Seeing through MIST Given a Small Fraction of an RSA Private Key

Colin D. Walter

Comodo Research Lab
10 Hey Street, Bradford, BD7 1DQ, UK
Colin.Walter@comodogroup.com www.comodogroup.com

Abstract. In smartcard encryption and signature applications, randomised algorithms are used to increase tamper resistance against attacks based on side channel leakage. MIST is one of these. As is the case with the classical m-ary and sliding windows exponentiation algorithms, the most significant half of the public modulus yields information which can be used to halve the number of key digits which need to be guessed to recover the secret key from a MIST side channel trace. Lattice based methods are used to reduce this to just one quarter of the least significant digits. This enables the strength of the MIST exponentiation algorithm to be guaged more accurately under several threat models.

Key words: Addition chains, division chains, randomized exponentiation, Mist, randomary exponentiation, RSA, side channel leakage, power analysis, SPA, DPA, SEMA, DEMA, blinding, smartcard.

1 Introduction

Smartcards have very limited scope for the inclusion of physical security measures. So tamper resistance should also be built into the component algorithms whenever possible. Because exponentiation is such a major process in many crypto-systems, including RSA, Diffie-Hellman and ECC, it is the main target for improved algorithmic methods. Initial side channel attacks made use of timing differences [8] due to conditional subtractions during modular multiplications [17]. Such differences are now easily avoided [19]. However, because unprotected hardware gates use different amounts of power depending upon whether or not they are switched to a different state, power usage by any chip is also data dependent, and so provides side channel leakage which may enable secret keys to be recovered. Although the first power attacks required averaging over a number of exponentiations in order to reduce the effects of noise and dependence on irrelevant data [9, 12], recent progress suggests that there is enough data in the trace of a single exponentiation to allow careful averaging to reveal the key [18]. In particular, one can average over subsets of digit-by-digit products within each long integer multiplication rather than the sequential traces of many exponentiations. In addition, electro-magnetic leakage seems to provide a very much more powerful means of obtaining key data than power variation [15, 16, 4, 1].

Such side channels and methods enable all the classical exponentiation methods to be attacked: both sliding windows and m-ary [7]. Consequently, a new breed of algorithms has had to be developed, namely the *randomised* exponentiation algorithms. Two such algorithms appeared at the CHES 2001 conference [11, 14] and further ones at, for example, the RSA 2002 [21] and CHES 2002 conferences [5, 6]. The first two seem to provide little extra security if one assumes squares and multiplies (adds and doubles) can usually be distinguished during a single exponentiation and no key blinding is used [13, 23]; they are also only suitable for elliptic curve (ECC) applications because of the requirement for the computation of inverses. However, the more recent algorithms seem to be more robust [22], and some of them can be applied easily to RSA because no inverses need be computed.

The purpose of this article is to contribute further to an assessment of the strength of the third of these randomised exponentiation algorithms, namely Mist [21]. First, elementary methods are applied to show that half of the digits generated by Mist suffice to reveal the key when the public modulus and exponent are known. The techniques of Boneh *et al.* [2] have been used formerly on classical representations of the secret key to show that only a quarter of the bits of the key need be known before algorithmic methods can be applied to determine the complete key in polynomial time. Here those techniques are adapted to the "randomary" representation of the key which Mist generates. The results are similar: a quarter of the randomary digits are sufficient to enable recovery of the full key in polynomial time.

These results are then applied to reduce the search space for keys which are partially revealed through side channel leakage. Although the conclusions show the computational effort to break the keys is still infeasible for standard key lengths, there is little room left for complacency. Key blinding or keys of at least 1024 bits would still appear to be necessary for good tamper resistance. By comparison, of course, the quality of side channel leakage which we assume is instantly fatal to the classical algorithms.

2 The Mist Algorithm

For notation, suppose that the RSA crypto-system has known public modulus N, public exponent E and private exponent D (after any required blinding), so that plaintext P and ciphertext C are related by $P = C^D \bmod N$. It is side channel leakage from this exponentiation that an attacker uses to extract the secret key D.

The Mist algorithm [20, 21] is a random-ary exponentiation method very similar to m-ary exponentiation, but it reverses the order of processing the digits of D, and the base m is varied randomly for each digit choice. The following version computes $P = C^D \bmod N$.

THE MIST EXPONENTIATION ALGORITHM

```
{ Pre-condition: D ≥ 0 }
Q ← C  ;
P ← 1  ;
While D > 0 do
Begin
   Choose a random base m from set S ;
   d ← D mod m ;
   If d ≠ 0 then P ← Q^d×P mod N ;
   Q ← Q^m mod N ;
   D ← D div m ;
   { Loop invariant: C^{D.Init} = Q^D × P mod N }
End ;
{ Post-condition: P = C^{D.Init} mod N }
```

The values of d are analogous to digits of D in representations where the base m is constant. Generally, the random choice of m is from a fixed set with known security and efficiency properties, such as $S = \{2,3,5\}$. When the random base set S consists of the single divisor 2, the method simplifies to the binary square-and-multiply algorithm in which the least significant exponent bit is processed first. In general, for a singleton set $S = \{m\}$, the algorithm simplifies to classical m-ary exponentiation but performed from right to left rather than from left to right. Space and time efficiency were shown in [21] to be comparable with 4-ary exponentiation when addition chains for computing Q^m compute Q^d en route rather than sequentially, and m is chosen with a suitable bias which favours $m = 2$ or $d = 0$.

The random choice of bases generates different exponentiation schemes on successive runs and so makes impossible the usual averaging process required for power analysis (SPA/DPA) [9] or electro-magnetic analysis (SEMA/DEMA) [15]. Hence, we assume that the attacker has extracted data from a single exponentiation which enables him to determine either i) which operations are squares and which are multiplications, or ii) which operations share an argument. In [22], the following results were established:

Theorem 1. ([22], Thm. 9) *Suppose squares and multiplies can be distinguished, but not individual reuse of operands. Then, for the choice of parameters given in* [22]*, the average number of exponents which can generate the same sequence of squares and multiplications as a given one for* D *is bounded below by* $D^{3/5}$.

Theorem 2. ([22], Thm. 8) *For the choice of parameters given in* [22]*, the average number of exponents with exponentiation schemes that have an operand sharing pattern identical to a given one for* D *is about* $D^{1/3}$.

Both of these theorems are established by counting the number of choices for the pairs (m, d). For example, addition chains for (2,1) and (3,0) are chosen to give them the same operand sharing patterns and square & multiply sequences.

Every time one of these patterns turns up, there is an ambiguity about the choice for D which effectively doubles the size of the search space for the correct D. In other cases there is a unique choice, or even three or four choices for (m, d) which give the pattern which the attacker has recognised. Simulations indicate that there is very little collapsing of the search space as a result of duplication in the reconstructed exponents. However, as one of the reviewers has pointed out, the size of the search space is not necessarily an accurate measure of the time to recover the correct key. If efficiency considerations dictate non-uniform selection criteria for d, then some exponents are more likely than others. So, on average, an intelligent search for the correct exponent would only have to traverse a fraction of the entire space. Nevertheless, if each d is equally likely, then half the space must be traversed. This latter is the case assumed in the illustrative results below, but it is easy to adapt the deductions to alternative situations.

If the digits and bases are indexed from 0 to $n-1$ in the order generated, then the following notation, initial conditions and properties hold, just as for a fixed base m:

$$\begin{array}{lcl} D_0 & = & D \\ d_i & \equiv & D_i \bmod m_i \\ D_{i+1} & = & D_i \text{ div } m_i \\ d_{n-1} & \neq & 0 \\ D_n & = & 0 \end{array} \tag{1}$$

$$\begin{array}{lcl} D_i & = & m_i D_{i+1} + d_i \\ D_i & = & ((\ldots(d_{n-1}m_{n-2} + d_{n-2})\ldots)m_{i+1} + d_{i+1})m_i + d_i \\ D & = & ((\ldots(d_{n-1}m_{n-2} + d_{n-2})\ldots)m_1 + d_1)m_0 + d_0 \end{array} \tag{2}$$

This indexing follows that of digits in the standard base m representation. Some additional related quantities are of use later:

$$\begin{array}{lcl} \mu_i & = & \prod_{j=0}^{i-1} m_j \\ \delta_i & = & ((\ldots(d_{i-1}m_{i-2} + d_{i-2})\ldots)m_1 + d_1)m_0 + d_0 \end{array} \tag{3}$$

which satisfy

$$\begin{array}{lcl} \delta_i & \equiv & D \bmod \mu_i \\ D_i & = & D \text{ div } \mu_i \\ D & = & \mu_i D_i + \delta_i \end{array} \tag{4}$$

Exponent blinding is assumed throughout, since this is probably essential to help address the vulnerability of applications for which the algorithm is considered necessary. So D will always represent the exponent actual used, with its digits as above, while D' will be the original unblinded secret key. Notation for the blinding is introduced in the next section.

3 Halving the Search Space

The starting point for both of the main results is the observation that $\phi(N)$ has a good approximation given in terms of N. If $N = PQ$ is the prime factorisation of N, the standard choice of P and Q to have the same number of bits allows an attacker to assume that $P < Q < 2P$. Then $2\sqrt{N} < P{+}Q < 3\sqrt{N/2}$ and so $\phi(N) = N{-}(P{+}Q){+}1$ is bounded by

$$N - 3\sqrt{N/2} + 1 \;<\; \phi(N) \;<\; N - 2\sqrt{N} + 1 \tag{5}$$

This interval has length less than $\frac{1}{8}\sqrt{N}$, so that three more than half the most significant bits of $\phi(N)$ can be determined trivially.

Suppose that the public and private exponents, E and D' respectively, are related by

$$D'{\times}E \;=\; 1{+}k\phi(N) \tag{6}$$

where it is reasonable to assume that D' is chosen to make $D' < \phi(N)$ so that $k < E$. However, to such D' a blinding factor should normally be added [8], giving the secret key

$$D \;=\; D'{+}r\phi(N) \tag{7}$$

which is actually used for an exponentiation. Here r is a random number, often of 32 bits. Then

$$D \;=\; \frac{1 + (k{+}rE)\phi(N)}{E} \tag{8}$$

Let B be an upper bound on such r. Then, in effect, the coefficient $k{+}rE$ of $\phi(N)$ is a random number in the range 0 to BE. The attacker just has to generate each of the $O(BE)$ possible values of the random coefficient of $\phi(N)$ in order to obtain a set containing an approximation to the value of D used in the exponentiation which he has observed. If there is no blinding, $r = 0$ is taken, and $B = 1$ will give the relevant results for that case.

For a given choice of r and k, let D_l and D_u denote the (irrational) lower and upper bounds on D determined by equations (5) and (8). So, for the assumptions made above,

$$D_u - D_l \;\;=\;\; \frac{k{+}rE}{E}(3\sqrt{1/2} - 2)\sqrt{N} \;\;<\;\; \frac{B}{8}\sqrt{N} \tag{9}$$

Next the attacker must generate all possibilities for the last half or so of the digits of D, taking enough of them to construct a D_j satisfying both

$$D_j \;\geq\; \sqrt{\frac{(k{+}rE)N}{E}} \quad\text{ and }\quad D_j \;\geq\; D_u{-}D_l \tag{10}$$

The first inequality is usually a consequence of the second. By (9), this is the case if $\sqrt{\frac{k+rE}{E}} \leq \frac{k+rE}{E}(3\sqrt{1/2} - 2)$, i.e. if $\frac{k+rE}{E} \geq (3\sqrt{1/2} - 2)^{-2}$, which holds for $r \geq 68$. So we can normally ignore it in the following estimates.

Depending on the threat model for side channel leakage, the digit choices will be restricted in some way. We will consider in more detail the two particular cases covered by Theorems 1 and 2. In each case, the ambiguities for each base/digit pair (m_i, d_i), $j \leq i \leq n{-}1$, were described in detail in [22]. The attacker's next problem is to identify which such choice is correct.

By (8) and the first inequality of (10), $D_j > \sqrt{D}$, so that (4) gives $\delta_j < \mu_j < \sqrt{D} < D_j$. Thus $\mu_j = D \operatorname{div} D_j$ by (4). By the second inequality of (10), $D_l/D_j < D/D_j < D_u/D_j \leq D_l/D_j + 1$. So, combining these,

Lemma 1. *For j as chosen above, $\mu_j = D \operatorname{div} D_j = \lfloor D_l/D_j \rfloor$ or $\lfloor D_u/D_j \rfloor$.*

These two quantities can be calculated from the assumed values of D_l and D_u and used to reject D_j if neither expression has the characteristic property of μ_j, namely being a product of elements from the chosen base set $S = \{2, 3, 5\}$. This should almost completely determine the correct D_j.

Our next task is to estimate how many values of D_j will be accepted by this process. It is reasonable to assume $\lfloor D_l/D_j \rfloor$ and $\lfloor D_u/D_j \rfloor$ are effectively random in terms of their prime factorisations. So we need to know the probability that a random number of size D/D_j will be a product of powers of only 2, 3 and 5.

Suppose $\mu = 2^x 3^y 5^z < K$. Then there are at most $\log_2 K$ possible choices for x, $\log_3 K$ for y and $\log_5 K$ for z, making a total of at most $(\log 2 \log 3 \log 5)^{-1}$ $(\log K)^3$ choices for μ. Differentiating with respect to K, this in turn means a maximum density of

$$3(\log 2 \log 3 \log 5)^{-1}(\log K)^2/K \tag{11}$$

for numbers of size K with the required form. So these forms are quite rare.

Combining (5) and (8) with the equality in (9) yields essentially

$$\frac{\sqrt{N} - 3\sqrt{1/2}}{3\sqrt{1/2} - 2} \quad < \quad \frac{D}{D_u - D_l} \quad < \quad \frac{\sqrt{N} - 2}{3\sqrt{1/2} - 2} \tag{12}$$

If D_j is chosen to be minimal such that the second inequality of (10) holds then $D_u - D_l \leq D_j \leq 5(D_u - D_l)$ because 5 is the maximum base. Thus, ignoring insignificant terms, the numbers of interest are bounded below by $D \operatorname{div} D_j \geq \lfloor D_l/5(D_u - D_l) \rfloor \geq \lfloor \frac{\sqrt{N} - 3\sqrt{1/2}}{5(3\sqrt{1/2} - 2)} \rfloor \approx \frac{5}{3}\sqrt{N}$ and bounded above by $D \operatorname{div} D_j \leq D_u/(D_u - D_l) \leq \frac{\sqrt{N} - 2}{3\sqrt{1/2} - 2} < 9\sqrt{N}$. So we may use $K = \frac{5}{3}\sqrt{N}$ in (11) to provide an upper bound on the density of $\{2, 3, 5\}$-powers in the region containing μ_j. If, instead, D_j is chosen minimal such that the first inequality of (10) holds then $D \operatorname{div} D_j > \frac{1}{5}\sqrt{D} \approx \frac{1}{5}\sqrt{rN}$ where $r < 68$. So again we must choose K in (11) to be $O(\sqrt{N})$.

However, by Theorems 1 and 2, there are up to ${D_j}^{3/5}$ or ${D_j}^{1/3}$ values of D_j with the right patterns to consider. Here $O(D_j) = O(D_u - D_l) = O(B\sqrt{N})$ by (9). Hence, by (11), the expected number of values D_j for which $D_l \operatorname{div} D_j$ or $D_u \operatorname{div} D_j$ has the right form is at most

$$O(B^{3/5}\{\log N\}^2 N^{-1/5}) \quad \text{or} \quad O(B^{1/3}\{\log N\}^2 N^{-1/3}) \tag{13}$$

respectively for each choice of k and r. In both cases, this is less than 1 for some expected sizes of N (192 bits in the case of ECC, say) and B (32 bits, say). This indicates that essentially only one solution is likely to exist, which will be the required one if k and r are chosen correctly. However, without the aid of any leaked data, an exhaustive search will have D_j cases to consider and will reduce the exponent of N in this expression to 0. Thus, any side channel which reduces the search space to below that of an exhaustive search will reduce the expected number of D_j generating the right form to at most $O(B\{\log N\}^2)$.

Once a feasible D_j is determined, the remaining part of D is straightforward to reconstruct iteratively. Using equation 2(i) repeatedly, each new value D_i $(j{-}1 \geq i \geq 0)$ is constructed from the preceding D_{i+1} and the divisibility of the associated μ_i by only 2, 3 or 5 is checked in the same way as for D_j in order to determine which choice of (m_i, r_i) is correct.

The effort for this is minimal at least for $i \approx j$. However, as i decreases, the probabilities change: $\mu_i = D$ div D_i has fewer and fewer factors, and so is more and more likely to be of the correct form. For $\mu_i \approx 3{\times}2^9$ there are only about ${\mu_i}^{3/5} \approx 54$ or ${\mu_i}^{1/3} \approx 9$ remaining choices for extending D_i to D in a way consistent with the side channel leakage. Also, only 79 values from the interval $[1 \,..\, 3{\times}2^9]$ give subsequent $\mu_{i'}$s with the right form. So there are very few incorrect choices which satisfy all the criteria and every plausible case can be investigated in full. For $\mu_i > 3{\times}2^9$, (11) shows that at most 1 in 12 numbers have the requisite form. The average base value $m_{i'}$ is 2.50 so that the previous t base values reduce D by a factor of about 2.50^t. So there are around $(2.50^t)^{3/5}$ or $(2.50^t)^{1/3}$ choices respectively for these t base/digit pairs. Any one of these has a probability at most 12^{-t} of being acceptable under the divisibility criterion. Hence at most about $(2.50^t)^{3/5}12^{-t}$ or $(2.50^t)^{1/3}12^{-t}$ of the incorrect choices will survive t base choices. These both tend to 0 as t increases. Hence, although one may temporarily have to consider some additional incorrect choices for D_i, these will disappear very quickly as i decreases. Indeed, with a probability of less than 1 in 12 of acceptability and at most 4 choices for extending D_{i+1} to D_i, the average number of further iterations which an incorrect value survives is less than $\sum_{i=1}^{\infty} i(\frac{4}{12})^i = \frac{3}{4}$. Thus, normally there will be fewer than two values of D_{i+1} which are under consideration for generating D_i – the correct one (if k and r were selected correctly) and at most one incorrect one. The computational cost of generating possible D from a correct D_j is therefore of the same order as that of applying the divisibility criterion j times: once for each $i < j$; and the computational cost for an incorrect D_j is just the cost of a constant number of applications of the divisibility criterion.

In total, there were $O(BE)$ values for (k,r), and hence for D_l and D_u, and $O(\{B\sqrt{N}\}^{3/5})$ or $O(\{B\sqrt{N}\}^{1/3})$ ways for choosing D_j from each D_l or D_u. Thus, from this method there will be respectively at most

$$O(EB^{8/5}N^{3/10}) \quad \text{or} \quad O(EB^{4/3}N^{1/6}) \tag{14}$$

possible values for D_j, and hence of D, which are generated and need checking. As noted above, almost all D_j will be rejected by the divisibility criterion on its first or second application and not lead to a viable D. These figures give the order of work involved in an attack. For standard RSA key sizes of 1024 or more bits and no other data to narrow the search space even further, this is still a computationally infeasible task for any E and no blinding ($B = 1$). Observe that increased security is obtained more cheaply by increasing the public exponent E or the size of the blinding factor B rather than the modulus N.

Other parameter choices and threat scenarios for MIST can be tested in a similar way to see if the search space still remains large enough to prevent a successful attack.

4 Quartering the Search Space

The lattice-based techniques of Coppersmith [3] produce complementary results to those in the previous section. They enable one to deduce the secret key from the least significant digits rather than the most significant ones. Existing relevant work for recovering secret keys from the public modulus and such partial knowledge of the key is exclusively centred on a bit-based view of the key. With bases 3 and 5 as possibilities, the bit-based view must be reformulated in a less base-dependent way. Fortunately, the generalisations involve few complications. So, analogously to the quarter of bits of D that Boneh *et al.* [2] use, a similar argument here requires on average about a quarter of the digit/base pairs to be established. Effectively, the computational effort must then be directed at an exhaustive search of a space whose size is a fractional power of D with exponent only a quarter of the value given in Theorems 1 or 2. The main result needed in the proof is the following:

Theorem 3. ([2], Cor 2.2) *Let $N = PQ$ be an n-bit RSA modulus. Let $\mu \geq 2^{n/4}$ be given and suppose $P_0 \stackrel{def}{\equiv} P \bmod \mu$ is known. Then it is possible to factor N in time polynomial in n.*

Both here and in [2], the choice made for μ is a product of base values. This makes μ a power of 2 in [2], but a product of powers of 2, 3 and 5 here. Suppose pairs (m_i, d_i) have been guessed correctly for $0 \leq i \leq j{-}1$ and some j. Then δ_j and μ_j are known, and satisfy $D \equiv \delta_j \bmod \mu_j$, as in equation 4(i). μ_j is the value which will be taken for μ in the theorem, so we choose j large enough that $\mu = \mu_j \geq 2^{n/4}$.

Rewriting equation (8) entirely in terms of N and P rather than N and $\phi(N)$ gives

$$DE \;=\; 1 + (k{+}rE)(N{-}P{-}N/P{+}1) \tag{15}$$

Reducing this modulo $\mu = \mu_j$, we see that $P_0 \equiv P \bmod \mu$ is a solution for x in the equation

$$\delta_j E \;\equiv\; 1 + (k{+}rE)(N{-}x{-}N/x{+}1) \mod \mu \tag{16}$$

and hence a root of

$$(k{+}rE)x^2 - (1{-}\delta_j E{+}(k{+}rE)(N{+}1))x + (k{+}rE)N \mod \mu \tag{17}$$

The coefficients here are all divisible by $k{+}rE$ because $(k{+}rE)\phi(N) = 1{-}DE \equiv 1{-}\delta_j E \mod \mu$. Suppose $g = \gcd\{k{+}rE, \mu\}$. Then, dividing (17) through by $k{+}rE$, $P_0 \equiv P \mod \mu$ is a root of the quadratic

$$x^2 - ((1{-}\delta_j E)/(k{+}rE){+}N{+}1)x + N \mod \mu g^{-1} \tag{18}$$

As before, the attack must consider separately every possible value for $k{+}rE$. g is easily computed once $k{+}rE$ is chosen, and so (18) is obtained.

Solutions are most easily obtained by first completing the square. $N{+}1$ and $(DE{-}1)/(k{+}rE) = \phi(N)$ are both even, so the coefficient of x in (18) is also even. In fact, $Q_0 \equiv Q \mod \mu$ is a second solution to (18) and the coefficient of x is then $-(P_0{+}Q_0)$. So, by taking $y = x{-}\frac{1}{2}((1{-}\delta_j E)/(k{+}rE){+}N{+}1)$ in (18), x is obtained from the solutions of an equation of the form

$$y^2 \equiv c \mod \mu g^{-1} \tag{19}$$

where c is easily computed. For solutions to exist, any power of 2, 3 or 5 which divides c must occur to an even power. If this is not the case, the wrong $k{+}rE$ must have been chosen, and so the next one should be selected. If it is the case, then that power can be removed from y easily and replaced later. So, without loss of generality, assume c is prime to each of 2, 3 and 5. Thus we are looking for a solution prime to μg^{-1}.

The solutions to (19) are obtained by solving modulo the maximal 2-, 3- and 5- powers which divide μg^{-1} and then using the Chinese Remainder Theorem to reconstruct the solutions modulo μg^{-1}. Solutions modulo the prime powers are obtained by lifting solutions progressively from lower powers of the prime. For higher powers of 2 than 2^2, Boneh *et al.* ([2], App. A) have previously noted that there are at most 4 solutions if there are solutions at all, and have provided a construction process for them. For an odd prime p, residues $\bmod\ p^t$ $(t{\geq}1)$ which are prime to p form a cyclic group of order $\phi(p^t)$ under multiplication (e.g. [10], Thm. 7.2.10). Hence (19) will have exactly two solutions, say $\pm y_t$, or none at all for $\mu g^{-1} = p^t$. Any solutions modulo p^{t+1} must then have the form $\pm(y_t{+}\delta_t p^t)$ for some δ_t in $[0 \mathinner{..} p{-}1]$. Each δ_t can be tried in turn to find the solutions. Once more, no solutions at any point means the wrong value for $k{+}rE$; otherwise there will be two.

Using the Chinese Remainder Theorem to combine the results modulo the powers of 2, 3 and 5, the number of solutions is the product of the number of solutions modulo the individual prime powers, namely $4{\times}2{\times}2 = 16$, assuming at least $2^3{\times}3{\times}5$ divides the modulus and each has at least one solution. These 16 solutions are all different because, by reducing them modulo each of the three prime powers, different solutions sets modulo the prime powers are recovered.

The effort in obtaining and combining the solutions in this way for a fixed $k{+}rE$ is proportional to the sum of the exponents of the powers of 2, 3 and 5 dividing μg^{-1}. It is thus a computation of order at most $\log(\mu g^{-1})$, and hence of order at most $\log \mu$. As j was chosen so that $\mu \geq 2^{n/4}$ and there are up to BE choices for $k{+}rE$, the solutions now provide all possible values for the P_0 in Theorem 3 for a computational effort of $O(nBE)$. Hence,

Theorem 4. *For a public RSA modulus N with n bits, public exponent E and exponent blinding using a random multiple of $\phi(N)$ which is bounded above by B, given the least significant randomary digits of D whose product of bases is at least $N^{1/4}$, it is possible to factorise N in time which is polynomial in n and linear in BE.*

As a corollary, it is possible to deduce the maximum effort required to deduce the secret key under the assumptions stated in Theorems 1 and 2. The number of partial keys which need to be tested is of order $(N^{1/4})^{3/5}$ and $(N^{1/4})^{1/3}$ respectively.

Corollary 1. *Under the threat scenarios described in Theorems* 1 *and* 2 *with known public modulus N and encryption key E, it is possible to factorise N in time which is a product of polynomial time in* $\log N$ *and linear time in $BEN^{3/20}$ or $BEN^{1/12}$ respectively.*

It is now clear that exponent blinding is important in the prevention of such attacks, and such blinding provides more tamper resistance per bit than increasing the key length. If similar but more powerful attacks than those described in [22] can be developed, then blinding will become strongly advisable if the key length is under 1024 bits. Furthermore, we have assumed the bases m_i were chosen at random. However, biasing the choice for efficiency or other reasons further reduces the time for executing the attack.

5 Conclusion

Two techniques have been demonstrated for using knowledge of the public RSA modulus and exponent in order to reduce the computational effort of recovering the secret key from partial knowledge of a randomary digit expansion of D generated by the MIST exponentiation algorithm. The first relies on the scarcity of large numbers which are products of powers of only 2, 3 and 5. The other generalises the well-known, lattice-based, "Bellcore Attack" method of Boneh, Durfee and Frankel [2].

Under threat models in which squares and multiplies or operand re-use can be recognised during a single exponentiation, these techniques do not undermine the confidence that power and EMR attacks (SPA/DPA and SEMA/DEMA) on the MIST exponentiation algorithm still appear to be computationally infeasible for standard key lengths and sensible implementations which include appropriate blinding. Of course, under the same assumptions, the classical sliding windows and m-ary exponentiation schemes provide no resistance whatsoever.

References

1. D. Agrawal, B. Archambeault, J. R. Rao & P. Rohatgi, *The EM Side-Channels*, Cryptographic Hardware and Embedded Systems – CHES 2002, B. Kaliski, Ç. Koç & C. Paar (editors), LNCS **2523**, Springer-Verlag, 2002, *to appear.*
2. D. Boneh, G. Durfee & Y. Frankel, *Exposing an RSA Private Key Given a Small Fraction of its Bits*, Advances in Cryptology – AsiaCrypt '98, K. Ohta & D. Pei (editors), LNCS **1514**, Springer-Verlag, 1998, 25–34.
3. D. Coppersmith, *Small Solutions to Polynomial equations and low exponent RSA vulnerabilities*, Journal of Cryptology **10** (1997), 233–260.
4. K. Gandolfi, C. Mourtel & F. Olivier, *Electromagnetic Analysis: Concrete Results*, Cryptographic Hardware and Embedded Systems – CHES 2001, Ç. Koç, D. Naccache & C. Paar (editors), LNCS **2162**, Springer-Verlag, 2001, 251–261.
5. J. C. Ha & S. J. Moon, *Randomized signed-scalar multiplication of ECC to resist power attacks*, Cryptographic Hardware and Embedded Systems – CHES 2002, B. Kaliski, Ç. Koç & C. Paar (editors), LNCS **2523**, Springer-Verlag, 2002, *to appear.*
6. K. Itoh, J. Yajima, M. Takenaka, & N. Torii, *DPA Countermeasures by improving the window method*, Cryptographic Hardware and Embedded Systems – CHES 2002, B. Kaliski, Ç. Koç & C. Paar (editors), LNCS **2523**, Springer-Verlag, 2002, *to appear.*
7. D. E. Knuth, *The Art of Computer Programming*, vol. **2**, "Seminumerical Algorithms", 2nd Edition, Addison-Wesley, 1981, 441–466.
8. P. Kocher, *Timing Attack on Implementations of Diffie-Hellman, RSA, DSS, and other systems*, Advances in Cryptology – CRYPTO '96, N. Koblitz (editor), LNCS **1109**, Springer-Verlag, 1996, 104–113.
9. P. Kocher, J. Jaffe & B. Jun, *Differential Power Analysis*, Advances in Cryptology – CRYPTO '99, M. Wiener (editor), LNCS **1666**, Springer-Verlag, 1999, 388–397.
10. R. Kumanduri & C. Romero, *Number Theory with Computer Applications*, Prentice Hall, 1998, ISBN 0-13-801812-X.
11. P.-Y. Liardet & N. P. Smart, *Preventing SPA/DPA in ECC Systems using the Jacobi Form*, Cryptographic Hardware and Embedded Systems – CHES 2001, Ç. Koç, D. Naccache & C. Paar (editors), LNCS **2162**, Springer-Verlag, 2001, 391–401.
12. T. S. Messerges, E. A. Dabbish & R. H. Sloan, *Power Analysis Attacks of Modular Exponentiation in Smartcards*, Cryptographic Hardware and Embedded Systems (Proc. CHES 99), C. Paar & Ç. Koç (editors), LNCS **1717**, Springer-Verlag, 1999, 144–157.
13. K. Okeya & K. Sakurai, *On Insecurity of the Side Channel Attack Countermeasure using Addition-Subtraction Chains under Distinguishability between Addition and Doubling*, Information Security and Privacy, L. Batten & J. Seberry (editors), LNCS **2384**, Springer-Verlag, 2002, 420–435.
14. E. Oswald & M. Aigner, *Randomized Addition-Subtraction Chains as a Countermeasure against Power Attacks*, Cryptographic Hardware and Embedded Systems – CHES 2001, Ç. Koç, D. Naccache & C. Paar (editors), LNCS **2162**, Springer-Verlag, 2001, 39–50.
15. J.-J. Quisquater & D. Samyde, *ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards*, Smart Card Programming and Security (e-Smart 2001), LNCS **2140**, Springer-Verlag, 2001, 200–210.
16. J.-J. Quisquater & D. Samyde, *Eddy current for Magnetic Analysis with Active Sensor*, Proc. e-Smart 2002, Nice, September 2002, 183–194.

17. C. D. Walter & S. Thompson, *Distinguishing Exponent Digits by Observing Modular Subtractions*, Topics in Cryptology – CT-RSA 2001, D. Naccache (editor), LNCS **2020**, Springer-Verlag, 2001, 192–207.
18. C. D. Walter, *Sliding Windows succumbs to Big Mac Attack*, Cryptographic Hardware and Embedded Systems – CHES 2001, Ç. Koç, D. Naccache & C. Paar (editors), LNCS **2162**, Springer-Verlag, 2001, 286–299.
19. C. D. Walter, *Precise Bounds for Montgomery Modular Multiplication and Some Potentially Insecure RSA Moduli*, Topics in Cryptology – CT-RSA 2002, B. Preneel (editor), LNCS **2271**, Springer-Verlag, 2001, 30–39.
20. C. D. Walter, *Improvements in, and relating to, Cryptographic Methods and Apparatus*, UK Patent Application 0126317.7, Comodo Research Laboratory, 2001.
21. C. D. Walter, *MIST: An Efficient, Randomized Exponentiation Algorithm for Resisting Power Analysis*, Topics in Cryptology – CT-RSA 2002, B. Preneel (editor), LNCS **2271**, Springer-Verlag, 2002, 53–66.
22. C. D. Walter, *Some Security Aspects of the MIST Randomized Exponentiation Algorithm*, Cryptographic Hardware and Embedded Systems – CHES 2002, B. Kaliski, Ç. Koç & C. Paar (editors), LNCS **2523**, Springer-Verlag, 2002, *to appear.*
23. C. D. Walter, *Breaking the Liardet-Smart Randomized Exponentiation Algorithm*, Proc. Cardis '02, USENIX, *to appear.*