B} = v$, we can easily solve this vector-matrix equation for x, which then yields $k_{p_1} k_{q_1}$ and $k_{p_2} k_{q_2}$.

With $d_p d_q$, $k_{p_1} k_{q_1}$ and $k_{p_2} k_{q_2}$ known, we now show how Λ_1 and Λ_2 can be computed. With Λ_1 and Λ_2 known, we can easily factor N_1 and N_2. We begin by considering the first equation in (7.45):

$$e^2 d_p d_q - N_1 k_{p_1} k_{q_1} = e(d_p + d_q) - 1 - k_{p_1} k_{q_1} \Lambda_1.$$

Everything in this equation is known except for $d_p + d_q$ and Λ_1, which are both small compared to N_1. This suggests we look for small solutions, modulo N_1, of the polynomial

$$f_{N_1}(x, y) = ex - k_{p_1}k_{q_1}y - 1 + e^2 d_p d_q,$$

since $(x_0, y_0) = (d_p + d_q, \Lambda_1)$ is a root of $f_{N_1}(x, y)$ modulo N_1. Defining the bounds

$$\begin{aligned} X &= 2N^{\delta} \\ Y &= 3N^{1/2}, \end{aligned} \tag{7.49}$$

notice that $|x_0| < X$ and $|y_0| < Y$. Therefore, from Theorem 2.10, we can compute $(x_0, y_0) = (d_p + d_q, \Lambda_1)$ provided N_1 is sufficiently large and $XY < N_1$. The last condition is satisfied when $\delta < 1/2 - \epsilon$, where the ϵ term corrects for the neglected constant terms in X and Y as well as the lower order terms ignored in the methods implicit in Theorem 2.10. We repeat the same process using the second equation in (7.45) to compute Λ_2. With Λ_1 and Λ_2 known, we compute the factorization of N_1 and N_2. Since all computations can be done in time polynomial in $\log_2(N)$, the result follows. ❏

When the public exponent is smaller than $N^{3/8}$ (*i.e.*, $\alpha < 3/8$), this attack is stronger than Jochemsz and May's attack given by (7.41).

7.5.2 Small n_k Attack

In this section, we present an attack on Dual Generalized Rebalanced-RSA when the security parameter in the key generation algorithm is small (*i.e.*, n_k is small). The attack uses continued fractions and is very similar to Attack 7.2. The main result is given in the following attack.

Attack 7.9. *For every $\epsilon > 0$ there exists N_0 such that for every $N_1, N_2 > N_0$ the following holds: Let $\{(e, N_1, N_2), (d_p, d_q, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual Generalized Rebalanced-RSA with public exponent $e = N^{\alpha}$, CRT-exponents $d_p, d_q < N^{\delta}$ and moduli having bitlength $n \geq 16$. Let $k_{p_2}, k_{q_2}, k_{p_1}, k_{q_1}$ be the constants in the CRT equations.*

If the public and private exponents satisfy

$$\alpha + \delta < \frac{5}{8} - \frac{\log_N(8\sqrt{7})}{2}, \tag{7.50}$$

we can compute a list containing $k_{p_1}k_{q_2}/k_{p_1}k_{q_1}$, in lowest terms. The length of this list is polynomial in $\log_2(N)$.

With $k_{p_1}k_{q_2}/k_{p_1}k_{q_1}$, in lowest terms, known, if the public and private exponents also satisfy

$$\alpha + \delta > \frac{1}{2} + \log_N(K') - \epsilon, \tag{7.51}$$

where $K' = \gcd(k_{p_2}k_{q_2}, k_{p_1}k_{q_1})$*, then N_1 and N_2 can be factored in time polynomial in* $\log_2(N)$*, provided that Assumption 2.13 holds.*

Justification: Recall from the key generation algorithm, Algorithm 7.2, that the Dual RSA moduli (for Generalized Rebalanced-RSA) satisfy

$$\begin{array}{ll} N_1 = p_1 q_1 & N_2 = p_2 q_2 \\ p_1 = p_a p_b + 1 & p_2 = p_{a'} p_b + 1 \\ q_1 = q_a q_b + 1 & q_2 = q_{a'} q_b + 1 \\ p_a = p_{a_1} p_{a_2} \cdots p_{a_{(k-1)}} p_{a_k} & p_{a'} = p_a k_{p_1}/p_{a_{i'}} \\ q_a = q_{a_1} q_{a_2} \cdots q_{a_{(k-1)}} q_{a_k} & q_{a'} = q_a k_{q_1}/q_{a_{j'}}, \end{array} \tag{7.52}$$

for some i' and j', where $k = \lceil (n/2 - n_e)/n_k \rceil$, and

$$\begin{aligned} k_{p_2} &= p_{a_{i'}} \\ k_{q_2} &= q_{a_{j'}}. \end{aligned} \tag{7.53}$$

Further, the size of each parameter is as shown in the following table.

Parameters	Bitlength	Size
p_1, q_1, p_2, q_2	$n/2$	$N^{1/2}$
p_{a_i}, q_{a_i}, k_{p_1}, k_{q_1}	n_k	$N^{\alpha+\delta-1/2}$
p_a, $p_{a'}$, q_a,$q_{a'}$	$n/2 - n_e$	$N^{1/2-\alpha}$
p_b, q_b	n_e	N^{α}

We proceed in the same way as the proof of Attack 7.2. Let $n \geq 16$. From the structure of N_1 and N_2, we observe that

$$\begin{aligned} \frac{N_1 - 1}{p_{a'} p_b q_{a'} q_b} &= \frac{p_a p_b q_a q_b + p_a p_b + q_a q_b}{p_{a'} p_b q_{a'} q_b} \\ &= \frac{p_{a_{i'}} q_{a_{j'}}}{k_{p_1} k_{q_1}} + \frac{p_a p_b + q_a q_b}{p_{a'} p_b q_{a'} q_b}, \end{aligned}$$

and

$$\begin{aligned}\frac{N_2-1}{p_{a'}p_bq_{a'}q_b} &= \frac{p_{a'}p_bq_{a'}q_b + p_{a'}p_b + q_{a'}q_b}{p_{a'}p_bq_{a'}q_b}\\ &= 1 + \frac{p_{a'}p_b + q_{a'}q_b}{p_{a'}p_bq_{a'}q_b}.\end{aligned}$$

Letting $A = (p_ap_b + q_aq_b)/(p_{a'}p_bq_{a'}q_b)$, $B = (p_{a'}p_b + q_{a'}q_b)/(p_{a'}p_bq_{a'}q_b)$, and noting that $p_{a_{i'}} = k_{p_2}$ and $q_{a_{j'}} = k_{q_2}$, we then have

$$\frac{N_1-1}{p_{a'}p_bq_{a'}q_b} = \frac{k_{p_2}k_{q_2}}{k_{p_1}k_{q_1}} + A, \tag{7.54}$$

$$\frac{N_2-1}{p_{a'}p_bq_{a'}q_b} = 1 + B. \tag{7.55}$$

Since p_a, $p_{a'}$, q_a and $q_{a'}$ are all $(n/2 - n_e)$-bit numbers and p_b and q_b are n_e-bit numbers, it follows that A satisfies

$$\begin{aligned}|A| &= \left|\frac{p_ap_b + q_aq_b}{p_{a'}p_bq_{a'}q_b}\right|\\ &< \frac{(2^{n/2-n_e})(2^{n_e}) + (2^{n/2-n_e})(2^{n_e})}{(\frac{1}{2}2^{n/2-n_e})(\frac{1}{2}2^{n_e})(\frac{1}{2}2^{n/2-n_e})(\frac{1}{2}2^{n_e})}\\ &= \frac{2\times 2^{n/2}}{(\frac{1}{2})^4\times 2^{n/2}\times 2^{n/2}}\\ &= \frac{2^5}{2^{n/2}}.\end{aligned} \tag{7.56}$$

Similarly, $|B| < 2^{5-n/2}$. Therefore, $|A|, |B| < 1$ when $n > 10$. Letting $K = (k_{p_2}k_{q_2})/(k_{p_1}k_{q_1})$, it then follows that dividing (7.54) by (7.55) yields

$$\begin{aligned}\frac{N_1-1}{N_2-1} &= \frac{K+A}{1+B}\\ &= (K+A)(1 - B + B^2 - B^3 + \cdots)\\ &= K + A - (K+A)(B - B^2 + B^3 - \cdots),\end{aligned} \tag{7.57}$$

so that

$$\left|\frac{N_1-1}{N_2-1} - K\right| = \left|A - (K+A)(B - B^2 + B^3 - \cdots)\right|. \tag{7.58}$$

Let's consider the size of the terms in the right-hand side of this equation. Since all the Dual RSA primes are balanced (*i.e.*, p_1, q_1, p_2, q_2 all have the same bitlength), it follows that $1/2 < k_{p_2}/k_{p_1} < 2$ and $1/2 < k_{q_2}/k_{q_1} < 2$, which gives the bound $1/4 < K < 4$. Combining this with $|A| < 1$, we have

$$|K + A| < 5,$$

as K is positive. Also, since $n \geq 16$, we have

$$\begin{aligned} |B^2| + |B^3| + |B^4| + \cdots &< \sum_{i=2}^{\infty} \left| \frac{2^5}{2^{n/2}} \right|^i \\ &= \frac{2^{n/2+10}}{2^n \left(2^{n/2} - 2^5\right)} \\ &< \frac{1}{5} \times \frac{2^5}{2^{n/2}}. \end{aligned} \tag{7.59}$$

Using these bounds, along with $|A| < 2^{5-n/2}$, we apply the triangle inequality twice to (7.58) to obtain

$$\begin{aligned} \left| \frac{N_1 - 1}{N_2 - 1} - K \right| &= \left| A - (K + A)(B - B^2 + B^3 - \cdots) \right| \\ &\leq |A| + |K + A| \times \left| B - B^2 + B^3 - \cdots \right| \\ &\leq |A| + |K + A| \times |B| + |K + A| \times \left| -B^2 + B^3 - \cdots \right| \\ &< \frac{2^5}{2^{n/2}} + 5\frac{2^5}{2^{n/2}} + 5\frac{1}{5}\frac{2^5}{2^{n/2}} \\ &= \frac{7 \times 2^5}{2^{n/2}} \\ &< \frac{224}{\sqrt{N}}. \end{aligned} \tag{7.60}$$

Letting $\alpha + \delta < 5/8 - \log_N(8\sqrt{7})/2$, so that

$$\begin{aligned} k_{p_1} k_{q_1} &= N^{2\alpha + 2\delta - 1} \\ &< \frac{N^{1/4}}{8\sqrt{7}}, \end{aligned}$$

it follows that

$$\begin{aligned} \left| \frac{N_1 - 1}{N_2 - 1} - \frac{k_{p_2} k_{q_2}}{k_{p_1} k_{q_1}} \right| &< \frac{224}{\sqrt{N}} \\ &< \frac{1}{2(k_{p_1} k_{q_1})^2}. \end{aligned} \tag{7.61}$$

Therefore, by Theorem 2.1 (continued fractions), we know that $k_{p_2}k_{q_2}/k_{p_1}k_{q_1}$, in lowest terms, will appear as one of the convergents in the continued fraction expansion of $(N_1-1)/(N_2-1)$. Let ${k_1}' = k_{p_2}k_{q_2}/K'$ and ${k_2}' = k_{p_1}k_{q_1}/K'$, where $K' = \gcd(k_{p_2}k_{q_2}, k_{p_1}k_{q_1})$. Thus, one of the convergents in $(N_1-1)/(N_2-1)$ will yield ${k_1}'$ and ${k_2}'$ (appearing as ${k_2}'/{k_1}'$). We compute each convergent as a candidate for ${k_2}'/{k_1}'$ and try to factor N_1 and N_2 using a lattice-based method.

Let's assume that we have the correct convergent (*i.e.*, we know k_1' and k_2'). The lattice-based method we use to factor the moduli is the same as that used in Attack 7.6 in Section 7.3.4. We briefly outline the method below.

Multiplying together the two RSA-CRT equations for each moduli, as was done in the previous attack, yields

$$\begin{array}{rcl} e^2d_pd_q - N_1k_{p_1}k_{q_1} & = & e(d_p+d_q) - 1 - k_{p_1}k_{q_1}\Lambda_1 \\ e^2d_pd_q - N_2k_{p_2}k_{q_2} & = & e(d_p+d_q) - 1 - k_{p_2}k_{q_2}\Lambda_2. \end{array} \tag{7.62}$$

Subtracting these equations, and rearranging gives

$$k_{p_1}k_{q_1}(N_1 - \Lambda_1) = k_{p_2}k_{q_2}(N_2 - \Lambda_2),$$

which when divided by K' yields

$${k_1}'(N_1 - \Lambda_1) = {k_2}'(N_2 - \Lambda_2),$$

where everything but Λ_1 and Λ_2 are known. Notice that except for the size of ${k_1}'$ and ${k_2}'$ (compared to ${k_1}'$ and ${k_2}'$), this is the same starting point for the proof of Attack 7.6. Here we have

$$|{k_1}'|, |{k_2}'| < N^{2\alpha+2\delta-1-\log_N(K')},$$

instead of

$$|{k_1}'|, |{k_2}'| < N^{\alpha+\delta-1-\log_N(k')}.$$

Adapting the result of Attack 7.6 to include this change, we find that both Λ_1 (indirectly through τ_1) and Λ_2 (directly) can be computed provided that N is sufficiently large and

$$\alpha + \delta > \frac{1}{2} + \log_N(K') - \epsilon.$$

Thus, since all computations can be done in time polynomial in $\log_2(N)$ and the number of candidates (convergents) that we need to test is polynomial in $\log_2(N)$, the result follows. ❑

Just as with the previous attack, when the public exponent is smaller than $N^{3/8}$ (*i.e.*, $\alpha < 3/8$), this attack is stronger than Jochemsz and May's attack given by (7.41). It is also stronger then the previous attack (Attack 7.8).

As shown by Verheul and van Tilborg [120] and Dujella [43], we can include an exhaustive search on each convergent to increase the bound of the sufficient condition. For an exhaustive search of size 2^ℓ, the bound can be increased by $\ell/2$ (see [120] or [43] for details). Allowing such a search gives us the following attack.

Attack 7.10. *For every $\epsilon > 0$ there exists N_0 such that for every $N_1, N_2 > N_0$ the following holds: Let $\{(e, N_1, N_2), (d_p, d_q, p_1, q_1, p_2, q_2)\}$ be a valid instance of Dual Generalized Rebalanced-RSA with public exponent $e = N^\alpha$, CRT-exponents $d_p, d_q < N^\delta$ and moduli with bitlength $n \geq 16$. Let $k_{p_2}, k_{q_2}, k_{p_1}, k_{q_1}$ be the constants in the CRT equations. If the public and private exponents satisfy*

$$\alpha + \delta < \frac{5}{8} + \frac{\ell}{2} - \frac{\log_N(8\sqrt{7})}{2}, \tag{7.63}$$

then we can compute a list containing $k_{p_1}k_{q_2}/k_{p_1}k_{q_1}$, in lowest terms. The size of the list is 2^ℓ.

Given $k_{p_1}k_{q_2}/k_{p_1}k_{q_1}$, in lowest terms, if the public and private exponents also satisfy

$$\alpha + \delta > \frac{1}{2} + \log_N(K') - \epsilon, \tag{7.64}$$

where $K' = \gcd(k_{p_2}k_{q_2}, k_{p_1}k_{q_1})$, then N_1 and N_2 can be factored in time polynomial in $m = \log_2(N)$ provided Assumption 2.13 holds. The overall runtime is polynomial in $\log_2(N)$ and linear in 2^ℓ.

The strongest attacks on Dual Generalized Rebalanced-RSA, in the limit of large moduli, are shown in Figure 7.1. All key choices (α and δ pairs) that lie below and the to the left the curves in the plot are unsafe. When the public key is greater than $N^{3/8}$, the strongest attack is Jochemsz and May's attack [70], which is the same for Dual Generalized Rebalanced-RSA and as it is for a single instance of Generalized Rebalanced-RSA. When the public exponent is smaller than $N^{3/8}$, the strongest result is given by Attack 7.9. Here, the special structure of the Dual RSA moduli allows for a stronger attack on Dual Generalized Rebalanced-RSA than on a single instance of Generalized Rebalanced-RSA. The area between the dotted and straight lines illustrates this difference in security. The result of Attack7.8, although not shown in the plot lies between the dotted and straight lines for public exponents smaller then $N^{3/8}$.

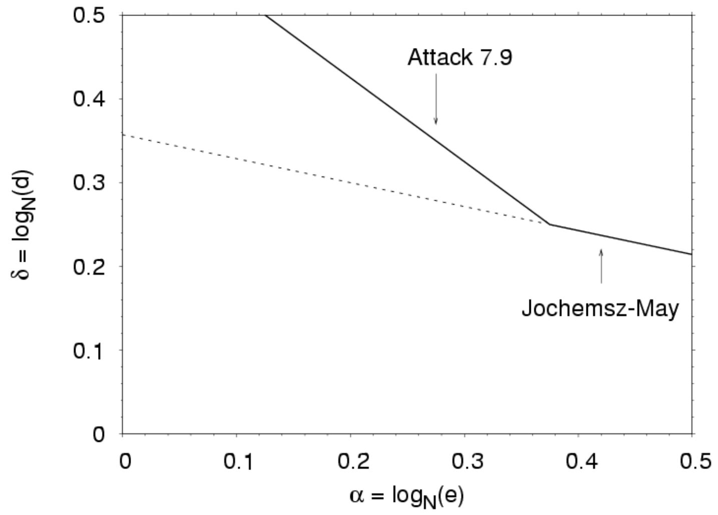


Figure 7.1: Unsafe parameter choices for Scheme III.

7.6 Summary of Attacks

In Table 7.2, we summarize all unsafe parameter choices, public and private exponents pairs, for the strongest known attacks against each of the three Dual RSA schemes. When the parameter restrictions are satisfied, an attack against Dual RSA exists that requires time polynomial in $\log_2(N)$ and linear in 2^ℓ.

7.7 Discussion

As discussed in [114], Dual RSA can be used in certain applications to reduce the key storage requirements. This reduction in memory requirements comes at the expense of increased time complexity for the key generation algorithms (see [114] for a discussion on this) and somewhat reduced security. As is demonstrated in this chapter, the number of parameter choices, public and

Table 7.2: Summary of unsafe Dual RSA parameters.

Scheme	Restrictions	Comments
I	$\alpha < 5/4 + \ell_n/2 - \delta$	$\delta < 1$ (§7.3.2, 7.3.4), Large N
	$\alpha < 1/4 + \ell_n/2$	$\delta \approx 1$ (§7.3.2, 7.3.4), Large N
	$\alpha > 1/2 - \ell_n$	known k_1 and k_2 (§7.3.2)
II	$\delta < 5/4 + \ell_n/2 - \alpha$	$\alpha < 1$ (§7.3.2, 7.3.4), Large N
	$\delta < 7/6 - \sqrt{1+6\alpha}/3$	$\alpha < 1$ (from [14]), Large N
	$\delta < 1/3$	$\alpha \approx 1$ (§7.4.1)
III	$\alpha + \delta < 5/8 + \ell/2 - 2/\log_2(N)$	$\alpha \leq 3/8$ (§7.5.2), Large N
	$4\alpha + 14\delta < 5$	$3/8 \leq \alpha < 1/2$ (from [70]), Large N
	$\delta < 2\ell/\log_2(N)$	Baby-step Giant-step (from [98])

private exponent pairs, that are insecure is greater when using Dual RSA as compared to using two different instances of RSA. This is due to the significant amount of structure imposed on the Dual RSA moduli, as is evidenced by attacks presented above that specifically exploit the special structure of the moduli.

As Dual RSA is a recently developed cryptosystem, more cryptanalysis is needed to properly assess its security.

Chapter 8

Conclusions

8.1 Summary and Discussion

In this thesis, we have analyzed the security of five different variants of RSA. In particular, using known factoring methods (ECM and NFS), continued fractions and lattice basis reduction techniques we have provided detailed analyses of the best attacks known (including some new attacks) on instances of RSA with certain special private exponents, multiple instances of RSA sharing a common small private exponent, Multi-prime RSA, Common Prime RSA and Dual RSA.

In Table 8.1, we give a high level summary of the security of each variant as compared to RSA for different types of attacks. In the table, we differentiate four types of attacks. Factoring attacks are attacks that factor the modulus given only the modulus and perhaps knowledge of any special structure that the primes might have. Both small d and partial key attacks

Table 8.1: Summary of security of each variant compared to RSA for various attack types. We use n/a to indicate that we did not consider this type of attack or that the type of attack was not applicable.

Ch.	Variant	Factoring	Small d	Partial Key	Special
3.	Small $\lvert d - \frac{a}{b}\lambda(N)\rvert$	same	same	n/a	weaker
4.	Many keys, small d	same	weaker	n/a	n/a
5.	Multi-prime RSA	weaker	stronger	stronger	n/a
6.	Common Prime RSA	weaker	stronger	n/a	n/a
7.	Dual RSA	weaker	weaker	n/a	weaker

are attacks that exploit the key equation. Small d attacks can be mounted on instances with small private exponents while partial key attacks require some knowledge of the private key. All other attacks are labelled as special. In the table, we only indicate whether the variant is stronger or weaker than RSA for a given type of attack. We do not indicate by how much the variant is stronger or weaker. Below, we give a brief summary for each variant.

In Chapter 3, we have shown that RSA with a private exponent that is sufficiently close to a rational multiple of $\lambda(N)$ is insecure. For private exponents satisfying $|d - \frac{a}{b}\lambda(N)| < N^{\delta}$, the attacks presented show that the private exponent d can be computed for δ as large as 0.284 when $b = 1$ or $a = 0$ (corresponding to Boneh and Durfee's lattice-based attack [13]). The bound on δ decreases with increasing b until it reaches zero when $b = 0.25$. The results of this chapter illustrate a class of weak keys for RSA.

In Chapter 4, we have shown that Wiener's small private exponent attack, when viewed as a lattice-based attack, becomes stronger when applied to multiple instances of RSA with the same small private exponent. In particular, a private exponent satisfying

$$d < \frac{1}{6} N^{\frac{1}{2} - \frac{1}{2(r+1)}},$$

can be recovered when r instances of RSA use this exponent. The lattice-based attack is only a heuristic, but it has been shown to work well in practice.

In Chapter 5, we have collected the best known attacks on Multi-prime RSA, including some new attacks appearing for the first time in this thesis. With the lone exception of factoring the modulus with the ECM, we find that all attacks on Multi-prime RSA become weaker as the number of primes in the modulus increases. Combining this with the reduced decryption costs, Multi-prime RSA seems to be a promising candidate as a replacement for RSA.

In Chapter 6, we have shown the existence of RSA instances that are more resistant to all the known small private exponent attacks (Wiener's continued fraction and Boneh and Durfee's lattice-based attacks). This variant of RSA, which we call Common Prime RSA, achieves this resistance at the cost of adding more structure to the RSA primes. In particular, the primes are constructed so that $\gcd(p-1, q-1)/2$ is a large prime. With a proper choice of parameters, we have shown that Common Prime RSA with private exponents as small as $N^{0.1}$ exist and are immune to all known attacks. Due to the significant additional structure imposed on the RSA primes, however, it is conceivable that stronger attacks may be found in the future.

Finally, in Chapter 7, we have analyzed the security of Dual RSA. In this variant, we have shown that the very special structure of the primes leads to several attacks that would, otherwise, not affect the security of two independent instances of RSA. In particular, we show that private exponents smaller than $N^{1/3}$ and public exponents smaller than $N^{1/4}$ are insecure (when the other exponent is the size of the modulus). Since the motivation for Dual RSA is to reduce memory requirements for the keys rather than to reduce computational costs, these restrictions on the exponent sizes may be acceptable in some circumstances.

8.2 Future Work

As mentioned in the Open Problems/Future Work part in several chapters in this thesis, there is a significant amount of additional work that can be done with respect to the variants considered here.

One promising direction is to re-examine the lattice-based attacks that use extensions of Coppersmith's methods to ensure that the optimal polynomials are considered. Until recently, the polynomials used in extensions of Coppersmith's methods were of small degree (*e.g.*, linear in each variable) and relatively simple. For example, the polynomial used by Boneh and Durfee [13] only had monomials $\{1, x, xy\}$. It was thought that more complex polynomials with higher degree would lead to bounds for the solutions that were meaningless (*i.e.*, negative or extremely small). The works of Ernst *et al.* [46] and Jochemsz and May [69, 70], however, have shown that more complex polynomials with higher degree sometimes lead to meaningful attacks. For example, the polynomial used by Jochemsz and May in [69] has monomials

$$\{1, x, x^2, y, z, xy, xz, yz\}.$$

Using the methods developed by Jochemsz and May [69] to compute bounds for any polynomial, all of the known lattice-based attacks should be re-examined to see if a more complex polynomial might improve the bounds. This process is, unfortunately, very tedious and time consuming at present.

In addition to the continued research into the security of the five variants of RSA in this thesis, we would like to highlight three research areas that extend beyond this thesis. In particular, the algebraic independence issue, sub-lattice optimization and Multi-power RSA. We consider each below.

Algebraic Independence

All multivariate extensions of Coppersmith's methods (for both integer and modular solutions) are only heuristic. This is due to the fact that there is no guarantee that the polynomials obtained from lattice basis reduction are algebraically independent. Indeed, it is an open question whether or not anything can be said about the algebraic independence of polynomials obtained from reduced basis vectors, given the original polynomials (original basis vectors).

While most experimental evidence in cryptographic applications shows that the polynomials are algebraically independent, it is not always the case (see [61] for example). As a first step in trying to answer this open problem, we feel that more experimental evidence needs to be acquired and analyzed. In particular, the lattices in which algebraically dependent polynomials occur with some frequency should be investigated.

Sub-lattice Optimizations

When using Coppersmith's methods, it is sometimes possible to obtain improved bounds by considering certain sub-lattices instead of the full lattice. Recall from Chapter 2 that the enabling equations for Coppersmith's methods (ensuring that the smallest polynomials are small enough to use Howgrave-Graham's result) can be written as

$$\mathrm{vol}(\mathcal{L}) < cN^d,$$

where c and d depend on the lattice dimension and the number of required small polynomials. The goal of using sub-lattices is to reduce the volume of the lattice $\mathrm{vol}(\mathcal{L})$ more than the term cN^d might be reduced. Unfortunately, computing the volume of the sub-lattice is non-trivial. This is because the sub-lattice is itself a lattice that is not full dimensional.

Boneh and Durfee [13] used geometrically progressive matrices to bound the volume of certain sub-lattices obtained by removing certain basis vectors (those that contributed to the volume more than others). Another technique, by Blömer and May [7], also removes basis vectors that significantly contribute to the volume of the lattice. But, instead of working with a non-full dimensional sub-lattice, they also remove corresponding columns in the basis matrix. The resulting lattice is not a sub-lattice of the original but it is full dimensional and admits a simple volume determination. Blömer and May then show that small vectors obtained by lattice reduction of the new lattice correspond to small vectors in the original lattice and they provide a bound on the difference in their sizes (based on the volume of the new full dimensional lattice). Both of these methods were only applied to one polynomial (with monomials $\{1, x, xy\}$).

To our knowledge, there has been no further work on using sub-lattices to improve the bounds of solutions to any other polynomial. Thus, it is an open problem if the bounds found using Jochemsz and May's technique for computing bounds for any polynomial (see [69]) are optimal, or if sub-lattices can be used to improve them. Many cryptographic attacks might be strengthened if the latter is true.

Multi-power RSA

In this thesis we have considered five variants of RSA. There are, of course, more variants. Another variant that is perhaps a good candidate to replace RSA is Takagi's variant (see [116, 117] or Appendix B.3), which is now referred to as Multi-power RSA (see [19]). This variant differs from RSA in several ways. First, the modulus is composed of two balanced primes with one prime having multiplicity greater than 1. That is, the modulus has the form $N = p^k q$ for some $k > 1$. Second, the public and private exponents are defined as inverses modulo $\text{lcm}(p-1, q-1)$ instead of modulo $\phi(N)$ or $\phi(N)/\gcd(p-1, q-1)$. Thus, both e and d are bounded by $N^{2/3}$. And last, decryption uses Hensel lifting and Chinese remaindering which makes the cryptosystem much more efficient than RSA.

The security of Multi-power RSA, when only considering factoring attacks, is the same as Multi-prime RSA with $k+1$ primes in the modulus. Unlike Multi-prime RSA, however, there are no known small private exponent or partial key exposure attacks for Multi-power RSA. Since most of these attacks on RSA (and Multi-prime RSA) exploit the fact that N is a good approximation of $\phi(N)$, they do not carry over to Multi-power RSA (where $N = p^k q$ is not a good approximation of $\text{lcm}(p-1, q-1)$). It remains an open question whether there exists a small private exponent or partial key exposure attack better than exhaustive search for Multi-power RSA.

Appendix A

Distribution of $g = \gcd(p-1, q-1)$

We have carried some experiments to determine (approximate) the distribution of $g = \gcd(p-1, q-1)$ for random primes p and q of the same bitlength. For several bitlengths of primes, we computed 100,000 random pairs of primes and computed $\gcd(p-1, q-1)$. The results (given as percentages) are given in Table A.1 for $g \leq 20$ and in Table A.2 for $20 < g \leq 100$.

	Bitlength of primes					
g	128	256	384	512	1024	average
2	49.5	49.7	49.8	49.5	49.5	49.6
4	12.4	12.4	12.4	12.3	12.4	12.4
6	14.7	14.6	14.6	14.7	14.8	14.7
8	3.1	3.1	3.0	3.2	3.15	3.1
10	3.1	3.2	3.1	3.2	3.1	3.2
12	3.8	3.5	3.6	3.5	3.6	3.6
14	1.4	1.3	1.4	1.3	1.4	1.4
16	0.7	0.7	0.8	0.8	0.8	0.8
18	1.6	1.6	1.6	1.7	1.6	1.6
20	0.7	0.7	0.8	0.8	0.8	0.8

Table A.1: Distribution of $2 \leq g \leq 20$. Percentages are rounded to one decimal place.

From the tables, we see that $g = 2$ with probability almost 50/100, $g \leq 6$ with probability about 76/100, $g \leq 20$ with probability about 91/100 and $g \leq 100$ with probability about 98/100.

	Bitlength of primes					
g	128	256	384	512	1024	average
22	0.477	0.513	0.508	0.522	0.4575	0.4955
24	0.954	0.955	0.897	0.897	0.935	0.9276
26	0.35	0.368	0.327	0.343	0.3225	0.3421
28	0.343	0.378	0.341	0.311	0.29	0.3326
30	0.934	0.967	0.945	0.967	0.9325	0.9491
32	0.17	0.193	0.22	0.186	0.2325	0.2003
34	0.192	0.183	0.193	0.209	0.1725	0.1899
36	0.442	0.433	0.417	0.428	0.4475	0.4335
38	0.157	0.166	0.164	0.125	0.1475	0.1519
40	0.217	0.2	0.199	0.18	0.195	0.1982
42	0.407	0.419	0.393	0.421	0.4125	0.4105
44	0.142	0.128	0.126	0.113	0.1575	0.1333
46	0.137	0.091	0.103	0.096	0.0925	0.1039
48	0.225	0.242	0.241	0.245	0.2275	0.2361
50	0.128	0.14	0.142	0.133	0.11	0.1306
52	0.089	0.081	0.072	0.072	0.0825	0.0793
54	0.187	0.186	0.182	0.169	0.175	0.1798
56	0.076	0.092	0.072	0.067	0.0775	0.0769
58	0.063	0.062	0.051	0.072	0.065	0.0626
60	0.232	0.235	0.217	0.265	0.2175	0.2333
62	0.054	0.061	0.054	0.064	0.0625	0.0591
64	0.044	0.064	0.045	0.051	0.04	0.0488
66	0.143	0.139	0.155	0.151	0.1625	0.1501
68	0.041	0.051	0.053	0.043	0.0525	0.0481
70	0.078	0.087	0.077	0.092	0.115	0.0898
72	0.112	0.109	0.105	0.098	0.09	0.1028
74	0.045	0.048	0.042	0.052	0.03	0.0434
76	0.029	0.044	0.043	0.03	0.025	0.0342
78	0.081	0.109	0.107	0.115	0.105	0.1034
80	0.06	0.052	0.05	0.057	0.0275	0.0493
82	0.027	0.034	0.034	0.026	0.0325	0.0307
84	0.092	0.115	0.114	0.095	0.135	0.1102
86	0.036	0.025	0.03	0.03	0.02	0.0282
88	0.032	0.04	0.032	0.031	0.045	0.036
90	0.084	0.113	0.095	0.091	0.1075	0.0981
92	0.025	0.022	0.034	0.036	0.02	0.0274
94	0.024	0.022	0.014	0.03	0.0125	0.0205
96	0.055	0.049	0.062	0.064	0.0525	0.0565
98	0.03	0.025	0.029	0.023	0.035	0.0284
100	0.036	0.033	0.032	0.025	0.035	0.0322

Table A.2: Distribution of $20 < g < 100$.

Appendix B

Some RSA Variants

Here we collect (and discuss) some variants of RSA which are relevant to this thesis but not directly considered.

B.1 Rebalanced RSA

RSA is typically used with a very small public and a very large private exponent. Thus, encryption costs are significantly smaller than decryption costs. Even when CRT decryption is used, the CRT-exponents d_p and d_q are, with high probability, very large modulo the primes.

In order to decrease decryption costs, Wiener [123] suggested that instead of choosing a small public exponent and computing the corresponding large private exponent (or CRT-exponents) that one can instead first choose small CRT-exponents and then compute the corresponding large public exponent. Thus, decryption costs are lowered at the cost of maximizing encryption costs. This variant is commonly referred to as *Rebalanced RSA* (*e.g.*, see [12, 19]), since there is a rebalancing of the encryption and decryption costs.

The first attack on Rebalanced RSA was by Qiao and Lam in 1998 [98]. The attack uses a Baby-Step Giant-Step method to factor the modulus with $O(\min\{\sqrt{d_p}, \sqrt{d_q}\})$ comparisons (*i.e.*, exponential in the bitlength of the modulus). The attack was for the special case of $d_p = d_q + 2$, but works for any instance of Rebalanced RSA.

In 2002, May [80] presented some polynomial time lattice-based attacks on Rebalanced RSA with small CRT-exponents when the RSA primes are unbalanced (*i.e.*, when p is significantly smaller than $N^{1/2}$). In particular, all of the attacks require that $p < N^{0.382}$.

In 2006, Bleichenbacher and May [6] presented an improved polynomial time attack on Rebalanced RSA with small CRT-exponents and unbalanced primes that works when one of the primes is smaller than $N^{0.468}$.

In 2006, Jochemsz and May [69] presented a polynomial time attack on Rebalanced RSA with small CRT-exponents and balanced primes when the difference between the CRT-exponents is known (attacking Qiao and Lam's scheme [98], which has a difference of two). The attack factors the modulus when $d_p < N^{0.099}$.

In 2007, Jochemsz and May [70] presented a polynomial time attack on Rebalanced RSA with balanced primes and arbitrary small CRT-exponents. The attack can factor the modulus when the CRT-exponents $d_p, d_q < N^{\delta}$ satisfy

$$\delta < \frac{5 - 4\alpha + 20\tau - 16\alpha\tau + 18\tau^2 - 12\alpha\tau^2}{14 + 56\tau + 66\tau^2 + 24\tau^3} - \epsilon, \tag{B.1}$$

for any $\epsilon > 0$ and $\tau \geq 0$. For $\alpha \approx 1$, which is expected if the CRT-exponents are chosen before computing e, the right-hand side of this inequality is maximized when $\tau \approx 0.381788$, which leads to the bound

$$\delta < 0.0734 - \epsilon. \tag{B.2}$$

In practice, Jochemsz and May have observed that this bound is actually pessimistic.

B.2 Generalized Rebalanced-RSA

Rebalanced RSA, as suggested by Wiener, allows for low decryption costs at the expense of high encryption costs. Building on the ideas of Sun and Yang [115], it is possible to have small CRT-exponents and a small public exponent (*i.e.*, $\delta, \alpha < N^{1/2}$). This notion of Rebalanced RSA with small public and CRT-exponents was independently introduced in 2005 by Sun and Wu [113, 112] and Galbraith, Heneghan and McKee [47]. Essentially, Sun and Wu, and Galbraith, Heneghan and McKee proposed key generation algorithms for Rebalanced RSA that can create instances with public exponents significantly smaller than N and CRT-exponents smaller than $N^{1/2}$. We call this variant *Generalized Rebalanced-RSA*.

The original security analyses in [112] and [47] both led to parameter suggestions that were insecure. The security was re-analyzed in 2005, independently by Galbraith, Heneghan and McKee in [48] and by Sun, Hinek and Wu in [111].

In 2006, Bleichenbacher and May [6] presented an attack on Generalized Rebalanced-RSA that can be factor the modulus when the CRT-exponents $d_p, d_q < N^\delta$ satisfy

$$\delta \leq \min\left(\frac{2}{5} - \frac{2}{5}\alpha, \frac{1}{4}\right). \tag{B.3}$$

In 2007, Jochemsz and May [70] presented an attack on Rebalanced RSA which can also be mounted on Generalized Rebalanced-RSA. The attack factors the modulus when the public exponent $e = N^\alpha$ satisfies $\alpha < 1/2$ and the CRT-exponents $d_p, d_q < N^\delta$ satisfy

$$\delta < \frac{5 - 4\alpha + 20\tau - 16\alpha\tau + 27\tau^2 - 30\alpha\tau^2 + 12\tau^2 - 24\alpha\tau^3}{14 + 56\tau + 66\tau^2 + 24\tau^3} - \epsilon, \tag{B.4}$$

for any $\epsilon > 0$ and $\tau \geq 0$. Since the key generation for Generalized Rebalanced-RSA ensures that $\alpha + \delta < 1/2$, we find that $\tau = 0$ maximizes the right-hand side of (B.4) for all relevant α. In this case, the bound on the CRT-exponents simplifies to

$$\delta < \frac{5 - 4\alpha}{14} - \epsilon, \tag{B.5}$$

or

$$4\alpha + 14\delta < 5 - \epsilon. \tag{B.6}$$

This is the strongest known attack on Generalized Rebalanced-RSA.

B.3 Multi-power RSA

In the patent for RSA [103], Rivest, Shamir and Adleman comment that

> In alternative embodiments, the present invention may use a modulus n which is a product of three or more primes (not necessarily distinct). Decoding may be performed modulo each of the prime factors of n and the results combined using "Chinese remaindering" or any equivalent method to obtain the result modulo n.

Thus, the most general form of an RSA modulus, from their description, is given by

$$N = p_1^{\gamma_1} p_2^{\gamma_2} \cdots p_r^{\gamma_r},$$

where $r \geq 2$ and $\gamma_i \geq 1$ for $i = 1, \ldots, r$. Depending on the number of primes and their multiplicities, we have different variants of RSA. When $r = 2$ and $\gamma_1 = \gamma_2 = 1$ we have RSA. When the total number of primes in the modulus, including multiplicities, is greater than 2 we have *Multi-factor RSA*. Notice that Multi-prime RSA is a special case of Multi-power RSA (in which $r > 2$ and $\gamma_i = 1$ for each of the primes). Another special case of Multi-factor RSA is when $r = 2$ and $\gamma_i > 1$ for at least one of the primes. We call this *Multi-power RSA*.

In 1998, Takagi [116, 117] proposed the first Multi-power RSA variant with a modulus of the form

$$N = p^s q,$$

where $s > 1$. Using Hensel lifting and Chinese remaindering, Takagi showed that decryption can be done much more efficiently as compared to normal RSA decryption.

In 2000, Lim, Kim, Yie and Lee [76] extended Takagi's cryptosystem to include moduli of the form

$$N = p^s q^t,$$

where $s > 1$ and $t = 1, 2,$ or 4 (depending on the value of $s + t$).

In all of the Multi-power variants, the public and private exponents are defined as inverses modulo $\mathrm{lcm}(p-1, q-1)$. Thus, we have the key equation

$$ed = 1 + k\,\mathrm{lcm}(p-1, q-1),$$

where k is some positive integer. Notice that both the public and private exponents are smaller than $N^{1/(s+t)}$.

Like RSA, the security of all of the Multi-power RSA variants is based on factoring the modulus. The size of the multiplicities (s and t) is determined by matching the expected runtime of the NFS and ECM to factor the modulus. Unlike RSA, however, there are very few known specialized attacks on Multi-power RSA (*e.g.*, small private exponent or partial key exposure attacks).

In 2002, Ciet *et al.* [25] extended all known small private attacks on RSA to small private exponent Multi-power RSA. However, all the attacks require that the public and private exponents be defined modulo $\phi(N)$ rather than $\mathrm{lcm}(p-1, q-1)$. Thus, the attacks are not relevant to Multi-power RSA.

In 2004, May [82] presented new attacks on small private exponent Multi-power RSA with moduli of the form $N = p^s q$. These attacks, like those of

Ciet *et al.*, also require that the public and private exponents be defined modulo $\phi(N)$ rather than $\mathrm{lcm}(p-1, q-1)$. In addition to these attacks, May also presents some lattice-based partial key exposure attacks for the CRT-exponent d_p when the public exponent is small. The attacks require knowledge of a $\frac{1}{s+1}$-fraction of the MSB or LSB of d_p. Other than factoring, this is the only known attack on Multi-power RSA.

Bibliography

[1] O. Acıiçmez, Ç. K. Koç, and J.-P. Seifert. On the power of simple branch prediction analysis. Cryptology ePrint Archive, Report 2006/351, 2006. [`http://eprint.iacr.org/`].

[2] M. Ajtai. The shortest vector problem in l_2 is NP-hard for randomized reductions. In *Proc. of 30th STOC*, pages 99–108. ACM, 1998.

[3] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. of 33rd STOC*, pages 601–610. ACM, 2001.

[4] M. Bellare and P. Rogaway. Optimal asymmetric encryption – how to encrypt with RSA. In *Advances in Cryptology - EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer-Verlag, 1995.

[5] S. M. Bellovin. The prehistory of public key cryptography, January 2006. [`http://www.cs.columbia.edu/~smb/nsam-160/`].

[6] D. Bleichenbacher and A. May. New attacks on RSA with small CRT-exponent. In *Public Key Cryptography - PKC 2006*, volume 3968 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, 2006.

[7] J. Blömer and A. May. Low secret exponent RSA revisited. In *Cryptography and Lattices, CaLC 2001*, volume 2146 of *Lecture Notes in Computer Science*, pages 4–19. Springer-Verlag, 2001.

[8] J. Blömer and A. May. New partial key exposure attacks on RSA. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 27–43. Springer-Verlag, 2003.

[9] J. Blömer and A. May. A generalized Wiener attack on RSA. In *Public Key Cryptography - PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 1–13. Springer-Verlag, 2004.

[10] J. Blömer and A. May. A tool kit for finding small roots of bivariate polynomials over the integers. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 251–267. Springer-Verlag, 2005.

[11] M. Blum. A potential danger with low-exponent modular encryption schemes: Avoid encrypting exactly the same message to several people! Unpublished, March 16, 1983.

[12] D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices of the American Mathematical Society*, 46(2):203–213, 1999.

[13] D. Boneh and G. Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. In *Advances in Cryptology - EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 1–11. Springer-Verlag, 1999.

[14] D. Boneh and G. Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. *IEEE Transactions on Information Theory*, 46(4):1339–1349, July 2000.

[15] D. Boneh, G. Durfee, and Y. Frankel. An attack on RSA given a small fraction of the private key bits. In *Advances in Cryptology - ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 25–34. Springer-Verlag, 1998.

[16] D. Boneh, G. Durfee, and Y. Frankel. Exposing an RSA private key given a small fraction of its bits, 2001. Revised and extended version of proceedings of ASIACRYPT '98 [15].
[`http://crypto.stanford.edu/~dabo/abstracts`].

[17] D. Boneh, G. Durfee, and N. A. Howgrave-Graham. Factoring $N = p^r q$ for large r. In *Advances in Cryptology – Proceedings of CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 326–337. Springer-Verlag, 1999.

[18] D. Boneh, A. Joux, and P. Q. Nguyễn. Why textbook ElGamal and RSA encryption are insecure. In *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 30–44. Springer-Verlag, 2000.

[19] D. Boneh and H. Shacham. Fast variants of RSA. *CryptoBytes*, 5(1):1–9, 2002.

[20] D. Boneh and R. Venkatesan. Breaking RSA may not be equivalent to factoring. In *Advances in Cryptology - EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 59–71. Springer-Verlag, 1099.

[21] D. R. L. Brown. Breaking RSA may be as difficult as factoring. Cryptology ePrint Archive, Report 2005/380, 2005. [`http://eprint.iacr.org/`].

[22] J. W. S. Cassels. *An Introduction to the Geometry of Numbers.* Springer-Verlag, second corrected edition, 1971.

[23] C.-Y. Chen, C.-C. Chang, and W.-P. Yang. Cryptanalysis of the secret exponent of the RSA scheme. *Journal of Information Science and Engineering*, 12:277–290, 1996.

[24] C.-Y. Chen, C.-Y. Ku, and D. C. Yen. Cryptanalysis of large RSA exponent by using the LLL algorithm. *Applied Mathematics and Computation*, 169:516–525, 2005.

[25] M. Ciet, F. Koeune, F. Laguillaumie, and J.-J. Quisquater. Short private exponent attacks on fast variants of RSA. UCL Crypto Group Technical Report Series CG-2002/4, Université Catholique de Louvain, 2002. [`http://www.dice.ucl.ac.be/crypto/tech_reports/`].

[26] C. C. Cocks. A note on 'non-secret encryption'. CESG Research Report, November 1973.

[27] H. Cohen. *A Course in Computational Algebraic Number Theory.* Springer-Verlag, second edition, 1995.

[28] H. Cohn and A. Kumar. Optimality and uniqueness of the Leech lattice among lattices, arXiv:math.MG/0403263. [`http://arxiv.org/abs/math.MG/0403263`].

[29] T. Collins, D. Hopkins, S. Langford, and M. Sabin. Public key cryptographic apparatus and method. United States Patent 5,848,159, December 8, 1998. (Filed January 16, 1997).

[30] Compaq Computer Corperation. Cryptography using Compaq multiprime technology in a parallel processing environment, 2000. [`ftp://ftp.compaq.com/pub/solutions/CompaqMultiPrimeWP.pdf`].

[31] J. Conway and N. Sloane. *Sphere Packings, Lattices and Groups*, volume 290 of *Grundlehren der mathematischen Wissenschaften.* Springer-Verlag, 3rd edition, 1999.

[32] D. Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 178–189. Springer-Verlag, 1996.

[33] D. Coppersmith. Finding a small root of a univariate modular equation. In *Advances in Cryptology - EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 155–165. Springer-Verlag, 1996.

[34] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997.

[35] J.-S. Coron. Finding small roots of bivariate integer polynomial equations revisited. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 492–505. Springer-Verlag, 2004.

[36] J.-S. Coron and A. May. Deterministic polynomial time equivalent of computing the RSA secret key and factoring. *Journal of Cryptology*, 20(1):39–50, January 2007.

[37] R. Crandall and C. Pomerance. *Prime Numbers - A Computational Perspective.* Springer, second edition edition, 2005.

[38] G. I. Davida. Chosen signature cryptanalysis of the RSA (MIT) public key cryptosystem. Technical Report TR–CS–82–2, Department of EECS, University of Wisconsin, Milwaukee, October 1982. [`http://www.uwm.edu/~davida/papers/chosen/`].

[39] J. M. DeLaurentis. A further weakness in the common modulus protocol for the RSA cryptoalgorithm. *Cryptologia*, 8(3):253–259, July 1984.

[40] D. E. Denning. Digital signatures with RSA and other public-key cryptosystems. *Communications of the ACM*, 27(4):388–392, April 1984.

[41] W. Diffie. The first ten years of public-key cryptography. *Proceedings of the IEEE*, 76(5):560–577, May 1988.

[42] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory IT-22*, pages 644–654, Nov. 1976.

[43] A. Dujella. Continued fractions and RSA with small secret exponent. *Tatra Mt. Math. Publ.*, 29:101–112, 2004.

[44] J. H. Ellis. The possibility of non-secret digital encryption. CESG Research Report, January 1970.

[45] J. H. Ellis. The story of non-secret encryption. CESG Research Report, 1987.

[46] M. Ernst, E. Jochemsz, A. May, and B. de Weger. Partial key exposure attacks on RSA up to full size exponents. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 371–387. Springer-Verlag, 2005.

[47] S. D. Galbraith, C. Heneghan, and J. F. McKee. Tunable balancing of RSA. In *Information Security and Privacy, 10th Australasian Conference, ACISP 2005*, volume 3574 of *Lecture Notes in Computer Science*, pages 280–292. Springer-Verlag, 2005.

[48] S. D. Galbraith, C. Heneghan, and J. F. McKee. Tunable balancing of RSA, 2005. Updated version of ACISP 2005 paper [47]. [`http://www.isg.rhul.ac.uk/~sdg/full-tunable-rsa.pdf`].

[49] M. Gardner. A new kind of cipher that would take millions of years to break. *Scientific American*, 237:120–124, August 1977.

[50] M. Girault. An identity-based identification scheme based on discrete logarithms modulo a composite number. In *Advances in Cryptology – Proceedings of EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 481–486. Springer-Verlag, 1991.

[51] M. Girault, P. Toffin, and B. Vallée. Computation of approximate L-th roots modulo n and application to cryptography. In *Advances in Cryptology – Proceedings of CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 403–418. Springer-Verlag, 1990.

[52] A. Granville and G. Martin. Prime number races. *American Mathematical Monthly*, 113:1–33, January 2006.

[53] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics.* Springer-Verlag, second corrected edition, 1993.

[54] P. M. Gruber and C. G. Lekkerkerker. *Geometry of Numbers*, volume 37 of *North-Holland Mathematical Library.* North-Holland, second edition, 1987.

[55] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers.* Oxford University Press, fourth edition, 1960.

[56] J. Håstad. On using RSA with low exponent in a public key network. In *Advances in Cryptology - CRYPTO '85*, volume 218 of *Lecture Notes in Computer Science*, pages 403–408. Springer-Verlag, 1986.

[57] J. Håstad. Solving simultaneous modular equations of low degree. *SIAM Journal on Computing*, 17(2):336–341, April 1988.

[58] M. J. Hinek. Lattice attacks in cryptography: A partial overview. CACR Technical Report CACR 2004-08, Centre for Applied Cryptographic Research, University of Waterloo, 2004. [`http://www.cacr.math.uwaterloo.ca/`].

[59] M. J. Hinek. New partial key exposure attacks on RSA revisited. CACR Technical Report CACR 2004–02, Centre for Applied Cryptographic Research, University of Waterloo, 2004. [`http://www.cacr.math.uwaterloo.ca/`].

[60] M. J. Hinek. (Very) large RSA private exponent vulnerabilities. CACR Technical Report CACR 2004-01, Centre for Applied Cryptographic Research, University of Waterloo, 2004. [`http://www.cacr.math.uwaterloo.ca/`].

[61] M. J. Hinek. Small private exponent partial key-exposure attacks on multiprime RSA. CACR Technical Report CACR 2005–16, Centre for Applied Cryptographic Research, University of Waterloo, 2005. [`http://www.cacr.math.uwaterloo.ca/`].

[62] M. J. Hinek. Another look at small RSA exponents. In *Topics in Cryptology - CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 82–98. Springer-Verlag, 2006.

[63] M. J. Hinek. On the security of multi-prime RSA. CACR Technical Report CACR 2006–16, Centre for Applied Cryptographic Research, University of Waterloo, 2006.
[http://www.cacr.math.uwaterloo.ca/].

[64] M. J. Hinek, M. K. Low, and E. Teske. On some attacks on multi-prime RSA. In *Selected Areas in Cryptography – SAC 2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 385–404. Springer-Verlag, 2003.

[65] M. J. Hinek and D. R. Stinson. An inequality about factors of multivariate polynomials. CACR Technical Report CACR 2006–15, Centre for Applied Cryptographic Research, University of Waterloo, 2006.
[http://www.cacr.math.uwaterloo.ca/].

[66] N. Howgrave-Graham. Finding small roots of univariate modular equations revisited. In *Coding and Cryptography, 6th IMA International Conference*, volume 1355 of *Lecture Notes in Computer Science*, pages 131–142. Springer-Verlag, 1997.

[67] N. Howgrave-Graham and J.-P. Seifert. Extending Wiener's attack in the presence of many decrypting exponents. In *Secure Networking - CQRE (Secure) '99*, volume 1740 of *Lecture Notes in Computer Science*, pages 153–166. Springer-Verlag, 1999.

[68] E. Jochemsz. An overview of known and new results on finding small roots of polynomials. Early version of [69], 2006.

[69] E. Jochemsz and A. May. A strategy for finding roots of multivariate polynomials with new applications in attacking RSA variants. In *Advances in Cryptology - ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 267–282. Springer-Verlag, 2006.

[70] E. Jochemsz and A. May. A polynomial time attack on standard RSA with private CRT-exponents smaller than $N^{0.073}$. Preprint, 2007.

[71] A. K. Lenstra. Generating RSA moduli with a predetermined portion. In *Advances in Cryptology - ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, pages 1–10. Springer-Verlag, 1998.

[72] A. K. Lenstra. Unbelievable security. Matching AES security using public key systems. In *Advances in Cryptology - ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 67–86, 2001.

[73] A. K. Lenstra and B. M. M. de Weger. Twin RSA. In *Progress in Cryptology - Mycrypt 2005*, volume 3715 of *Lecture Notes in Computer Science*, pages 222–228. Springer-Verlag, 2005.

[74] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[75] C. H. Lim and P. J. Lee. Security and performance of server-aided RSA computation protocols. In *Advances in Cryptology – Proceedings of CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 70–83. Springer-Verlag, 1995.

[76] S. Lim, S. Kim, I. Yie, and H. Lee. A generalized Takagi-cryptosystem with a modulus of the form $p^r q^s$. In *Progress in Cryptology - INDOCRYPT 2000*, volume 1977 of *Lecture Notes in Computer Science*, pages 283–294. Springer-Verlag, 2000.

[77] L. Lovász. *An Algorithmic Theory of Numbers, Graphs and Convexity*, volume 50 of *CBMS*. SIAM, 1986.

[78] Maplesoft. Maple 10. [`http://www.maplesoft.com`].

[79] J. Martinet. *Perfect Lattices in Euclidean Spaces*. Springer-Verlag, 2003. English version of the Éditions Masson 1996 French edition.

[80] A. May. Cryptanalysis of unbalanced RSA with small CRT-exponent. In *Advances in Cryptology - CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 242–256. Springer-Verlag, 2002.

[81] A. May. Computing the RSA secret key is deterministic polynomial time equivalent to factoring. In *Advances in Cryptology - CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 213–219. Springer-Verlag, 2004.

[82] A. May. Secret exponent attacks on RSA-type schemes with moduli $n = p^r q$. In *Public Key Cryptography - PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 218–230. Springer-Verlag, 2004.

[83] J. McKee and R. G. E. Pinch. Further attacks on server-aided RSA cryptosystems, 1998.
[`http://citeseer.ist.psu.edu/388295.html`].

[84] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. Discrete Mathematics and Its Applications. CRC Press, 1996.

[85] R. C. Merkle. Secure communications over insecure channels. *Communications of the ACM*, pages 294–299, April 1978. Submitted 1975.

[86] D. Micciancio. The shortest vector problem is NP-hard to approximate within some constant. *SIAM Journal on Computing*, 30(6):2008–2035, 2001. A preliminary version appeared in Proc. of the 39th (FOCS '98), pages 92–98, 1998.

[87] M. Mignotte. An inequality about factors of polynomials. *Mathematics of Computation*, 20(128):1153–1157, October 1974.

[88] G. L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13:300–317, 1976.

[89] J. W. Milnor and D. Husemöller. *Symmetric Bilinear Forms*. Springer-Verlag, 1973.

[90] J. H. Moore. Protocol failures in cryptosystems. *Proceedings of the IEEE*, 76(5):594–602, May 1988.

[91] J. A. Muir. Techniques of side channel cryptanalysis. Master's thesis, University of Waterloo, 2001.

[92] P. Q. Nguyễn and D. Stehlé. Low-dimensional lattice basis reduction revisited. In *Algorithmic Number Theory : 6th International Symposium, ANTS-VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 338–357. Springer-Verlag, 2004.

[93] P. Q. Nguyễn and D. Stehlé. Floating-point LLL revisited. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 215–233. Springer-Verlag, 2005.

[94] P. Q. Nguyễn and J. Stern. The two faces of lattices in cryptology. In *Cryptography and Lattices – Proceedings of CALC '01*, volume 2146 of *Lecture Notes in Computer Science*, pages 146–180. Springer-Verlag, 2001.

[95] C. D. Olds. *Continued Fractions*. New Mathematical Library. The L. W. Singer Company, 1963.

[96] E. Oswald and B. Preneel. A survey on passive side-channel attacks and their countermeasures for the NESSIE public-key cryptosystems, January 2003.
[`http://www.cosic.esat.kuleuven.ac.be/nessie/reports/phase2/`].

[97] J. A. Proos. *Imperfect Decryption and Partial Information Attacks in Cryptography.* PhD thesis, University of Waterloo, 2003.

[98] G. Qiao and K.-Y. Lam. RSA signature algorithm for microcontroller implementation. In *Smart Card Research and Applications, CARDIS '98*, volume 1820 of *Lecture Notes in Computer Science*, pages 353–356. Springer-Verlag, 1998.

[99] J.-J. Quisquater and C. Couvreur. Fast decipherment algorithm for RSA public key cryptosystem. *Electronics Letters*, 18(21):905–907, October 1982.

[100] M. O. Rabin. Digitized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology, Laboratory for Computer Science, January 1979.

[101] R. L. Rivest and B. Kaliski. RSA problem, December 2003. To appear in *Encyclopedia of Cryptography and Security.*
[`http://theory.lcs.mit.edu/~rivest/publications.html`].

[102] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[103] R. L. Rivest, A. Shamir, and L. M. Adleman. Cryptographic communications system and method. United States Patent 4,405,829, September 20, 1983. (Filed December 14, 1977).

[104] RSA Laboratories. Public Key Cryptography Standards PKCS #1 v2.1: RSA cryptography standard, June 2001.
[`http://www.rsasecurity.com/rsalabs/`].

[105] C. P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53:201–224, 1987.

[106] V. Shoup. NTL: A library for doing number theory, version 5.3.1, 2002. [`http://shoup.net/ntl/`].

[107] V. Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, 2005.

[108] C. L. Siegel. *Lectures on the Geometry of Numbers*. Springer-Verlag, 1989.

[109] G. J. Simmons. A "weak" privacy protocol using the RSA crypto algorithm. *Cryptologia*, 7(2):180–181, April 1983.

[110] Statistics Canada. Electronic commerce and technology. *The Daily*, April 20, 2006. [`http://www.statcan.ca/`].

[111] H.-M. Sun, M. J. Hinek, and M.-E. Wu. On the design of rebalanced-RSA. CACR Technical Report CACR 2005-35, Centre for Applied Cryptographic Research, University of Waterloo, 2005. [`http://www.cacr.math.uwaterloo.ca/`].

[112] H.-M. Sun and M.-E. Wu. An approach towards rebalanced RSA-CRT with short public exponent. Cryptology ePrint Archive, Report 2005/053, 2005. [`http://eprint.iacr.org/`].

[113] H.-M. Sun and M.-E. Wu. Design of rebalanced RSA-CRT for fast encryption. Information Security Conference, 2005.

[114] H.-M. Sun, M.-E. Wu, W.-C. Ting, and M. J. Hinek. Dual RSA and its applications, 2006. Submitted.

[115] H.-M. Sun and C.-T. Yang. RSA with balanced short exponents and its application to entity authentication. In *Public Key Cryptography - PKC 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 199–215. Springer-Verlag, 2005.

[116] T. Takagi. Fast RSA-type cryptosystem modulo $p^k q$. In *Advances in Cryptology - CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 318–326. Springer-Verlag, 1998.

[117] T. Takagi. A fast RSA-type public-key primitive modulo $p^k q$ using Hensel lifting. *IEICE Transactions*, 87-A(1):94–101, 2004.

[118] B. Vallée, M. Girault, and P. Toffin. How to guess ℓ-th roots modulo n by reducing lattice bases. In *Proceedings of Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, 6th International Conference, AECC-6, 1988*, volume 357 of *Lecture Notes in Computer Science*, pages 427–442. Springer-Verlag, 1989.

[119] S. A. Vanstone and R. J. Zuccherato. Short RSA keys and their generation. *J. Cryptology*, 8(2):101–114, March 1995.

[120] E. R. Verheul and H. C. A. van Tilborg. Cryptanalysis of 'less short' RSA secret exponents. *Appl. Algebra Eng. Commun. Comput.*, 8(5):425–435, July 1997.

[121] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, 1999.

[122] A. G. Voyiatzis. An introduction to side-channel cryptanalysis of RSA. *ACM Crossroads*, 11(3), Spring 2005. [http://www.acm.org/crossroads/xrds11-3/].

[123] M. J. Wiener. Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory*, 36(3):553–558, May 1990.

[124] M. J. Williamson. Non-secret encryption using a finite field. CESG Research Report, January 1974.

[125] M. J. Williamson. Thoughts on cheaper non-secret encryption. CESG Research Report, August 1976.

[126] P. Zimmerman. Integer factoring records, May 2005. [http://www.loria.fr/~zimmerma/records/factor.html].