ON SOME ATTACKS ON MULTI-PRIME RSA

M JASON HINEK, MO KING LOW, AND EDLYN TESKE

Abstract. Using more than two factors in the modulus of the RSA cryptosystem has the arithmetic advantage that the private key computations can be speeded up using Chinese remaindering. At the same time, with a proper choice of parameters, one does not have to work with a larger modulus to achieve the same level of security in terms of the difficulty of the integer factorization problem. However, numerous attacks on specific instances on the RSA cryptosystem are known that apply if, for example, the decryption or encryption exponent are chosen too small, or if partial knowledge of the private key is available. Little work is known on how such attacks perform in the multi-prime case. It turns out that for most of these attacks it is crucial that the modulus contains exactly two primes. They become much less effective, or fail, when the modulus factors into more than two distinct primes.

1. Introduction

The RSA cryptosystem, due to Rivest, Shamir and Adleman [RSA78], is one of the most popular public key cryptosystems and widely used to ensure privacy and authenticity of electronic data. In the first place, the security of RSA is based on the difficulty of the following Integer Factorization Problem (IFP): given an integer N that is the product of two primes p and q of approximately the same size, find p and q. As a consequence, the choice of RSA parameters to achieve a certain level of security is based on the estimated current and future performance of integer factorization algorithms. On the other hand, numerous attacks have been developed that are unrelated to the IFP and show vulnerabilities of specific instances of the RSA cryptosystem. Boneh [Bon99] gives an excellent survey on this matter.

Little work has been reported on how such attacks apply to multi-prime RSA, i.e., where N is a product of more than two primes. Of most practical interest are the cases of 3- and 4-prime RSA. Commercial implementations of 3-prime RSA [CHLS97] exist. The use of more than two primes in the RSA cryptosystem has the advantage that the private key operations can be speeded up using the Chinese Remainder Theorem. An easy calculation shows that compared with 2-prime RSA, the theoretical speed-up is by a factor of 9/4 for 3-prime RSA, and 4 for 4-prime RSA. (In practice, a speed-up of 1.73 for 3-prime RSA has been achieved [BS02].) This is under the assumption that the size of the modulus N is the same in all systems. In fact, as long as the running time for the Elliptic Curve Method to find the smallest factor of $N = \prod_{i=1}^{r} p_i$ exceeds the running time for the Number Field Sieve to factor N, multi-prime RSA does not require a larger modulus. For example, we can conclude from [Len01, Tab. 3] that currently a 2048-bit modulus $N = pq$ offers roughly the same level of security against factoring algorithms as a 2048-bit modulus for 3-prime RSA.

In this report, we take selected attacks discussed in [Bon99] and examine whether and how efficiently they can be applied to multi-prime RSA. We now briefly describe multi-prime RSA in a simplified

Date: July 18, 2002.
1991 *Mathematics Subject Classification.* Primary 94A60. Secondary 11Y16.

version, which works about just the same as RSA. Throughout this paper, we use the notation as follows. Let r be an integer ≥ 2, and let N be the product of r pairwise distinct large primes $p_1, \ldots, p_r$ of roughly the same size. For convenience, if $r = 2$ we work with $p := p_1$ and $q := p_2$. The bitsize of N is always denoted by n. Let $\phi(N) = \prod_{i=1}^{r}(p_i - 1)$ be the order of the multiplicative group $\mathbb{Z}_N^*$, and let e, d be two integers satisfying $ed \equiv 1 \pmod{\phi(N)}$. We call N the *modulus*, e the *encryption* (or: *public*) *exponent* and d the *decryption* (or: *private*) *exponent*. The pair (N, e) is the *public key*, the pair (N, d) is the *private key*. To encrypt a message $M \in \mathbb{Z}_N^*$, one computes the ciphertext $C = M^e \mod N$, which then can be decrypted by computing $C^d = M^{ed} = M \mod N$. When RSA is used for signing, the private key is used to generate a digital signature $S = M^d \mod N$ of the message $M \in \mathbb{Z}_N^*$, and the public key is used for verification, where it is checked that $S^e \equiv M \pmod{N}$. As said, this description of the RSA encryption and signature algorithms is highly simplified. In practice, proper padding and, for signature generation, hashing are indispensable.

Just as in the case of 2-prime RSA, factoring the modulus is equivalent to exposing the private key for 3- and 4-prime RSA (see Section 3). In particular, this means that the common modulus attack can be mounted in these cases as well. On the other hand, most of the attacks discussed in [Bon99] make such extensive use of the fact that the modulus is a product of exactly two primes that they become much less effective in the multi-prime case, or don't seem to work at all. For example, in Section 4 we study low private exponent attacks based on continued fractions and on lattice reduction. For the continued fraction attack, we show that the bound on the size of the decryption exponent for the attack to work is $O(N^{1/2r})$, and a similar behaviour can be observed for the lattice based attacks. Another example is the following: while 2-prime RSA with small encryption exponent can leak half of the most significant bits of the decryption exponent, this fraction goes down to $1/r$ for r-prime RSA. See Section 5. In 2-prime RSA, if the public exponent is small, a polynomial time method by Boneh, Durfee and Frankel exists that completely recovers the private exponent once it is partially exposed. However, in the 3- and 4-prime case these methods become totally ineffective, as we show in Section 5.1. A partial key exposure attack for a medium public exponent (i.e., $e < \sqrt{N}$), also by Boneh, Durfee and Frankel, fails in the multi-prime case because instead of solving a quadratic congruence, solving congruences of degree r where not all coefficients are known is required. Another, weaker, partial key exposure attack by the same authors carries over, but is applicable only for public exponents up to $N^{1/r}$ rather than $\sqrt{N}$. See Section 5.2. These results suggest that multi-prime RSA does not only allow for faster decryption using Chinese remaindering, but is also somewhat more secure than 2-prime RSA.

Our paper is organized as follows. First we introduce and discuss precise conditions on the relative sizes of the prime factors; these conditions correspond to the usual assumptions in the 2-prime case. We also provide some background needed for lattice-based attacks. Then, in Section 3, we discuss the equivalence of factoring and finding the secret key. Low decryption exponent attacks are the subject of Section 4. In Section 5 we study partial key exposure attacks. We conclude with Section 6.

2. Preliminaries

We introduce some more notation. We call the congruence $ed \equiv 1 \pmod{\phi(N)}$ the *public/private key relation*. If $0 < d, e < \phi(N)$, there exists a unique integer k, $0 < k < \min\{e, d\}$, such that

$$ed - k\phi(N) = 1 \ . \tag{2.1}$$

This equation is called the *public/private key equation*, and in the sequel, k is always the value defined through (2.1).

Throughout this work we make the following assumptions on the primes in the RSA modulus. First, the primes are labeled in increasing order. That is,

$$p_i < p_{i+1} \tag{2.2}$$

for all $i \geq 1$. Second, the primes satisfy

$$4 < N^{1/r}/2 < p_1 < N^{1/r} < p_r < 2N^{1/r} \ . \tag{2.3}$$

This second assumption guarantees that we have *balanced primes*. That is, all primes in the modulus are of roughly the same size and thus are equally hard to find by the Elliptic Curve Method. On the other hand, other factoring algorithms (esp. the Number Field Sieve) cannot exploit (2.2), (2.3) for speed-ups.

For an r-prime RSA modulus N, (2.2) and (2.3) imply that

$$N - \phi(N) < (2r-1)N^{1-1/r} \ . \tag{2.4}$$

This is because with Euler's phi function,

$$\phi(N) = N - \sum_{i=1}^{r} N/p_i + \sum_{\substack{i,j=1 \\ i<j}}^{r} N/(p_i p_j) - \sum_{\substack{i,j,k=1 \\ i<j<k}}^{r} N/(p_i p_j p_k) + \cdots + (-1)^r \ , \tag{2.5}$$

we have $N - \phi(N) < \sum_{i=1}^{r} N/p_i$. Now, since $p_r > N^{1/r}$ and $p_i \geq p_1 > N^{1/r}/2$ for $i = 1, \ldots, r-1$, we have that $N/p_r < N^{1-1/r}$, and $N/p_i < 2N^{1-1/r}$ for $i = 1, \ldots, r-1$. Combining these inequalities gives (2.4). Further, since N is an n-bit modulus, we can express (2.4) as

$$N - \phi(N) < (2r)2^{n(1-1/r)} = 2^{n-n/r+1+\log_2 r} \ . \tag{2.6}$$

Some of the attacks we consider use lattices and lattice reduction algorithms. We now give some notation and facts, for which we follow [DN00] and [BD00].

Let $u_1, \ldots, u_w \in \mathbb{Z}^m$ with $w \leq m$. The set $L = \{\sum_{i=1}^{w} a_i u_i \mid a_i \in \mathbb{Z}\}$ of all integer linear combinations of the u_i's is a lattice. It is called the lattice *spanned* by $\langle u_1, \ldots, u_w \rangle$. Further, if the vectors $u_1, \ldots, u_w$ are linearly independent over $\mathbb{Z}$, then $\langle u_1, \ldots, u_w \rangle$ is called a *basis* of L. There are an infinite number of bases for each lattice L. All bases for a given lattice share two common parameters: the lattice rank and the lattice volume. The lattice *rank* (or *dimension*) is the number, w, of vectors in the basis. The lattice *volume* (or *determinant*), denoted by $\mathrm{vol}(L)$, is the w-dimensional volume of the parallelepiped spanned by the u_i's. If $w = m$ the lattice is said to have full rank. In the full rank case, the volume is equal to the absolute value of the determinant of any basis.

Given any basis $\langle u_1, \ldots, u_w \rangle$ of a lattice, we can use the LLL lattice reduction algorithm [LLL82] to create a new basis such that the vectors, say $\langle b_1, \ldots, b_w \rangle$, of the new basis are in some way "small". This new basis is called an *LLL-reduced* basis. The main features of any *LLL*-reduced basis $\langle b_1, \ldots, b_w \rangle$ of a lattice L spanned by $\langle u_1, \ldots, u_w \rangle$ in $\mathbb{Z}^m$ are:

(1) $\|b_1\| \leq 2^{w/2}\mathrm{vol}(L)^{1/w}$ and $\|b_2\| \leq 2^{(w-1)/2}\mathrm{vol}(L)^{1/(w-1)}$.
(2) The basis can be computed in time $O(mw^5 \log(\max\{\|u_i\|_\infty\}))$.

We will treat the LLL lattice reduction algorithm as a black box.

Boneh, Durfee and Frankel showed how to use lattice reduction to factor a 2-prime RSA modulus given that the $n/4$ least significant bits of one of the factors are known.

Theorem 2.1 ([BDF98]). *Let $N = pq$ be an n-bit* RSA *modulus. Let $s \geq 2^{n/4}$ be given and suppose $p \bmod s$ is known. Then it is possible to factor N in time polynomial in n.*

The runtime of the algorithm needed to factor the modulus in Theorem 2.1 will be denoted by $T_{\mathrm{C}}(n)$. The C is in reference to the fact that the above result follows from Coppersmith's work on finding small roots of bivariate polynomials [Cop97].

3. Equivalence of Factoring and Exposing the Private Key

In 2-prime RSA, computing the decryption exponent d from the public key is equivalent to factoring N. This result can be extended to $r = 3, 4$. Given the public and private exponents, we find a multiple of $\phi(N)$ from (2.1). This multiple $k\phi(N)$ is fed into a Las Vegas algorithm that produces a factor of N by calculating non-trivial square roots of 1 modulo N. This works as follows: We write $k\phi(N) = 2^t u$ with u odd. Then, for a randomly chosen integer $w \in [2, N-2]$ we compute $w^{2^s u}$ for $s = 0, 1, \ldots,$ until $w^{2^s u} \equiv 1 \pmod N$. If that happens already for $s = 0$, the algorithm outputs FAILURE. If $s \geq 1$ and $w^{2^{s-1}u} \not\equiv -1 \pmod N$, then $w^{2^{s-1}u}$ is a non-trivial square root of 1 mod N and thus $\gcd(N, w^{2^{s-1}u} + 1)$ is a non-trivial factor of N. If $w^{2^{s-1}u} \equiv -1 \pmod N$, the output is FAILURE.

In the 2-prime case, an analysis in [Sti95] shows that the probability of finding a (prime) factor of N is at least 0.5. We proceed analogously for an r-prime modulus N ($r \geq 2$), and estimate the number of integers $w \in [2, N-2]$ that cause FAILURE. For this, we write $p_i - 1 = 2^{h_i} q_i$ with q_i odd. Then there exist $\prod_{i=1}^r q_i$ integers $w \in [1, N-1]$ for which $w^u \equiv 1 \pmod N$, and at most $(\prod_{i=1}^r q_i) \sum_{i=0}^{h-1} 2^{ri}$ integers $w \in [1, N-1]$ for which $w^{2^{s-1}u} \equiv -1 \pmod N$ for some integer $s \in [1, t]$, where $h = \min\{h_i : i = 1, \ldots, r\}$. The total number of values of w that cause FAILURE can then easily be bounded by $N/2^{r-1}$ (see [Sti95, Low02] for details). Thus, the probability that the above algorithm outputs a factor of N is at least $1 - 1/2^{r-1}$. In the 2-prime case, disclosure of one factor leads to the complete factorization of N.

3.1. **3-Prime RSA.** The probability of finding a factor p of N with the above algorithm is at 0.75. Then either p or N/p is prime, and the problem of factoring N can be reduced to the 2-prime case.

3.2. **4-Prime RSA.** The probability of finding a factor x of N is at least 0.875. However, it may occur that both x and N/x are composite. We consider this event a FAILURE since then the 4-prime case cannot be reduced to a smaller-prime case: we cannot determine a multiple of $\phi(x)$ or $\phi(N/x)$. To guarantee a sufficient probability that a *prime* factor is revealed, we put $x_s = \gcd(w^{2^{s-1}u} + 1, N)$ and j such that $h_j = \max\{h_i : i = 1, \ldots, 4\}$, and we give a lower bound on the number of $w \in [1, N-1]$ for which $x_s = p_j$ for some integer $s \in [1, t]$. From the Chinese Remainder Theorem,

$$\left.\begin{array}{ll} w^{2^s u} \equiv 1 & \pmod N \\ \text{and} & \\ x_s = p_j & \end{array}\right\} \iff \left\{\begin{array}{ll} w^{2^{s-1}u} \equiv -1 & \pmod{p_j} \\ \text{and} & \\ w^{2^{s-1}u} \equiv \;\;\, 1 & \pmod{p_i} \text{ for } i \neq j\ . \end{array}\right. \tag{3.1}$$

Note that if our Las Vegas algorithm chooses an integer w for which (3.1) holds for some s, then its output will indeed be x_s. Now let $s = h_j$. Then the number of w modulo p_j for which $w^{2^{s-1}u} \equiv -1 \pmod{p_j}$ is $2^{s-1}q_j = (p_j - 1)/2$. Since $s = h_j \geq h_i$, the number of w modulo p_i $(i \neq j)$ for which $w^{2^{s-1}u} \equiv 1 \pmod{p_i}$ is at least $(p_i - 1)/2$. Thus, the number of w mod N for which (3.1) holds for $s = h_j$ is at least $\phi(N)/16$. By (2.5), $\phi(N) > N - 7N^{3/4}$, so that the probability that the output of our algorithm is a prime factor is at least $\frac{1}{16} - \frac{7}{16N^{1/4}}$, which is > 0.0624 for $N > 2^{49}$. Of course, this lower bound is far away from being sharp, but sufficient for our purpose: we expect to find a prime factor of N with probability at least $1 - 10^{-6}$ after trying at most 215 values of w. Once a prime factor of N is found, the 4-prime case can be reduced to the 3-prime case.

4. Low Private Exponent Attacks

An advantage of choosing a low private exponent, d, is that decryption or digital signatures can be made faster (the cost for modular exponentiation grows linearly in $\log d$). This is important especially in constrained environments such as smartcards. But a private exponent that is too small is insecure because it leads to the factorization of the modulus, as we discuss in Sections 4.1 and 4.2.

4.1. **Continued Fractions Attack.** A result of Wiener [Wie90] states that if $N = pq$ and the private exponent $d < \frac{1}{3}N^{1/4}$, then, given the public key (N, e) the prime factors of N can be recovered in time polynomial in N. The underlying attack is based on the following property of approximation using continued fractions: If α, β are integers and x is a real number, and $|\alpha/\beta - x| < 1/(2\beta^2)$, then α/β is a convergent of x [HW60]. Convergents of x are computed using the continued fraction algorithm, which is based on Euclid's algorithm.

We generalize Wiener's attack to the multi-prime case. Let k be as in (2.1). Then

$$0 < \frac{k}{d} - \frac{e}{N} = \frac{k(N - \phi(N)) - 1}{dN} .$$

Using (2.4) and that $k < d$, we find that the right-hand side is strictly less than $(2r-1)/N^{1/r}$. Now

$$(2r-1)/N^{1/r} < 1/(2d^2) \qquad \Longleftrightarrow \qquad d < N^{1/2r}/\sqrt{2(2r-1)} .$$

Thus, if the last inequality holds, then on applying the continued fraction algorithm to e/N, we will obtain k/d as one of the convergents. To obtain the private exponent, an adversary simply has to test all convergents k'/d' of e/N. In fact, only the convergents with even index need to be tested (where the expansion begins with $q_0 = \lfloor e/N \rfloor$), since these convergents are exactly those for which $k'/d' > e/N$ [Old63]. Note that all convergents, as well as k/d, are in their lowest terms. Now one substitutes k', d' into (2.1) to produce ϕ', a candidate for $\phi(N)$. In the 2-prime case one now can solve the quadratic equation $(x-p)(x-q) = x^2 - (N - \phi' + 1)x + N = 0$. If $\phi' = \phi(N)$, the roots of this equation will be p, q. If N is an r-prime modulus with $r > 2$, we use that $\phi(N)$ is divisible by 2^r. Only if this is the case for a candidate ϕ', one proceeds with testing d'. For this, let M be a random element in $\mathbb{Z}_N$. If $(M^e)^{d'} \equiv M \pmod{N}$, then there is a high probability that $d' = d$, which can be verified using the techniques from Section 3.

Since the continued fraction algorithm is polynomial time, and the number of convergents of the fraction e/N is $O(\log N)$, and the test for each candidate k'/d' costs polynomial time, the decryption exponent can be found in polynomial time. We thus have proven

Theorem 4.1. *Let N be an r-prime modulus and $d < N^{1/2r}/\sqrt{2(2r-1)}$. Given (N, e), the decryption exponent can be recovered in time polynomial in N.*

Experimental results show that the attack is effective even if d is slightly larger than $N^{1/2r}/\sqrt{2(2r-1)}$. Using Shoup's NT Library [Sho], we implemented Wiener's attack and applied it in the 3-prime case. We generated three 512-bit primes to produce a modulus of 1534 to 1536 bits. Thus, for d of 256 and more bits, the bound of Theorem 4.1 does not hold. Nevertheless, among 1000 trials with randomly chosen primes and 256-bit private exponents, the correct value of k/d was among the convergents of e/N in 344 cases. For 257-bit d, the success rate still was 93/1000, while for 258 and 259 bits the rates were 23/1000 and 7/1000, respectively. (The number of convergents was always less than 1000. Among the successful runs, the correct value of d was found after an average of 150 convergents.) Nevertheless, the continued fraction attack is outperformed by the lattice-based attacks by Boneh and Durfee, which we discuss next.

We conclude by noting that just as in the 2-prime case, a method of defense against the continued fraction attack is to work with a public exponent $e' = e + t\phi(N)$ for some large t. In the 2-prime case, $e' > N^{3/2}$ is sufficient to defeat the attack. This bound decreases as r increases. In fact, generalizing Wiener's analysis [Wie90] to the r-prime case gives the following theorem.

Theorem 4.2. *Let N be an RSA modulus and $e > N^{(r+1)/r}$. Given (N, e), the attack underlying Theorem 4.1 cannot reveal the decryption exponent, independent of the size of d.*

4.2. **Lattice Based Attacks.** A low private exponent attack by Boneh and Durfee [BD00] uses lattice reduction techniques. This attack renders 2-prime RSA insecure when the private exponent is less than N^δ, where, in the most efficient variant of the attack, $\delta = 0.292$ as $N \to \infty$. We show how the Boneh-Durfee attack and a modified approach due to Blömer and May [BM01] generalize to r-prime RSA, and obtain corresponding asymptotic upper bounds on the private exponent. Moreover, for $r = 2, 3, 4$ and various fixed sizes of N, we give explicit upper bounds on the private exponent for which the attack is guaranteed to work.

Following [BD00], with slight modifications, we begin with the public/private key equation $ed - k\phi(N) = 1$. Letting $s = \phi(N) - N$ and $A = N$, we get $ed - k(A + s) = 1$, and reduction modulo e yields $-k(A+s) \equiv 1 \pmod{e}$. Our aim is to solve this equivalence relation with two unknowns. In fact, if we knew s, then we could immediately compute the private exponent $d = e^{-1} \mod \phi(N)$, and factor N. We let $\alpha, \delta \in \mathbb{R}^+$ such that

$$e = N^\alpha \qquad \text{and} \qquad d < N^\delta (= e^{\delta/\alpha}) \ .$$

Then

$$k = (ed-1)/\phi(N) < ed/\phi(N) < 2ed/N = 2e^{1+(\delta-1)/\alpha} \ , \tag{4.1}$$

where we used that $\phi(N) > N/2$. Also, by (2.4), we have $|s| < (2r-1)e^{(1-1/r)/\alpha}$. Letting $a_r = 1 - 1/r$ we obtain

$$|s| < (2r-1)e^{a_r/\alpha} \ . \tag{4.2}$$

Thus, with $\epsilon_2 = \ln 2$ and $\epsilon_r = \ln(2r-1)$, we are trying to solve the following *small inverse problem*: find integers k and s satisfying

$$-k(A+s) \equiv 1 \pmod{e} \ , \quad |k| < e^{1+(\delta-1)/\alpha+\epsilon_2} \quad \text{and} \quad |s| < e^{a_r/\alpha+\epsilon_r} \ . \tag{4.3}$$

In order to be able to efficiently find $s = \phi(N) - N$ among the solutions of the small inverse problem, we need an efficient method to find all solutions, and we need that the number of solutions is sufficiently small.

4.2.1. *Solving The Small Inverse Problem.* In the following, identify a polynomial and its associated coefficient vector. In particular, we define the *norm* of a polynomial $h(x,y) = \sum_{i,j} a_{i,j}x^i y^j$ by $\|h(x,y)\| = (\sum_{i,j} |a_{i,j}|^2)^{1/2}$.

The small inverse problem (4.3) can be restated as follows: given a polynomial $f(x,y) = x(A+y)-1$, find (x_0, y_0) such that

$$f(x_0,y_0) \equiv 0 \pmod{e} \ , \quad |x_0| < X \ , \quad |y_0| < Y \ , \tag{4.4}$$

where $X = e^{1+\frac{\delta-1}{\alpha}+\epsilon_2}$, and $Y = e^{\frac{a_r}{\alpha}+\epsilon_r}$. Notice that $(-k, s)$ is a root of $f(x,y) \bmod e$. Now, a result of Howgrave-Graham allows us to transform the modular equation in (4.4) into an integer equation.

Theorem 4.3 ([HG97]). *Let $h(x,y) \in \mathbb{Z}[x,y]$ be a polynomial which is a sum of at most w monomials, and let $X, Y \in \mathbb{N}$. Suppose that $h(x_0,y_0) \equiv 0 \pmod{e^m}$ for some positive integer m, where $|x_0| < X$ and $|y_0| < Y$. Then, if $\|h(xX, yY)\| < e^m/\sqrt{w}$, the equation $h(x_0,y_0) = 0$ holds over the integers.*

Our goal now is to construct a small norm polynomial that has $(-k, s)$ as a root modulo e^m for some m. To this end, given a positive integer m, define the polynomials

$$g_{i,k}(x,y) := x^i f^k(x,y) e^{m-k} \quad \text{and} \quad h_{j,k}(x,y) := y^j f^k(x,y) e^{m-k} \ . \tag{4.5}$$

Now let (x_0, y_0) be a root of $f(x,y)$ modulo e. Then $g_{i,k}(x_0,y_0) \equiv h_{j,k}(x_0,y_0) \equiv 0 \pmod{e^m}$ for all $i,j,k \geq 0$. Thus, if we construct a lattice L using $g_{i,k}$ and $h_{j,k}$ as basis vectors, for various i,j,k, then all polynomials in the lattice have (x_0,y_0) as a root modulo e^m. Now recall from Section 2 that the first two polynomials b_1 and b_2 in an LLL-reduced basis satisfy $\|b_1\| \leq 2^{w/2}\mathrm{vol}(L)^{1/w}$ and $\|b_2\| \leq 2^{(w-1)/2}\mathrm{vol}(L)^{1/(w-1)}$. Therefore, if our lattice L satisfies

$$2^{(w-1)/2}\mathrm{vol}(L)^{1/(w-1)} < e^m/\sqrt{w} \ , \tag{4.6}$$

then b_1 and b_2 have norm small enough to satisfy Theorem 4.3, so that all small roots of b_1 and b_2 modulo e^m are also roots over the integers. This gives us two bivariate equations over the integers, which have at least one common small root, (x_0,y_0). Further, if these two polynomials are also algebraically independent, we can use the resultant of b_1 and b_2 to find their common roots (see [BD00] for details).

We now focus on generating a lattice with sufficiently small volume.

4.2.2. *The Boneh-Durfee Lattice* [BD00, Section 4]. Given integers $m \geq 1$ and $t \geq 0$, we construct the lattice as follows. For each $k = 0, \ldots, m$, use $g_{i,k}(xX, yY)$ for $i = 0, \ldots, m-k$ and $h_{j,k}(xX, yY)$ for $j = 0, \ldots, t$ as the basis vectors, with $g_{i,k}$ and $h_{j,k}$ as in (4.5). The lattice and basis will be denoted by $L_{BD}(m,t)$ and $B_{BD}(m,t)$, respectively. By construction, $L_{BD}(m,t)$ has full rank, and so the volume of the lattice equals the absolute value of determinant of the matrix generated by $B_{BD}(m,t)$. The dimension of the lattice is

$$w = (m+1)(m+2)/2 + t(m+1) \ .$$

Also, since this matrix is triangular, the determinant computation is straightforward and we have $\mathrm{vol}(L_{BD}(m,t)) = \det_x \cdot \det_y$, where $\det_x$ and $\det_y$ are the determinants of the submatrices corresponding to the $g_{i,k}$ and the $h_{j,k}$, respectively. We have

$$\det_x = e^{m(m+1)(m+2)/3} \cdot X^{m(m+1)(m+2)/3} \cdot Y^{m(m+1)(m+2)/6}, \tag{4.7}$$

$$\det_y = e^{tm(m+1)/2} \cdot X^{tm(m+1)/2} \cdot Y^{t(m+1)(m+t+1)/2}. \tag{4.8}$$

Substituting $X = e^{1+\frac{\delta-1}{\alpha}+\epsilon_2}$ and $Y = e^{\frac{a_r}{\alpha}+\epsilon_r}$ into (4.7) and (4.8) we obtain

$$\begin{aligned} \det_x &= e^{(2+\frac{\delta-1}{\alpha}+\epsilon_2+(\frac{a_r}{\alpha}+\epsilon_r)/2)m(m+1)(m+2)/3}, \\ \det_y &= e^{(2+\frac{\delta-1}{\alpha}+\epsilon_2)tm(m+1)/2+(\frac{a_r}{\alpha}+\epsilon_r)t(m+1)(m+t+1)/2}. \end{aligned}$$

Thus, $\mathrm{vol}(L_{BD}(m,t)) = e^C$, where

$$\begin{aligned} C = C(m,t,\alpha,\delta,r) &= (2+(\delta-1)/\alpha+\epsilon_2+(a_r/\alpha+\epsilon_r)/2)m(m+1)(m+2)/3 \\ &\quad +(2+(\delta-1)/\alpha+\epsilon_2)tm(m+1)/2+(a_r/\alpha+\epsilon_r)t(m+1)(m+t+1)/2. \end{aligned}$$

We now need to find values for m and t so that (4.6) holds. That is, we require m and t to satisfy

$$e^{C(m,t,\alpha,\delta,r)} < \frac{e^{m(w-1)}}{2^{(w-1)^2/2}w^{(w-1)/2}}. \tag{4.9}$$

Finding integer values for m and t that allow for the largest δ in (4.9) is a difficult problem, as (4.9) is highly nonlinear, α and e are variables, and we require $m \geq 1$ and $t \geq 0$. In the case $r = 2$, Boneh and Durfee [BD00] approximate the right-hand side of (4.9) by $e^{m(w-1)}$ and then find that $t = t_{\mathrm{opt}} = m(1-2\delta)/2$ minimizes the left-hand side of $C(m,t,\alpha,\delta,r) - m(w-1) < 0$. By letting $m \to \infty$ in this inequality, they obtain an upper bound $\delta_{\max}$ on δ such that, for N large, the attacks works if $d < N^{\delta_{\max}}$. For $\alpha = 1$, they obtain $\delta_{\max} = 7/6 - \sqrt{7}/3 \approx 0.2847$. Using [May02, Lemma 7] and [BD00], we can generalize this asymptotic bound and find that for arbitrary $\alpha > 0$ and $r = 2, 3, 4, \ldots$,

$$\delta_{\max}(\alpha, a_r) = 1 + \frac{1}{3}a_r - \frac{2}{3}\sqrt{a_r\,(a_r + 3\,\alpha)}. \tag{4.10}$$

(Recall that $a_r = 1 - 1/r$.) Next, we want to find upper bounds for δ for explicit values of N and α. For this, in a tedious but elementary computation we transform (4.9) to an inequality of the form $\delta < G(m,t,\alpha,r,N)$ (recall that $e = N^\alpha$) and numerically maximize G over $(m,t) \in \mathbb{N} \times \mathbb{N}_0$, for $r = 2, 3$, and 4. We show the results for $\alpha = 1$ (which represents the most common case in practice) in Table 1. We see that for r fixed and N increasing, the bounds converge to the asymptotic bounds (4.10), which are given in the penultimate row. Moreover, just as in the case of Wiener's continued fractions attack whose bounds are given in the last row, the upper bounds on δ decrease as the number of prime factors of N increases.

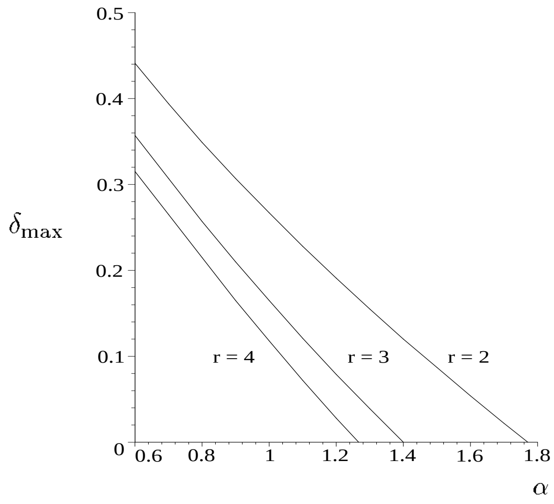
Table 2 illustrates how $\delta_{\max}$ varies with α. Here we give data for the 3-prime case with a 2000-bit modulus, which reflects the typical behaviour. We see that, roughly, $\delta \sim 1/\alpha$. Hence, with increasing public exponent $e = N^\alpha$, a smaller private exponent $d = N^\delta$ is required for the attack to work, up to a point (here: $\alpha = 1.4$) where the attack is not guaranteed to work for any d. For 2-prime RSA and 2000-bit N, we have $\delta_{\max}(\alpha = 0.6) = 0.441$, while for 4-prime RSA and 2000-bit N, we have $\delta_{\max}(\alpha = 0.6) = 0.315$. See Figure 1(a) for a plot. Comprehensive data covering the 1000- to 10000-bit range for N are given in [Hin02].

$\alpha = 1$	2-prime RSA				3-prime RSA				4-prime RSA			
N (bits)	$\delta_{\max}$	t_{opt}	m_{opt}	w_{opt}	$\delta_{\max}$	t_{opt}	m_{opt}	w_{opt}	$\delta_{\max}$	t_{opt}	m_{opt}	w_{opt}
1000	0.257	4	23	396	0.156	2	24	375	0.109	1	24	350
2000	0.267	6	31	720	0.165	3	32	660	0.118	2	33	663
4000	0.273	9	45	1495	0.170	4	41	1071	0.123	3	44	1170
6000	0.275	10	50	1836	0.172	5	49	1525	0.125	3	47	1320
8000	0.277	12	59	2550	0.173	6	57	2059	0.126	4	57	1943
10000	0.278	12	59	2912	0.174	7	65	2673	0.127	4	59	2070
$\vdots$	$\vdots$				$\vdots$				$\vdots$			
∞	0.285	∞	∞	∞	0.180	∞	∞	∞	0.132	∞	∞	∞
Wiener	0.250				0.167				0.125			

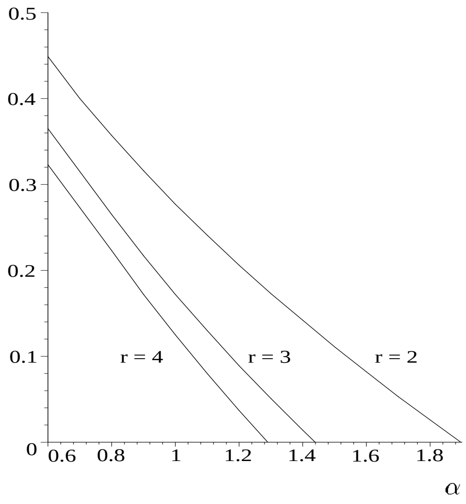
TABLE 1. Upper bounds for δ with $\alpha = 1$ using the Boneh-Durfee lattice. Last row: bounds from Section 4.1.

α	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5
$\delta_{\max}$	0.357	0.307	0.257	0.210	0.165	0.121	0.079	0.039	0.000	-0.038
t_{opt}	0	0	1	2	3	4	5	7	8	9
m_{opt}	25	27	30	31	32	33	33	37	37	37
w_{opt}	26	28	62	96	132	170	204	304	342	380

TABLE 2. Upper bounds for δ for 2000-bit 3-prime N using the Boneh-Durfee lattice.



(a) Boneh-Durfee Attack



(b) Blömer-May Attack

FIGURE 1. Upper bound on δ for 2-, 3-, and 4-prime RSA with 2000-bit N.

4.2.3. *Extending The Boneh-Durfee Lattice.* By construction, the matrix that represents the basis $B_{BD}(m,t)$ for the Boneh-Durfee lattice is triangular. Thus, the volume of the lattice is simply the product of the diagonal elements. Boneh and Durfee [BD00] observed that some of these diagonal elements contribute more to the volume than others, and suggested to remove those. This complicates the computation of the lattice volume, since one now has to work with a lattice that does not have full rank. But using the concept of geometrically progressive matrices, one nevertheless can bound the volume of the new lattice. Applied to the case $r = 2$ and $\alpha = 1$, this gives a larger upper bound on δ, namely $\delta_{\max} = 1 - \sqrt{2}/2 \approx 0.2929$. As before, this bounds holds as $N \to \infty$.

4.2.4. *The Blömer-May Lattice.* Another extension of the Boneh-Durfee lattice is given by Blömer and May [BM01]. Again, certain rows of $B_{BD}(m, t)$ are removed that contribute too much to the volume. But now also certain columns are removed, to ensure that the new basis has full rank and the matrix is triangular. Blömer and May find that, asymptotically, $\delta_{\max} = (\sqrt{6} - 1)/5 \approx 0.2899$ for 2-prime N with $\alpha = 1$. As before, we can generalize this and find that for arbitrary $\alpha > 0$ and $r = 2, 3, \ldots$ the Blömer-May lattice yields an asymptotic upper bound on δ that is given by

$$\delta_{\max}(\alpha, a_r) = 1 - \frac{6}{5}\, a_r - \frac{3}{5}\,\alpha + \frac{2}{5}\sqrt{\alpha^2 - a_r\,\alpha + 4\,a_r^2}\ . \tag{4.11}$$

As with the Boneh-Durfee lattice, we present some bounds on δ when $\alpha = 1$ for 2-, 3-, and 4-prime RSA for various modulus sizes in Table 3. These were determined in the same manner as for the Boneh-Durfee attack. Upper bounds on δ for varying α are shown in Figure 1(b). From Table 3 and more extensive data in [Hin02] we see that this attack allows for larger δ than Boneh and Durfee's attack in Section 4.2.2, but behaves in the same general way.

$\alpha = 1$	2-prime				3-prime				4-prime			
N (bits)	$\delta_{\max}$	t_{opt}	m_{opt}	w_{opt}	$\delta_{\max}$	t_{opt}	m_{opt}	w_{opt}	$\delta_{\max}$	t_{opt}	m_{opt}	w_{opt}
1000	0.271	14	30	465	0.167	10	43	484	0.120	8	62	567
2000	0.277	19	39	800	0.172	14	54	825	0.125	11	68	828
4000	0.281	26	51	1404	0.175	19	68	1380	0.128	16	86	1479
6000	0.283	31	60	1952	0.177	23	80	1944	0.129	19	98	1980
8000	0.284	36	69	2590	0.177	26	89	2430	0.129	21	106	2354
10000	0.285	39	74	3000	0.178	28	95	2784	0.130	23	115	2784
$\vdots$	$\vdots$				$\vdots$				$\vdots$			
∞	0.290	∞	∞	∞	0.181	∞	∞	∞	0.132	∞	∞	∞

TABLE 3. Upper bounds for δ with $\alpha = 1$ using the Blömer-May lattice.

4.3. **Chinese Remainder Theorem Attack.** As the previous sections show, the private exponent must not be chosen too small. To still achieve fast private key computations, one can perform modular exponentiations by $d_i := d \mod (p_i - 1)$ modulo the prime factors p_i of N and then recombine the results using Chinese remaindering. Of course, the smaller d_i, the faster the private key computations. However, although the attacks of the previous sections do not apply to small d_i, one can find a factor of N in time $O(\sqrt{d_{\text{SL}}}\log^2 N)$, where d_{SL} denotes the *second* largest value among the d_i. This attack is an immediate generalization of the corresponding attack in the 2-prime case.

Let $m = \log d_{\mathrm{SL}}$, and let g be a random number modulo N. Let

$$G(X) = \prod_{k=0}^{2^{\lceil m/2 \rceil}-1} \left(g^{e \cdot 2^{m/2} \cdot k} X - g \right) \mod N \ .$$

Using fast Fourier transform techniques, for $j = 1, 2, \ldots, 2^{m/2}$ evaluate $G(X)$ at $X_j = g^{ej}$. Now assume $d_i \leq d_{\mathrm{SL}}$. Then, since d_i can be written in the form $d_i = 2^{m/2} A_i + B_i$ for some $0 \leq A_i < 2^{m/2}$, $0 < B_i \leq 2^{m/2}$, and since $ed_i \equiv 1 \pmod{(p_i - 1)}$, we find that

$$g^{e(2^{m/2} \cdot A_i + B_i)} - g = g^{ed_i} - g \equiv 0 \pmod{p_i} \ ,$$

so that $\gcd(G(g^{eB_i}), N) > 1$. If $d_i \not\equiv d_l \pmod{2^{m/2}}$ for $i \neq l$, then $G(g^{eB_i}) \not\equiv 0 \pmod{p_l}$ and the event $\gcd(G(g^{eB_i}), N) > 1$ indeed reveals a prime factor of N.

Using FFT methods to evaluate a polynomial of degree σ at σ points with coefficients modulo N requires time $O(\sigma \log \sigma)$ for the transforms and $O(\sigma \log^2 N)$ or the σ multiplications [Tur82]. Thus, in running time $O(2^{m/2} \log^2 N) = O(\sqrt{d_{\mathrm{SL}}} \log^2 N)$ the modulus can be factored.

5. Partial Key Exposure Attacks

A *partial key exposure* attack is an attack in which some bits of the decryption exponent are known and are used to try to extract the rest of the bits. We discuss selected partial key exposure attacks on 2-prime RSA when the encryption exponent is either of small or medium size, and extend these attacks to r-prime RSA with $r > 2$.

Let us first point out that the assumption of partial knowledge of the private exponent is quite realistic. For example, if the public exponent is small, then 2-prime RSA leaks half of the most significant bits of the private exponent [Bon99]. More generally, r-prime RSA leaks $1/r$ of the most significant bits of d. This works as follows. Assume we know k from (2.1). Let $d_k = \lfloor (kN+1)/e \rfloor$. Then, with (2.1) and since $1 \leq k < e$,

$$d_k - d \leq \frac{k}{e}(N - \phi(N)) < N - \phi(N) \ .$$

Applying (2.6) to the right-hand side, we get

$$d_k - d < 2^{n - n/r + 1 + \log_2 r} \ .$$

That is, at least $n/r - 1 - \log_2(r)$ of the most significant bits of d_k and d agree. Of course, an attacker does not know k, but since $1 \leq k < e$, there are only $e-1$ candidates for d_k. Thus, r-prime RSA with an n-bit modulus leaks $n/r + O(1)$ of the most significant bits of d once the correct value of d_k is identified. In many cases, the latter can be done using techniques discussed later in this section.

Boneh [Bon99] points out that if $e = 3$, then always $k = 2$. This is also true for multi-prime RSA.

Lemma 5.1. *Let N be the product of two or more distinct primes each greater than 3, and $d \equiv 3^{-1} \pmod{\phi(N)}$. Then $3d - 2\phi(N) = 1$.*

Proof. Let $N = \prod_{i=1}^{r} p_i$ with $p_i > 3$ prime. Then $p_i - 1 \not\equiv 2 \pmod 3$ for $i = 1, \ldots, r$. Also, since the inverse of 3 modulo $\phi(N)$ exists, we have $\gcd(p_i - 1, 3) = 1$ for $i = 1, \ldots, r$ and thus $p_i - 1 \not\equiv 0 \pmod 3$ for all i. Therefore, $p_i - 1 \equiv 1 \pmod 3$ for all i, and hence $\phi(N) = \prod_{i=1}^{r}(p_i - 1) \equiv 1$

(mod 3). Now, reducing (2.1) (with $e = 3$) modulo 3 gives $-k \equiv 1 \pmod 3$, or $k \equiv 2 \pmod 3$. Since $1 \le k < 3$, this implies $k = 2$. □

Consequently, when $e = 3$ in an r-prime RSA system, about $1/r$ of the most significant bits of the decryption exponent can be immediately disclosed. Although this is less bits than in the 2-prime case, the conclusion is that $e = 3$ (and small values of e in general) should be avoided in r-prime RSA as well.

5.1. **Low Public Exponent Attack.** When the public exponent is small (such that a running time $O(e \log e)$ is manageable) and the $n/4$ *least* significant bits of d are known, the remaining bits of the private exponent of a 2-prime RSA system can be recovered.

In brief, this attack works as follows (see [BDF98]). Assume the $n/4$ least significant bits of d are known. Let $d_0 = d \bmod 2^{n/4}$. Reducing (2.1) modulo $2^{n/2}$ and substituting $\phi(N) = N-(p+N/p)+1$ yields the congruence

$$k'x^2 + (ed_0 - 1 - k'(N+1))x + k'N \equiv 0 \pmod{2^{n/4}} \ , \tag{5.1}$$

where $k' = k$ and $x = p \mod 2^{n/4}$. If we knew $p \mod 2^{n/4}$, then we could apply Theorem 2.1 to factor N in time $T_{\mathrm{C}}(n)$. Thus, in the actual attack, for each integer k', $1 \le k' < e$, one computes all solutions $\mod 2^{n/4}$ of (5.1) and attempts to factor N with each such candidate for $p \mod 2^{n/4}$.

For the attack to be effective, it is important that the number of solutions of (5.1) is small if $k = k'$. Steinfeld and Zheng [SZ01] give a bound on this number that is exponential in the number of common least significant bits of the prime factors p and q of N, and which leads to the following result.

Theorem 5.1 ([SZ01]). *Let t_{p-q} denote the number of common least significant bits of p and q. Then, given the $n/4$ least significant bits of d, an attack by Boneh, Durfee, and Frankel* [BDF98] *factors N in time $O(T_{\mathrm{BDF}}(n))$, where*

$$T_{\mathrm{BDF}}(n) \le \begin{cases} 2e\log e \cdot 2^{t_{p-q}+1} T_{\mathrm{C}}(n) \ , & \text{if } 2(t_{p-q}-1) < n/4 \ , \\ 2e\log e \cdot 2^{n/8} T_{\mathrm{C}}(n) \ , & \text{if } 2(t_{p-q}-1) \ge n/4 \ . \end{cases}$$

The attack works best if $p \not\equiv q \pmod 4$, in which case a total number of at most $4e\lceil \log_2 e \rceil$ candidates for $p \mod 2^{n/4}$ need to be tested (cf. [BDF98]).

We now consider this attack in the 3-prime case. Again, we start from (2.1), substitute $\phi(N)$ using (2.5), replace p_3 by $N/(p_1p_2)$ and write $k' = k$ to obtain that $(x, y) = (p_1 \mod 2^{n/4}, p_2 \mod 2^{n/4})$ is a solution of the congruence $F_{k'}(x, y) \equiv 0 \pmod{2^{n/4}}$, where

$$F_{k'}(x, y) = k'x^2y^2 - k'(x^2y + xy^2) + ((ed_0 - 1) - k'(N-1))xy + k'N(x+y) - k'N \ . \tag{5.2}$$

With k' unknown, the idea is as above, namely to try all solutions $(x, y) \mod 2^{n/4}$ for each $1 \le k' < e$, by applying a lattice reduction algorithm to factor N. For this approach to be of any use, the number of candidates for $(p_1 \mod 2^{n/4}, p_2 \mod 2^{n/4})$ must be small enough such that exhaustive search is feasible. To estimate the number of solutions of $F_{k'}(x, y) \equiv 0 \pmod{2^{n/4}}$, let

$$F(x, y) = x^2y^2 + ax^2y + bxy^2 + cxy + dx + ey + f \ , \tag{5.3}$$

and $h_x(y) = by^2 + cy + d$ and $h_y(x) = ax^2 + cx + e$. Then a straightforward computation shows that for integers x_0, y_0 and for $\nu \in \mathbb{N}$,

$$F(x + x_0 2^\nu, y + y_0 2^\nu) \equiv F(x, y) + 2^\nu(x_0 h_x(y) + y_0 h_y(x)) \pmod{2^{\nu+1}} \ .$$

Now, if $(\hat{x}, \hat{y})$ is a particular solution to $F(x, y) \equiv 0 \pmod{2^\nu}$, this solution can possibly be lifted to a solution modulo $2^{\nu+1}$, and any lifted solution (if it exists) will be of the form $(x, y) = (\hat{x} + x_0 2^\nu, \hat{y} + y_0 2^\nu)$, where $x_0, y_0 \in \{0, 1\}$. In fact, it is easy to see (cf. [Hin02]) that

$$F(\hat{x} + x_0 2^\nu, \hat{y} + y_0 2^\nu) \equiv 0 \pmod{2^{\nu+1}} \text{ iff } x_0 h_x(\hat{y}) + y_0 h_y(\hat{x}) \equiv F(\hat{x}, \hat{y})/2^\nu \pmod 2 \ . \tag{5.4}$$

We now make the following substitutions in (5.3):

$$a = b = -1 \ , \ c = \phi(N) - N + 1 \ , \ d = e = N \ , \ f = -N \ ,$$

and denote the corresponding function by $\overline{F}(x, y)$. Then the corresponding $\overline{h}_x(y)$ and $\overline{h}_y(x)$ satisfy $\overline{h}_x(y) \equiv y^2 + 1 \pmod 2$ and $\overline{h}_y(x) \equiv x^2 + 1 \pmod 2$. From this, we easily obtain

Lemma 5.2 ([Hin02]). *If $(\hat{x}, \hat{y})$ is a solution of $\overline{F}(x, y) \equiv 0 \pmod{2^\nu}$ with $\hat{x}, \hat{y} \equiv 1 \pmod 2$ then $(\hat{x}, \hat{y})$ lifts to either 4 solutions modulo $2^{\nu+1}$ or does not lift to any solution modulo $2^{\nu+1}$.*

This lemma together with (5.4) allows us to generate all solutions of $\overline{F}(x, y) \equiv 0 \pmod{2^\nu}$ for any $\nu > \eta$ provided we know all solutions modulo 2^η. Now observe that, conveniently, $k\overline{F}(x, y) \equiv F_k(x, y) \pmod{2^{n/4}}$ and $p, q \equiv 1 \pmod 2$. Thus, if $k = 2^t u$ where u odd, then every solution to $\overline{F}(x, y) \equiv 0 \pmod{2^{n/4-t}}$ is also a solution to $F_k(x, y) \equiv 0 \pmod{2^{n/4}}$. This allows us to conjecture a lower bound for the number of solutions to the latter congruence.

Conjecture 5.1. *For a random 3-prime RSA modulus N, the number of solutions to $F_k(x, y) \equiv 0 \pmod{2^{n/4}}$ is exponential in n.*

Rationale. Let $(\hat{x}, \hat{y})$ be a solution of $\overline{F}(x, y) \equiv 0 \pmod{2^\nu}$, for some $\nu \in \mathbb{N}$. This solution lifts to 4 solutions mod $2^{\nu+1}$ if and only if $\overline{F}(\hat{x}, \hat{y})/2^\nu$ is even. Which, under the reasonable assumption that $\overline{F}(\hat{x}, \hat{y})/2^\nu$ behaves like a random integer, happens with probability 1/2. Thus, *on average*, each $(\hat{x}, \hat{y})$ lifts to 2 solutions, and, on average, the number of solutions modulo $2^{\nu+1}$ will be twice as high as the number of solutions modulo 2^ν. Thus the average number of solutions modulo $2^{\nu+1}$ will be 2^ν times the number of solutions modulo 2. □

Using (5.4) and Lemma 5.2, we carried out some experiments to obtain experimental evidence on the average number of solutions of (5.2). We generated a random 3-prime RSA modulus N, and then computed all solutions of $F_k(x, y) \equiv 0 \pmod{2^\nu}$ for $2 \le \nu \le 10$, starting with the known solution $(1, 1)$ modulo 2. This was done for 2000 and 1500 values of N, using public exponents $e = 3$ and $e = 2^{16} + 1$, respectively. The results of these experiments are given in the Table 4. We see that indeed, the number of solutions to $F_k(x, y) \equiv 0 \pmod{2^\nu}$ is $\Omega(2^\nu)$. This supports our conjecture that the number of solutions to $F_k(x, y) \equiv 0 \pmod{2^{n/4}}$ is exponential in the bitsize of the RSA modulus N, which makes the Boneh-Durfee-Frankel attack [BDF98] completely ineffective for 3-prime RSA.

Extending the attack to 4-prime RSA yields similar results. Here we look for solutions to the multivariate modular equation in three unknowns: $G_{k'}(x, y, z) \equiv 0 \pmod{2^{n/4}}$, where

$$\begin{aligned} G_{k'}(x, y, z) \quad = \quad & k'(x^2y^2z^2 - (x^2y^2z + x^2yz^2 + xy^2z^2) + (x^2yz + xy^2z + xyz^2)) \\ & + (ed_0 - 1 - k'(N+1))xyz + k'N(xy + xz + yz) - k'N(x + y + z) + k'N \ . \end{aligned}$$

Note that $(p_1 \bmod 2^{n/4}, p_2 \bmod 2^{n/4}, p_3 \bmod 2^{n/4})$ is a solution of $G_{k'}(x, y, z) \equiv 0 \pmod{2^{n/4}}$ if $k' = k$. Here we find, for the corresponding function $\overline{G}(x, y, z)$, that

	Number of Solutions for $e = 3$ (2000 trials)			Number of Solutions for $e = 2^{16} + 1$ (1500 trials)		
Modulo	ave $\pm\ \sigma$	min	max	ave $\pm\ \sigma$	min	max
2^2	4 ± 0	4	4	4 ± 0	4	4
2^3	16 ± 0	16	16	16 ± 0	16	16
2^4	64 ± 0	64	64	58 ± 13	16	64
2^5	205 ± 64	64	256	191 ± 64	64	256
2^6	592 ± 188	256	768	554 ± 302	64	1024
2^7	1431 ± 705	256	2304	1504 ± 1175	256	4096
2^8	3162 ± 1936	1024	6144	3868 ± 4081	256	16384
2^9	6856 ± 4853	1024	15360	9610 ± 13405	1024	65536
2^{10}	14273 ± 11306	4096	43008	*	1024	73728*

TABLE 4. Number of solutions of $F_k(x,y) \equiv 0 \pmod{2^\nu}$ with random 150-bit 3-prime RSA modulus N, for $2 \le \nu \le 10$. σ denotes the standard deviation. (*) when the number of solutions exceeded 73728, the implementation terminated with a memory error.

Lemma 5.3 ([Hin02]). *If $(\hat{x}, \hat{y}, \hat{z})$ is a solution of $\overline{G}(x,y,z) \equiv 0 \pmod{2^\nu}$ with $\hat{x}, \hat{y}, \hat{z} \equiv 1 \bmod 2$ then $(\hat{x}, \hat{y}, \hat{z})$ either lifts to 8 solutions modulo $2^{\nu+1}$ or does not lift to any solution modulo $2^{\nu+1}$.*

We also obtain similar experimental results, this time conducted with random 200-bit 4-prime moduli N and $e = 3$ and $\nu \le 6$. We conclude that also in the 4-prime case the number of solutions of the corresponding modular equation is exponential in the size of the RSA modulus and the partial key attack is not feasible for 4-prime RSA.

Given the result by Steinfeld and Zheng, it is interesting to examine a possible dependence between the number of solutions to $\overline{F}(x,y) \equiv 0 \pmod{2^\nu}$ and common least significant bits of the p_i. To this end, for $l = 1, \ldots, 10$, we generated 100 random 150-bit 3-prime RSA moduli for which p_1, p_2, p_3 had exactly the l least significant bits in common. For $e = 3$ and $\nu = 1, \ldots, 10$, and $e = 2^{16} + 1$ and $\nu = 1, \ldots, 9$, we counted the number of solutions of $\overline{F}(x,y) \equiv 0 \pmod{2^\nu}$. Table 5 shows our experimental results for $e = 3$ and $\nu = 10$, which captures the general behaviour of the data.

There is obviously a pattern, but certainly the number of solutions is not determined by the number of common least significant bits alone, as it was the case for 2-prime RSA.

5.2. **Medium Public Exponent Attack.** Next we discuss an attack by Boneh, Durfee, and Frankel [BDF98] that applies when $e < \sqrt{N}$ and a certain fraction of the *most* significant bits of d is known. It first computes a small-size interval that contains the unique integer k that satisfies (2.1), and then uses these candidates to determine d. This works as follows.

Using (2.4), we can immediately generalize Lemma 4.2 and Theorem 4.1 from [BDF98] to the r-prime case ($r \ge 2$).

Lemma 5.4. *Let $N = \prod_{i=1}^r p_i$ and suppose d_0 is given such that, for positive integers c_1 and c_2, $|e(d - d_0)| < c_1 N$ and $ed_0 < c_2 N^{1+1/r}$. Let $\Delta = 2(2r-1)c_2 + 2c_1$ and $\tilde{k} = (ed_0 - 1)/N$. Then the integer k satisfying (2.1) lies in $(\tilde{k} - \Delta, \tilde{k} + \Delta) \cap \mathbb{Z}$.*

	# common bits									
# solutions	1	2	3	4	5	6	7	8	9	10
4096	0	0	0	64	55	53	42	46	54	52
6144	56	55	50	0	0	0	0	0	0	0
10240	0	0	0	22	29	27	29	22	16	20
12288	11	0	0	0	0	0	0	0	0	0
18432	14	19	26	0	0	0	0	0	0	0
24576	6	0	0	0	0	0	0	0	0	0
34816	0	7	24	14	16	20	29	32	30	28
36864	12	0	0	0	0	0	0	0	0	0
43008	1	19	0	0	0	0	0	0	0	0

TABLE 5. Frequency of the number of solutions of $\overline{F}(x,y) \equiv 0 \pmod{2^{10}}$ with respect to the number of common least significant bits of p_1, p_2, p_3.

Proof. With $\tilde{k} = (ed_0 - 1)/N$ and $k = (ed-1)/\phi(N)$ we have

$$|k - \tilde{k}| = \left|(ed_0 - 1)\left(\frac{1}{\phi(N)} - \frac{1}{N}\right) + \frac{e(d-d_0)}{\phi(N)}\right| < c_2 N^{1+1/r}\left(\frac{N-\phi(N)}{N\phi(N)}\right) + c_1 \frac{N}{\phi(N)} \;.$$

Using (2.4) and that $\phi(N) > N/2$, we obtain $|k-\tilde{k}| < 2(2r-1)c_2 + 2c_1$. □

Theorem 5.2. *Let t be an integer, $0 \le t \le n/r$, and assume $2^t < e < 2^{t+1}$. Given the t most significant bits of d, there is an efficiently computable set of integers A_r such that $k \in A_r$ and $|A_r| \le 16((2r-1)2^{1/r} + 1)$.*

Proof. Given the t most significant bits of d, we can construct an integer $d_0 < 2^n$ such that $|d - d_0| < 2^{n-t}$. Then, since $2^{n-1} < N < 2^n$, we have $|e(d-d_0)| < 2^{n+1} < 4N$ and $ed_0 < 2^{n+n/r+1} < 2^{2+1/r}N^{1+1/r}$. Thus, Lemma 5.4 holds with $c_1 = 4$ and $c_2 = 2^{2+1/r}$. So let $\Delta_r = 8(2r-1)2^{1/r} + 8$ and $\tilde{k} = (ed_0 - 1)/N$ and $A_r = (\tilde{k} - \Delta_r, \tilde{k} + \Delta_r) \cap \mathbb{Z}$. Then $k \in A_r$ and $|A_r| \le 2\Delta_r$. □

Note that for $r = 2$, the cardinality of A_2 differs from the value given in [BDF98], which is 40. On the one hand, we were able to derive a sharper bound on $N - \phi(N)$ from our assumption (2.3) on the relative sizes of the prime factors (while (2.3) is actually equivalent with the corresponding condition (1) of [BDF98]). But on the other hand, for the n-bit number N we used the estimate $2^{n-1} < N$ rather than $2^n < N$.

Not only does the range of suitable e in Theorem 5.2 decrease for $r > 2$, we also run into problems in the attempt to identify the correct value of k among the integers in A_r.

In the 2-prime case, if $n/4 \le t \le n/2$ and if the public exponent is prime, one proceeds as follows ([BDF98]): Let $k' \in A_2$. If $k' \ge e$, discard k'. Otherwise, let $s = N + 1 + (k')^{-1} \bmod e$, and compute a root x_0 of $x^2 - sx + N \equiv 0 \pmod e$. If $k' = k$ then, from (2.1), $(k')^{-1} \equiv -\phi(N) \pmod e$ and $s \equiv N + 1 - \phi(N) \pmod e$. Thus, $s \equiv p + q \pmod e$ and $x_0 \equiv p \pmod e$ or $x_0 \equiv q \pmod e$ so that Theorem 2.1 applies (observe that $e > 2^{n/4}$) and it is possible to factor N in time polynomial in n. If $k \ne k'$, then Theorem 2.1 fails to factor N and k' can be discarded. This technique can also be adapted to the case that e is composite with known prime factorization.

In the r-prime case ($r > 2$), the above approach to test the elements of A_r does not work. To find p_i modulo e in a similar way, we would need to solve the degree-r congruence $\prod_{i=1}^{r}(x - p_i) \equiv 0 \pmod{e}$. However, not all coefficients in this equation are known! For example, if $r = 3$, we might want to put $s = N-1+(k')^{-1} \mod e$. Then, if $k = k'$, we have $s \equiv p_1p_2+p_1p_3+p_2p_3-(p_1+p_2+p_3) \pmod{e}$. But this leaves us only with the congruence $x^3-(p_1+p_2+p_3)x^2+(s+p_1+p_2+p_3)x-N \equiv 0 \pmod{e}$, with $\sum_{i=1}^{3} p_i$ unknown.

That means that 2-prime RSA can be broken if the public exponent is prime or of known prime factorization, if e has $t+1$ bits and the t most significant bits of the private exponent are known and $n/4 \leq t \leq n/2$, but for multi-prime RSA with $r > 2$ factors no corresponding attack is known.

Another, weaker, result from [BDF98] that requires knowledge of significantly more bits of the private exponent does generalize to the r-prime case. This result does not require knowledge of the prime factorization of e, but we need that k is not significantly smaller than e. (The latter is not a significant restriction, see [BDF98]).

Theorem 5.3. *Let N be an r-prime RSA modulus and suppose $2^t < e < 2^{t+1}$ where $0 \leqslant t \leqslant n/r$ is an integer. Also, assume that $k > \epsilon \cdot e$ for $1 > \epsilon > 0$. Given the $n-t$ most significant bits of d, the remaining bits of d can be recovered in time $O(rn^3/\epsilon)$.*

Proof. Using the $n-t$ most significant bits of d we can construct an integer d_0 such that $d - d_0 < 2^t$. Let $\tilde{k} = (ed_0 - 1)/N$. Then Theorem 5.2 applies with $A_r = (\tilde{k} - \Delta_r, \tilde{k} + \Delta_r) \cap \mathbb{Z}$ where $\Delta_r = 16((2r-1)2^{1/r}+1)$. So we know that $k \in A_r$. Now, for each $k' \in A_r$ that is coprime with e, we do the following: for each integer $v \in [0, 1/\epsilon)$, let $d' = d_0 + k'v$; then check if d' is the correct private exponent, by testing if $(M^e)^{d'} \equiv M \pmod{N}$ for random $M \in \mathbb{Z}_N$. This method succeeds because if $k = k'$, then $d = d_0 + k(d-d_0)/k$, and $(d-d_0)/k < 1/\epsilon$ by our assumption on k and construction of d_0. Testing each candidate (k', v) through modular exponentiations of n-bit integers takes $O(n^3)$ time. There are $1/\epsilon$ candidates for v' and $O(r)$ candidates for k'. Hence, the total running time to recover d is $O(rn^3/\epsilon)$. ∎

Theorem 5.3 implies that, paradoxically, the smaller e is, the more bits of d are required to completely recover the private exponent. For example, if e is approximately $N^{1/3}$, then we need 2/3 of the most significant bits of d to mount the attack, while if e is close to $N^{1/4}$, as many as 3/4 of the bits of d are needed. Note that the bound on the public exponent for which the attack works becomes smaller with more prime factors in the modulus.

6. Conclusion

We have examined how selected attacks on RSA can be extended to the multi-prime case, and how they perform in the new setting. We have shown that as the numbers of primes factors in the modulus increases, the attacks become more complex, which results in that the attacks apply in fewer instances, or become totally ineffective, or do not seem to extend at all. While our results suggest that thus multi-prime RSA is less vulnerable to current attacks on RSA, it is important to now look for attacks that specifically exploit the fact that the modulus factors into three, or four, primes.

Acknowledgements. The authors would like to thank Dan Boneh for help with Section 4.3, and Shuhong Gao for help with Section 5.1. We also thank Glenn Durfee and Don Coppersmith for helpful explanations.

References

[BD00] 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, 2000.

[BDF98] D. Boneh, G. Durfee, and Y. Frankel. Exposing an RSA private key given a small fraction of its bits. In *Advances in Cryptology - ASIACRYPT '98*, volume 1514 of *Lecture Notes In Computer Science*, pages 25–34. Springer-Verlag, 1998. Revised and extended version available from http://crypto.stanford.edu/~dabo/pubs.html.

[BM01] J. Blömer and A. May. Low secret exponent RSA revisited. In *Cryptography and Lattices - Proceedings of CALC '01*, volume 2146 of *Lecture Notes In Computer Science*, pages 4–19. Springer-Verlag, 2001.

[Bon99] D. Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices of the American Mathematical Society*, 46(2):203–213, 1999.

[BS02] D. Boneh and H. Shacham. Fast variants of RSA. *CryptoBytes (The technical newsletter of RSA laboratories)*, 5(1):1–9, 2002.

[CHLS97] T. Collins, D. Hopkins, S. Langford, and M. Sabin. Public Key Cryptography Apparatus and Method. US Patent #5,848,159, Jan. 1997.

[Cop97] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997.

[DN00] G. Durfee and P. Q. Nguyen. Cryptanalysis of the RSA schemes with short secret exponent from Asiacrypt '99. In *Advances in Cryptology - ASIACRYPT 2000*, volume 1976 of *Lecture Notes In Computer Science*, pages 14–29. Springer-Verlag, 2000.

[HG97] N.A. Howgrave-Graham. Finding small roots of univariate modular equations revisited. In *Cryptography and Coding*, volume 1355 of *Lecture Notes In Computer Science*, pages 131–142. Springer-Verlag, 1997.

[Hin02] M. J. Hinek. Low public exponent partial key and low private exponent attacks on multi-prime RSA. Master's thesis, University of Waterloo, Dept. of Combinatorics and Optimization, 2002.

[HW60] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers.* Oxford University Press, fourth edition, 1960.

[Len01] 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. Springer-Verlag, 2001.

[LLL82] A. Lenstra, H. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[Low02] M.K. Low. Attacks on multi-prime RSA with low private exponent or medium-sized public exponent. Master's thesis, Univ. of Waterloo, Dept. of Combinatorics and Optimization, 2002.

[May02] A. May. Cryptanalysis of unbalanced RSA with small CRT-exponent. In *Advances in Cryptology - CRYPTO 2002*, Lecture Notes In Computer Science. Springer-Verlag, 2002.

[Old63] C. D. Olds. *Continued Fractions.* Random House, Inc., 1963.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[Sho] V. Shoup. Number theory library (NTL), Version 5.2. http://www.shoup.net/ntl.

[Sti95] D. R. Stinson. *Cryptography : Theory and Practice.* CRC Press LLC, 1995.

[SZ01] R. Steinfeld and Y. Zheng. An advantage of low-exponent RSA with modulus primes sharing least significant bits. In *Proceedings RSA Conference 2001, Cryptographer's Track*, volume 2020 of *Lecture Notes in Computer Science*, pages 52–62. Springer-Verlag, 2001.

[Tur82] J. W. M. Turk. Fast arithmetic operations on numbers and polynomials. In H.W. Lenstra, Jr. and R. Tijdeman, editors, *Computational Methods in Number Theory, Part I.* Mathematisch Centrum, Amsterdam, 1982.

[Wie90] M. J. Wiener. Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory*, 36(3):553–558, 1990.

University of Waterloo, Department of Combinatorics and Optimization, Waterloo, Ontario, Canada N2L 3G1

E-mail address: {mjhinek,eteske}@uwaterloo.ca, mklow@fastmail.ca