Making RSA–PSS Provably Secure Against Non-Random Faults

Gilles Barthe[1], François Dupressoir[1], Pierre-Alain Fouque[2], Benjamin Grégoire[4], Mehdi Tibouchi[3], and Jean-Christophe Zapalowicz[4]

[1] IMDEA Software Institute
gilles.barthe@imdea.org,francois.dupressoir@imdea.org
[2] Université de Rennes 1 and Institut universitaire de France
pierre-alain.fouque@ens.fr
[3] NTT Secure Platform Laboratories
tibouchi.mehdi@lab.ntt.co.jp
[4] INRIA
benjamin.gregoire@inria.fr,jean-christophe.zapalowicz@inria.fr

Abstract. RSA–CRT is the most widely used implementation for RSA signatures. However, deterministic and many probabilistic RSA signatures based on CRT are vulnerable to fault attacks. Nevertheless, Coron and Mandal (Asiacrypt 2009) show that the randomized PSS padding protects RSA signatures against *random faults*. In contrast, Fouque et al. (CHES 2012) show that PSS padding does not protect against certain *non-random faults* that can be injected in widely used implementations based on the Montgomery modular multiplication. In this article, we prove the security of an infective countermeasure against a large class of non-random faults; the proof extends Coron and Mandal's result to a strong model where the adversary can force the faulty signatures to be a multiple of one of the prime factors of the RSA modulus. Such non-random faults induce more complex probability distributions than in the original proof, which we analyze using careful estimates of exponential sums attached to suitable rational functions. The security proof is formally verified using appropriate extensions of EasyCrypt, and provides the first application of formal verification to provable (i.e. reductionist) security in the context of fault attacks.

Keywords: Fault Attacks, PSS, RSA–CRT, Infective countermeasure, Formal Verification, EasyCrypt

1 Introduction

Signature schemes are among the most widely used constructions in cryptography. Although there is much interest in signature schemes based on elliptic curves, RSA signatures are still widely used. Moreover, many implementations of RSA, including OpenSSL and implementations for embedded devices such as smartcards, use the well-known Chinese Remainder Theorem (CRT) technique for computing modular exponentiations more efficiently: exponentiations using the CRT can be expected to be 4 times faster than those using full-size exponents. However, when unprotected, RSA–CRT is vulnerable to the so-called Bellcore attack, first introduced by Boneh, DeMillo and Lipton [8], and later refined [3,31]. An adversary who knows the padded message and can inject a fault in one of the half exponentiations can efficiently factor the public modulus using a single faulty signature and a GCD computation.

Many countermeasures have been proposed to mitigate this vulnerability, including extra computations and sanity checks of intermediate and final results (see [26]). The simplest such protection is to verify the signature before releasing it. This is reasonably cheap since the public exponent e is usually small. Another approach is to use an extended modulus, as in Shamir's trick [27] and its later refinements which also protect CRT recombination using Garner's formula [7,13,29,14]. Finally, redundant exponentiation algorithms [19,26] such as the Montgomery Ladder can be used. Regardless of the approach, RSA–CRT fault countermeasures tend to be rather costly: for

example, Rivain's countermeasure [26,20] has a stated overhead of 10% compared to an unprotected implementation, and is purportedly more efficient than previous works [19,29,20].

Boneh et al.'s original fault attack does not apply to RSA signatures with probabilistic encoding functions, but some extensions of it were proposed to attack randomized ad-hoc padding schemes such as ISO 9796-2 and EMV [15,17]. At Asiacrypt 2009, Coron and Mandal [16] paved the way of provable security against side-channel attack in a practical setting by proving that RSA–PSS is secure against *random* faults in the random oracle model. Injecting a fault on the half-exponentiation modulo the second factor q of N produces a result that can be modeled as uniformly distributed modulo q, and the result of such a fault cannot be used to break RSA–PSS signatures. It is tempting to conclude that using RSA–PSS should enable signers to dispense with costly RSA–CRT countermeasures. However, Fouque et al. [18] show that it is possible to break RSA–PSS using certain *non-random* faults if the result is not checked. Indeed, they obtain a key recovery attack with a few faulty signatures on CRT implementations of RSA–PSS that use the state-of-the-art modular multiplication algorithm of Montgomery [23]. Thus, even with PSS, it remains important to check the signature before releasing it.

Infective Countermeasures. Checking results before release is a simple and practical security measure, but it is not sufficient by itself, since simple tests can be easily bypassed by flipping the outcome of a comparison [2,28]. Infective countermeasures are an alternate approach in which results are released all the time, but become gibberish when faulty computations occur: a fault (usually not controlled by the adversary) results in a random value, which consequently makes the faulty signature random. From a security point of view, since faults may not be random, we may not be able to prove that the faulty output is fully random. However, one may ask that the output be independent of secret information even in the presence of non-random faults. Infective countermeasures have been used before by Canetti and Goldwasser [11] to deal with fault-injecting adversaries when decrypting ciphertexts in a distributed manner. One such countermeasure for RSA–CRT was proposed by Boscher, Handschuh and Trichina [9]. In their technique, the signer computes the signature S and recomputes $y' = S^e \bmod N$ to check the signature against the padded message y, before returning $S + y'_p - (y \bmod p) + y'_q - (y \bmod q)$ if $y' = y$, and an error otherwise. Even if the adversary bypasses the verification $y' = y$, the output signature mixes the fault and correct signature in a non-trivial way. Still, this countermeasure was later attacked by Trichina and Korkikyan [28] for deterministic padding schemes. We tackle the problem of masking faulty signatures so as to prevent the exploitation of faults and protect validity checks.

Our contributions. In this paper we generalize the fault model from [18] and consider a very powerful adversary able to inject *non-random* faults. In particular, our adversaries are able to force the signature to be a multiple of one of the prime factors. We consider a simple countermeasure (see Fig. 1) that uses infective techniques, mixing additional randomness into faulty signatures in a *provably secure way*. In practice, we show that our random infection masks faulty signatures enough for us to prove the security of RSA–PSS under the RSA assumption in the random oracle model if enough additional randomness is provided. Concretely, we sample a random value r' and add $r' \cdot (y - y')$ to the signature mod N, where y is the original padded message and y' is the padded message recovered from the signature. When the signature is computed correctly, $(y - y')$ is zero and the correct signature is returned. If the signature is faulty, we show that the masked output is statistically close to uniform and hence leaks no secret information. We prove such results in two key lemmas corresponding to [16, Lemmas 1, 2]. Since our faults are non-random, the probability distributions are more complex; we use careful estimates of exponential sums attached to

corresponding rational functions to establish their regularity. We only analyze this countermeasure when the validity check is performed in the standard way (by computing the public permutation), but our random infection might also be used to protect other checks such as Rivain's [26,20]. In the same way, although we use RSA–CRT as a motivating example, our fault model is in fact independent of the way the modular exponentiation is implemented, and is not limited to fault attacks on RSA–CRT. A discussion of the faults we model can be found in Section 3.

Figure 1 Protected signing algorithm

1: **function** SIGN(sk, pk, m) ▷ $sk = (d_p, d_q, a_p, a_q, N)$, $pk = (e_p, e_q, b_p, b_q, N)$
2: $r \leftarrow \{0,1\}^{k_0}$ ▷ Start of PSS padding
3: $\omega \leftarrow \mathcal{H}(m, r)$
4: $st \leftarrow \mathcal{G}(\omega) \oplus (r \,||\, 0^{k_g - k_0})$
5: $y \leftarrow \mathsf{os2ip}(0 \,||\, \omega \,||\, st)$
6: $\sigma_p \leftarrow y^{d_p} \bmod p$ ▷ Signature computation
7: $\sigma_q \leftarrow y^{d_q} \bmod q$
8: $\sigma \leftarrow (a_p \cdot \sigma_p + a_q \cdot \sigma_q) \bmod N$
9: $y'_p \leftarrow \sigma^{e_p} \bmod p$ ▷ Internal recomputation of the padded message
10: $y'_q \leftarrow \sigma^{e_q} \bmod q$
11: $y' \leftarrow (b_p \cdot y'_p + b_q \cdot y'_q) \bmod N$
12: $r' \leftarrow \{0,1\}^{\rho} \backslash \{0\}$ ▷ Infective countermeasure
13: $\sigma' \leftarrow \sigma + r' \cdot (y - y') \bmod N$
14: **return** $\mathsf{i2osp}(\sigma')$

The second contribution of the paper is a formal proof of security of the countermeasure using EasyCrypt [4], a computer-aided framework that has previously been used to reason about the security of cryptographic constructions—but was never applied to fault attacks and countermeasures. Our proof is the first application of formal verification to provable security against fault attacks, as other works [12,24,25] applying formal verification to fault attacks are focused on proving the correctness of the countermeasures (that is, that the protected program either returns the same result as the original program, or fails), but do not provide any provable security guarantees. Apart from increasing our confidence in the effectiveness of the countermeasure, our formal proof reveals a glitch in the proof of Coron and Mandal [16], and also paves the way for formally verifying the effectiveness of the countermeasures on standard implementations of PKCS probabilistic signing, in the same way that [1] uses EasyCrypt to prove security of an implementation of PKCS encryption.

2 Related work

Christofi *et al* [12] use program verification techniques for proving Vigilant's countermeasure for CRT-RSA. Their approach is based on a transformation that takes a source program p under scrutiny and outputs a program $\hat{p}$ whose code embeds all possible faults. Informally, the program $\hat{p}$ contains for each program point where a fault can be forced a switch statement that models the faults allowed at that program point; the case analysis of the switch statement triggers which fault will be forced, and hence is performed on an additional argument of the program. By proving that the transformed program is correct for all possible inputs, one thus obtains a guarantee that the program is correct for all faults. In their work, Christofi *et al* take as correctness criterion that the transformed program either returns a value that matches the value returned by the original

program on the same input, or else returns an error. Intuitively, their criterion entails that the program obtained by transformation either outputs the correct result, or detects that there is a fault during execution and aborts. While it is a natural guarantee to seek, their theorem does not constitute a proof of security in the sense of provable security, but rather a heuristic to validate a countermeasure. Using their approach, Christofi *et al* validate with the Frama-C platform the correctness (in the sense mentioned above) of Vigilant's countermeasure for CT-RSA.

Rauzy and Guilley [25] develop symbolic methods to analyze fault attacks against RSA-CRT implementations. Broadly speaking, they formalize arithmetic computations as algebraic expressions, and develop an axiomatic theory for (dis)proving equalities between two expressions. Their theory includes all the equations from the theory of rings, equations for the modulo operator, and some specific results, such as Euler's theorem and the Chinese Remainder Theorem. The central component of their tool is a procedure that takes an expression and returns a simplified form. Given an expression e that represents an arithmetic computation, their tool computes the set of expressions that represent faulty variants of e and tests for all expressions $\hat{e}$ in this set if the expression $gcd(N, e - \hat{e})$ simplifies to p or q, where N is the RSA modulus and p and q are its prime factors. If some expression $\hat{e}$ is found, then the algorithm is considered insecure; on the contrary, the algorithm is considered secure if no expression e that passes the test is found. Their tool is useful to find fault attacks on an algorithm, or as a heuristics to gain some confidence in an algorithm, but only provides guarantees of security against a very restricted class of attackers. Moreover, it is specialized to deterministic signature schemes and cannot deal with randomized paddings like PSS.

Moro et al [24] focus on the specific class of instruction skip attacks, in which an adversary forces to skip the execution of a targetted instruction. These attacks apply to several architectures, and can be realized for instance by introducing glitches into the clock signal of the microcontroller. The approach pursued by Moro et al is to make program execution fault-tolerant using redundancy. In order to achieve acceptable performance, their redundancy scheme is designed to account for some inner characteristics of fault attacks; for instance, their model reflects the hardness of forcing several faults within a small number of clock cycles. Concretely, they focus on the Thumb-2 instruction set that is available on ARM architectures, and provide for each instruction a possible replacement for execution in the presence of instruction skip faults. Replacements are sequences of instructions which are given a faulty semantics; Moro et al use a model-checker to establish the equivalence between execution of the instruction without faults and executing the replacement sequence of instructions with instruction skip faults. The strength of their approach lies in its generality: namely, the transformation can be applied to arbitrary programs, and significantly improve resistance against instruction skip attacks. However, their approach is not suitable to obtain the strong guarantees required by provable security.

3 Our results

Instead of considering the many possible faults an adversary could inject in Fig. 1, we give the adversary access to two distinct oracles (Fig. 2) that compute valid signatures and generalize faulty signatures, as justified in Section 3. As discussed, our fault model is independent of the algorithm used to compute modular exponentiation. We therefore simplify notations by using more standard definitions for public and secret key, where a public key pk is composed of a public exponent e and a modulus N, and a secret key sk is composed of a private exponent d and a modulus N.

Throughout the security proof, we consider a fixed k that serves as the size of the modulus and signatures. In particular, we assume that the modulus is balanced, that is $N = p \cdot q$ is such that

$2^{k-1} \leq N < 2^k$ and $2^{k/2-1} \leq p < q < 2^{k/2}$. We also assume that public exponents produced by the key generation algorithm are upper bounded by some constant e_{max} much smaller than 2^k (in practice, $2^{16}+1$ is often used). PSS padding is computed using two hash functions $\mathcal{H}$, outputting bitstrings of length k_h, and $\mathcal{G}$, producing bitstrings of length k_g, where $k_h + k_g + 1 = k$. In addition, the padding scheme uses a random salt of length $k_0 < k_g$. For simplicity, we model $\mathcal{H}$ as a function from $\{0,1\}^* \times \{0,1\}^{k_0}$ to $\{0,1\}^{k_h}$, and $\mathcal{G}$ as a function from $\{0,1\}^{k_h}$ to $\{0,1\}^{k_g}$. This is done without loss of generality. In algorithm and game descriptions, we denote with i2osp and os2ip the conversions between integers and their binary representations. For simplicity, i2osp always produces a bitstring of length k.

Figure 2 Oracles in our fault model

oracle $\mathcal{S}(m)$
- $r \leftarrow \{0,1\}^{k_0}$
- $\omega \leftarrow \mathcal{H}(m, r)$
- $st \leftarrow \mathcal{G}(\omega) \oplus (r \,||\, 0^{k_g - k_0})$
- $y \leftarrow \mathsf{os2ip}(0 \,||\, \omega \,||\, st)$
- $\sigma \leftarrow y^d \bmod N$
- **return** $\mathsf{i2osp}(\sigma)$

oracle $\mathcal{V}(m, \sigma)$
- $s \leftarrow \mathsf{os2ip}(\sigma)$
- **if** $0 \leq s < N$ **then**
  - $y \leftarrow s^e \bmod N$
  - $b \,||\, \omega \,||\, st \leftarrow \mathsf{i2osp}(y)$
  - $r \,||\, \gamma \leftarrow st \oplus \mathcal{G}(\omega)$
  - $\omega' \leftarrow \mathcal{H}(m, r)$
  - **return** $b = 0 \wedge \omega = \omega' \wedge \gamma = 0^{k_g - k_0}$
- **else**
  - **return** $\perp$

oracle $\mathcal{F}(m, \epsilon, a)$
- $r \leftarrow \{0,1\}^{k_0}$
- $\omega \leftarrow \mathcal{H}(m, r)$
- $st \leftarrow \mathcal{G}(\omega) \oplus (r \,||\, 0^{k_g - k_0})$
- $y \leftarrow \mathsf{os2ip}(0 \,||\, \omega \,||\, st)$
- $\sigma \leftarrow y^d \bmod N$
- $r' \leftarrow \{0,1\}^{\rho} \backslash \{0\}$
- $\sigma' = \epsilon \cdot \sigma + a \cdot p + r' \cdot y \bmod N$
- **return** $\mathsf{i2osp}(\sigma')$

We reduce the $\mathcal{UF}$-$\mathcal{CMA}$ security of the faulty signature scheme presented in Fig. 2, when the adversary is given access to the faulty signature oracle, along with the valid signature oracle and the random oracles $\mathcal{H}$ and $\mathcal{G}$, to the one-way security of RSA. We consider a forgery valid even if it was produced by the faulty signature oracle. In the rest of this paper, we use $\mathcal{S}$ to denote the valid signature oracle, $\mathcal{F}$ to denote the faulty signature oracle, $\mathcal{K}$ to denote the RSA key generation algorithm, and $\mathcal{V}$ for the PSS verification algorithm. Subscripts identify the game in which a particular oracle appears. We denote with $\mathcal{Q}^{\mathcal{X}}$ the set of query-response pairs for queries made to oracle $\mathcal{X}$ so far.

Figure 3 Initial and Final Games

game $\mathcal{UF}$-$\mathcal{CMA}$
- $(e, d, N) \leftarrow \mathcal{K}()$
- $(m, s) \leftarrow \mathcal{A}^{\mathcal{S}, \mathcal{F}, \mathcal{H}, \mathcal{G}}(e, N)$
- $b \leftarrow \mathcal{V}(m, s)$
- $win \leftarrow b \wedge (m, s) \notin \mathcal{Q}^{\mathcal{S}}$
- **return** win

game $\mathcal{OW}$-$\mathcal{RSA}$
- $(e, d, N) \leftarrow \mathcal{K}()$
- $x^* \leftarrow [0..N)$
- $y^* \leftarrow x^{*e} \bmod N$
- $x \leftarrow \mathcal{I}^{\mathcal{H}, \mathcal{G}}(e, N, y^*)$
- **return** $x = x^*$

Verification \ Signature	No	$\sigma_p = 0$	$\sigma_q = 0$	$\sigma = 0$
No	σ	$ap + r'y \bmod N$	$aq + r'y \bmod N$	$r'y \bmod N$
$y'_p = 0$	$\sigma + ap + r'y \bmod N$	$ap + r'y \bmod N$	$aq + r'y \bmod N$	$r'y \bmod N$
$y'_q = 0$	$\sigma + aq + r'y \bmod N$	$ap + r'y \bmod N$	$aq + r'y \bmod N$	$r'y \bmod N$
$y' = 0$	$\sigma + r'y \bmod N$	$ap + r'y \bmod N$	$aq + r'y \bmod N$	$r'y \bmod N$

Table 1. Output of $\mathcal{F}$ under various faults with a a random value.

Theorem 1 ($\mathcal{UF}$-$\mathcal{CMA}$ security of protected PSS in the presence of faults). *Given a CMA adversary $\mathcal{A}$ against the faulty signature scheme $(\mathcal{K}, \mathcal{S}, \mathcal{F}, \mathcal{V})$ that makes at most $q_{\mathcal{H}}$ queries to $\mathcal{H}$, $q_{\mathcal{G}}$ queries to $\mathcal{G}$, $q_{\mathcal{S}}$ queries to $\mathcal{S}$ and $q_{\mathcal{F}}$ queries to $\mathcal{F}$, we build a one-way inverter $\mathcal{I}$ such that*

$$\Pr[\mathcal{UF}\text{-}\mathcal{CMA} : win] \leq \Pr[\mathcal{OW}\text{-}\mathcal{RSA} : x = x^*] + \epsilon_0$$

with

$$\epsilon_0 = \frac{(q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot (q_{\mathcal{H}} + q_{\mathcal{G}} + q_{\mathcal{S}} + q_{\mathcal{F}}) + (q_{\mathcal{G}} + 1) \cdot q_{\mathcal{F}} \cdot 3 + 1}{2^{k_h}} + \frac{(q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot (q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}}) + q_{\mathcal{F}} \cdot (q_{\mathcal{H}} + 1) + q_{\mathcal{H}} + q_{\mathcal{S}}}{2^{k_0}} + \frac{q_{\mathcal{H}} + 1}{2^{\frac{k}{2}-2}} + q_{\mathcal{F}} \cdot \frac{(4e_{max}^2 + 1) \cdot k^2}{2^{\rho - k/2}}$$

Remark 1. This theorem allows us to conclude directly with a security claim for PSS when e_{max} is reasonably small (typically $2^{16} + 1$), $\rho \geq k/2 + 200$, and the modulus is not too large (see Remark 2).

Fault model justification. In this section, we justify our fault model, described by oracle $\mathcal{F}$ in Fig. 2. Our faulty signature oracle computes the correct padded message y, samples r' and returns $\sigma' = \epsilon \cdot y^d + a \cdot p + r' \cdot y \bmod N$, with $a \in \mathbb{Z}/N\mathbb{Z}$ and $\epsilon \in \{0, 1\}$ chosen by the adversary. We allow multiple faults to be injected, but only during the RSA–CRT computations (lines 6-8 and 9-11 of the protected signing Fig. 1). In other words, we do not consider faults during the computation of the padding (lines 2-5) or while the countermeasure is being applied (lines 12-13). This assumption can be justified by the fact that RSA–CRT computations are usually the privileged target of fault attacks since the secret exponent as well as the secret primes are involved. Sometimes, faults can be injected on public parameters [10,6], however such faults modify computation with secret values, σ_p and σ_q during CRT recombinaison and so this is treated in our model. In this model, we do not consider safe errors which consider specific implementations.

Our fault model can be viewed as a generalization of the various faults presented by Fouque et al. [18]: the Null faults (forcing a small register to 0), the Constant faults (forcing a small register to a constant) and the Zero High-Order Bits faults (forcing part of a small register to 0). When applied during the RSA–CRT computations to a precise small register, these faults may allow the adversary to factor the RSA modulus. They apply to any padding scheme, including randomized padding schemes such as PSS. More precisely the purpose of these faults is to transform a signature into a multiple or close multiple of one of the modulus's prime factors. In the context of the proof, we wish to give our attacker as much power as possible. Thus we let him set the faulty signature to 0 or to a multiple of one of the prime factors. We also let him apply these faults to the public exponentiation used to check the validity of the signature. We let the adversary inject these faults independently of the particular algorithm or implementation of modular exponentiation.

Table 1 describes the form of the signature under various null faults. As discussed, we let the adversary fault the private exponentiation σ and its CRT components σ_p and σ_q, as well as the public exponentiation used for checking y' and its CRT components y'_p and y'_q. Note that modeling a as a uniform random integer module N would already let us capture all possible faults that yield a multiple of p. By letting the adversary choose the value of a, we are in fact able to capture more faults, some of which may be interesting and practical. Considering the symmetry between p and q, all results in this table are of the form $\epsilon \cdot y^d + a \cdot p + r' \cdot y \bmod N$.

4 Statistical Lemmas

We need several results on the regularity of the probability distributions related to the infective countermeasure. Recall that the *statistical distance* between a random variable X on a finite set S and the uniform distribution is defined as:

$$\Delta_1(X) = \frac{1}{2} \cdot \sum_{s \in S} \left| \Pr[X = s] - \frac{1}{|S|} \right|.$$

We say that X is δ-*statistically close to uniform* when $\Delta_1(X) \leq \delta$.

Our proof of Lemma 3 relies on exponential sums in $\mathbb{Z}/m\mathbb{Z}$, so we first fix some notations and recall useful standard results. For any integer m, we denote by $\mathbf{e}_m$ the additive character $\mathbb{Z}/m\mathbb{Z} \to \mathbb{C}^*$ given by $\mathbf{e}_m(x) = \exp(2i\pi x/m)$. The following results hold.

Proposition 1 (Orthogonality). *For all $x \in \mathbb{Z}/m\mathbb{Z}$, we have:*

$$\sum_{c=0}^{m-1} \mathbf{e}_m(cx) = \begin{cases} 0 & \text{if } c \not\equiv 0 \pmod m, \\ m & \text{if } c \equiv 0 \pmod m. \end{cases}$$

Lemma 1 ([30, Problem 11.c]). *For any modulus[5] $m \geq 60$ and any non negative integers h, k, we have:*

$$\sum_{c=1}^{m-1} \left| \sum_{x=k}^{k+h} \mathbf{e}_m(cx) \right| \leq (m-1) \log m.$$

Lemma 2 (Weil [21]). *Consider a prime modulus p. For all polynomials $g(X), h(X) \in \mathbb{F}_p[X]$ such that the rational function $f(X) = h(X)/g(X)$ is not constant on $\mathbb{F}_p$, the bound:*

$$\left| \sum_{\substack{x \in \mathbb{F}_p \\ g(x) \neq 0}} \mathbf{e}_p\big(f(x)\big) \right| \leq \big(\max(\deg g, \deg h) + v - 1\big) \cdot p^{1/2}$$

holds, where v is the number of distinct zeros of $g(X)$ in the algebraic closure of $\mathbb{F}_p$.

We now discuss our key statistical lemmas. The first one ensures that the faulty signature $\sigma' = \epsilon\sigma + ap - r'y$ is indistinguishable from a uniform random if the nonce r' is large enough. We write x instead of r' in the rest of this section.

[5] We will assume that this bound on m holds for all moduli involved in our computations below, i.e. $p, q > 60$, which is of course satisfied for all RSA moduli in practice.

Lemma 3. *Let $N = pq$ be a k-bit balanced RSA modulus (i.e. $N = pq$ for primes p, q such that $60 < q < p < 2q$) and e the public exponent, $0 \leq y < 2^{k-1}$ a random integer and x a random nonzero ρ-bit integer. Let $a \in \mathbb{Z}/N\mathbb{Z}$ and $\epsilon \in \{0, 1\}$ two fixed values. The statistical distance between the distribution of $\sigma' = \epsilon y^d + ap + xy \bmod N$ and the uniform distribution modulo N is bounded as:*

$$\Delta_1(\sigma') \leq \frac{(4e^2+1)N^{1/2}(\log^2 N)}{2^\rho - 1}.$$

Proof. The full proof is provided in Appendix A; more precisely, this result follows directly from Lemma 7 with $\mathcal{X} = [1, 2^\rho)$ and $\mathcal{Y} = [0, 2^{k-1})$. For the reader's convenience, we provide a short proof sketch below.

The idea of the proof is to express the statistical distance as an exponential sum over $\mathbb{Z}/N\mathbb{Z}$. Indeed, let $s = \sigma' - ap$. Using the orthogonality property of additive characters (Prop. 1), the probability that s can be written as $\epsilon y^d + xy \bmod N$, with $x \in \mathcal{X}$ and $y \in \mathcal{Y}^* = \mathcal{Y} \cap (\mathbb{Z}/N\mathbb{Z})^*$ is:

$$\Pr\left[s = \epsilon y^d + xy\right] = \frac{1}{\#\mathcal{X}\#\mathcal{Y}^*} \sum_{(x,y)\in\mathcal{X}\times\mathcal{Y}^*} \frac{1}{N} \sum_{c=0}^{N-1} \mathbf{e}_N\left(c\Big(x - \frac{s - \epsilon y^d}{y}\Big)\right)$$

since the equation is equivalent to $x = (s - \epsilon y^d)/y$. The contribution of $c = 0$ is exactly $1/N$, and if we put it aside we get:

$$\left|\Pr_{(x,y)\in\mathcal{X}\times\mathcal{Y}^*}\left[s = \epsilon y^d + xy\right] - \frac{1}{N}\right| \leq \frac{\max_{1\leq c\leq N-1}|B_c|}{N\#\mathcal{X}\#\mathcal{Y}^*} \sum_{c=1}^{N-1}\left|\sum_{x\in\mathcal{X}} \mathbf{e}_N(cx)\right| \text{ where } B_c = \sum_{y\in\mathcal{Y}^*} \mathbf{e}_N\Big(c\frac{\epsilon y^d - s}{y}\Big)$$

Then, the sum in c and x can be bounded as $N \log N$ using Lemma 1, whereas the sums B_c are exponential sums for rational functions, and can be bounded using Lemma 2, a famous theorem by Weil related to the Riemann hypothesis for function fields. There are a few subtle points to deal with: the sums B_c are in fact incomplete exponential sums (they are like Weil sums but restricted to an interval), and they are for a composite modulus (while Weil's theorem applies to prime moduli). Nevertheless, we can prove that $|B_c| \leq 4e^2 N^{1/2} \log N$ for all c, and conclude that:

$$\left|\Pr_{(x,y)\in\mathcal{X}\times\mathcal{Y}^*}\left[s = \epsilon y^d + xy\right] - \frac{1}{N}\right| \leq \frac{4e^2 N^{1/2}\log^2 N}{\#\mathcal{X}\#\mathcal{Y}^*}.$$

Since the result we want is for $(x, y) \in \mathcal{X} \times \mathcal{Y}$ rather than $\mathcal{X} \times \mathcal{Y}^*$, we have to account for the difference, which yields a slightly worse bound in $4e^2 + 1$ rather than $4e^2$: overall, we find the statistical distance to be bounded as

$$\sum_{s\in\mathbb{Z}/N\mathbb{Z}}\left|\Pr_{(x,y)\in\mathcal{X}\times\mathcal{Y}}\left[s = \epsilon y^d + xy\right] - \frac{1}{N}\right| \leq N \cdot \frac{(4e^2+1)N^{3/2}\log^2 N}{\#\mathcal{X}\#\mathcal{Y}} \leq \frac{(4e^2+1)N^{1/2}\log^2 N}{2^{\rho-1} - 1/2}$$

as required, since $\#\mathcal{Y} \geq N/2$. □

Remark 2. Concretely, this result means that, for a small public exponent e, it suffices to choose ρ slightly larger than half of the size of the modulus N to obtain a distribution that is statistical close to uniform. For example, if $e \leq 2^{16} + 1$ and N is a typical RSA modulus (less than 20000 bits, say), then taking ρ to be 200 bits larger than half of the length of N ensures that $\delta_1 < 2^{-128}$.

Moreover, we conjecture that *much smaller* values of ρ are in fact sufficient to obtain a close to uniform distribution and make our security proof go through (e.g. 200 bits rather than $k/2+200$ bits). Indeed, assuming the Generalized Riemann Hypothesis (GRH), we can prove that this is the case for $\epsilon = 0$ (see Appendix C) using estimates based on multiplicative character sums rather than exponential sums, and we conjecture that a similar estimate holds for $\epsilon = 1$ as well.

The security proof requires another statistical lemma which ensures that the adversary has a negligible probability of querying the correct value $\omega \leftarrow \mathcal{H}(M, r)$ given a faulty signature. This lemma can be easily derived from Lemma 8 proved in appendix B.

Lemma 4. *Let N, e be as in Lemma 3 and $a \in \mathbb{Z}/N\mathbb{Z}$, $\epsilon \in \{0,1\}$ be two fixed values, and assume that $\rho \geq k_h + k/2 + 2\log_2 k + \log_2(2e^2+1)$. For any choice of $\sigma' \in \mathbb{Z}/N\mathbb{Z}$ such that $s = \sigma' - ap \in (\mathbb{Z}/N\mathbb{Z})^*$ and any k_h-bit value ω', the probability that a solution $(x, y) \in [1, 2^\rho) \times [0, 2^{k-1})$ of the equation $\sigma' = \epsilon y^d + ap + xy \bmod N$ satisfies that the most significant k_h bits $\omega \in [0, 2^{k_h})$ of y coincides with ω' is bounded as:*

$$\Pr\left[\omega = \omega' | \sigma'\right] \leq \frac{3}{2^{k_h}}.$$

Remark 3. The probability that s is not invertible can be easily bounded by $3/\sqrt{N}$ for balanced RSA modulus. Concretely, this result means that, for a small public exponent e, we can choose ρ slightly larger than half of the size of the modulus N and k_h.

5 Security proof

The hash functions $\mathcal{G}$ and $\mathcal{H}$ are modelled as random oracles. For clarity, we display the initial definition of $\mathcal{H}$ on the left in Fig. 4. The initial definition of $\mathcal{G}$ is similar. We assume two global maps h and g are used to build the random oracles. Our proof works mostly by transforming the random oracle $\mathcal{H}$. We therefore display the code for $\mathcal{H}$ for each transition, only displaying other oracles when they suffer non-trivial changes. The proof, including all intermediate games, is fully formalized in EasyCrypt.

Game 0. We initially transform both random oracles to keep track of the first caller to make a particular query. It can be either the adversary (Adv), the signature oracle (Sig), or the faulty signature oracle (FSig). Calls made by the experiment when checking the validity of the forgery do not need to tag their query as they are the last queries made to the random oracles and do not need to update its state. We also extend the internal state of $\mathcal{H}$ with an additional field for use later in the proof, and currently set to a default value $\perp$.

Games 1 and 2. In Game 1, we anticipate a call to $\mathcal{G}$ on the output of $\mathcal{H}$ every time $\mathcal{H}$ is called. When $\mathcal{H}$ is called by either one of the signing oracles, we return the result of that call to $\mathcal{G}$ as well as the result of the current $\mathcal{H}$ query, allowing broad simplifications to the signing oracles.

In Game 2, we deal with collisions on r and ω values in the signing oracles. In later steps of the proof, we will need the control-flow of the faulty signature oracle to be completely independent from both r and ω, and we modify the oracle to allow these later transformations. Fresh queries are treated normally. Non-fresh queries made by the signing oracles are resampled as fresh if the previous query had been made by the faulty signature oracle. Non-fresh queries made by the faulty signature oracle are resampled, but not stored into the state. Game 1 is perfectly indistinguishable

Figure 4 Initial transition: extending state

```
oracle H(m, r)
    if (m, r) ∉ dom(h) then
        h[m, r] ← {0,1}^{k_h}
    return h[m, r]
```

```
oracle H_0(m, r)
    if (m, r) ∉ dom(h) then
        ω ← {0,1}^{k_h}
        h[m, r] ← (ω, c, ⊥)
    return π_1(h[m, r])
```

$$\Pr[\mathcal{UF}\text{-}\mathcal{CMA}^{\mathcal{A},\mathcal{K},\mathcal{S},\mathcal{F},\mathcal{V}} : win] = \Pr[\mathit{Game0} : win]$$

Figure 5 Games 1 and 2: anticipating calls to $\mathcal{G}$ and removing signing collisions

```
oracle H_1(c, m, r)
    if (m, r) ∉ dom(h) then
        ω ← {0,1}^{k_h}
        h[m, r] ← (ω, c, ⊥)
        st ← G(c, ω)
    else
        ω ← π_1(h[m, r])
        if c = Adv then
            st ← ⊥
        else
            st ← G(c, ω)
    return (ω, st)
```

```
oracle H_2(c, m, r)
    if (m, r) ∉ dom(h) ∨ c = FSig ∨
       (c = Sig ∧ π_2(h[m, r]) = FSig)  then
        ω ← {0,1}^{k_h}
        st ← {0,1}^{k_g}
        if c ≠ FSig ∨ (m, r) ∉ dom(h) then
            h[m, r] ← (ω, c, ⊥)
        if c ≠ FSig ∨ ω ∉ dom(g) then
            g[ω] ← (st ⊕ (r || 0^{k_g − k_0}), c)
    else
        ω ← π_1(h[m, r])
        if c = Adv then
            st ← ⊥
        else
            (ω, st) ← ⊥
    return (ω, st)
```

$$\Pr[\mathit{Game0} : win] \leq \Pr[\mathit{Game2} : win] + \frac{(q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot (q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}})}{2^{k_0}} + \frac{(q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot (q_{\mathcal{G}} + q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}})}{2^{k_h}}$$

from Game 0, and Game 2 can be distinguished from Game 1 if either i. the fresh r used in $\mathcal{H}$-queries made by the signing oracles collides with a previously used r (with probability at most $(q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot (q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot 2^{-k_0}$); changes introducing this failure are marked in red (lines 2, 5 and 6); ii. or the fresh ω used in $\mathcal{G}$-queries made by the signing oracles collides with a previously used ω (with probability at most $(q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot (q_{\mathcal{G}} + q_{\mathcal{H}} + q_{\mathcal{S}} + q_{\mathcal{F}}) \cdot 2^{-k_h}$); changes introducing this failure are marked in blue (lines 4, 7 and 8). Note that the value stored in g at line 8 in $\mathcal{H}_2$ is uniformly distributed since st is.

Game 3. Given that $\mathcal{H}$ now samples both bitstrings that compose the final padded message, we compute the entire signature in $\mathcal{H}$ when called by either one of the signing oracles. We transform the experiment to sample an integer x^* and compute $y^* = x^{*e} \bmod N$ to serve as one-way challenge. We embed it in the state when replying to $\mathcal{H}$ queries made by the adversary. Everything up to this point has been set up so that the signing oracles can simply use $\pi_3(h[m, r])$ as the padded message for m with salt r. Game 3 includes this simplification.

We introduce additional notation for clarity in the rest of the proof. Consider the function:

$$f_{(e,N),y^*,c} : \sigma \mapsto \begin{cases} y^* \cdot \sigma^e \bmod N \text{ if } c = \mathsf{Adv} \\ \sigma^e \bmod N \qquad \text{otherwise} \end{cases}$$

For a set $X \subseteq \mathbb{Z}/N\mathbb{Z}$, we denote by $\mathsf{pim}_{(e,N),y^*,c}(X)$ the uniform distribution on the set $S = \left\{\sigma \in \mathbb{Z}/N\mathbb{Z} \mid f_{(e,N),y^*,c}(\sigma) \in X\right\}$.

Figure 6 Games 3 and 4: Embedding the one-way challenge and oracle queries in $\mathcal{H}$

```
1:  oracle H_3(c, m, r)
2:      if (m, r) ∉ dom(h) ∨ c = FSig ∨
           (c = Sig ∧ π_2(h[m, r]) = FSig)  then
3:          σ ← pim_{(e,N),y*,c}([0..2^{k-1}))
4:          y ← f_{(e,N),y*,c}(σ)
5:          b || ω || st ← i2osp(y)
6:          if c ≠ FSig ∨ (m, r) ∉ dom(h) then
7:              h[m, r] ← (ω, c, σ)
8:          if c ≠ FSig ∨ ω ∉ dom(g) then
9:              g[ω] ← (st ⊕ (r || 0^{k_g − k_0}), c)
10:     else
11:         ω ← π_1(h[m, r])
12:         if c = Adv then
13:             st ← ⊥
14:         else
15:             (ω, st) ← ⊥
16:     return (ω, st)
```

```
1:  oracle H_4(c, m, r)
2:      if (m, r) ∉ dom(h) ∨ c = FSig then
3:          σ ← pim_{(e,N),y*,c}([0..2^{k-1}))
4:          y ← f_{(e,N),y*,c}(σ)
5:          b || ω || st ← i2osp(y)
6:          if c ≠ FSig then
7:              h[m, r] ← (ω, c, σ)
8:              g[ω] ← (st ⊕ (r || 0^{k_g − k_0}), c)
9:      else
10:         ω ← π_1(h[m, r])
11:         if c = Adv then
12:             st ← ⊥
13:         else
14:             (ω, st) ← ⊥
15:     return (ω, st)
```

$$\Pr[\mathit{Game2} : \mathit{win}] \leq \Pr[\mathit{Game3} : \mathit{win}] + 2^{-\frac{k}{2}+2} \qquad \Pr[\mathit{Game3} : \mathit{win}] \leq \Pr[\mathit{Game4} : \mathit{win}] + \frac{q_{\mathcal{H}} \cdot q_{\mathcal{S}}}{2^{k_0}} + \frac{q_{\mathcal{G}} \cdot q_{\mathcal{F}} \cdot 3}{2^{k_h}}$$

Game 3 is indistinguishable from Game 2 exactly when x^* is invertible. Therefore, the probability that the adversary distinguishes the two games is exactly $\frac{p+q-1}{p \cdot q}$. We have $p + q - 1 \leq 2^{\frac{k}{2}+1}$ and $2^{k-1} \leq p \cdot q$ and we can therefore bound the probability of this simulation failing by $2^{-\frac{k}{2}+2}$. Since

the invertibility of x^* is important in some later steps, we in fact let $\mathcal{H}$ compute a response only when x^* is invertible. In later stages, since x^* is not public, we instead check the invertibility of y^*, which is equivalent. For simplicity, we omit discussions regarding this detail in the rest of this section.

Game 4. In this game, we stop keeping track of the random oracle queries made by the faulty signature oracle. This is an important step towards being able to apply Lemma 3, which only discusses the statistical distance between two distributions on σ', rather than (ω, σ'). Note that, in Coron and Mandal's proof, Lemma 3 is applied before this transition, in a context in which its premises are not fulfilled.

By removing data about random oracle queries, we introduce observable changes in the game's behaviour whenever the adversary queries $\mathcal{H}$ with an r that was used previously in a faulty signature query, or whenever the adversary queries $\mathcal{G}$ with an ω that was used previously in a faulty signature query. We bound the probability of the adversary guessing an ω value using Lemma 4. Since the view of the adversary does not depend on r values sampled by the faulty signature oracle (see Fig. 7), the probability of the adversary guessing an r value used in generating a faulty signature is easily bounded.

Game 5. Our main goal at this stage is to show that faulty signatures are in fact indistinguishable from uniform randomness and can be simulated without using the random oracles. Once this is done, we will be able to resume the proof of security following more standard PSS proofs.

We now use Lemma 3 to completely simulate faulty signature oracle queries. We focus on the faulty signature oracle, inlining and simplifying $\mathcal{H}$ knowing that $c = \mathsf{FSig}$. On the left, we display the simplified faulty signature oracle from Game 4 for reference.

Figure 7 Game 5: sampling faulty signatures

1: **oracle** $\mathcal{F}_4(m, \epsilon, a)$
2: $\quad r \leftarrow \{0,1\}^{k_0}$
3: $\quad \sigma \leftarrow \mathsf{pim}_{(e,N),y^*,c}\left(\left[0..2^{k-1}\right)\right)$
4: $\quad y \leftarrow \sigma^e \bmod N$
5: $\quad r' \leftarrow \{0,1\}^{\rho} \setminus 0$
6: $\quad \sigma' \leftarrow \epsilon \cdot \sigma + a \cdot p - r' \cdot y$
7: $\quad$ **return** $\mathsf{i2osp}(\sigma')$

1: **oracle** $\mathcal{F}_5(m, \epsilon, a)$
2: $\quad r \leftarrow \{0,1\}^{k_0}$
3: $\quad \sigma' \leftarrow [0..N)$
4: $\quad$ **return** $\mathsf{i2osp}(\sigma')$

$$\Pr[\mathit{Game4} : \mathit{win}] \leq \Pr[\mathit{Game5} : \mathit{win}] + \frac{q_{\mathcal{F}} \cdot (4e_{max}^2 + 1) \cdot k^2}{2^{\rho - k/2}}$$

We make use of elementary properties of the statistical distance and Lemma 3 to bound the probability of distinguishing Games 5 and 6. Note that sampling σ in $\mathsf{pim}_{(e,N),y^*,c}\left(\left[0..2^{k-1}\right)\right)$ and applying the public RSA permutation to obtain y is perfectly equivalent to sampling y in $\left[0..2^{k-1}\right)$.

Game 6. With the faulty signature oracle simplified away, we can now focus on simulating the signature oracle. From now on, the c argument to $\mathcal{H}$ can no longer be FSig. More generally, it is impossible for any entry in h or g to be tagged with FSig.

The signature oracle we have defined at this point is not a valid simulator as it does not run in polynomial time. To ensure that it does, we replace the sampling operation at line 3 in Fig. 6

(right) with the loop displayed on the left of Fig. 8 to sample σ. The adversary can distinguish the two games whenever the loop finishes in a state where y does not start with a 0 bit.

Figure 8 Game 6 and inverter: sampling σ in polynomial time

```
1: while (!0 ≤ y < 2^{k-1}) ∧ i < k_0 do
2:     σ ← [0..N)
3:     y ← f_{(e,N),y*,c}(σ)
4:     i ← i + 1
```

```
1: oracle I(e, N, y*)
2:     (m, s) ← A^{H_7, G_7, S_7, F_7}(e, N)
3:     σ ← os2ip(s)
4:     y ← σ^e mod N
5:     b || ω || st ← i2osp(y)
6:     r || γ ← st ⊕ g[ω]
7:     (ω', Adv, u) ← h[m, r]
8:     return σ · u^{-1}
```

$$\Pr[\mathit{Game5} : \mathit{win}] \leq \Pr[\mathit{Game6} : \mathit{win}] + \frac{q_{\mathcal{H}}+q_{\mathcal{S}}}{2^{k_0}} \qquad \Pr[\mathit{Game6} : \mathit{win}] \leq \Pr[\mathcal{OW}\text{-}\mathcal{RSA}^{\mathcal{I}} : x = x^*] + \frac{1}{2^{k_h}} + \frac{q_{\mathcal{H}}}{2^{\frac{k}{2}-1}}$$

At each iteration of the loop, the σ sampled is invalid with probability at most $\frac{1}{2}$. The probability that *all* iterations produce an invalid σ is therefore bounded by $\frac{1}{2^{k_0}}$, since all samples are independent. $\mathcal{H}_7$ may now be queried $q_{\mathcal{H}} + q_{\mathcal{S}}$ times, allowing us to conclude.

Reduction All the oracles are simulated without using any secret data. We now focus on building an inverter. The adversary can win in two disjoint cases:

- either the $\mathcal{H}$-query made by the verification algorithm is fresh (this occurs with probability at most 2^{-k_h}),
- or the $\mathcal{H}$-query made by the verification algorithm was previously made by the adversary. If the query was made by the signature oracle, the forgery cannot be fresh.

In the latter case, the one-way challenge can then be recovered by the inverter shown on the right of Fig. 8. The key observation is that, in case of a successful forgery, we have $y = \sigma^e \bmod N$ (line 4) and $y = y^* \cdot u^e \bmod N$ (by invariant on h). By definition of y^* and the morphism and injectivity properties of RSA, we therefore have $\sigma = x^* \cdot u$. We need to also consider the case where a u stored in the h map by the adversary is not invertible, which occurs with probability at most $q_{\mathcal{H}} \cdot 2^{-k/2+1}$.

The final bound is obtained by transitively using the individual transition bounds.

6 Formalization

We have formalized the security proof using EasyCrypt. The formal proof contains the sequence of games outlined in the previous Section and justifications for all transitions between games. Lemmas 3 and 4 are stated as axioms in our formalization. Formally verifying the lemmas in EasyCrypt, or any other proof assistant, would first require to build a verified library for additive characters that covers at least the results used in our proof, and is outside the scope of this work.

A tarball containing the formal proof and the pre-release version of EasyCrypt used for the proof can be found at https://www.easycrypt.info/downloads/ches14/faultyPSS.tar.bz2.

Overview of EasyCrypt. EasyCrypt [4] is a tool-assisted framework that supports the verification of code-based game-playing proofs. Games are written in a core probabilistic procedural imperative

language, and using a module system to account for instance for adversaries. For the clarity of presentation, we simply view an adversary as an abstract procedure.

Reasoning about game-based proofs is based on two main tools. The first tool is a probabilistic relational Hoare logic pRHL that allows to relate two games. Its judgments are of the form $\{\Phi\}G_1 \sim G_2\{\Psi\}$ where Φ and Ψ are binary relations on states, G_1 and G_2 are games. Informally, the judgment is valid if the two subdistributions obtained by executing G_1 and G_2 on two initial memories that are related by Φ are themselves related by Ψ. The main use of pRHL is for performing transitions between two games. In particular, when the relation Ψ simply corresponds to equivalence between states, one can conclude that for every event E, we have $Pr[G1 : E] = Pr[G2 : E]$. In the more general case where the relation Ψ corresponds to equivalence between states up to a failure event F, one can conclude from the valid judgment that for every event E,

$$Pr[G1 : E] \leq Pr[G2 : E] + Pr[G2 : F]$$

The second tool is a probabilistic Hoare logic pHL that allows to (upper or lower) bound the probability of an event in a game. Its judgments are of the form $\{\Phi\}G\{\Psi\} \diamond p$ where Φ and Ψ are predicates on states (events), G is a game, p is a probability expression and $\diamond$ is a comparison operator $=, \leq, \geq$. Informally, the judgment is valid if the probability of Ψ in the subdistribution obtained by executing G on an initial memory satisfying Φ is equal (or upper bounded, or lower bounded, depending on the comparison operator $\diamond$) by p. The main use of pHL is for bounding the probability of failure events.

The combination of pRHL and pHL provides a powerful framework that captures the most common patterns of reasoning used in cryptographic proofs. We refer to [5,4] for more information.

References

1. J. B. Almeida, M. Barbosa, G. Barthe, and F. Dupressoir. Certified computer-aided cryptography: efficient provably secure machine code from high-level implementations. In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM CCS 13*, pages 1217–1230, Berlin, Germany, Nov. 4–8, 2013. ACM Press.
2. R. J. Anderson and M. G. Kuhn. Low cost attacks on tamper resistant devices. In B. Christianson, B. Crispo, T. M. A. Lomas, and M. Roe, editors, *Security Protocols Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136. Springer, 1997.
3. C. Aumüller, P. Bier, W. Fischer, P. Hofreiter, and J.-P. Seifert. Fault attacks on RSA with CRT: Concrete results and practical countermeasures. In B. S. Kaliski Jr., Çetin Kaya. Koç, and C. Paar, editors, *CHES 2002*, volume 2523 of *LNCS*, pages 260–275, Redwood Shores, California, USA, Aug. 13–15, 2002. Springer, Berlin, Germany.
4. G. Barthe, B. Grégoire, S. Heraud, and S. Z. Béguelin. Computer-aided security proofs for the working cryptographer. In P. Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 71–90, Santa Barbara, CA, USA, Aug. 14–18, 2011. Springer, Berlin, Germany.
5. G. Barthe, B. Grégoire, and S. Zanella-Béguelin. Formal certification of code-based cryptographic proofs. In *36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009*, pages 90–101, New York, 2009. ACM.
6. A. Berzati, C. Canovas-Dumas, and L. Goubin. Public key perturbation of randomized RSA implementations. In S. Mangard and F.-X. Standaert, editors, *CHES 2010*, volume 6225 of *LNCS*, pages 306–319, Santa Barbara, California, USA, Aug. 17–20, 2010. Springer, Berlin, Germany.
7. J. Blömer, M. Otto, and J.-P. Seifert. A new CRT-RSA algorithm secure against Bellcore attacks. In *ACM Conference on Computer and Communications Security*, pages 311–320, 2003.
8. D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of eliminating errors in cryptographic computations. *Journal of Cryptology*, 14(2):101–119, 2001.
9. A. Boscher, H. Handschuh, and E. Trichina. Fault resistant RSA signatures: Chinese remaindering in both directions. *IACR Cryptology ePrint Archive*, 2010:38, 2010.

10. E. Brier, D. Naccache, P. Q. Nguyen, and M. Tibouchi. Modulus fault attacks against RSA-CRT signatures. In B. Preneel and T. Takagi, editors, *CHES 2011*, volume 6917 of *LNCS*, pages 192–206, Nara, Japan, Sept. 28 – Oct. 1, 2011. Springer, Berlin, Germany.
11. R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In J. Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 90–106, Prague, Czech Republic, May 2–6, 1999. Springer, Berlin, Germany.
12. M. Christofi, B. Chetali, L. Goubin, and D. Vigilant. Formal verification of a CRT-RSA implementation against fault attacks. *J. Cryptographic Engineering*, 3(3):157–167, 2013.
13. M. Ciet and M. Joye. Practical fault countermeasures for Chinese remaindering based cryptosystems. In L. Breveglieri and I. Koren, editors, *FDTC*, pages 124–131, 2005.
14. J.-S. Coron, C. Giraud, N. Morin, G. Piret, and D. Vigilant. Fault attacks and countermeasures on Vigilant's RSA-CRT algorithm. In *FDTC*, pages 89–96, 2010.
15. J.-S. Coron, A. Joux, I. Kizhvatov, D. Naccache, and P. Paillier. Fault attacks on RSA signatures with partially unknown messages. In *CHES*, pages 444–456, 2009.
16. J.-S. Coron and A. Mandal. PSS is secure against random fault attacks. In M. Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 653–666, Tokyo, Japan, Dec. 6–10, 2009. Springer, Berlin, Germany.
17. J.-S. Coron, D. Naccache, and M. Tibouchi. Fault attacks against EMV signatures. In *CT-RSA*, pages 208–220, 2010.
18. P.-A. Fouque, N. Guillermin, D. Leresteux, M. Tibouchi, and J.-C. Zapalowicz. Attacking RSA-CRT signatures with faults on Montgomery multiplication. *J. Cryptographic Engineering*, 3(1):59–72, 2013.
19. C. Giraud. An RSA implementation resistant to fault attacks and to simple power analysis. *IEEE Trans. Computers*, 55(9):1116–1120, 2006.
20. D.-P. Le, M. Rivain, and C. H. Tan. On double exponentiation for securing RSA against fault analysis. In J. Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 152–168, San Francisco, CA, USA, Feb. 25–28, 2014. Springer, Berlin, Germany.
21. R. Lidl and H. Niederreiter. *Finite Fields*. Cambridge University Press, 1996.
22. H. L. Montgomery. *Topics in multiplicative number theory*. Springer-Verlag, 1971.
23. P. L. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44:519–521, 1985.
24. N. Moro, K. Heydemann, E. Encrenaz, and B. Robisson. Formal verification of a software countermeasure against instruction skip attacks. *Journal of Cryptographic Engineering*, pages 1–12, 2014.
25. P. Rauzy and S. Guilley. A formal proof of countermeasures against fault injection attacks on CRT-RSA. *Journal of Cryptographic Engineering*, pages 1–13, 2013.
26. M. Rivain. Securing RSA against fault analysis by double addition chain exponentiation. In M. Fischlin, editor, *CT-RSA 2009*, volume 5473 of *LNCS*, pages 459–480, San Francisco, CA, USA, Apr. 20–24, 2009. Springer, Berlin, Germany.
27. A. Shamir. Improved method and apparatus for protecting public key schemes from timing and fault attacks. Patent Application, 1998. WO 1998/052319 A1.
28. E. Trichina and R. Korkikyan. Multi fault laser attacks on protected CRT-RSA. In L. Breveglieri, M. Joye, I. Koren, D. Naccache, and I. Verbauwhede, editors, *FDTC*, pages 75–86. IEEE Computer Society, 2010.
29. D. Vigilant. RSA with CRT: A new cost-effective solution to thwart fault attacks. In E. Oswald and P. Rohatgi, editors, *CHES 2008*, volume 5154 of *LNCS*, pages 130–145, Washington, D.C., USA, Aug. 10–13, 2008. Springer, Berlin, Germany.
30. I. M. Vinogradov. *Elements of number theory*. Dover, 1954.
31. S.-M. Yen, S.-J. Moon, and J. Ha. Permanent fault attack on the parameters of RSA with CRT. In R. Safavi-Naini and J. Seberry, editors, *ACISP 03*, volume 2727 of *LNCS*, pages 285–296, Wollongong, NSW, Australia, July 9–11, 2003. Springer, Berlin, Germany.

A Proof of Lemma 3

Fix two positive integers $X, Y \leq N$ and some integer constant $h \in [0, N - Y)$, and consider the sets $\mathcal{X} = [1, X] \subset \mathbb{Z}/N\mathbb{Z}$, $\mathcal{Y} = [h, Y + h) \subset \mathbb{Z}/N\mathbb{Z}$ and $\mathcal{Y}^* = \mathcal{Y} \cap (\mathbb{Z}/N\mathbb{Z})^*$. $\mathcal{X}$ and $\mathcal{Y}$ are of cardinality X, Y respectively, and we define $Y^* = \#\mathcal{Y}^*$. Note that $Y \geq Y^* \geq Y - p - q$ since there are fewer than $p + q$ non-invertible elements in $\mathbb{Z}/N\mathbb{Z}$.

We want to estimate the regularity of the distribution of elements of the form $xy + \epsilon y^d \in \mathbb{Z}/N\mathbb{Z}$ for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. To do so, for any $s \in \mathbb{Z}/N\mathbb{Z}$, we denote by $T(s)$ the number of representations

of s as a sum of the form $xy + \epsilon y^d$ for $(x, y) \in \mathcal{X} \times \mathcal{Y}$, and by $T^*(s)$ the corresponding number for $(x, y) \in \mathcal{X} \times \mathcal{Y}^*$. In other words:

$$T(s) = \#\{(x, y) \in \mathcal{X} \times \mathcal{Y} \mid s = xy + \epsilon y^d\}; \qquad T^*(s) = \#\{(x, y) \in \mathcal{X} \times \mathcal{Y}^* \mid s = xy + \epsilon y^d\}.$$

Our goal is to prove that for almost all $s \in \mathbb{Z}/N\mathbb{Z}$, $T(s)$ is close to its average value XY/N, subject to suitable bounds on X and Y. To do so, we first estimate $T^*(s)$ using exponential sum techniques.

Lemma 5. *For all $s \in \mathbb{Z}/N\mathbb{Z}$, we have:*

$$\left| T^*(s) - \frac{XY^*}{N} \right| \leq 4e^2 N^{1/2} \log^2 N.$$

Proof. Note that for $y \in \mathcal{Y}^*$ (hence invertible), we have $s = xy + \epsilon y^d$ if and only if $x = (s - \epsilon y^d)/y$ in $\mathbb{Z}/N\mathbb{Z}$. As a result, Lemma 1 ensures that:

$$\begin{aligned} T^*(s) &= \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}^*} \frac{1}{N} \sum_{c=0}^{N-1} \mathbf{e}_N\left(c\Big(x - \frac{s - \epsilon y^d}{y}\Big)\right) = \frac{1}{N} \sum_{c=0}^{N-1} \sum_{x \in \mathcal{X}} \mathbf{e}_N(cx) \sum_{y \in \mathcal{Y}^*} \mathbf{e}_N\Big(c \cdot \frac{\epsilon y^d - s}{y}\Big) \\ &= \frac{XY^*}{N} + \frac{1}{N} \sum_{c=1}^{N-1} \sum_{x \in \mathcal{X}} \mathbf{e}_N(cx) \sum_{y \in \mathcal{Y}^*} \mathbf{e}_N\Big(c \cdot \frac{\epsilon y^d - s}{y}\Big), \end{aligned}$$

by putting aside the contribution of $c = 0$. Now define:

$$B = \max_{1 \leq c \leq N-1} |B_c| \quad \text{where} \quad B_c = \sum_{y \in \mathcal{Y}^*} \mathbf{e}_N\Big(c \cdot \frac{\epsilon y^d - s}{y}\Big).$$

We then have:

$$\left| T^*(s) - \frac{XY^*}{N} \right| \leq \frac{B}{N} \sum_{c=1}^{N-1} \left| \sum_{x \in \mathcal{X}} \mathbf{e}_N(cx) \right| \leq B \log N, \tag{1}$$

by Lemma 1. Therefore, estimating $T^*(s)$ reduces to obtaining an appropriate bound on B, which we will do using Weil's theorem (Lemma 2), which provides a bound on sums of the form:

$$S_\ell(\varphi) = \sum_{\substack{x \in \mathbb{F}_\ell \\ \varphi(x) \neq \infty}} \mathbf{e}_p\big(\varphi(x)\big)$$

for rational functions φ on the finite field $\mathbb{F}_\ell$, provided that φ is not constant. Now write:

$$\begin{aligned} B_c &= \sum_{w \in (\mathbb{Z}/N\mathbb{Z})^*} \mathbf{e}_N\Big(c \cdot \frac{\epsilon w^d - s}{w}\Big) \cdot [w \in \mathcal{Y}] = \sum_{w \in (\mathbb{Z}/N\mathbb{Z})^*} \mathbf{e}_N\Big(c \cdot \frac{\epsilon w^d - s}{w}\Big) \cdot \sum_{y \in \mathcal{Y}} \frac{1}{N} \sum_{c'=0}^{N} \mathbf{e}_N\big(c'(w - y)\big) \\ &= \frac{1}{N} \sum_{c'=0}^{N} \sum_{y \in \mathcal{Y}} \mathbf{e}_N(-c'y) \sum_{w \in (\mathbb{Z}/N\mathbb{Z})^*} \mathbf{e}_N\Big(c \cdot \frac{\epsilon w^d - s}{w} + c'w\Big) \\ &= \frac{1}{N} \sum_{c'=0}^{N} \sum_{y \in \mathcal{Y}} \mathbf{e}_N(-c'y) \sum_{z \in (\mathbb{Z}/N\mathbb{Z})^*} \mathbf{e}_N\Big(\frac{c'z^{2e} + \epsilon cz - cs}{z^e}\Big) \end{aligned}$$

using the change of variable $z^e = w$. If we pick integers u, v such that $up + vq = 1$, we see that the sum in z decomposes as:

$$\begin{aligned}\sum_{z\in(\mathbb{Z}/N\mathbb{Z})^*} \mathbf{e}_N\Big(\frac{c'z^{2e} + \epsilon cz - cs}{z^e}\Big) &= \sum_{z\in(\mathbb{Z}/N\mathbb{Z})^*} \mathbf{e}_N\Big((up+vq)\cdot\frac{c'z^{2e} + \epsilon cz - cs}{z^e}\Big)\\ &= \sum_{z_p\in\mathbb{F}_p^*} \mathbf{e}_p\Big(vf_{c,c'}(z_p) \bmod p\Big) \sum_{z_q\in\mathbb{F}_q^*} \mathbf{e}_p\Big(uf_{c,c'}(z_q) \bmod q\Big)\\ &= S_p(vf_{c,c'} \bmod p)\cdot S_q(uf_{c,c'} \bmod q)\end{aligned}$$

where the rational function $f_{c,c'}$ is given by $f_{c,c'}(z) = (c'z^{2e} + \epsilon cz - cs)/z^e$.

Now, if $c \not\equiv 0 \pmod p$, $vf_{c,c'} \bmod p$ is a non constant rational function in $\mathbb{F}_p(z)$ of degree at most $2e$ whose denominator vanishes in exactly one point of $\overline{\mathbb{F}_p}$, so Lemma 2 ensures $\big|S_p(vf_{c,c'} \bmod p)\big| \le 2e\sqrt{p}$. On the other hand, if $c \equiv 0 \pmod p$, we have:

$$S_p(vf_{c,c'} \bmod p) = \sum_{z\in\mathbb{F}_p^*} \mathbf{e}_p(vc'z^e) = \sum_{w\in\mathbb{F}_p^*} \mathbf{e}_p(vc'w) = \begin{cases} p-1 & \text{if } c' \equiv 0 \pmod p,\\ -1 & \text{otherwise.}\end{cases}$$

And the corresponding results hold for $S_q(uf_{c,c'})$. As a result, to bound B_c, we should distinguish the case when c is invertible mod N from the case where it is a multiple of p or q. When c is invertible, we directly have:

$$\begin{aligned}|B_c| &\le \frac{1}{N}\sum_{c'=0}^{N}\Big|\sum_{y\in\mathcal{Y}}\mathbf{e}_N(-c'y)\Big|\cdot\big|S_p(vf_{c,c'} \bmod p)\big|\cdot\big|S_q(uf_{c,c'} \bmod q)\big|\\ &\le \frac{1}{N}\sum_{c'=0}^{N}\Big|\sum_{y\in\mathcal{Y}}\mathbf{e}_N(-c'y)\Big|\cdot 2e\sqrt{p}\cdot 2e\sqrt{q} \le \frac{4e^2\sqrt{N}\cdot N\log N}{N} = 4e^2N^{1/2}\log N\end{aligned}$$

by Lemma 1. On the other hand, suppose that c is a multiple of p. We have:

$$\begin{aligned}|B_c| &= \Bigg|\frac{-1}{N}\sum_{\substack{c'\in\mathbb{Z}/N\mathbb{Z}\\ c'\not\equiv 0 \mod p}}\sum_{y\in\mathcal{Y}}\mathbf{e}_N(-c'y)\cdot S_q(uf_{c,c'} \bmod q) + \frac{p-1}{N}\sum_{\substack{c'\in\mathbb{Z}/N\mathbb{Z}\\ c'\equiv 0 \mod p}}\sum_{y\in\mathcal{Y}}\mathbf{e}_N(-c'y)\cdot S_q(uf_{c,c'} \bmod q)\Bigg|\\ &\le \frac{1}{N}\cdot N\log N\cdot 2e\sqrt{q} + \frac{p}{N}\sum_{c''=0}^{q-1}\Big|\sum_{y\in\mathcal{Y}}\mathbf{e}_q(-c''y)\Big|\cdot 2e\sqrt{q} \le 2e\sqrt{q}\cdot\log N + 2e\sqrt{q}\cdot\log q < 4e^2N^{1/2}\log N,\end{aligned}$$

by applying Lemma 1 once for the modulus N and once for q. The same bound holds when c is a multiple of q, so that $B \le 4e^2N^{1/2}\log N$. Together with (1), this concludes the proof. □

Lemma 6. *For all $s \in \mathbb{Z}/N\mathbb{Z}$, the difference $T(s) - T^*(s)$ is non negative, vanishes if s is invertible, and is bounded as follows otherwise:*

$$T(s) - T^*(s) \le \begin{cases} X + 2XY/N + Y/p + Y/q & \text{if } s = 0,\\ XY/N + Y/p & \text{if } s \neq 0 \text{ but } s \equiv 0 \bmod p,\\ XY/N + Y/q & \text{if } s \neq 0 \text{ but } s \equiv 0 \bmod q.\end{cases}$$

In particular, $T(s) - T^*(s) \le 3X + p + q$ for all s.

Proof. Indeed, if s is invertible, the equation $s = xy + \epsilon y^d = y\big(x + \epsilon y^{d-1}\big)$ can have no solution with $y \notin (\mathbb{Z}/N\mathbb{Z})^*$. Hence $T(s) = T^*(s)$.

Similarly, suppose that s is zero mod p but not mod q. Then all solutions (x, y) to the equation must satisfy $y \not\equiv 0 \mod q$, so solutions with $y \notin \mathcal{Y}^*$ are of the form (x, py_0) with $y_0 \in [h/p, (Y+h)/p)$. Moreover, for each given value of y_0, we have $x \equiv -py_0^{d-1} \mod q$, so the reduction of x mod q is fixed and there can thus be at most $X/q + 1$ solutions. Therefore, in total, we have:

$$T(s) - T^*(s) \leq \Big(\frac{X}{q} + 1\Big)\frac{Y}{p} = \frac{XY}{N} + \frac{Y}{p}$$

as required, and the corresponding result when s is zero mod q but not mod p follows for the same reason.

Finally, when $s = 0$, we can distinguish three types of solutions with $y \notin \mathcal{Y}^*$: i. those for which y is zero mod p but not mod q, and there are at most $XY/N + Y/p$ of them as above; ii. those for which y is zero mod q but not mod p, and there are at most $XY/N + Y/q$ of them; iii. and those with $y = 0$, in which case all values of x satisfy the equation. Hence:

$$T(0) - T^*(0) \leq \frac{XY}{N} + \frac{Y}{p} + \frac{XY}{N} + \frac{Y}{q} + X$$

as required. And the bound by $3X + p + q$ follows from the fact that $Y \leq N$. □

Lemma 7. *Suppose that N is a balanced RSA modulus (i.e. $N = pq$ for primes p, q such that $60 < q < p < 2q$). Then, the following holds:*

$$\sum_{s \in \mathbb{Z}/N\mathbb{Z}} \left| T(s) - \frac{XY}{N} \right| \leq (4e^2 + 1)N^{3/2} \log^2 N. \tag{2}$$

In particular, the statistical distance to uniform of elements of the form $xy + \epsilon y^d$ in $\mathbb{Z}/N\mathbb{Z}$ with $(x, y) \in \mathcal{X} \times \mathcal{Y}$ is bounded by $(4e^2 + 1)N^{3/2}(\log^2 N)/(XY)$, and is therefore negligible for $XY \gg e^2 N^{3/2+\delta}$ for any $\delta > 0$.

Proof. We decompose the sum from (2) as:

$$\sum_{s \in \mathbb{Z}/N\mathbb{Z}} \left| T(s) - \frac{XY}{N} \right| \leq \sum_{s \in \mathbb{Z}/N\mathbb{Z}} \left| T(s) - T^*(s) \right| + \sum_{s \in \mathbb{Z}/N\mathbb{Z}} \left| T^*(s) - \frac{XY^*}{N} \right| + \sum_{s \in \mathbb{Z}/N\mathbb{Z}} \left| \frac{XY^*}{N} - \frac{XY}{N} \right|,$$

and denote by S_1, S_2, S_3 the three sums on the right-hand side. We have:

$$S_3 = X \cdot (Y - Y^*) \leq X \cdot \big(N - \varphi(N)\big) \leq X \cdot (p + q) \leq X \cdot 3q \leq 3N^{3/2}$$

since $X \leq N$. Moreover, by Lemma 5, we have $S_2 \leq N \cdot 4e^2 N^{1/2} \log^2 N = 4e^2 N^{3/2} \log^2 N$. And Lemma 6 ensures that:

$$S_1 \leq \sum_{s \notin (\mathbb{Z}/N\mathbb{Z})^*} (3X + p + q) \leq (p + q) \cdot (3X + p + q) \leq 3\sqrt{N} \cdot 4N = 12N^{3/2}.$$

Thus, the sum from (2) is bounded by:

$$S_1 + S_2 + S_3 \leq 4e^2 N^{3/2}\big(\log^2 N + 15\big) \leq (4e^2 + 1)N^{3/2} \log^2 N$$

since $\log^2 N > \log^2 60 > 15$. This concludes the proof. □

B Proof of Lemma 4

Suppose $\mathcal{Y}$ is of the form $[0, Y)$ with $Y = mY_0$ for some integers $m, Y_0 \geq 1$, and write $\mathcal{Y}_\omega = [\omega Y_0, (\omega+1)Y_0)$ for $\omega = 0, 1, \ldots, m-1$. We claim that for XY_0 large enough, for each $s \in (\mathbb{Z}/N\mathbb{Z})^*$, the number of solutions of the equation $s = xy + \epsilon y^d$ with $(x, y) \in \mathcal{X} \times \mathcal{Y}_\omega$ is roughly independent of ω. More precisely, the following holds.

Lemma 8. *Suppose N as in Lemma 3 and assume that $\alpha = (4e^2+2)N^{3/2}\log^2 N/(XY_0) < 1/2$. Then, for all $s \in (\mathbb{Z}/N\mathbb{Z})^*$ and $0 \leq \omega < m$, the probability that a solution $(x, y) \in \mathcal{X} \times \mathcal{Y}$ of the equation $s = xy + \epsilon y^d$ satisfies $y \in \mathcal{Y}_\omega$ is bounded as:*

$$\left| \Pr\left[y \in \mathcal{Y}_\omega \mid (x, y) \in \mathcal{X} \times \mathcal{Y} \text{ and } s = xy + \epsilon y^d \right] - \frac{1}{m} \right| \leq \frac{4\alpha}{m}.$$

Proof. Let us denote by $T_\omega(s)$ the number of solutions x, y of the equation $s = xy + \epsilon y^d$ such that $(x, y) \in \mathcal{X} \times \mathcal{Y}_\omega$, and by $T^*_\omega(s)$ the number of solutions such that, in addition, y is invertible. We have:

$$\Pr\left[y \in \mathcal{Y}_\omega \mid (x, y) \in \mathcal{X} \times \mathcal{Y} \text{ and } s = xy + \epsilon y^d \right] = \frac{T_\omega(s)}{T(s)} = \frac{T^*_\omega(s)}{T^*(s)}$$

where the second equality holds by Lemma 6. And by Lemma 5, we have:

$$\begin{aligned} \left| T^*(s) - \frac{XY}{N} \right| &\leq \left| T^*(s) - \frac{XY^*}{N} \right| + \frac{X}{N}(Y - Y^*) \leq 4(e^2+1)N^{1/2}\log^2 N + 1 \cdot 3\sqrt{N} \\ &\leq 4(e^2+2)N^{1/2}\log^2 N \leq \alpha \frac{XY_0}{N}, \end{aligned}$$

and similarly, again by Lemma 5, $\left| T^*_\omega(s) - XY_0/N \right| \leq \alpha XY_0/N$. Thus, we get:

$$\frac{1-\alpha}{m+\alpha} = \frac{\frac{XY_0}{N} \cdot (1-\alpha)}{\frac{XY}{N} \cdot (1+\alpha/m)} \leq \frac{T^*_\omega(s)}{T^*(s)} \leq \frac{\frac{XY_0}{N} \cdot (1+\alpha)}{\frac{XY}{N} \cdot (1-\alpha/m)} = \frac{1+\alpha}{m-\alpha}.$$

It follows that:

$$-\frac{(m+1)\alpha}{m(m+\alpha)} = \frac{1-\alpha}{m+\alpha} - \frac{1}{m} \leq \frac{T^*_\omega(s)}{T^*(s)} - \frac{1}{m} \leq \frac{1+\alpha}{m-\alpha} - \frac{1}{m} = \frac{(m+1)\alpha}{m(m-\alpha)}.$$

And finally, if we bound $(m+1)/m$ above by 2 and α by $1/2$, we obtain:

$$\left| \frac{T^*_\omega(s)}{T^*(s)} - \frac{1}{m} \right| \leq \frac{2\alpha}{m - 1/2} = \frac{4\alpha}{2m-1} \leq \frac{4\alpha}{m}$$

as required. □

C Statistical indistinguishability for smaller ρ

We prove that under GRH, Lemma 5 holds *on average* with a much better bound for $\epsilon = 0$ (and we conjecture that the same is true for $\epsilon = 1$). We formulate the bound on $(\mathbb{Z}/N\mathbb{Z})^*$ for simplicity, but one can easily extend the result to $\mathbb{Z}/N\mathbb{Z}$ with the approach of Lemma 7.

Lemma 9. *Assume that the Generalized Riemann Hypothesis holds, and let $N, \mathcal{X}, \mathcal{Y}$ be as in Appendix A. For any $\delta > 0$ there exists a constant $c_\delta > 0$ such that:*

$$\sum_{s \in (\mathbb{Z}/N\mathbb{Z})^*} \left| \Pr_{\substack{(x,y) \in \mathcal{X} \times \mathcal{Y} \\ xy \in (\mathbb{Z}/N\mathbb{Z})^*}} \big[s = xy\big] - \frac{1}{\varphi(N)} \right| \leq c_\delta N^\delta \sqrt{\frac{\varphi(N)}{XY}}.$$

In particular, xy is statistically close to uniform in $(\mathbb{Z}/N\mathbb{Z})^*$ for $XY \gg N^{1+3\delta}$.

Proof. The proof uses multiplicative characters χ of $\mathbb{Z}/N\mathbb{Z}$ rather than additive ones. The deviation from the average at a given $s \in (\mathbb{Z}/N\mathbb{Z})^*$ can be written as:

$$\Pr[s = xy] - \frac{1}{\varphi(N)} = \frac{1}{\varphi(N)XY} \sum_{\chi \neq \chi_0} \sum_{(x,y) \in \mathcal{X} \times \mathcal{Y}} \chi(xy)\overline{\chi(s)}$$

where χ_0 is the trivial character. It follows that:

$$\sum_{s \in (\mathbb{Z}/N\mathbb{Z})^*} \left| \Pr[s = xy] - \frac{1}{\varphi(N)} \right|^2 = \frac{1}{(\varphi(N)XY)^2} \sum_{\chi, \chi' \neq \chi_0} \sum_{x,y,x',y'} \chi(xy)\overline{\chi'(x'y')} \sum_s (\overline{\chi}\chi')(s),$$

and the sum in s on the right-hand side vanishes for $\chi \neq \chi'$ and evaluates to $\varphi(N)$ otherwise. Hence:

$$\sum_{s \in (\mathbb{Z}/N\mathbb{Z})^*} \left| \Pr[s = xy] - \frac{1}{\varphi(N)} \right|^2 = \frac{1}{\varphi(N) \cdot (XY)^2} \sum_{\chi \neq \chi_0} \big|S(\chi)\big|^2,$$

where we write $S(\chi) = \sum_{x \in \mathcal{X}} \chi(x) \sum_{y \in \mathcal{Y}} \chi(y)$. Now since $\mathcal{X}$ is an interval of the form $[1, X]$, it is classical that GRH implies, for any $\delta > 0$, $\left| \sum_{x \in \mathcal{X}} \chi(x) \right| \leq c_\delta X^{1/2} N^\delta$ for some constant $c_\delta > 0$ (see e.g. [22, Eq. (13.2)]). Hence:

$$\sum_{s \in (\mathbb{Z}/N\mathbb{Z})^*} \left| \Pr[s = xy] - \frac{1}{\varphi(N)} \right|^2 \leq \frac{c_\delta^2 N^{2\delta}}{\varphi(N) X \cdot Y^2} \sum_\chi \sum_{y,y'} \chi(y)\overline{\chi(y')} = \frac{c_\delta^2 N^{2\delta}}{XY},$$

from which the stated result follows using the Cauchy–Schwarz inequality. □