A Third is All You Need: Extended Partial Key Exposure Attack on CRT-RSA with Additive Exponent Blinding

Yuanyuan Zhou[1][0000−0002−8703−219X], Joop van de Pol[2], Yu Yu[3][0000−0002−9278−4521], and François-Xavier Standaert[1][0000−0001−7444−0285]

[1] Crypto Group, ICTEAM Institute, UCLouvain, Louvain-la-Neuve, Belgium
{zhou.yuanyuan@gmail.com, fstandae@uclouvain.be}
[2] Unaffiliated
csjhvdp@my.bristol.ac.uk
[3] Shanghai Jiao Tong University, Shanghai 200240, China
yuyu@yuyu.hk

Abstract. At Eurocrypt 2022, May et al. proposed a partial key exposure (PKE) attack on CRT-RSA that efficiently factors N knowing only a $\frac{1}{3}$-fraction of either most significant bits (MSBs) or least significant bits (LSBs) of private exponents d_p and d_q for public exponent $e \approx N^{\frac{1}{12}}$. In practice, PKE attacks typically rely on the side-channel leakage of these exponents, while a side-channel resistant implementation of CRT-RSA often uses additively blinded exponents $d'_p = d_p + r_p(p-1)$ and $d'_q = d_q + r_q(q-1)$ with unknown random blinding factors r_p and r_q, which makes PKE attacks more challenging.

Motivated by the above, we extend the PKE attack of May et al. to CRT-RSA with additive exponent blinding. While admitting $r_p e \in (0, N^{\frac{1}{4}})$, our extended PKE works ideally when $r_p e \approx N^{\frac{1}{12}}$, in which case the entire private key can be recovered using only $\frac{1}{3}$ known MSBs or LSBs of the blinded CRT exponents d'_p and d'_q. Our extended PKE follows their novel two-step approach to first compute the key-dependent constant k' ($ed'_p = 1 + k'(p-1)$, $ed'_q = 1 + l'(q-1)$), and then to factor N by computing the root of a univariate polynomial modulo $k'p$. We extend their approach as follows. For the MSB case, we propose two options for the first step of the attack, either by obtaining a single estimate $k'l'$ and calculating k' via factoring, or by obtaining multiple estimates $k'l'_1, \ldots, k'l'_z$ and calculating k' probabilistically via GCD.
For the LSB case, we extend their approach by constructing a different univariate polynomial in the second step of the LSB attack. A formal analysis shows that our LSB attack runs in polynomial time under the standard Coppersmith-type assumption, while our MSB attack either runs in sub-exponential time with a reduced input size (the problem is reduced to factor a number of size $e^2 r_p r_q \approx N^{\frac{1}{6}}$) or in probabilistic polynomial time under a novel heuristic assumption. Under the settings of the most common key sizes (1024-bit, 2048-bit, and 3072-bit) and blinding factor lengths (32-bit, 64-bit, and 128-bit), our experiments verify the

validity of the Coppersmith-type assumption and our own assumption, as well as the feasibility of the factoring step.
To the best of our knowledge, this is the first PKE on CRT-RSA with experimentally verified effectiveness against 128-bit unknown exponent blinding factors. We also demonstrate an application of the proposed PKE attack using real partial side-channel key leakage targeting a Montgomery Ladder exponentiation CRT implementation.

Keywords: Partial Key Exposure · Additive Blinding · CRT-RSA · Coppersmith method

1 Introduction

1.1 Partial Key Exposure Attacks on (CRT-)RSA

As one of the longest-serving and the most widely used public-key cryptosystems, RSA can use the Chinese Remainder Theorem (CRT) optimization to speed up exponentiation operations, which is especially favored by power-constrained embedded systems. To mitigate side-channel attacks [35], a real implementation of (CRT-)RSA often adopts blinding countermeasures such as message blinding, modulus blinding and exponent blinding [15, 19, 22, 35, 40], of which additive exponent blinding [35] was referred to by the BSI as the "classical exponent blinding" [11] and is widely deployed, e.g., in open source cryptographic libraries MbedTLS [44], Libgcrypt [37], and Botan [9]. Throughout this work, we consider CRT-RSA with additive exponent blinding as the main target of our research.

At Eurocrypt'96, Coppersmith presented a novel lattice-based method to find small solutions of univariate modular polynomials with some applications to cryptanalysis of RSA [17], and he [16] extended this method to bivariate equations to factor RSA modulus N with half MSBs of one of its prime factors. Boneh et al. [6] introduced PKE attacks to recover the full RSA private key using a few consecutive MSBs or LSBs of the private key based on the Coppersmith method. Many subsequent works continued the research on PKE, but most do not take into account the widely deployed exponent blinding countermeasure [3, 7, 23, 28, 38, 42, 43, 50, 56–59], with only a couple of exceptions to the best of our knowledge [24, 32]. Fouque et al. [24] investigated PKE attacks on an additively blinded private exponent with up to 32-bit blinding factors and very small e values for RSA without CRT. They showed that non-consecutive known bits—which could realistically result from SCA (Side-Channel Analysis) leakage of sliding-window exponentiation implementations—of the additively blinded private exponent could be used to recover the private exponent. However, their attack relies on the fact that both the public exponent e and the blinding factor are small to enable brute force, and requires multiple traces to gradually retrieve the entire private key. In particular, the retrieval requires several (50 in their experiments) instantiations of the recovered non-consecutive exponent bits (when $e = 3$) or 2-bit consecutive exponent bits (when $e = 2^{16} + 1$), each providing a $\frac{1}{64}$ to $\frac{1}{16}$ (depending on the public exponent, key length) non-consecutive or

consecutive portion of the blinded exponents. Further, it works with very small e values (the cases of $e = 3$ and $2^{16} + 1$ were experimentally verified) and up to 32-bit blinding factors. Joye and Lepoint [32] also focused on PKE on RSA with additive exponent blinding. They constructed trivariate or bivariate polynomials based on Coppersmith's lattice techniques to recover the whole key with partial known blinded exponent MSBs/LSBs. Their attack requires in the best case more than half of the known MSBs or LSBs of the blinded private exponent, which was experimentally verified. In addition to the aforementioned works that assume some **partial** bits of the (blinded) private exponent(s) known with certainty, another line of works considers the case where **all** bits of the blinded private exponent(s) are classified, but where they can be wrongly classified with some error probability, e.g. [4, 52–54]. In summary, previous PKE attacks on (CRT-)RSA with additive exponent blinding mostly consider restricted cases of e and small blinding factors using special-structured partial key leakages from multiple observations, or need more than half of blinded private exponent from a single observation of the partial key leakage, or need a full recovery of the (blinded) exponent bits with errors.

1.2 Our contribution

In this work, we aim to answer the question: *how and to which extent can we extend the PKE attack to CRT-RSA with additive exponent blinding (ideally using as few as possible consecutive MSBs/LSBs of the private CRT exponent from a single observation of the partial key leakage) both theoretically and empirically?*

Our contributions in this context are summarized as follows.

First, we extend the PKE attack on CRT-RSA with unblinded exponents (d_p and d_q) [43] to that with additively blinded ones, i.e., $d'_p = d_p + r_p(p-1)$ and $d'_q = d_q + r_q(q-1)$ for unknown random blinding factors r_p and r_q.[1] Instead of restricting r_p and r_q to 32 bits or any lengths within the reach of brute force [24], we allow the blinding factors up to some exponential size N^γ, and concretely 128 bits or even more. Our extended PKE works ideally when the products of public exponent $e = N^\alpha$ and the blinding factors (i.e., $e \cdot r_p$ and $e \cdot r_q$ in the LSB case; $e \cdot r_p^{\frac{2}{3}}$ and $e \cdot r_q^{\frac{2}{3}}$ in the MSB case) are roughly $N^{\frac{1}{12}}$, i.e., $\alpha + \gamma \approx \frac{1}{12}$ or $\alpha + \frac{2}{3}\gamma \approx \frac{1}{12}$. In this setting, the attackers/evaluators need only a $\frac{1}{3}$-fraction of MSBs or LSBs of d'_p and d'_q to recover the entire private key. In general, as for [43], our extended PKE attack works in the small e regime (albeit with more MSBs/LSBs), i.e., $0 < e < N^{\frac{1}{4}-\gamma}$, which arguably covers most common choices in practice (e.g., NIST mandates $e \in (2^{16}, 2^{256})$ [45]).

Second, we formally analyze the asymptotic time complexity of the proposed PKE attack. In the LSB case, it runs in polynomial time under the standard

[1] We argue that the extension is non-trivial as, intuitively (from an information theoretic point of view), the random unknown blinding factors (r_p and r_q) effectively reduce the information on d_p, d_q, k, and l that can be obtained directly from partial key exposure.

Coppersmith-type heuristic assumption. In the MSB case, it either runs in sub-exponential time with a reduced input size (i.e., the number to factor in this case is of size $e^2 \cdot r_p \cdot r_q$, significantly smaller than the RSA modulus N), or runs in probabilistic polynomial time (PPT) under a novel assumption that certain values corresponding to the blinded exponents are coprime with a certain probability, but with the additional cost that, for one of the exponents, multiple observations of the blinded execution are needed. We implement the proposed PKE attack for the most common key sizes (1024-bit, 2048-bit, and 3072-bit) and blinding factor lengths (32-bit, 64-bit, and 128-bit), and our extensive experiments not only confirm the effectiveness of the Coppersmith-type heuristic (in the LSB case), but also that of our own assumption and the feasibility of the sub-exponential (in $e^2 \cdot r_p \cdot r_q$) time complexity (in the MSB case). In particular, the MSB attack succeeds with the probability that randomly distributed numbers are coprime (using multiple observations of the blinded exponents), or completes mostly in seconds or minutes for most parameter settings (using single observation of the blinded exponents). In the most challenging case occasionally observed for 2048-bit (resp., 3072-bit) key with 128-bit additive blinding exponent, it finishes in less than 1.5 (resp., 50) hours. To the best of our knowledge, this is the first[2] PKE on CRT work that experimentally verifies the effectiveness using 128-bit additive exponent blinding factors. The experiment parameters also provide hands-on references (such as lattice dimensions, and required running time) to the attackers/evaluators for applying PKE attacks.

Third, the extended PKE attack can be combined with various forms of side-channel attacks, e.g., cold boot attacks [27,61], cache-timing micro-architectural attacks [2, 62], and timing, power or electromagnetic analysis attacks on CRT-RSA [13,29,35,51,64], to significantly enhance the applicability of these attacks (by reducing the goal of full key recovery to obtaining only a $\frac{1}{3}$-fraction of leakage). Note that this PKE attack can tolerate some errors of the partial key leakage,[3] i.e., combining the error-free MSBs or LSBs of d'_p from one observation of the partial key leakage, and those of d'_q from another one. To this end, we demonstrate an application of the extended PKE attacks on deep learning-based partial side-channel leakage from a typical real-world target, i.e., a Montgomery Ladder exponentiation CRT implementation with 2048-bit key and 64-bit additive exponent blinding on a 45 nm secure microcontroller with RSA co-processor.

Several preliminaries and proofs in this work closely follow the corresponding versions in [43]. In those cases, the contributions of May et al. are included here in full such that this work is self-contained and easier to read.

1.3 Organization of the paper

The rest of this paper is organized as follows. Section 2 introduces the necessary background information about the Coppersmith method and the state-of-the-art

[2] The work [54] estimates the effort for 128-bit blinding factors with error probability 0.05, but only verifies the estimates experimentally for 32-bit blinding factors.

[3] It is a commonly used strategy also for error-tolerant lattice-based attacks on (EC)DSA as pointed out in [1].

PKE attack on unblinded CRT exponents published at Eurocrypt'22. Section 3 provides a mathematical proof of the extended PKE attack in the additive blinding scenario. Afterwards, we experimentally verify the effectiveness and time complexity of this approach in Section 4. Finally, in Section 5, an application of the extended PKE attack is demonstrated using partial side-channel key leakage of a typical real-world Montgomery Ladder exponentiation implementation with additive exponent blinding countermeasure.

2 Preliminaries

2.1 Notations

By 'log' we denote the base 2 logarithm, 'ln' denotes the natural logarithm, and $\zeta(.)$ denotes Riemann's zeta function. We use capital letters for random variables and small caps for their realizations. We use capital bold font letters for matrices (e.g., $\mathbf{M}$) and small bold caps for vectors (e.g., $\mathbf{v}$). We use $|\det \mathbf{M}|$ to denote the absolute value of the *determinant* of the matrix $\mathbf{M}$, which corresponds to the determinant of the lattice spanned by this matrix.

2.2 Coppersmith's method

Similar to many previous PKE works, we rely on Coppersmith's method to find small modular roots of multivariate polynomials [18]. We first give a simple introduction to the multivariate Coppersmith's lattice-based method.

Let $f \in \mathbb{Z}[x_1, ..., x_j]$ be a j-variate polynomial over the integers with maximum degree δ in each variable separately, this polynomial has a small root $r = (r_1, ..., r_j)$ modular an integer M and let $U_i \in \mathbb{Z}$ denote some known bounds. The goal is to find integers $|r_i| \leq U_i$ such that $f(r_1, ..., r_j) = 0$ in polynomial time.

To this end, a series of so-called *shift-polynomials* with a chosen sufficiently big positive integer $m \in \mathbb{N}$ and the indices $i_0, ..., i_j \in \mathbb{N}$ is constructed as below:

$$s_{[i_0,...,i_j]}(x_1, ..., x_j) = f^{i_0}(x_1, ..., x_j) \cdot x_1^{i_1} \cdot ... \cdot x_n^{i_j} \cdot M^{m-i_0}.$$

Those polynomials have the root r modulo M^m by construction. A subset of the constructed shift-polynomials $s_{[i_0,...,i_j]}(U_1x_1, ..., U_jx_j)$ is selected to generate an l-dimensional lattice $\mathcal{L}$ with their coefficient vectors such that the lattice $\mathcal{L}$ has a triangular basis matrix $\mathbf{B}$. Given a large enough chosen integer m and the determinant of $\mathcal{L}$ fulfilling the *enabling condition*, i.e., $|\det \mathbf{B}| \leq M^{ml}$, then a collection of j polynomials $p_1(x_1, ..., x_j), ..., p_j(x_1, ..., x_j)$ can be computed in polynomial time. This is due to the fact that the coefficient vectors of the polynomials $p_j(U_1x_1, ..., U_jx_j)$ are elements of $\mathcal{L}$, which is generated by the coefficient vectors of the polynomials $s_i(U_1x_1, ..., U_jx_j)$. Therefore, the computation of $p_1(x_1, ..., x_j), ..., p_j(x_1, ..., x_j)$ can be done with the widely used LLL lattice basis reduction algorithm [36]. Those computed polynomials have the root r not only modulo M^m but also over the integers as proved by Howgrave-Graham in [30] as stated below.

Lemma 1 (Howgrave-Graham [30]). *Let $h(x_1, ..., x_k) \in \mathbb{Z}[x_1, ..., x_k]$ be a polynomial in at most ω monomials. Suppose that $h(r_1, ..., r_k) \equiv 0 \mod M^m$ for some positive integer m. Also let $|r_i| < X_i$ for $1 \leq i \leq k$ and*

$$\|h(x_1X_1, ..., x_kX_k)\| < \frac{M^m}{\sqrt{\omega}}.$$

Then, $h(r_1, ..., r_k) = 0$ holds over the integers.

In the case of $j = 1$, i.e., the polynomial f is univariate, the root r can be straightforwardly resolved from p_1 using standard techniques, e.g., Newton's method. In the case of $j > 1$, if such (computed) polynomials generate an ideal $(p_1, ..., p_j)$ of zero-dimensional variety, we can use a Gröbner basis to resolve their root, it means that the RSA modulus N can be efficiently factored in our context. However, the existence of such a zero-dimensional variety relies on the standard Coppersmith-type heuristic assumption (see also [43, Assumption 1]) as below:

Assumption 1 *In the multivariate setting, Coppersmith's method yields polynomials that generate an ideal of zero-dimensional variety.*

It is worth mentioning that it is essential to experimentally verify the validity of this assumption because it might fail in some cases, e.g., in the small e regime of the TK attack [57] as pointed out in [43].

2.3 PKE attack on unblinded CRT exponents

We first briefly recall the PKE attack on unblinded CRT private exponents in [43]. We denote an RSA public key as (N, e), where $N = pq$ and $e = N^\alpha$. In practice, p and q usually have the same bit-length, so they are bounded as $p, q = \Theta(\sqrt{N})$. The unblinded CRT exponents are marked as d_p and d_q, without loss of generality, which are assumed to be full-size, i.e., $d_p, d_q = \Theta(\sqrt{N})$. The CRT private key exponentiation is executed as

$$M = (C^{d_q} \bmod q) + q \cdot \left((q^{-1} \bmod p) \cdot \left((C^{d_p} \bmod p) - (C^{d_q} \bmod q)\right) \bmod p\right) \ ,$$

where C is the ciphertext to be decrypted and M is the plaintext. The definition of PKE attacks on RSA is, that the RSA private key can be fully recovered in polynomial time given only a constant fraction of the secret exponent(s). This recent work put forward the state-of-the-art PKE attack on CRT exponents in small e regime from $\frac{1}{2}$ known LSBs of d_p (or d_q, see [7, Fig. 4]) to $\frac{1}{3}$ known LSBs (or MSBs) of both d_p and d_q (see [43, Fig. 1]) when $e \approx N^{\frac{1}{12}}$. It is a novel two-step (both steps can finish in polynomial time) approach as below:

- **Step 1:** Compute CRT key constants k ($ed_p = 1 + k(p - 1)$) and l ($ed_q = 1 + l(q - 1)$) based on the known parts of d_p and d_q and the public key (N, e). If the MSBs of d_p and d_q are known, this step is trivial to solve

a quadratic polynomial equation. In the LSB case (the LSBs are known), Coppersmith's lattice-based method [18] is involved so this step relies on a standard Coppersmith-type heuristic assumption as mentioned above in 1. They experimentally verified this heuristic in addition to the mathematical proof.

– **Step 2:** Recover the unknown bits of d_p and subsequently the factor p of N using the previously calculated k. The authors considered it as an extension of Howgrave-Graham's *approximate divisor* algorithm [31] to the case of *approximate divisor multiples* for some known multiple k of an unknown divisor p of N. They mathematically proved that this unknown divisor can be recovered in polynomial time.

3 Extended PKE attack on additively blinded CRT exponents

As discussed in Subsection 1.2, the problem to be solved in this work is: to which extent can one reveal the whole private key when only part of the additively blinded CRT exponent bits is disclosed. In the following, we describe the extended PKE attack on additively blinded CRT exponents based on [43].

May et al. only considered the PKE attack on unblinded CRT private exponents in [43], we extend their work to cover the additively blinded CRT private exponents, and it is called EPKE (Extended Partial Key Exposure) attack in the rest of this paper.

Using the additive blinding factors r_p and r_q, the blinded CRT exponents are $d'_p = d_p + r_p(p-1)$ and $d'_q = d_q + r_q(q-1)$, where $r_p, r_q = N^\gamma$. Let ($d_p'^{(M)}$, $d_q'^{(M)}$) be the MSBs of d'_p and d'_q, and ($d_p'^{(L)}$, $d_q'^{(L)}$) be the LSBs. So, we have:

$$d'_p = d_p'^{(M)} 2^i + d_p'^{(L)},$$
$$d'_q = d_q'^{(M)} 2^i + d_q'^{(L)}.$$

Hereafter, we label it as the MSB case if ($d_p'^{(M)}$, $d_q'^{(M)}$) are known (e.g., by side-channel attacks), or as the LSB case if ($d_p'^{(L)}$, $d_q'^{(L)}$) are known. In addition to the Coppersmith-type assumption required for the LSB case in [43], our algorithm in the MSB case has two options with different requirements:

– Given only parts of two blinded exponents d'_p and d'_q, a factoring step of a number roughly $\frac{1}{6}$ the size of N is required, leading to a sub-exponential time complexity.
– Given parts of one blinded exponent d'_p and multiple blinded exponents $d'_{q,i}$, the time complexity remains polynomial, with a success probability according to the additional heuristic Assumption 2 defined below (which will be experimentally verified).

Assumption 2 *For z distinct blinded exponents $d'_{q,i} = d_q + r_{q,i}(q-1)$, the corresponding values $l'_i = l + r_{q,i}e$ are coprime with probability $1/\zeta(z)$.*

According to [20, (1.2) and (3.1)], this assumption holds if the values $l'_i = l + r_{q,i}e$ are distributed uniformly. However, the assumption that these values are distributed uniformly is too strong. In any case, Assumption 2 is sufficient and is experimentally validated as described in Section 4.

Our main result of EPKE regarding factoring N is the extension of [43, **Theorem 1**] as follows:

Theorem 1. *Let (N, e) be the public key and N be large enough with $e = N^\alpha$. Given the MSBs $(d_p'^{(M)}, d_{q,1}'^{(M)}, \ldots, d_{q,z}'^{(M)})$ or the LSBs $(d_p'^{(L)}, d_q'^{(L)})$ of the additively blinded CRT exponents (d'_p, d'_q) with blinding factors $r_p, r_{q,i} = N^\gamma$. If the unknown parts of d'_p and $d'_{q(,i)}$ are upper bounded by N^δ and $\delta \geq \gamma$ (which already holds for commonly used up to 128-bit additive blinding factors), where*

$$\gamma \leq \delta < \min\{\frac{1}{4} + \alpha + \gamma, \frac{1}{2} - 2\alpha - \gamma\}$$

for the MSB case, or

$$\delta < \min\{\frac{1}{4} + \alpha + \gamma, \frac{1}{2} - 2\alpha - 2\gamma\}$$

for the LSB case, then N can be factored

- *in polynomial time under Coppersmith's heuristic assumption (see Assumption 1) for the LSB case,*
- *in sub-exponential time $\exp(c(\ln(N^{2\alpha+2\gamma}))^t(\ln\ln(N^{2\alpha+2\gamma}))^{1-t})$ for the MSB case ($z = 1$) with constants c and t dependent on the underlying integer factorization algorithm, or*
- *in probabilistic polynomial time under Assumption 2 for the MSB case ($z > 1$).*

Proof outline. The additively blinded CRT exponents d'_p, d'_q satisfy the equations

$$ed'_p = 1 + k'(p-1), \tag{1}$$

$$ed'_q = 1 + l'(q-1), \tag{2}$$

where $k' = k + r_p e$ and $l' = l + r_q e$ as we have

$$ed_p = 1 + k(p-1),$$

$$ed_q = 1 + l(q-1).$$

Similar to [43], we use a two-step method to factor N.

Step 1: Given the unknown parts of d'_p and d'_q being upper bounded by $N^{\frac{1}{2}-2\alpha-\gamma}$ (MSB case) or $N^{\frac{1}{2}-2\alpha-2\gamma}$ (LSB case),

- in Section 3.1 we prove that k' can be calculated in sub-exponential time $\exp(c(\ln(N^{2\alpha+2\gamma}))^t(\ln\ln(N^{2\alpha+2\gamma}))^{1-t})$ using a single known MSBs of d'_q in the MSB case,

- in Section 3.2 we prove that k' can be calculated in probabilistic polynomial time using multiple known MSBs of d'_q in the MSB case under Assumption 2, and
- in Section 3.3 we prove that k' can be calculated in polynomial time under the Coppersmith-type heuristic Assumption 1 for multivariate polynomials in the LSB case.

Step 2: With the computed k' and the unknown MSBs or LSBs of d'_p bounded by $N^{\frac{1}{4}+\alpha+\gamma}$, we provide an algorithm for factoring N in polynomial time based on the novel result of May et al. of *approximate divisor multiples* [43, **Theorem 3**]. ■

Note that the proof outline for Theorem 1 and the proofs of Lemmas 2, 4 and 5 follow the work of May et al. [43] and only differ in handling those two random blinding factors r_p and r_q.

3.1 Step 1a: Computing (k', l') from MSB using factoring.

In the MSB case, to compute k' using known MSBs of two blinded exponents d'_p and d'_q, the basic idea is to compute the product $k'l'$ followed by a factorization of this product to find the candidates of k' using the modular sum of $k'+l' \mod e$. The time complexity here is determined by the factorization of the product $k'l'$, which is sub-exponential according to [8, **Formula 3.10**]. The time complexities of the state-of-the-art fastest known integer factorization algorithms are of the form $\exp(c(\ln n)^t(\ln\ln n)^{1-t}))$ for some constants $c > 0$ and $0 < t < 1$, in which n is the number to be factored. For QS (Quadratic Sieve) and ECM (Elliptic-Curve Method) factorization algorithms, $t = \frac{1}{2}$; for NFS (Number Field Sieve), $t = \frac{1}{3}$. Note that, in practice, this sub-exponential time complexity is affordable because of the small size of e and commonly used small sizes (very often 64 bits or even 32 bits, at most 128 bits) of exponent blinding factors r_p and r_q. We also experimentally verify this in Section 4. Most of the experiments take seconds or minutes for the factorization step and occasionally need up to one and a half hours (resp., fifty hours) for 2048-bit (resp., 3072-bit) key with a regular PC.

Lemma 2 ((k', l') from MSB). *Let (N, e) be the public key and N be large enough with $e = N^\alpha$. Given the MSBs ($d_p'^{(M)}$, $d_q'^{(M)}$), if the unknown parts ($d_p'^{(L)}$, $d_q'^{(L)}$) are upper bounded by N^δ, where*

$$\gamma \leq \delta < \frac{1}{2} - 2\alpha - \gamma,$$

then (k', l') can be computed in time $\exp(c(\ln(N^{2\alpha+2\gamma}))^t(\ln\ln(N^{2\alpha+2\gamma}))^{1-t})$.

Proof. According to Equations (1) and (2), we have

$$k'p = k' - 1 + ed'_p,$$
$$l'q = l' - 1 + ed'_q.$$

Multiplying those two equations, we get

$$k'l'N = (k'-1)(l'-1) + ed'_p(l'-1) + ed'_q(k'-1) + e^2 d'_p d'_q. \tag{3}$$

Let

$$\widetilde{A} = \frac{2^{2i} e^2 d_p'^{(M)} d_q'^{(M)}}{N}.$$

$\widetilde{A}$ can be easily computed with the known ($d_p'^{(M)}$, $d_q'^{(M)}$). In the following we demonstrate that $\lceil \widetilde{A} \rceil = k'l' - o(1)$ with $\delta < \frac{1}{2} - 2\alpha - \gamma$. Therefore, the estimation of the product $k'l'$ equals $\lceil \widetilde{A} \rceil$ given a sufficiently large N.

From Equation (3) we have

$$\begin{aligned} k'l'N - \widetilde{A}N &= (k'-1)(l'-1) + ed'_p(l'-1) + ed'_q(k'-1) + e^2 d'_p d'_q - \widetilde{A}N \\ &= (k'-1)(l'-1) + ed'_p(l'-1) + ed'_q(k'-1) \\ &\quad + 2^i e^2 (d_p'^{(M)} d_q'^{(L)} + d_p'^{(L)} d_q'^{(M)}) + e^2 d_p'^{(L)} d_q'^{(L)}. \end{aligned}$$

Considering

$$d'_p, d'_q = \Theta(N^{\frac{1}{2}+\gamma}), d_p'^{M}, d_q'^{M} = \Theta(N^{\frac{1}{2}+\gamma-\delta}), d_p'^{(L)}, d_q'^{(L)}, 2^i = \Theta(N^{\delta}), k', l' = \Theta(N^{\alpha+\gamma}),$$

we have

$$\begin{aligned} k'l'N - \widetilde{A}N &= \mathcal{O}(N^{2\alpha+2\gamma}) + \mathcal{O}(N^{2\alpha+\frac{1}{2}+2\gamma}) + \mathcal{O}(N^{2\alpha+\frac{1}{2}+\gamma+\delta}) + \mathcal{O}(N^{2\alpha+2\delta}) \\ &= \mathcal{O}(N^{2\alpha+\frac{1}{2}+\gamma+\delta}), \end{aligned}$$

and further we obtain

$$k'l' - \widetilde{A} = \mathcal{O}(N^{\delta+2\alpha+\gamma-\frac{1}{2}}) = o(1).$$

In conclusion, the product $k'l'$ can be calculated in polynomial time $\mathcal{O}(\log^2 N)$.

Further we can deduce from Equation (3) that

$$k' + l' \equiv 1 - k'l'(N-1) \mod e, \tag{4}$$

where the right-hand side can be computed with the obtained product $k'l'$ and the public key (N, e).

To compute k', the product $k'l'$ is factored using the state-of-the-art integer factorization algorithms [8] to get all its factors in sub-exponential time $\exp(c(\ln(N^{2\alpha+2\gamma}))^t(\ln\ln(N^{2\alpha+2\gamma}))^{1-t})$. Then, the search for a combination of those factors that satisfies Equation 4 will reveal two values corresponding to k' and l'. Putting all together, the time complexity of computing k' in the MSB case is $\exp(c(\ln(N^{2\alpha+2\gamma}))^t(\ln\ln(N^{2\alpha+2\gamma}))^{1-t})$. ■

As opposed to the proof in [43], it is not trivial to recover k' and l' from one product $k'l'$ in polynomial time. Trying to use Equation (4) to compute a univariate polynomial with k' and l' as roots will fail because, unlike $0 < k + l < 2e$, there is no small interval that $k' + l'$ will fall into to derive $k' + l'$ from $k' + l' \mod e$, which is due to the blinding factors r_p and r_q. Since $k = k' \mod e$ and $l = l' \mod e$, it may seem feasible to find k and l instead, but the method from [43] will not work as it is non-trivial to derive the required product kl from $kl \mod e = k'l' \mod e$. Constructing a multivariate polynomial as in the LSB case to recover k and l using Coppersmith's method is also non-trivial, as the modulus e is not large enough compared to roots k and l. Regardless, the factoring method is fast enough in practice shown by the experimental results in Section 4.

In the end, two candidates for k' are found because the modular sum cannot tell which of those two possible values is k' and which is l'. The fact that two candidates are found for k' means that Step 2 may have to be repeated for both candidates. However, it is not described in [43] how the two values k and l can be distinguished in the MSB case, while they are two equivalent roots of a univariate polynomial (as opposed to the LSB case, where they together form one root of a bivariate polynomial).

3.2 Step 1b: Computing $(k', l'_1, \ldots, l'_z)$ from MSB using GCD.

This section describes how to compute, in the MSB case, k' using known MSBs of one blinded exponent d'_p and multiple blinded exponents $d'_{q,i}$ in heuristic probabilistic polynomial time. Rather than using factorization methods to determine the factors of $k'l'$, it is possible to use the fact that l' is different in every execution of the CRT exponentiation due to the random blinding factor. An attacker can recover partial information on two different exponentiation executions with blinded d_q, i.e., $d'_{q,1}$ and $d'_{q,2}$, compute the products $k'l'_1$ and $k'l'_2$, and recover k' by computing the GCD (Greatest Common Divisor). It may be that l'_1 and l'_2 are not coprime, in which case the GCD will comprise $k' \cdot f$ where $f = \gcd(l'_1, l'_2)$. This happens with some probability, and can be solved by capturing additional exponentiation executions and successively computing the GCD with more products that include k'. Alternatively, the additional factor f is likely small (experimentally verified as shown in Figures 5-7) in practice and can be recovered using a small brute-force with only two instantiations. The attacker can guess f, derive the corresponding k', l'_1, and l'_2, and verify whether both pairs satisfy Equation (4).

Lemma 3 ($(k', l'_1, \ldots, l'_z)$ from MSB). *Let (N, e) be the public key and N be large enough with $e = N^\alpha$. Given the MSBs ($d_p'^{(M)}$, $d_{q,1}'^{(M)}$, …, $d_{q,z}'^{(M)}$), if the unknown parts ($d_p'^{(L)}$, $d_{q,1}'^{(L)}$, …, $d_{q,z}'^{(L)}$) are upper bounded by N^δ, where*

$$\gamma \leq \delta < \frac{1}{2} - 2\alpha - \gamma,$$

then $(k', l'_1, \ldots, l'_z)$ can be computed in time $\mathcal{O}(\log^2 N)$ with probability $1/\zeta(z)$ under Assumption 2.

Proof. In this case, to compute k', we use one observation $d_p'^{(M)}$ combined with z observations $d_{q,1}'^{(M)}, \ldots, d_{q,z}'^{(M)}$. To this end, we first calculate all the products $k'l'_1, \ldots, k'l'_z$, using the process described in the proof of Lemma 2 and then compute $k^* = \gcd(k'l'_1, \ldots, k'l'_z)$. According to Section 3.1, all the products $k'l'_i$ can be calculated in polynomial time $\mathcal{O}(\log^2 N)$ using the already known MSBs ($d_p'^{(M)}$, $d_{q,1}'^{(M)}$, $\ldots$, $d_{q,z}'^{(M)}$) and public key (N, e). Now, $k^* = k'$ if and only if $\gcd(l'_1, \ldots, l'_z) = 1$, which, according to Assumption 2 occurs with probability $1/\zeta(z)$. Upon obtaining k', it can be verified that this is correct by computing a corresponding l'_i and verify that they satisfy Equation (4). The complexity of the GCD computation is $\mathcal{O}((\log z) \cdot M((2\alpha+2\gamma)\log N)\log((2\alpha+2\gamma)\log N)$ considering the state-of-the-art quasi-linear time recursive algorithm [55, **Theorem 4**]. Putting all together, the time complexity of computing k' in the MSB case using multiple known MSBs is $\mathcal{O}(\log^2 N)$. ■

3.3 Step 1c: Computing (k', l') with known LSBs.

As mentioned above, in the LSB case, the approach to computing k' using Coppersmith's lattice-based method relies on the standard heuristic Assumption 1. In Section 4, we will verify the efficiency of this heuristic method in practice.

Lemma 4 ((k', l') from LSB). *Let (N, e) be the public key and N be large enough with $e = N^\alpha$. Given the LSBs ($d_p'^{(L)}$, $d_q'^{(L)}$), if the unknown parts ($d_p'^{(M)}$, $d_q'^{(M)}$) are upper bounded by N^δ, where*

$$\delta < \frac{1}{2} - 2\alpha - 2\gamma,$$

then (k', l') can be computed in polynomial time under Assumption 1.

Proof. Replacing d'_p, d'_q in Equation (3) with $d_p'^{(M)}2^i + d_p'^{(L)}$, $d_q'^{(M)}2^i + d_q'^{(L)}$ respectively, we get

$$k'l'N \equiv (k'-1)(l'-1) + ed_p'^{(L)}(l'-1) + ed_q'^{(L)}(k'-1) + e^2 d_p'^{(L)} d_q'^{(L)} \mod 2^i e,$$

Denote

$$A \equiv -e^2 d_p'^{(L)} d_q'^{(L)} + ed_p'^{(L)} + ed_q'^{(L)} - 1,$$

then we obtain

$$k'l'(N-1) - k'(ed_q'^{(L)} - 1) - l'(ed_p'^{(L)} - 1) + A \equiv 0 \mod 2^i e. \tag{5}$$

A polynomial is constructed using Equation (5) as below:

$$f(x, y) = (N-1)xy - (ed_q'^{(L)} - 1)x - (ed_p'^{(L)} - 1)y + A,$$

which has the root (k', l') modulo $2^i e$ and all the coefficients can be easily calculated. To compute k' and l' using Coppersmith's lattice-based method, similar to [43], we transform f into another polynomial g that has at least one *small* coefficient (i.e., it does not grow as function of N). More precisely, we construct g by replacing the coefficient of x by $\gcd(ed_q'^{(L)} - 1, 2^i)$ and multiplying all other coefficients of f with

$$\left(\frac{ed_q'^{(L)} - 1}{\gcd(ed_q'^{(L)} - 1, 2^i)}\right)^{-1} \mod 2^i e.$$

We know that k', l' are upper bounded by $N^\gamma e$, and it is also known that, under Assumption 1, all roots (x_0, y_0) of g modulo $2^i e$ that satisfy $|x_0|, |y_0| < N^\gamma e$ can be solved in polynomial time if

$$(N^\gamma e)^2 < (2^i e)^{\frac{2}{3}}. \tag{6}$$

Because $e = N^\alpha$ and $2^i = \Theta(N^{\frac{1}{2}+\gamma-\delta})$, the inequality 6 is actually asymptotically equivalent to

$$\delta < \frac{1}{2} - 2\alpha - 2\gamma,$$

which completes the proof. ■

3.4 Step 2: Factoring N with computed k'.

After computing k' as described in Sections 3.1 and 3.2 for the MSB case or in Section 3.3 for the LSB case, the second step is to factor N in polynomial time using the computed k' and the known part of d_p'. Please keep in mind that, in the MSB case when factoring is used, this step occasionally has to be performed twice because we do not know which of the two factors obtained in Step 1 corresponds to k' (as opposed to l'). Our Step 2 is similar to [43] and based on their **Theorem 3** as below:

Theorem 2 (May-Nowakowski-Sarkar [43]). *Suppose we are given a polynomial $f(x) = x + a$ and integers $k, N \in \mathbb{N}$, where $k = N^\mu$ for some $\mu \geq 0$. Let $p > N^\beta \in \mathbb{N}, \beta \in [0, 1]$ be an unknown divisor of N. In time polynomial in $\log N, \log k$ and $\log a$, we can compute all integers x_0, satisfying*

$$f(x_0) \equiv 0 \mod kp \quad \text{and} \quad |x_0| \leq N^{\beta^2+\mu}.$$

To factor N with the computed k' and the known parts of (d_p', d_q'), we have the following Lemma 5, which is a direct application of Theorem 2.

Lemma 5. *Let (N, e) be the public key and N be large enough with $e = N^\alpha$. Given the value k' and the MSBs $(d_p'^{(M)}, d_q'^{(M)})$ or the LSBs $(d_p'^{(L)}, d_q'^{(L)})$. If the*

unknown parts of (d'_p, d'_q) *are upper bounded by* N^δ *for* $\delta < \frac{1}{4} + \alpha + \gamma$*, then* N *can be factored in time polynomial in* $\log N, \log k'$ *and* $\log a$*, where*

$$a = (ed_p'^{(M)}2^i + k' - 1) \cdot (e^{-1} \mod k'N)$$

in the MSB case, or

$$a = \left(\frac{ed_p'^{(L)} + k' - 1}{\gcd(2^i e, k'N)}\right) \cdot \left(\left(\frac{2^i e}{\gcd(2^i e, k'N)}\right)^{-1} \mod \frac{k'N}{\gcd(2^i e, k'N)}\right)$$

in the LSB case.

Proof. In the MSB case, according to

$$ed'_p = 1 + k'(p-1) \quad \textit{and} \quad d'_p = d_p'^{(L)} + d_p'^{(M)}2^i$$

we have

$$ed_p'^{(L)} + ed_p'^{(M)}2^i + k' - 1 = k'p.$$

We construct a polynomial $f_{MSB}(x) = x + a \mod k'p$ with known coefficient

$$a = (ed_p'^{(M)}2^i + k' - 1) \cdot (e^{-1} \mod k'N)$$

and root $x_0 = d_p'^{(L)}$.

Since $k' = \Theta(N^{\alpha+\gamma})$ and $p = \Theta(\sqrt{N})$, it is concluded from [43, **Theorem 3**] that the unknown part $d_p'^{(L)}$ can be solved in polynomial time if $d_p'^{(L)} < N^{\frac{1}{4}+\alpha+\gamma}$. This condition is already fulfilled because $d_p'^{(L)} \leq N^\delta$. The last step to factor N is to get $p = \gcd(f_{MSB}(d_p'^{(L)}), N)$.

In the LSB case, similarly we construct a polynomial

$$f_{LSB}(x) = x + (ed_p'^{(L)} + k' - 1) \cdot ((2^i e)^{-1} \mod k'N),$$

however, it is slightly different from the MSB case that the modular multiplicative inverse $(2^i e)^{-1} \mod k'N$ does not always exist because $\gcd(2^i e, k'N) = 1$ does not always hold. To this end, we slightly change the polynomial to

$$f_{LSB}(x) = x + \left(\frac{ed_p'^{(L)} + k' - 1}{\gcd(2^i e, k'N)}\right) \cdot \left(\left(\frac{2^i e}{\gcd(2^i e, k'N)}\right)^{-1} \mod \frac{k'N}{\gcd(2^i e, k'N)}\right),$$

in this way, it is guaranteed that the modular multiplicative inverse $\left(\frac{2^i e}{\gcd(2^i e, k'N)}\right)^{-1} \mod \frac{k'N}{\gcd(2^i e, k'N)}$ exists. The rest is the same as the MSB case to factor N to get $p = \gcd(f_{LSB}(d_p'^{(M)}), N)$ and that finishes our proof. ■

It is worth noting that our EPKE also does not work if $e \geq N^{\frac{1}{4}-\gamma}$, unless factoring is easy, according to [43, **Corollary 1**].

Table 1. Summary of the EPKE experiments - MSB case with single $d_q'^{(M)}$

Len(N)	Len(e)	Len(r_p, r_q)	Len(UnknownLSB)	Step 1a Factoring time	Step 2 Lattice Dim.	LLL time
1024	64	32	336/**352**	<1 s	21	1s
1024	43	64	347/**362**	<1 s	21	1s
1024	17*	128	347/**401**	2s	21	2s
2048	149	32	665/**693**	42s	21	4s
2048	128	64	677/**704**	175s	21	4s
2048	85	128	697/**725**	340s	21	4s
3072	235	32	1008/**1034**	1787s	31	60s
3072	213	64	1014/**1045**	5993s	31	60s
3072	171	128	1032/**1066**	6651s	31	60s

4 Experimental results

As aforementioned, it is critical to experimentally verify the validity of Assumption 1 in the LSB case, as well as the validity of Assumption 2 and the sub-exponential time of factoring $k'l'$ in the MSB case. To assess the effectiveness of the EPKE attacks, we first conduct the EPKE attacks to recover the entire private exponent with $\frac{1}{3}$ known blinded MSBs or LSBs of d_p' and d_q', with different key and additive blinding factor lengths. To this end, we consider the commonly used key length of 1024-bit, 2048-bit and 3072-bit, the commonly used additive blinding factor length of 32-bit, 64-bit and 128-bit, and both MSB and LSB cases. Rather than executing the EPKE attacks on real or simulated side-channel leakage, known keys are generated and a fraction of the known bits of the blinded private exponents are used directly. Since our bounds depend on both the length of e and the length of the additive blinding factor, we choose the length of e based on the length of the additive blinding factor such that $r_p^{\frac{2}{3}} e \approx N^{\frac{1}{12}}$ in the MSB case and such that $r_p e \approx N^{\frac{1}{12}}$ in the LSB case. In these cases, we can use only $\frac{1}{3}$-part of known bits to recover the full key. More precisely, $\alpha = \frac{1}{12} - \frac{2}{3}\gamma$ according to our bounds $\min\{\frac{1}{4} + \alpha + \gamma, \frac{1}{2} - 2\alpha - \gamma\}$ for the MSB case, and $\alpha = \frac{1}{12} - \gamma$ according to our bounds $\min\{\frac{1}{4} + \alpha + \gamma, \frac{1}{2} - 2\alpha - 2\gamma\}$ for the LSB case. If $\gamma = 0$, i.e., without exponent blinding, our bounds are the same as [43] for both MSB and LSB cases, i.e., our bounds are the generalization of their work. The experimental EPKE attack results are summarized in Table 1 and Table 2 for the MSB case and Table 3 for the LSB case. The experiments are repeated 100 times for each setting by randomly generating a CRT key pair including e, so the time values in each table correspond to the average time over 100 experiments of each setting. We have implemented the experiments in SAGE 9.5 and YAFU [5] 2.08 factorization toolkit (Ubuntu 20.04.4) with an Intel® Core™ i5-7500 CPU 3.40GHz.

The fourth column shows two values, the ones in bold font correspond to the theoretical bounds, and the others correspond to the experimental results. In the MSB case, with a public exponent e of size $\alpha = \frac{1}{12} - \frac{2}{3}\gamma$, we nearly reach the asymptotic bound using a $\min\{\frac{1}{4} + \alpha + \gamma, \frac{1}{2} - 2\alpha - \gamma\}$-part MSBs of blinded CRT exponents (d_p', d_q') to recover the entire private key.

Table 2. Summary of the EPKE experiments - MSB case with multiple $d_q'^{(M)}$

Len(N)	Len(e)	Len(r_p, r_q)	Len(UnknownLSB)	Step 1b Success Prob.
1024	64	32	336/**352**	0.67/0.89/0.93/0.97/0.98/0.99/0.98/0.98/1.00
1024	43	64	347/**362**	0.65/0.89/0.91/0.98/0.97/0.99/0.99/1.00/0.99
1024	17*	128	347/**401**	0.65/0.78/0.94/0.99/0.96/0.99/0.99/1.00/1.00
2048	149	32	665/**693**	0.67/0.79/0.89/0.95/0.99/1.00/0.98/0.99/1.00
2048	128	64	677/**704**	0.73/0.86/0.92/0.93/0.98/1.00/1.00/1.00/1.00
2048	85	128	697/**725**	0.73/0.86/0.92/0.95/0.99/1.00/0.99/1.00/1.00
3072	235	32	1008/**1034**	0.69/0.81/0.93/0.98/0.98/0.99/1.00/1.00/1.00
3072	213	64	1014/**1045**	0.72/0.82/0.94/0.98/0.98/0.99/1.00/1.00/1.00
3072	171	128	1032/**1066**	0.67/0.81/0.93/1.00/0.99/1.00/1.00/1.00/1.00

Table 3. Summary of the EPKE experiments - LSB case

Len(N)	Len(e)	Len(r_p, r_q)	Len(UnknownMSB)	Step 1c Lattice Dim.	LLL time	Step 2 Lattice Dim.	LLL time
1024	53	32	320/**341**	121	216s	21	<1 s
1024	21	64	320/**341**	121	202s	21	<1 s
1024	17*	128	191/**222**	121	461s	21	4s
2048	139	32	648/**682**	121	487s	21	4s
2048	107	64	647/**682**	121	470s	21	4s
2048	43	128	649/**682**	121	419s	21	4s
3072	224	32	978/**1024**	121	1104s	31	90s
3072	192	64	978/**1024**	121	1110s	31	90s
3072	128	128	976/**1024**	121	991s	31	87s

In the LSB case, with a public exponent e of size $\alpha = \frac{1}{12} - \gamma$, we succeeded in computing $(k',\ l')$ in each performed experiment, affirming the validity of Assumption 1. We closely reach the asymptotically bound using a $\min\{\frac{1}{4} + \alpha + \gamma, \frac{1}{2} - 2\alpha - 2\gamma\}$-part LSBs of blinded CRT exponents $(d_p',\ d_q')$ to recover the entire private key. We need more known bits to reveal the key compared with the MSB case.

It has to be mentioned that the fourth row (1024-bit key with 128-bit blinding factor) in both MSB and LSB cases is different from the other settings. Because the chosen e should be 1 (resp., $N^{-\frac{1}{24}}$) for the MSB (resp., LSB) case if we want to use $\frac{1}{3}$ known MSBs/LSBs of blinded CRT exponents $(d_p',\ d_q')$ to recover the full private key. However, those two e values are not realistic, instead, we choose a very widely-used (e.g., $2^{16} + 1$) size of e, i.e., 17-bit, which complies with the NIST's recommendation of e and is also close to those two values. Our experiments suggest that in the MSB case the required known bits are close to the optimum value, i.e., $\frac{1}{3}$ MSBs of blinded exponents. While in the LSB case we need more than $\frac{2}{3}$ LSBs of blinded exponents, because the used e value is too far away from the expected value $e = N^{-\frac{1}{24}}$.

Another critical point to be verified is the required sub-exponential factoring time to compute k' in Step 1a (see Section 3.1), as shown in Figures 1-3, clearly it can be observed that the required factoring time for all the settings is certainly affordable even with an average PC. Mostly, the factoring finishes within seconds or minutes, but on very few occasions it requires between one hour and one and a half hours (resp., fourteen hours and fifty hours) for 2048-bit (resp., 3072-

bit) key with 128-bit blinding. It is worth noting that the 1024-bit key with the 32-bit blinding factor case requires slightly more time than the 1024-bit key with the 64-bit blinding factor case, because the former case uses SAGE's internal factorization function while all other cases utilize YAFU's factorization functions.

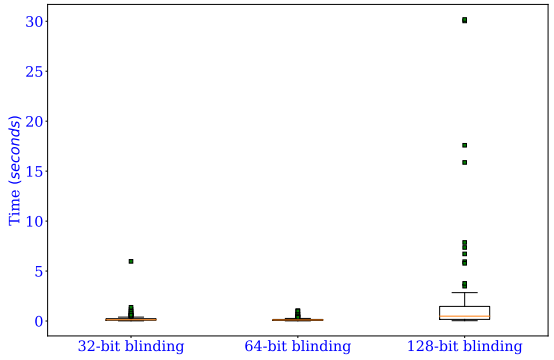

Fig. 1. Step 1a Factoring Time for 1024-bit key.

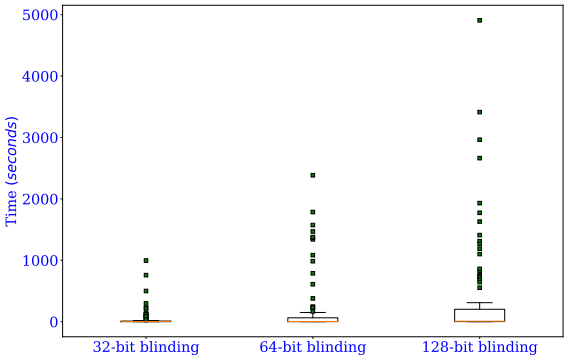

Fig. 2. Step 1a Factoring Time for 2048-bit key.

Finally, the success probability of the probabilistic method to obtain k' in Step 1b (see Section 3.2) is shown in Figure 4. For all parameter sets in Tables 1-3, 100 RSA key pairs were generated, and for each key pair, 1,000 sets of values

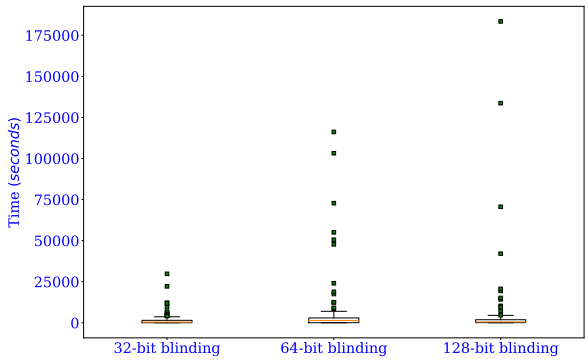

Fig. 3. Step 1a Factoring Time for 3072-bit key.

$(l'_1, \ldots, l'_{10})$ were generated based on blinded exponents. The number of values required to obtain a GCD of 1 was computed for each set, in order to determine the estimated success probability of the attack depending on z. Our experimental results in Table 2 also validate the estimated probability here as part of the full attack path, as the last column indicates the successful EPKE attack probability with $2 \sim 10$ different randomly generated l'_z values for each setting of key size and blinding factor length. As can be observed, the successful EPKE attack probability is already above 0.65 using only two l'_z values, and it reaches up to almost 1 using five l'_z values. Therefore, we experimentally verify the validity of Assumption 2. In addition, as mentioned above, the f factor can be recovered using only two l'_z values with a small brute-force because it is expected to be small in practice. We also empirically confirm that the f factor is indeed pretty small as shown in Figures 5-7. Mostly it is smaller than 100 for different key sizes and exponent blinding factor lengths, the brute-force effort is affordable to compute k' with only two l'_z values to achieve the successful EPKE attack probability of 1.

A somewhat surprising observation is, from a PKE attack perspective, comparing to without additive exponent blinding, CRT-RSA with additive exponent blinding of size N^γ and public exponents $e \leq N^{\frac{1}{12}-\gamma}$ becomes easier to attack with larger exponent blinding factors. For instance, CRT-RSA with 2048-bit N and widely used 17-bit public exponent e. Without exponent blinding, the PKE attack can efficiently break the scheme with roughly 48% (492 bits) of the LSBs of the CRT exponents. On the contrary, given a 153-bit exponent blinding, then roughly only 33% (389 bits) are required. However, from an SCA attack point of view, it is other way around. The additive exponent blinding limits an attacker to use only a single attack trace to get the partial key information instead of using multiple attack traces without exponent blinding.

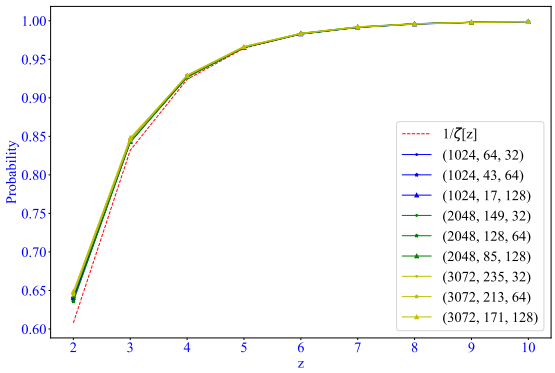

Fig. 4. Step 1b Estimated versus assumed probability that $l'_1, \ldots, l'_z$ are coprime.

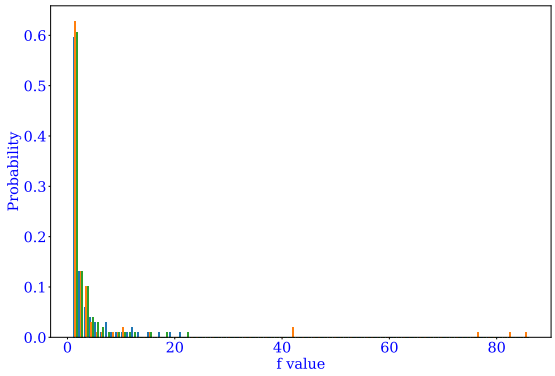

Fig. 5. Step 1b Histogram of f values with two l'_z values for 1024-bit key.

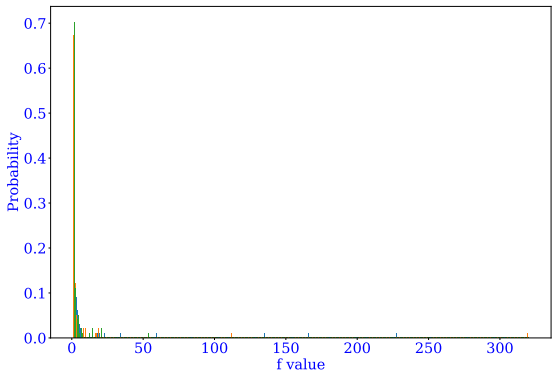

Fig. 6. Step 1b Histogram of f values with two l'_z values for 2048-bit key.

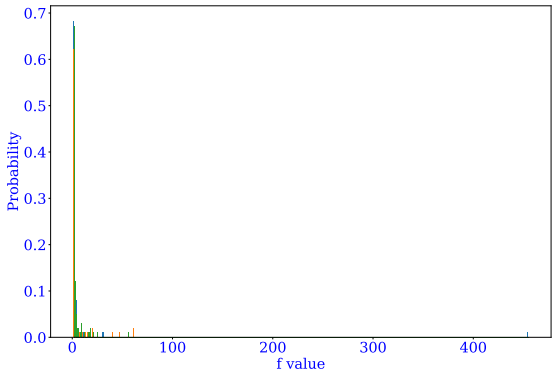

Fig. 7. Step 1b Histogram of f values with two l'_z values for 3072-bit key.

In both cases, we verified the effectiveness of the proposed EPKE attacks using partially known blinded CRT exponents to disclose the entire private key. Next, we will demonstrate an application of this EPKE attack using the obtained partial side-channel key leakage via profiled attacks in a realistic context.

5 A use case of EPKE on real SCA partial key leakage

In the following, we first introduce our profiled attacks-based experimental verification methodology concerning the EPKE attacks for an attacker/evaluator to disclose the full private key based on partially recovered CRT exponents. It includes the metrics used for training neural networks and the knowledge of points of interest (POIs) assumption. In the end, we present the experimental results rendering this verification methodology.

5.1 Deep learning profiled attack

Since the seminal work of Kocher [35], side-channel analysis (SCA) has been a powerful and de facto tool to evaluate the physical security of various cryptographic implementations, especially on embedded devices. There is a rich body of literature about SCA on RSA implementations. Some of these works [13, 15, 48, 60, 64] concentrated on RSA implementations with additive blinding. In particular, Carbone et al. [13] conducted profiled attacks, which is considered the most powerful SCA, on an RSA implementation with all aforementioned blinding countermeasures on a CC EAL4+ certified IC. Zaid et al. [64] improved the profiled attack by introducing a new ensembling loss function.

Since the introduction of template attacks [14], the SCA community has considered profiled attacks as the most powerful side-channel attacks. Profiled attacks have two stages, i.e., the profiling stage and the attack stage. In the profiling stage, an attacker/evaluator uses a profiling device (and has control of the key or at least knows the key) to model the leakage characteristic of the target key-dependent sensitive data (exponent bits in this work) with the side-channel traces of the target implementation. The built leakage characteristic models for every possible target sensitive data value are the outcome of the profiling stage. In the attack (also called online) stage, the victim device is used to measure the side-channel traces of the target implementation. Afterwards, the attack traces are matched with the previously built leakage characteristic models of the target sensitive data (exponent bits in this work). For each attack trace, a probability is computed for each possible guessed key unit value (2 possible values for one exponent bit) per each trace. Finally, for each hypothesised key unit value, the probabilities for all the attack traces are combined using, e.g., the maximum likelihood method to get a combined probability of that specific guessed key unit value. The combined probability of each possible hypothesised key unit value is compared to find the highest one. The hypothesised key unit value with the highest probability is considered the recovered key unit. This attack process is repeated for all key units to reveal the complete key. In this work, the

combination of probabilities is skipped because only single-trace attacks are in scope due to the exponent blinding countermeasure.

We use deep learning profiled attacks (DL) to experimentally investigate the efficiency of our proposed extended PKE attack in the additive exponent blinding scenario. In this regard, the deep learning model is used to extract the required partial MSBs or LSBs of blinded CRT exponents as the input to our extended PKE attack to reveal the entire private key. We use a published MLP (Multi-Layer Perceptron) [39] neural network model for our DL profiled attacks. The structure of this model is very simple and shallow, but it showed pretty good performance. We will describe the parameters-setting in Section 5.6.

Unlike the classical profiled attacks (e.g., template attacks), DL makes no assumption of the leakage characteristic. It exploits the features (sample points for side-channel traces) to classify the labels (sensitive data in the SCA context) using neural networks (details followed in Subsection 5.6). The training process of neural networks (i.e., the profiling) aims to construct a classifier function $F(.) : \mathbb{R}^d \rightarrow \mathbb{R}^{|S|}$. The input trace $\mathbf{l} \in R^d$ is mapped to the output vector $\mathbf{p} \in R^{|S|}$ of scores via this function. During the training, the backpropagation method [25,34] is used for each training batch to update the trainable parameters of the neural network model aiming at minimizing the loss, which is computed to quantize the classification error over each training batch. Then in the attack stage, the built trained model (i.e., $F(.)$ with all the final updated trainable parameters) is used to classify each attack trace to obtain its probability vector $\mathbf{p}[s_g]$. Afterwards, the final probability vector $\mathbf{p}[g]$ of each key candidate g is calculated using all the attack traces. Note that, in this work only a single attack trace is used to decide the final probability of each attacked exponent bit. The key candidate $g^* = \text{argmax } \mathbf{p}[g]$ is considered the right one.

5.2 Knowledge of POIs assumption

There is an implicit assumption about the knowledge of the POIs to apply the profiled attacks to CRT exponentiation implementations. That is, the attackers/evaluators can determine the rough timing interval of each exponent bit calculation in the side-channel traces. It is feasible for most of security products in a grey-box testing context via SPA (Simple Power Analysis)/SEMA (Simple Electromagnetic Analysis) and CPA (Correlation Power Analysis)/CEMA (Correlation Electromagnetic Analysis) (or similar techniques) as shown in [13, 21]. For instance, one can vary the key length, perform correlation analyses on the input and output data, make use of the design information such as the location of the RSA co-processor in the glue logic area or temporarily switch off some side-channel countermeasures like jitters. The additive exponent blinding countermeasure will not make this harder, because it only adds more patterns corresponding to exponent bit calculations in the side-channel traces. If the patterns of one exponent bit calculation without exponent blinding can be identified in a side-channel trace, it is the same to identify the exponent bit calculation patterns given the presence of an additive exponent blinding countermeasure.

5.3 Metrics

We use the most common machine learning metric accuracy [25] to monitor and evaluate our deep learning profiled attacks. Its definition is the successful classification rate obtained over a dataset. Accordingly, the training accuracy, the validation accuracy and the test accuracy correspond to the reached successful classification rates respectively over the training, the validation and the test sets. The training and validation accuracy metrics are used to monitor the performance of the neural network training, and we use the test accuracy to evaluate the trained model. The accuracy is suitable for our experiments because we focus on the successful classification rate of each exponent bit, and we use a balanced dataset (the number of profiling traces of each class is the same) to avoid the potential deceptive impact of the accuracy metric. Concerning the metrics used to train the neural networks, we use the Negative Log-Likelihood (NLL) loss function [12] for DL profiled attacks. It is a loss function calculated as $-\log y$ used in multi-class classification, where y is a prediction corresponding to the ground-truth label after the softmax [25] activation function is applied. The loss for a mini-batch is computed by taking the mean or sum of all items in the batch. Because it is proved that minimizing the NLL loss is equivalent to maximizing the Perceived Information [10, 49] and thus minimizing the online attack complexity, thanks to the recent work [41].

5.4 Montgomery Ladder Exponentiation implementation

Our target is a Montgomery Ladder exponentiation CRT-RSA implementation on a modern 45 nm secure microcontroller equipped with an RSA co-processor running at 100 MHz. This is a typical real-world target from a side-channel attack viewpoint due to the implemented side-channel countermeasures, that is, SPA/DPA-resistant atomic Montgomery Ladder exponentiation with additive message and exponent blinding and multiplicative modulus blinding. Moreover, the 32-bit CPU also has variable internal clock, random branch insertion, memory encryption and physical address scrambling countermeasures to enhance the side-channel resistance.

Algorithm 1 illustrates the implemented left-to-right Montgomery Ladder exponentiation [33]. It is a well-known and widely used SPA-resistant regular exponentiation algorithm without using dummy operations to defeat SPA/SEMA and safe-error attacks [63]. We view the n-bit private exponent as a binary vector $\mathbf{d} = (d_{n-1}, ..., d_0)$ (where d_0 is the LSB). N is the modulus used for the exponentiation, and C is the message to be decrypted using the private exponent.

5.5 EPKE attack on SCA partial key leakage verification strategy

To represent a realistic scenario of EPKE attack on CRT using SCA partial key leakage, we follow the verification strategy as below:

1. Determine the Montgomery Ladder exponentiation interval using the SPA technique as discussed in Section 5.2.

Algorithm 1 Montgomery Ladder.

Require: $C, N, \mathbf{d} = (d_{n-1}, ..., d_0)$
Ensure: $M = C^d \mod N$
1: $R_0 \leftarrow 1$
2: $R_1 \leftarrow C$
3: **for** $i = n-1$ *downto* 0 **do**
4: $\quad R_{\neg d_i} \leftarrow R_{d_i} \times R_{\neg d_i} \mod N$
5: $\quad R_{d_i} \leftarrow R_{d_i} \times R_{d_i} \mod N$
6: **end for**
7: **return** R_0

2. Measure a set of 5,000 profiling traces focusing on the exponentiation interval using a CRT key pair labelled as K_{Prof0}.
3. Randomly generate ten different CRT key pairs with 128-bit e values (considering a 64-bit additive blinding factor used by the target CRT implementation) to measure the attack traces for the MSB case, denote those ten attack key pairs as $K_{MSB1}, K_{MSB2}, ..., K_{MSB10}$. Similarly, for the LSB case, randomly generate another ten different CRT key pairs $K_{LSB1}, K_{LSB2}, ...,$ K_{LSB10} with 107-bit e values to acquire the corresponding attack traces.
4. Measure ten attack traces for each of those 20 attack keys.
5. Train the neural network model using the profiling traces to save the best model with the highest test accuracy.
6. Perform the profiled attack on all the attack traces using the saved best neural network model to recover the MSBs or LSBs of CRT exponents. More precisely, recover 411 ($= 1024+64-677$, according to the sixth row in Table 1) MSBs of d'_p and d'_q for each attack trace with $K_{MSB1}, K_{MSB2}, ..., K_{MSB10}$. Similarly, recover 441 ($= 1024+64-647$, according to the sixth row in Table 3) LSBs of d'_p and d'_q for each attack trace with $K_{LSB1}, K_{LSB2}, ..., K_{LSB10}$.
7. Conduct the EPKE attack using the recovered 411 MSBs of d'_p and d'_q to disclose the full private key for $K_{MSB1}, K_{MSB2}, ..., K_{MSB10}$ and verify the required sub-exponential time. To this end, we benefit from the previously mentioned combination of the recovered MSBs of d'_p from one attack trace and the recovered MSBs of d'_q from another attack trace. For each MSB attack key K_{MSBi}, we have 100 different combinations, using those combinations to tolerate potential SCA errors. We do the same for the LSB attack keys. In this case we need to verify the validity of the Coppersmith-type heuristic assumption.

We acquired power consumption traces with a Lecroy WaveRunner 8254 oscilloscope at a sampling rate of 500 MS/s. For each exponentiation execution, we triggered the oscilloscope at its end and recorded the processing of the entire exponentiation. Each trace consists of 16,000,000 sample points. Each exponent bit processing corresponds to two Montgomery modular multiplications as shown in Algorithm 1, and it contains about 6,074 sample points illustrated in Figure 8.

While we recorded the entire exponentiation for this experiment, it is worth mentioning that an extra advantage of our EPKE attack is that the recovered MSBs or LSBs of d'_p and the MSBs or LSBs of d'_q can come from different observations of the partial key leakage. An attacker can, at his own convenience, capture and combine the error-free MSBs or LSBs of d'_p from one observation of the partial key leakage, and those of d'_q from another one, while the remaining parts can be arbitrarily erroneous or not captured at all (i.e., only capture one-third of the execution of one of the two CRT exponentiations).

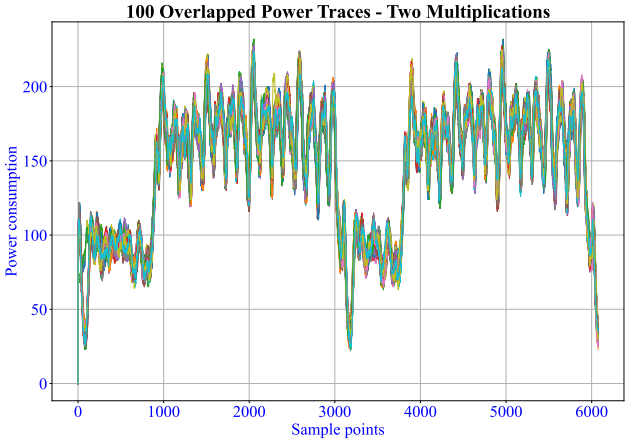

Fig. 8. 100 Overlapped Power Traces of One Exponent Bit Processing.

5.6 Attack results.

We have implemented the DL profiled attacks in Python and PyTorch [46] version 1.10.1 with an NVIDIA GTX 1080Ti GPU. We use the *Adadelta* optimizer [65] and utilize the adaptive learning rate policy *ReduceLROnPlateau* to gradually decrease the learning rate (with the default Adadelta optimizer initial learning rate of 1.0) with a factor of 0.05 if the training stagnates. We use a batch size of 512 and 100 as the number of epochs. All profiling and attack traces are normalized using the *StandardScalar* function from the Scikit-learn [47] library by removing the mean and scaling to unit variance. 20% of profiling traces compose a validation set. A validation data set is crucial to DL performance because it provides a way of instantly detecting over-fitting [26]. The DL attacks utilize the trained model with the highest validation accuracy. As mentioned above, we use the NLL loss for the model training. Table 4 summarizes the details of the used DL model and the corresponding hyperparameters.

Table 4. MLP model details

MLP
nb_epoch = 100 batch_size_training = 512
Dense(50, activation="relu", input_shape=(nb_samples,)) BatchNormalization() Dense(100, activation="relu") BatchNormalization() Dense(2, activation="softmax") compile(loss='categorical-crossentropy', optimizer='adadelta', metrics=['accuracy'])
learning_rate_policy = ReduceLROnPlateau(optimizer, 'min', factor=0.05, verbose=True)

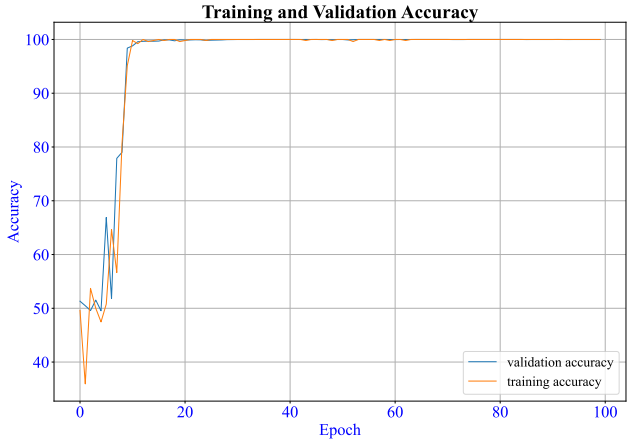

Fig. 9. Training and Validation Accuracy.

We first train the model using the profiling traces to save the best model based on the highest validation accuracy depicted in Figure 9, in which the best model corresponds to a validation accuracy of 100%.

The saved best model is then used to retrieve the 411 (resp., 441) MSBs (resp., LSBs) of d'_p and d'_q for all the 10 MSB-case (resp., LSB-case) attack traces with attack key K_{MSBi} (resp., K_{LSBi}). For each attack key, there are 100 combinations of recovered MSBs (resp., LSBs) of d'_p from one attack trace and recovered MSBs (resp., LSBs) of d'_q from another attack trace. We conduct the EPKE attacks using those combinations to recover the prime factor p as described in Section 3.4. The second column of Tables 5 and 6 presents the number of successful recovery of the prime factor out of 100 combinations for each attack key. The last three (resp., four) columns indicate the time cost and lattice dimensions of the two-step EPKE attack on the MSB (resp., LSB) cases. In the third column of Table 5, we present the average factoring time to compute k' and the corresponding minimum and maximum time (in the brackets).

Table 5. Summary of the real SCA data EPKE experiments - MSB case

Attack Key	Success Nr.	Step 1a Factoring time	Step 2 Lattice Dim.	LLL time
K_{MSB1}	26	110s ([0.93s, 599.7s])	21	6s
K_{MSB2}	35	90s ([0.5s, 1379.7s])	21	6s
K_{MSB3}	15	176s ([1.0s, 1817.7s])	21	6s
K_{MSB4}	3	10s ([6.1s, 16.4s])	21	6s
K_{MSB5}	9	202s ([1.5s, 1211.8s])	21	6s
K_{MSB6}	4	251s ([59.7s, 477.9s])	21	6s
K_{MSB7}	4	65s ([2.1s, 235.7s])	21	6s
K_{MSB8}	4	212s ([1.6s, 775.2s])	21	6s
K_{MSB9}	4	969s ([764.4s, 1373.7s])	21	6s
K_{MSB10}	12	21s ([2.6s, 60.7s])	21	6s

Table 6. Summary of the real SCA data EPKE experiments - LSB case

Attack Key	Success Nr.	Step 1c Lattice Dim.	LLL time	Step 2 Lattice Dim.	LLL time
K_{LSB1}	6	121	388s	21	2s
K_{LSB2}	2	121	404s	21	2s
K_{LSB3}	4	121	407s	21	3s
K_{LSB4}	3	121	421s	21	3s
K_{LSB5}	8	121	410s	21	3s
K_{LSB6}	5	121	407s	21	3s
K_{LSB7}	9	121	399s	21	3s
K_{LSB8}	12	121	405s	21	3s
K_{LSB9}	16	121	442s	21	3s
K_{LSB10}	9	121	433s	21	3s

The results further confirm the effectiveness of our EPKE attack in a realistic context, that is, disclosing a fraction of blinded CRT exponents via SCA followed by an EPKE attack to reveal the entire private key. In addition, the SCA experiments show that the proposed EPKE attack can tolerate slight SCA errors as demonstrated by our proposed verification strategy, i.e., combining the recovered error-free partial MSBs or LSBs of d'_p from one trace and MSBs or LSBs of d'_q from another one. When considering real-world CRT implementations, SCA attacks will often result in errors, which is why it is essential for an attacker/evaluator to be able to apply (E)PKE attacks.

6 Conclusion and Future Work

Many existing PKE works focused on implementations of RSA and its CRT variant without exponent blinding countermeasures. The state-of-the-art PKE attack on CRT without exponent blinding [43] can recover the whole CRT private key with only a third MSBs or LSBs of CRT exponents when the public exponent $e \approx N^{\frac{1}{12}}$. In this work, we showed that it can be extended to CRT implementations with additive exponent blinding, which is the most widely deployed exponent blinding countermeasure in real-world products. We proposed an extended PKE attack to recover the full CRT private key using partial disclosed MSBs or LSBs of additively blinded CRT exponents. It follows a two-step

approach, first computing the key-dependent constant k' and then factoring the N using partial MSBs or LSBs of blinded CRT exponents d'_p and d'_q.

The mathematical proof and time-complexity analyses suggest that a third MSBs or LSBs of blinded CRT exponents d'_p and d'_q are enough to recover the entire CRT private key: in polynomial time based on the standard Coppersmith-type heuristic assumption in the LSB case when $e \approx N^{\frac{1}{12}-\gamma}$, as well as in sub-exponential (yet practically feasible) time or in probabilistic polynomial time under a heuristic assumption in the MSB case when $e \approx N^{\frac{1}{12}-\frac{2}{3}\gamma}$. Under different settings of typical key size and exponent blinding factor length, our extensive experiments verify the validity of the Coppersmith-type heuristic for the LSB case, as well as the affordability of the required sub-exponential time and the heuristic assumption for the MSB case. More precisely, in practice, using an average PC, the required sub-exponential time is mostly in seconds and minutes, the worst-case occasionally observed is less than one and a half hours.

Moreover, as an application of the proposed EPKE attack in real life, utilizing real SCA partial key leakage of a real-world SPA/DPA-resistant Montgomery Ladder CRT implementation on a 45 nm secure microcontroller with an RSA co-processor, our SCA experimental results suggest that the EPKE attack can tolerate slight SCA errors of the recovered MSBs or LSBs of the blinded CRT exponents. Such errors are expected to occur in a realistic scenario, especially when only a single attack trace is available. It sheds some light on the error-tolerant potential of the EPKE attack, and points out a possible future direction of our work, i.e., how to improve the EPKE to tolerate more generic SCA errors of the recovered MSBs or LSBs, e.g., in the binary symmetric model where every recovered exponent bit will be "flipped" with a crossover probability (i.e., the Bernoulli distribution).

Acknowledgements We would like to express our gratitude to the anonymous reviewers for their insightful comments. François-Xavier Standaert is a senior research associate of the Belgian Fund for Scientific Research (F.R.S.-FNRS). This work has been funded in parts by the ERC project SWORD (Grant Number 724725). Yu Yu was supported by the National Key Research and Development Program of China (Grant Nos. 2020YFA0309705 and 2018YFA0704701) and the National Natural Science Foundation of China (Grant Nos. 62125204 and 61872236). Yu Yu also acknowledges the support from the XPLORER PRIZE.

References

1. Albrecht, M.R., Heninger, N.: On bounded distance decoding with predicate: Breaking the "lattice barrier" for the hidden number problem. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 528–558. Springer, Heidelberg, Germany, Zagreb, Croatia (Oct 17–21, 2021). https://doi.org/10.1007/978-3-030-77870-5_19
2. Aldaya, A.C., García, C.P., Tapia, L.M.A., Brumley, B.B.: Cache-timing attacks on RSA key generation. IACR TCHES **2019**(4), 213–242 (2019). https://doi.

org/10.13154/tches.v2019.i4.213-242, https://tches.iacr.org/index.php/TCHES/article/view/8350

3. Aono, Y.: A new lattice construction for partial key exposure attack for RSA. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 34–53. Springer, Heidelberg, Germany, Irvine, CA, USA (Mar 18–20, 2009). https://doi.org/10.1007/978-3-642-00468-1_3
4. Bauer, S.: Attacking exponent blinding in RSA without CRT. In: Schindler, W., Huss, S.A. (eds.) COSADE 2012. LNCS, vol. 7275, pp. 82–88. Springer, Heidelberg, Germany, Darmstadt, Germany (May 3–4, 2012). https://doi.org/10.1007/978-3-642-29912-4_7
5. bbuhrow: YAFU, Automated integer factorization Version: 2.0.8 (2022), https://github.com/bbuhrow/yafu, available at https://github.com/bbuhrow/yafu
6. Bellare, M., Halevi, S., Sahai, A., Vadhan, S.P.: Many-to-one trapdoor functions and their relation to public-key cryptosystems. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 283–298. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 23–27, 1998). https://doi.org/10.1007/BFb0055735
7. Blömer, J., May, A.: New partial key exposure attacks on RSA. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 27–43. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 2003). https://doi.org/10.1007/978-3-540-45146-4_2
8. Bos, J., Stam, M. (eds.): Computational Cryptography: Algorithmic Aspects of Cryptology. London Mathematical Society Lecture Note Series, Cambridge University Press (2021). https://doi.org/10.1017/9781108854207
9. Botan: Botan, a Crypto and TLS for Modern C++ library, Version: 2.19.1 (2022), https://github.com/randombit/botan, available at https://github.com/randombit/botan/blob/master/src/lib/pubkey/rsa/rsa.cpp
10. Bronchain, O., Hendrickx, J.M., Massart, C., Olshevsky, A., Standaert, F.X.: Leakage certification revisited: Bounding model errors in side-channel security evaluations. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 713–737. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 18–22, 2019). https://doi.org/10.1007/978-3-030-26948-7_25
11. BSI: BSI AIS46 RSA SCA Resistance Guideline, Minimum Requirements for Evaluating Side-Channel Attack Resistance of RSA, DSA and Diffie-Hellman Key Exchange Implementations Version: 1.0 (2013), https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_46_BSI_guidelines_SCA_RSA_V1_0_e_pdf.pdf?__blob=publicationFile&v=1, available at https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_46_BSI_guidelines_SCA_RSA_V1_0_e_pdf.pdf?__blob=publicationFile&v=1
12. Cagli, E., Dumas, C., Prouff, E.: Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without preprocessing. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 45–68. Springer, Heidelberg, Germany, Taipei, Taiwan (Sep 25–28, 2017). https://doi.org/10.1007/978-3-319-66787-4_3
13. Carbone, M., Conin, V., Cornélie, M.A., Dassance, F., Dufresne, G., Dumas, C., Prouff, E., Venelli, A.: Deep learning to evaluate secure RSA implementations. IACR TCHES **2019**(2), 132–161 (2019). https://doi.org/10.13154/tches.v2019.i2.132-161, https://tches.iacr.org/index.php/TCHES/article/view/7388

14. Chari, S., Rao, J.R., Rohatgi, P.: Template attacks. In: Kaliski Jr., B.S., Koç, Çetin Kaya., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 13–28. Springer, Heidelberg, Germany, Redwood Shores, CA, USA (Aug 13–15, 2003). https://doi.org/10.1007/3-540-36400-5_3
15. Clavier, C., Feix, B., Gagnerot, G., Roussellet, M., Verneuil, V.: Horizontal correlation analysis on exponentiation. In: Soriano, M., Qing, S., López, J. (eds.) ICICS 10. LNCS, vol. 6476, pp. 46–61. Springer, Heidelberg, Germany, Barcelona, Spain (Dec 15–17, 2010)
16. Coppersmith, D.: Finding a small root of a bivariate integer equation; factoring with high bits known. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 178–189. Springer, Heidelberg, Germany, Saragossa, Spain (May 12–16, 1996). https://doi.org/10.1007/3-540-68339-9_16
17. Coppersmith, D.: Finding a small root of a univariate modular equation. In: Maurer, U.M. (ed.) EUROCRYPT'96. LNCS, vol. 1070, pp. 155–165. Springer, Heidelberg, Germany, Saragossa, Spain (May 12–16, 1996). https://doi.org/10.1007/3-540-68339-9_14
18. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology **10**(4), 233–260 (Sep 1997). https://doi.org/10.1007/s001459900030
19. Coron, J.S.: Resistance against differential power analysis for elliptic curve cryptosystems. In: Koç, Çetin Kaya., Paar, C. (eds.) CHES'99. LNCS, vol. 1717, pp. 292–302. Springer, Heidelberg, Germany, Worcester, Massachusetts, USA (Aug 12–13, 1999). https://doi.org/10.1007/3-540-48059-5_25
20. Diaconis, P., Erdös, P.: On the distribution of the greatest common divisor. In: A festschrift for Herman Rubin, pp. 56–61. Institute of Mathematical Statistics (2004)
21. Diop, I., Linge, Y., Ordas, T., Liardet, P.Y., Maurine, P.: From theory to practice: horizontal attacks on protected implementations of modular exponentiations. Journal of Cryptographic Engineering **9**(1), 37–52 (Apr 2019). https://doi.org/10.1007/s13389-018-0181-1
22. Dugardin, M., Schindler, W., Guilley, S.: Stochastic methods defeat regular RSA exponentiation algorithms with combined blinding methods. J. Math. Cryptol. **15**(1), 408–433 (2021). https://doi.org/10.1515/jmc-2020-0010, https://doi.org/10.1515/jmc-2020-0010
23. Ernst, M., Jochemsz, E., May, A., de Weger, B.: Partial key exposure attacks on RSA up to full size exponents. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 371–386. Springer, Heidelberg, Germany, Aarhus, Denmark (May 22–26, 2005). https://doi.org/10.1007/11426639_22
24. Fouque, P.A., Kunz-Jacques, S., Martinet, G., Muller, F., Valette, F.: Power attack on small RSA public exponent. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 339–353. Springer, Heidelberg, Germany, Yokohama, Japan (Oct 10–13, 2006). https://doi.org/10.1007/11894063_27
25. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), http://www.deeplearningbook.org
26. Guyon, I.: A scaling law for the validation-set training-set size ratio. In: AT & T Bell Laboratories (1997)
27. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys. In: van Oorschot, P.C. (ed.) USENIX Security 2008. pp. 45–60. USENIX Association, San Jose, CA, USA (Jul 28 – Aug 1, 2008)

28. Heninger, N., Shacham, H.: Reconstructing RSA private keys from random key bits. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 1–17. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2009). https://doi.org/10.1007/978-3-642-03356-8_1
29. Hlavác, M.: Known-plaintext-only attack on RSA-CRT with Montgomery multiplication. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 128–140. Springer, Heidelberg, Germany, Lausanne, Switzerland (Sep 6–9, 2009). https://doi.org/10.1007/978-3-642-04138-9_10
30. Howgrave-Graham, N.: Finding small roots of univariate modular equations revisited. In: Darnell, M. (ed.) 6th IMA International Conference on Cryptography and Coding. LNCS, vol. 1355, pp. 131–142. Springer, Heidelberg, Germany, Cirencester, UK (Dec 17–19, 1997)
31. Howgrave-Graham, N.: Approximate integer common divisors. In: Silverman, J.H. (ed.) Cryptography and Lattices, International Conference, CaLC 2001, Providence, RI, USA, March 29-30, 2001, Revised Papers. Lecture Notes in Computer Science, vol. 2146, pp. 51–66. Springer (2001). https://doi.org/10.1007/3-540-44670-2_6, https://doi.org/10.1007/3-540-44670-2_6
32. Joye, M., Lepoint, T.: Partial key exposure on RSA with private exponents larger than N. In: Ryan, M.D., Smyth, B., Wang, G. (eds.) Information Security Practice and Experience - 8th International Conference, ISPEC 2012, Hangzhou, China, April 9-12, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7232, pp. 369–380. Springer (2012). https://doi.org/10.1007/978-3-642-29101-2_25, https://doi.org/10.1007/978-3-642-29101-2_25
33. Joye, M., Yen, S.M.: The Montgomery powering ladder. In: Kaliski Jr., B.S., Koç, Çetin Kaya., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 291–302. Springer, Heidelberg, Germany, Redwood Shores, CA, USA (Aug 13–15, 2003). https://doi.org/10.1007/3-540-36400-5_22
34. Kelley, H.J.: Gradient theory of optimal flight paths. Ars Journal **30**(10), 947–954 (1960)
35. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) CRYPTO'96. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 18–22, 1996). https://doi.org/10.1007/3-540-68697-5_9
36. Lenstra, A., Lenstra, H., Lovász, L.: Factoring polynomials with rational coefficients. Math. Ann. **261**, 515–534 (1982)
37. Libgcrypt: Libgcrypt, the gnu crypto library, Version: 1.9.0 (2021), https://github.com/gpg/libgcrypt, available at https://github.com/gpg/libgcrypt/blob/master/cipher/rsa.c
38. Lu, Y., Zhang, R., Lin, D.: New partial key exposure attacks on CRT-RSA with large public exponents. In: Boureanu, I., Owesarski, P., Vaudenay, S. (eds.) ACNS 14. LNCS, vol. 8479, pp. 151–162. Springer, Heidelberg, Germany, Lausanne, Switzerland (Jun 10–13, 2014). https://doi.org/10.1007/978-3-319-07536-5_10
39. Maghrebi, H.: Deep learning based side-channel attack: a new profiling methodology based on multi-label classification. Cryptology ePrint Archive, Report 2020/436 (2020), https://eprint.iacr.org/2020/436
40. Mangard, S., Oswald, E., Popp, T.: Power analysis attacks - revealing the secrets of smart cards. Springer (2007)
41. Masure, L., Dumas, C., Prouff, E.: A comprehensive study of deep learning for side-channel analysis. IACR TCHES **2020**(1), 348–375 (2019). https://doi.

org/10.13154/tches.v2020.i1.348-375, https://tches.iacr.org/index.php/TCHES/article/view/8402
42. May, A., Nowakowski, J., Sarkar, S.: Partial key exposure attack on short secret exponent CRT-RSA. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13090, pp. 99–129. Springer (2021). https://doi.org/10.1007/978-3-030-92062-3_4, https://doi.org/10.1007/978-3-030-92062-3_4
43. May, A., Nowakowski, J., Sarkar, S.: Approximate divisor multiples - factoring with only a third of the secret crt-exponents. In: Dunkelman, O., Dziembowski, S. (eds.) Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13277, pp. 147–167. Springer (2022). https://doi.org/10.1007/978-3-031-07082-2_6, https://doi.org/10.1007/978-3-031-07082-2_6
44. MbedTLS: MbedTLS, a TLS and SSL library, Version: 3.1.0 (2021), https://github.com/Mbed-TLS/mbedtls, available at https://github.com/Mbed-TLS/mbedtls/blob/development/library/rsa.c
45. NIST: FIPS PUB 186-4, Digital Signature Standard (DSS) (FIPS 186-4) (2013), https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf, available at https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf
46. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. pp. 8024–8035 (2019), https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html
47. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
48. Perin, G., Chmielewski, L.: A semi-parametric approach for side-channel attacks on protected RSA implementations. In: Homma, N., Medwed, M. (eds.) Smart Card Research and Advanced Applications - 14th International Conference, CARDIS 2015, Bochum, Germany, November 4-6, 2015. Revised Selected Papers. Lecture Notes in Computer Science, vol. 9514, pp. 34–53. Springer (2015). https://doi.org/10.1007/978-3-319-31271-2_3, https://doi.org/10.1007/978-3-319-31271-2_3
49. Renauld, M., Standaert, F.X., Veyrat-Charvillon, N., Kamel, D., Flandre, D.: A formal study of power variability issues and side-channel attacks for nanoscale devices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 109–128. Springer, Heidelberg, Germany, Tallinn, Estonia (May 15–19, 2011). https://doi.org/10.1007/978-3-642-20465-4_8
50. Sarkar, S., Venkateswarlu, A.: Partial key exposure attack on CRT-RSA. In: Meier, W., Mukhopadhyay, D. (eds.) INDOCRYPT 2014. LNCS, vol. 8885, pp. 255–264.

Springer, Heidelberg, Germany, New Delhi, India (Dec 14–17, 2014). https://doi.org/10.1007/978-3-319-13039-2_15
51. Schindler, W.: A timing attack against RSA with the Chinese remainder theorem. In: Koç, Çetin Kaya., Paar, C. (eds.) CHES 2000. LNCS, vol. 1965, pp. 109–124. Springer, Heidelberg, Germany, Worcester, Massachusetts, USA (Aug 17–18, 2000). https://doi.org/10.1007/3-540-44499-8_8
52. Schindler, W., Itoh, K.: Exponent blinding does not always lift (partial) spa resistance to higher-level security. In: Lopez, J., Tsudik, G. (eds.) ACNS 11. LNCS, vol. 6715, pp. 73–90. Springer, Heidelberg, Germany, Nerja, Spain (Jun 7–10, 2011). https://doi.org/10.1007/978-3-642-21554-4_5
53. Schindler, W., Wiemers, A.: Power attacks in the presence of exponent blinding. Journal of Cryptographic Engineering **4**(4), 213–236 (Nov 2014). https://doi.org/10.1007/s13389-014-0081-y
54. Schindler, W., Wiemers, A.: Generic power attacks on RSA with CRT and exponent blinding: new results. Journal of Cryptographic Engineering **7**(4), 255–272 (Nov 2017). https://doi.org/10.1007/s13389-016-0146-1
55. Stehlé, D., Zimmermann, P.: A binary recursive gcd algorithm. In: Buell, D.A. (ed.) Algorithmic Number Theory, 6th International Symposium, ANTS-VI, Burlington, VT, USA, June 13-18, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3076, pp. 411–425. Springer (2004). https://doi.org/10.1007/978-3-540-24847-7_31, https://doi.org/10.1007/978-3-540-24847-7_31
56. Takayasu, A., Kunihiro, N.: Partial key exposure attacks on RSA: Achieving the boneh-durfee bound. In: Joux, A., Youssef, A.M. (eds.) SAC 2014. LNCS, vol. 8781, pp. 345–362. Springer, Heidelberg, Germany, Montreal, QC, Canada (Aug 14–15, 2014). https://doi.org/10.1007/978-3-319-13051-4_21
57. Takayasu, A., Kunihiro, N.: Partial key exposure attacks on CRT-RSA: Better cryptanalysis to full size encryption exponents. In: Malkin, T., Kolesnikov, V., Lewko, A.B., Polychronakis, M. (eds.) ACNS 15. LNCS, vol. 9092, pp. 518–537. Springer, Heidelberg, Germany, New York, NY, USA (Jun 2–5, 2015). https://doi.org/10.1007/978-3-319-28166-7_25
58. Takayasu, A., Kunihiro, N.: Partial key exposure attacks on RSA: achieving the boneh-durfee bound. Theor. Comput. Sci. **761**, 51–77 (2019). https://doi.org/10.1016/j.tcs.2018.08.021, https://doi.org/10.1016/j.tcs.2018.08.021
59. Takayasu, A., Lu, Y., Peng, L.: Small crt-exponent RSA revisited. J. Cryptol. **32**(4), 1337–1382 (2019). https://doi.org/10.1007/s00145-018-9282-3, https://doi.org/10.1007/s00145-018-9282-3
60. Walter, C.D.: Sliding windows succumbs to big mac attack. In: Koç, Çetin Kaya., Naccache, D., Paar, C. (eds.) CHES 2001. LNCS, vol. 2162, pp. 286–299. Springer, Heidelberg, Germany, Paris, France (May 14–16, 2001). https://doi.org/10.1007/3-540-44709-1_24
61. Wang, T., Cui, X., Ni, Y., Yu, D., Cui, X., Qu, G.: A practical cold boot attack on RSA private keys. In: 2017 Asian Hardware Oriented Security and Trust Symposium, AsianHOST 2017, Beijing, China, October 19-20, 2017. pp. 55–60. IEEE Computer Society (2017). https://doi.org/10.1109/AsianHOST.2017.8353995, https://doi.org/10.1109/AsianHOST.2017.8353995
62. Yarom, Y., Genkin, D., Heninger, N.: CacheBleed: A timing attack on OpenSSL constant time RSA. In: Gierlichs, B., Poschmann, A.Y. (eds.) CHES 2016. LNCS, vol. 9813, pp. 346–367. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–19, 2016). https://doi.org/10.1007/978-3-662-53140-2_17

63. Yen, S., Joye, M.: Checking before output may not be enough against fault-based cryptanalysis. IEEE Trans. Computers **49**(9), 967–970 (2000). https://doi.org/10.1109/12.869328, https://doi.org/10.1109/12.869328
64. Zaid, G., Bossuet, L., Habrard, A., Venelli, A.: Efficiency through diversity in ensemble models applied to side-channel attacks. IACR TCHES **2021**(3), 60–96 (2021). https://doi.org/10.46586/tches.v2021.i3.60-96, https://tches.iacr.org/index.php/TCHES/article/view/8968
65. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. CoRR **abs/1212.5701** (2012), http://arxiv.org/abs/1212.5701