Modulus Fault Attacks Against RSA-CRT Signatures

Éric Brier[1], David Naccache[2], Phong Q. Nguyen[2,3], and Mehdi Tibouchi[2]

[1] Ingenico
1, rue Claude Chappe, BP 346, F-07503 Guilherand-Granges, France
eric.brier@ingenico.com
[2] École normale supérieure
Département d'informatique, Groupe de Cryptographie
45, rue d'Ulm, F-75230 Paris CEDEX 05, France
{david.naccache, phong.nguyen, mehdi.tibouchi}@ens.fr
[3] INRIA

Abstract. RSA-CRT fault attacks have been an active research area since their discovery by Boneh, DeMillo and Lipton in 1997. We present alternative key-recovery attacks on RSA-CRT signatures: instead of targeting one of the sub-exponentiations in RSA-CRT, we inject faults into the *public modulus* before CRT interpolation, which makes a number of countermeasures against Boneh *et al.*'s attack ineffective.

Our attacks are based on orthogonal lattice techniques and are very efficient in practice: depending on the fault model, between 5 to 45 faults suffice to recover the RSA factorization within a few seconds. Our simplest attack requires that the adversary knows the faulty moduli, but more sophisticated variants work even if the moduli are unknown, under reasonable fault models. All our attacks have been fully validated experimentally with fault-injection laser techniques.

Keywords: Fault Attacks, Digital Signatures, RSA, CRT, Lattices.

1 Introduction

1.1 RSA-CRT Signatures

RSA [26] is the most widely used signature scheme. To sign a message m, the signer first applies an encoding function μ to m, and then computes the signature $\sigma = \mu(m)^d \bmod N$. To verify the signature σ, the receiver checks that $\sigma^e = \mu(m) \mod N$. The Chinese Remainder Theorem (CRT) is often used to speed up signature generation by a factor of about 4. This is done by computing:

$$\sigma_p = \mu(m)^{d \bmod p-1} \mod p \quad \text{and} \quad \sigma_q = \mu(m)^{d \bmod q-1} \mod q$$

and deriving σ from (σ_p, σ_q) using the CRT.

1.2 Fault Attacks on RSA-CRT Signatures

Back in 1997, Boneh, DeMillo and Lipton [7] showed that RSA-CRT implementations are vulnerable to fault attacks. Assuming that the attacker can induce a fault when σ_q is computed while keeping the computation of σ_p correct, one gets:

$$\sigma_p = \mu(m)^{d \bmod p-1} \mod p \quad \text{and} \quad \sigma_q \neq \mu(m)^{d \bmod q-1} \mod q$$

hence:

$$\sigma^e = \mu(m) \mod p \quad \text{and} \quad \sigma^e \neq \mu(m) \mod q$$

which allows the attacker to factor N by computing $\gcd(\sigma^e - \mu(m) \bmod N, N) = p$. This attack applies to any deterministic padding function μ, such as RSA PKCS#1 v1.5 or Full-Domain Hash [3], or probabilistic signatures where the randomizer used to generate the signature is sent along with the signature, such as PFDH [14]. Only probabilistic signature schemes such that the randomness remains unknown to the attacker may be safe, though some particular cases have been attacked as well [13].

In 2005, Seifert [27] introduced a new type of RSA fault attacks, by inducing faults on the RSA public modulus. The initial attack [27] only allowed to bypass RSA verification, but key-recovery attacks were later discovered by Brier *et al.* [9], and improved or extended in [19,6,4,5]. These key-recovery attacks only apply to RSA without CRT, and they require significantly more faults than Boneh *et al.*'s attack, at least on the order of 1000 faulty signatures.

1.3 Our contribution

We present new alternative key-recovery attacks on RSA-CRT signatures: instead of targeting one of the RSA-CRT sub-exponentiations, we inject faults into the *public modulus* like in Seifert's attack. This makes typical countermeasures against Boneh *et al.*'s attack ineffective against the new attacks.

Our attacks are based on the orthogonal lattice techniques introduced by Nguyen and Stern [21] in 1997. They are very effective in practice: they disclose the RSA factorization within a few seconds using only between 5 to 45 faulty signatures. The exact running time and number of faulty signatures depend on the fault model.

For instance, in our simplest attack, the running time is a fraction of a second using only 5 faulty signatures, but the attacker is assumed to know the faulted moduli for the 5 different messages. However, our attack can be extended to the case where the attacker no longer knows the faulted moduli, using at most 45 faulty signatures, under the following two fault models: either the faulted moduli only differ from the public modulus on a single byte of unknown position and unknown value, or the faulted moduli may differ from the public modulus by many bytes, but the differences are restricted to the least significant bits, up to half of the modulus size.

All our attacks have been fully validated with physical experiments with laser shots on a RISC microcontroller.

1.4 Related work

Many countermeasures have been proposed to protect against Boneh *et al.*'s attack and its numerous generalizations, but they often focus on the exponentiation process. The previously mentioned fault attacks [9,19,6,4,5] on RSA using faulty moduli only apply to standard RSA without CRT, and they use non-lattice techniques. Our attack seems to be the first attack on RSA-CRT with faulted moduli.

It should be pointed out, however, that a number of protected RSA-CRT implementations also protect the CRT recombination. This is for example the case of [1,11,16,8,30,25].

More generally, as we observe in §5, using the technique known as Garner's formula for CRT recombination does thwart the attack introduced in this paper. Since this formula is often used in practice, typical implementations conforming to RSA standards like PKCS#1 and IEEE P1363 should in principle be immune to this attack.

1.5 Roadmap

In §2, we describe the basic attack where the faulty moduli are assumed to be known to the attacker. In §3, we extend the attack to realistic fault models in which the faulty moduli are no longer known to the attacker. In §4, we describe physical experiments with laser shots on a RISC microcontroller to validate the attack. Finally, in §5, we suggest possible countermeasures against this attack.

2 The New Attack

2.1 Overview

Consider again the generation of RSA-CRT signatures. To obtain the signature σ of a message m padded as $\mu(m)$, the signer computes the mod-p and mod-q parts:

$$\sigma_p = \mu(m)^d \mod p \quad \text{and} \quad \sigma_q = \mu(m)^d \mod q$$

and returns the signature:

$$\sigma = \sigma_p \cdot \alpha + \sigma_q \cdot \beta \mod N \tag{1}$$

where α, β are the pre-computed Chinese Remainder coefficients $\alpha = q \cdot (q^{-1} \bmod p)$ and $\beta = p \cdot (p^{-1} \bmod q)$.

Assume that an adversary can obtain the correct signature σ, and also a signature σ' of the same padded message $\mu(m)$ after corrupting the modulus N before the CRT step (1). In other words, the attacker gets σ as before but also σ' defined as:

$$\sigma' = \sigma_p \cdot \alpha + \sigma_q \cdot \beta \mod N' \quad \text{for some } N' \neq N$$

Suppose further, for the moment, that the adversary is able to recover the faulty modulus N': we will see in §3 how this not-so-realistic hypothesis can be lifted in a more practical setting. Then, by applying the Chinese Remainder Theorem to σ and σ', the adversary can compute

$$v = \sigma_p \cdot \alpha + \sigma_q \cdot \beta \mod N \cdot N'.$$

But if we denote the bit length of N by n, then $N \cdot N'$ is a $2n$-bit integer, whereas α, β are of length n and σ_p, σ_q of length $n/2$, so v is really a linear combination of α and β in $\mathbb{Z}$:

$$v = \sigma_p \cdot \alpha + \sigma_q \cdot \beta.$$

That alone does not suffice to factor N, but several such pairs (σ, σ') provide multiple linear combinations of the (unknown) integers α, β with relatively small coefficients. Then lattice reduction techniques allow us to recover the coefficients σ_p and σ_q, and hence obtain the factorization of N by GCDs. The following sections describe this process in detail.

2.2 Applying Orthogonal Lattice Techniques

We assume that the reader is familiar with cryptanalysis based on lattices (see [20,23] for more information), particularly the orthogonal lattices introduced by Nguyen and Stern [21]: if L is a lattice in $\mathbb{Z}^n$, we let $L^\perp$ be the lattice formed by all vectors in $\mathbb{Z}^n$ which are orthogonal to all vectors of L. If an attacker obtains ℓ pairs (σ, σ'), he can compute as before a vector $\boldsymbol{v} = (v_1, \ldots, v_\ell)$ of $3n/2$-bit integers satisfying an equation of the form:

$$\boldsymbol{v} = \alpha \boldsymbol{x} + \beta \boldsymbol{y} \tag{2}$$

where $\boldsymbol{x}, \boldsymbol{y}$ are unknown vectors with $n/2$-bit components and α, β are the (unknown) CRT coefficients relative to p and q. Lattice reduction can exploit such a hidden linear relationship as follows.

Using standard techniques [21,22], it is possible to compute a reduced basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\ell-1}\}$ of the lattice $\boldsymbol{v}^\perp \subset \mathbb{Z}^\ell$ of vectors orthogonal to $\boldsymbol{v}$ in $\mathbb{Z}^\ell$. In particular we get:

$$\alpha \langle \boldsymbol{b}_j, \boldsymbol{x} \rangle + \beta \langle \boldsymbol{b}_j, \boldsymbol{y} \rangle = 0 \quad \text{for } j = 1, 2, \ldots, \ell - 1.$$

Now, observe that the smallest nonzero solution $(u, v) \in \mathbb{Z}^2$ of the equation $\alpha \cdot u + \beta \cdot v = 0$ is $\pm(\beta, -\alpha)/g$, where $g = \gcd(\alpha, \beta)$ is heuristically expected to be very small, which implies that $|u|, |v| \geq \Omega(N)$ where the $\Omega()$ constant is very small. For each $j = 1, 2, \ldots, \ell - 1$, there are thus two possibilities:

Case 1: $\langle \boldsymbol{b}_j, \boldsymbol{x}\rangle = \langle \boldsymbol{b}_j, \boldsymbol{y}\rangle = 0$, in which case $\boldsymbol{b}_j$ belongs to the lattice $L = \{\boldsymbol{x}, \boldsymbol{y}\}^\perp$ of vectors in $\mathbb{Z}^\ell$ orthogonal to both $\boldsymbol{x}$ and $\boldsymbol{y}$;

Case 2: $\langle \boldsymbol{b}_j, \boldsymbol{x}\rangle$ and $\langle \boldsymbol{b}_j, \boldsymbol{y}\rangle$ have absolute value $\geq \Omega(N)$ with a very small $\Omega()$ constant. Since $\boldsymbol{x}, \boldsymbol{y}$ both have norm at most $\sqrt{\ell N}$, this implies $\|\boldsymbol{b}_j\| \geq \Omega(\sqrt{N/\ell})$ by Cauchy-Schwarz.

Since the lattice $L = \{\boldsymbol{x}, \boldsymbol{y}\}^\perp$ is of rank $\ell - 2$, Case 1 cannot hold for all $\ell - 1$ linearly independent vectors $\boldsymbol{b}_j$, so that the longest one $\boldsymbol{b}_{\ell-1}$ should be in Case 2, and hence $\|\boldsymbol{b}_{\ell-1}\| \geq \Omega(\sqrt{N/\ell})$. On the other hand, the other vectors form a lattice of rank $\ell - 2$ and volume:

$$V = \mathrm{vol}(\mathbb{Z}\boldsymbol{b}_1 \oplus \cdots \oplus \mathbb{Z}\boldsymbol{b}_{\ell-2}) \approx \frac{\mathrm{vol}(\boldsymbol{v}^\perp)}{\|\boldsymbol{b}_{\ell-1}\|} = \frac{\|\boldsymbol{v}\|}{\|\boldsymbol{b}_{\ell-1}\|} \leq \frac{\sqrt{\ell} \cdot N^{3/2}}{\Omega(\sqrt{N/\ell})} = O(\ell N)$$

which can heuristically be expected to behave like a random lattice. In particular, we should have:

$$\|\boldsymbol{b}_j\| = O(\sqrt{\ell - 2} \cdot V^{1/(\ell-2)}) = O(\ell^{1/2+1/(\ell-2)} \cdot N^{1/(\ell-2)}) \quad \text{for } j = 1, 2, \ldots, \ell - 2.$$

This length is much smaller than $\sqrt{N/\ell}$ as soon as $\ell \geq 5$. Assuming that this is case, $\boldsymbol{b}_j$ should thus be in Case 1 for $j = 1, 2, \ldots, \ell - 2$. This means that those vectors generate a sublattice $L' = \mathbb{Z}\boldsymbol{b}_1 \oplus \cdots \oplus \mathbb{Z}\boldsymbol{b}_{\ell-2}$ of full rank in $L = \{\boldsymbol{x}, \boldsymbol{y}\}^\perp$.

Taking orthogonal lattices, we get $(L')^\perp \supset L^\perp = \mathbb{Z}\boldsymbol{x} \oplus \mathbb{Z}\boldsymbol{y}$. Therefore, $\boldsymbol{x}$ and $\boldsymbol{y}$ belong to the orthogonal lattice $(L')^\perp$ of L'. Let $\{\boldsymbol{x'}, \boldsymbol{y'}\}$ be a reduced basis of that lattice. We can enumerate all the lattice vectors in $(L')^\perp$ of length at most $\sqrt{\ell N}$ as linear combinations of $\boldsymbol{x'}$ and $\boldsymbol{y'}$. The Gaussian heuristic suggests that there should be roughly:

$$\frac{\pi(\sqrt{\ell N})^2}{\mathrm{vol}((L')^\perp)} = \frac{\pi \ell N}{V} = O(1)$$

such vectors, so this is certainly feasible. For all those vectors $\boldsymbol{z}$, we can compute $\gcd(\boldsymbol{v} - \boldsymbol{z}, N)$. We will thus quickly find $\gcd(\boldsymbol{v} - \boldsymbol{x}, N)$ among them, since $\boldsymbol{x}$ is a vector of length $\leq \sqrt{\ell N}$ in $(L')^\perp$. But by definition of $\boldsymbol{v}$ we have:

$$\boldsymbol{v} = \boldsymbol{x} \mod p \quad \text{and} \quad \boldsymbol{v} = \boldsymbol{y} \mod q$$

so $\gcd(\boldsymbol{v} - \boldsymbol{x}, N) = p$, which reveals the factorization of N.

2.3 Attack Summary

Assume that, for $\ell \geq 5$ padded messages $\mu(m_i)$, we know a correct signature σ_i and a signature σ'_i computed with a faulty modulus N'_i. Then, we can heuristically recover the factorization of N as follows.

1. For each i, compute the integer $v_i = \mathrm{CRT}_{N,N'_i}(\sigma_i, \sigma'_i)$. They form a vector $\boldsymbol{v} = (v_1, \ldots, v_\ell) \in \mathbb{Z}^\ell$.
2. Compute an LLL-reduced [17] basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\ell-1}\}$ of the lattice $\boldsymbol{v}^\perp \subset \mathbb{Z}^\ell$ of vectors in $\mathbb{Z}^\ell$ orthogonal to $\boldsymbol{v}$. This is done by applying LLL to the lattice in $\mathbb{Z}^{1+\ell}$ generated by the rows of the following matrix:

$$\begin{pmatrix} \kappa v_1 & 1 & & 0 \\ \vdots & & \ddots & \\ \kappa v_\ell & 0 & & 1 \end{pmatrix}$$

   where κ is a suitably large constant, and removing the first component of each resulting vector [21].
3. The first $\ell - 2$ vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\ell-2}$ generate a lattice $L' \subset \mathbb{Z}^\ell$ of rank $\ell - 2$. Compute an LLL-reduced basis $\{\boldsymbol{x'}, \boldsymbol{y'}\}$ of the orthogonal lattice $(L')^\perp$ to that lattice. Again, this is done by applying LLL to the lattice in $\mathbb{Z}^{\ell+2+\ell}$ generated by the rows of

$$\begin{pmatrix} \kappa' b_{1,1} & \cdots & \kappa' b_{\ell-2,1} & 1 & & 0 \\ \vdots & & \vdots & & \ddots & \\ \kappa' b_{1,\ell} & \cdots & \kappa' b_{\ell-2,\ell} & 0 & & 1 \end{pmatrix}$$

and keeping the last ℓ components of each resulting vector.

4. Enumerate the vectors $\boldsymbol{z} = a\boldsymbol{x'} + b\boldsymbol{y'} \in (L')^{\perp}$ of length at most $\sqrt{\ell N}$, and for each such vector $\boldsymbol{z}$, compute $\gcd(\boldsymbol{v} - \boldsymbol{z}, N)$ using all components, and return any nontrivial factor of N.

2.4 Simulation Results

Since the attack is heuristic, it is important to evaluate its experimental performances. To do so, we have implemented a simulation of the attack in SAGE [29]: for a given modulus N, we compute the vector $\boldsymbol{v}$ corresponding to a series of ℓ signatures on random messages and apply the lattice attack, attempting to recover a factor of N.

Table 1 shows the measured success probabilities for various values of ℓ and modulus sizes. It confirms the heuristic prediction that 5 faulty signatures should always suffice to factor N. It turns out that even 4 signatures are enough in almost half the cases.

Experimental running times are given in Table 2. The whole attack takes a few dozen milliseconds on a standard PC. The number of vectors to test as part of the final exhaustive search step is about 20 in practice, which is done very quickly.

Number of faulty signatures ℓ	4	5	6
1024-bit moduli	48%	100%	100%
1536-bit moduli	45%	100%	100%
2048-bit moduli	46%	100%	100%

Table 1. Attack success probability as a function of the number of faulty signatures and the size of N. Each parameter set was tested with random faults on 500 random moduli of the given size.

Modulus size	1024	1536	2048
Average search space $\pi\ell N/V$	24	23	24
Average total CPU time	16 ms	26 ms	34 ms

Table 2. Efficiency of the attack with $\ell = 5$ faulty signatures and various modulus sizes. Each parameter set was tested with random faults on 500 random moduli of the given size. Timings for a SAGE implementation, on a single 2.4 GHz Core2 CPU core.

3 Extending the Attack to Unknown Faulty Moduli

As mentioned in §2.1, in its basic form, the attack requires the recovery of the faulty moduli N'_i in addition to the corresponding faulty signatures σ'_i. This is not a very realistic assumption, since a typical implementation does not output the public modulus along with each signature.

To work around this limitation, we would like to reconstruct the vector $\boldsymbol{v}$ of integer values needed to run the attack from signatures alone, without the knowledge of the faulty moduli—possibly at the cost of requiring a few more faulty signatures.

This can actually be achieved in various ways depending on the precise form of the faults inflicted to the modulus. We propose solutions for the following two realistic fault models:

1. The faulty moduli N'_i differ from N on a single (unknown) byte. This is known to be possible using power glitches or laser shots.
2. The differences between the faulty moduli N'_i and N are located on the least significant half: the errors on the least significant bits can be up to half of the modulus size. It is easy to obtain such faults with a laser or a cold boot attack.

3.1 Single Byte Faults

In this model, the attacker is able to obtain a certain number $\ell' \geq 5$ of pairs (σ_i, σ'_i) where $\sigma_i = \alpha x_i + \beta y_i \bmod N$ is a valid signature and $\sigma'_i = \alpha x_i + \beta y_i \bmod N'_i$ is the same signature computed with a faulty modulus. The faulty moduli N'_i are not known, but they only differ from N on a single byte whose position and value is unknown.

This type of fault can for example occur when attacking the transfer of the modulus to memory on a smart card with an 8-bit processor, or when using a laser attack with a sufficiently focused beam.

For a 1024-bit modulus N, for example, there are $128 \times 255 \approx 2^{15}$ possible faulty moduli. It can thus seem like a reasonable approach to try and run the attack with all possible faults. However, since this should be done with 5 signatures, this results in a search space of size $\approx (2^{15})^5 = 2^{75}$ which is prohibitive.

This kind of exhaustive search can be made practical, though, if we take into account the fact that the CRT value $v_i = \mathrm{CRT}_{N,N'_i}(\sigma_i, \sigma'_i)$ satisfies:

$$v_i = \alpha x_i + \beta y_i \leq N \cdot (p+q) = N^{3/2}\left(\sqrt{\frac{p}{q}} + \sqrt{\frac{q}{p}}\right) < (2N)^{3/2}$$

since $p/q \in (1/2, 2)$. Now, for a given value of σ'_i, there are only very few possible target moduli N^*_i differing from N on a single byte such that $v^*_i = \mathrm{CRT}_{N,N^*_i}(\sigma_i, \sigma'_i) < (2N)^{3/2}$: often only one or two, and almost never more than 20. We only need to run the attack with those target v^*_i's until we find a factor.

Experimentally, for a 1024-bit modulus, the average base 2 logarithm of the number of possible v^*_i's is about 2.5, so if an attacker has 5 pairs (σ_i, σ'_i) in this model, they can expect to try all vectors $\boldsymbol{v}$ in a search space of around 12.5 bits, *i.e.* run the attack a few thousand times, for a total running time of under 2 minutes. This is already quite practical.

If more pairs are available, the attacker can keep the 5 pairs for which the number of possible v^*_i's is the smallest. This reduces the search space accordingly. In Table 3, we show how the exhaustive search space size and the expected running time evolve with the number of signatures in a typical example.

Number of pairs ℓ'	5	7	10	15	20	25
Search space size (bits)	11.6	9.8	7.2	6.2	4.2	2.6
Total attack time (seconds)	49	14	2.4	1.2	0.29	0.10

Table 3. Exhaustive search space size for the vector $\boldsymbol{v}$ of CRT values, and expected attack running time, depending on the number of pairs (σ_i, σ'_i) available to the attacker. Measured for a family of random single byte faults on a 1024-bit modulus. Timings are given for the SAGE implementation as above.

3.2 Faults on Many Least Significant Bits

In this model, the attacker is able to obtain $\ell = 5$ signature families of the form $(\sigma_i, \sigma'_{i,1}, \ldots, \sigma'_{i,k})$, where the σ_i's are correct signatures:

$$\sigma_i = \alpha x_i + \beta y_i \bmod N$$

and the $\sigma'_{i,j}$'s are faulty signatures of the form:

$$\sigma'_{i,j} = \alpha x_i + \beta y_i \bmod N'_{i,j} \quad 1 \le i \le \ell,\ 1 \le j \le k.$$

In other words, for each one of the ℓ different messages, the attacker learns the reduction of the CRT value $v_i = \alpha x_i + \beta y_i$ modulo N, as well as modulo k different unknown faulty moduli $N'_{i,j}$. Additionally, it is assumed that all $N'_{i,j}$ differ from N only on the least significant bits, but the number of distinct bits can be as large as half of the modulus size: we assume that$|N - N'_{i,j}| < N^\delta$ for a certain constant $\delta < 1/2$.

This is a reasonable fault model for a laser attack: it suffices to target a laser beam on the least significant bits of N to produce this type of faults.

To run the attack successfully, the attacker needs to recover the CRT values v_i. This can be done with high probability when the number of available faults k for a given message is large enough. The simplest approach is based on a GCD computation.

Indeed, fix an index $i \in \{1, \ldots, \ell\}$, and write $N'_{i,j} = N + \varepsilon_j$, $v_i = u$, $\sigma_i = u_0$ and $\sigma'_{i,j} = u_j$. The attacker knows the u_j's and wants to recover u.

Now, observe that there are integers t_j such that u satisfies $u = u_0 + t_0 \cdot N$ and $u = u_j + t_j \cdot (N + \varepsilon_j)$. In particular, for $j = 1, \ldots, k$ we can write:

$$(t_j - t_0) \cdot N + (u_j - u_0) + t_j \cdot \varepsilon_j = 0. \tag{3}$$

This implies that $u_j - u_0 \equiv t_j \cdot \varepsilon_j \pmod N$. However, we have $t_j \cdot \varepsilon_j < N^{1/2+\delta} \ll N$, so that the congruence is really an equality in $\mathbb{Z}$. In view of (3), this implies that all t_j's are in fact equal, and hence:

$$t_0 \cdot \varepsilon_j = u_0 - u_j \quad 1 \le j \le k.$$

If the errors ε_j on the modulus are co-prime, which we expect to happen with probability $\approx 1/\zeta(k)$, we can then deduce t_0 as the GCD of all values $u_0 - u_j$, and this gives:

$$u = u_0 + t_0 \cdot N = u_0 + N \cdot \gcd(u_0 - u_1, \ldots, u_0 - u_k).$$

As seen in Table 4, the success probability is in practice very close to $1/\zeta(k)$ regardless of the size of errors.

It is probably possible to further improve the success probability by trying to remove small factors from the computed GCD $g = \gcd(u_0 - u_1, \ldots, u_0 - u_k)$ to find t_0 when $g > \sqrt{N}$, but we find that the number of required faults is already reasonable without this computational refinement.

Indeed, recall that $\ell = 5$ CRT values are required to run the attack. If k faults are obtained for each of the ℓ messages, the probability that these ℓ CRT values can be successfully recovered with this GCD approach is $\zeta(k)^{-\ell}$. This is greater than 95% for $k = 7$, and 99% for $k = 9$.

k (faults per message)	3	5	7	9
$1/\zeta(k)$	.832	.964	.992	.998
100-bit errors	83.2%	96.8%	99.0%	99.8%
200-bit errors	83.4%	96.2%	99.2%	99.8%
400-bit errors	82.7%	96.6%	99.1%	99.8%
Average CPU time	.73 ms	.75 ms	.79 ms	.85 ms

Table 4. Success probabilities of the GCD method for CRT value recovery, depending on the number of available faults on a given message. Tested with random 1024-bit moduli. In the simulation, errors ε_j are modeled as uniformly random signed integers of the given size, and 10,000 of them were generated for each parameter set.

We can also mention an alternate, lattice-based approach to recovering the CRT value u. The relation between the different quantities above can be written in vector form as:

$$u_0\mathbf{1} = \boldsymbol{u} + t_0\boldsymbol{e}$$

where $\mathbf{1} = (1, \ldots, 1)$, $\boldsymbol{u} = (u_1, \ldots, u_k)$ and $\boldsymbol{e} = (\varepsilon_1, \ldots, \varepsilon_k)$.

Then, since $u_0 \approx N$ is much larger than $\|t_0\boldsymbol{e}\| \approx N^{1/2+\delta}$, short vectors orthogonal to $\boldsymbol{u}$ will be orthogonal to both $\mathbf{1}$ and $\boldsymbol{e}$. More precisely, we can heuristically expect that when k is large enough ($k \gtrsim 2/(1-2\delta)$), the first $k-2$ vectors of a reduced basis of $\boldsymbol{u}^\perp$ will be orthogonal to $\mathbf{1}$ and $\boldsymbol{e}$.

Taking orthogonal lattices again, we can thus obtain a reduced basis $\{\boldsymbol{x}, \boldsymbol{y}\}$ of a two-dimensional lattice containing $\mathbf{1}$ and $\boldsymbol{e}$ (and of course $\boldsymbol{u}$). Since $\mathbf{1}$ is really short, we always find that $\boldsymbol{x} = \mathbf{1}$ in practice. Then, it happens quite often that $\boldsymbol{y}$ can be written as $\lambda\mathbf{1} \pm \boldsymbol{e}$, in which case t_0 is readily recovered as the absolute value of the second coordinate of $\boldsymbol{u}$ in the basis $\{\boldsymbol{x}, \boldsymbol{y}\}$.

However, this fails when $\mathbb{Z}\mathbf{1} \oplus \mathbb{Z}\boldsymbol{e}$ is a proper sublattice of $\mathbb{Z}\boldsymbol{x} \oplus \mathbb{Z}\boldsymbol{y} = \mathbb{Z}^k \cap (\mathbb{Q}\mathbf{1} \oplus \mathbb{Q}\boldsymbol{e})$, namely, when there is some integer $d > 1$ such that all errors ε_j are congruent mod d. Thus, we expect the success probability of this alternate approach to be $1/\zeta(k-1)$, which is slightly less than with the GCD approach.

4 Practical Experiments

Practical experiments for validating the new attack were done on an 8-bit 0.35μm RISC microcontroller with no countermeasures. As the microprocessor had no arithmetic coprocessor the values σ_p and σ_q were pre-computed by an external program upon each fault-injection experience and fed into the attacked device. The target combined σ_p and σ_q using multiplications and additions (using Formula 1) as well as the final modular reduction.

The location and spread of the faults were controlled by careful beam-size and shot-instant tuning. The reader is referred to Appendix B for a description of the physical setting (common to the experiments reported in [18]).

We conducted several practical experiments corresponding to three different scenarios, roughly corresponding to the fault models considered in §2.1, §3.1 and §3.2 respectively. Let us describe these experiments in order.

4.1 First Scenario: Known Modulus

In this case, we considered 5 messages for a random 1024-bit RSA modulus N. For each message m_i, we obtained a correct signature σ_i, as well as a faulty-modulus signature σ'_i where the faulty modulus N'_i was also read back from the microcontroller.

Therefore, we were exactly in the setting described in §2.1, and could apply the algorithm from §2.3 directly: apply the Chinese Remainder Theorem to construct the vector $\boldsymbol{v}$ of CRT values and run the lattice-based attack to recover a factor of N.

The implementation of the attack used the same SAGE code as the simulation from §2.4. In our experimental case, the ball of radius $\sqrt{N\ell}$ contained only about 10 vectors of the double orthogonal lattice, and the whole attack revealed a factor of N in less than 20 milliseconds.

4.2 Second Scenario: Unknown Single Byte Fault

In this case, we tried to replicate a setting similar to the one considered in §3.1. We considered 20 messages and a random 1024-bit RSA modulus N. For each message m_i, we obtained a correct signature σ_i, as well as faulty-modulus signatures σ'_i with undisclosed faulty modulus N'_i generated by targeting a single byte of N with the laser.

We had to eliminate some signatures, however, because in some cases, errors on the modulus turned out to exceed 8 bits.[4] After discarding those, we had 12 pairs (σ_i, σ'_i) left to carry out the approach described in §3.1.

The first step in this approach is to find, for each i, all values v_i^* of the form $\mathrm{CRT}_{N,N_i^*}(\sigma_i, \sigma'_i)$ (N_i^* differing from N only on one byte) that are small enough to be correct candidate CRT values. Unlike the setting of §3.1, we could not assume that bit-differences were aligned on byte boundaries: we had to test a whole 1016×255 candidate moduli[5] N_i^* for each i . Therefore, this search step was a bit costly, taking a total of 11 minutes and 13 seconds. Additionally, due to the higher number of candidate moduli, the number of candidate CRT values v_i^* was also somewhat larger than in §3.1, namely:

$$7, 17, 3, 9, 15, 5, 14, 44, 44, 17, 10, 55$$

for our 12 pairs respectively. Keeping only the 5 indices with the smallest number of candidates, we obtained $3 \times 5 \times 7 \times 9 \times 10 = 9450$ possible CRT value vectors $\boldsymbol{v}^*$.

We then ran the lattice-based attack on each of these vectors in order until a factor of N was found. The factor was found at iteration number 2120, after a total computation time of 43 seconds.

4.3 Third Scenario: Unknown Least Significant Bytes Faults

In this case, we considered 10 messages for a random 1024-bit N. For each message m_i, we obtained a correct signature σ_i, as well as 10 faulty-modulus signature $\sigma'_{i,j}$ with undisclosed faulty modulus N'_i. The laser beam targeted the lower order bytes of N but with a large aperture, generating multiple faults stretching over as much as 448 modulus bits.

In practice, we only used the data $(\sigma_i, \sigma'_{i,1}, \ldots, \sigma'_{i,10})$ for the first 5 messages, discarding the rest. Then, we reconstructed the CRT values v_i using the GCD technique of §3.2:

$$v_i = \sigma_i + N \cdot \gcd(\sigma_i - \sigma'_{i,1}, \ldots, \sigma_i - \sigma'_{i,10}) \quad 1 \leq i \leq 5$$

and applied the lattice-based attack on the resulting vector $\boldsymbol{v}$. This revealed a factor of N in 16 milliseconds.

We also tried the same attack using a fewer number of the $\sigma'_{i,j}$'s, and found that it still worked when taking only 4 of those values in the computation of v_i:

$$v_i = \sigma_i + N \cdot \gcd(\sigma_i - \sigma'_{i,1}, \ldots, \sigma_i - \sigma'_{i,4}) \quad 1 \leq i \leq 5$$

but failed if we took 3 instead. Considering that $1/\zeta(3)^5 \approx .40$ and $1/\zeta(4)^5 \approx .67$, this is quite in line with expectations.

5 Countermeasures and Further Research

Probabilistic and stateful signature schemes are usually secure against this attack, since they make it difficult to obtain two signatures on the same padded message. However, all deterministic schemes are typically vulnerable, including those in which the attacker doesn't have full access to the signed message, provided that the target device can be forced to compute the same signature twice.

A natural countermeasure is to use a CRT interpolation formula that does not require N, such as Garner's formula, computed as follows:

[4] Note that in a real-world attack, it might not be possible to detect such overly spread out faults: hence, this particular technique should be used preferably when faults are *known* to affect only single bytes (*e.g.* in a glitch attack), whereas the technique from the next section is better suited to laser attacks as aperture control is much less of an issue.

[5] There are duplicates among those, corresponding to perturbations of 7 consecutive bits or less, but we did not attempt to avoid testing them several times, as this can only improve the search by a small constant factor while introducing significant complexity in the code.

$$\boxed{\begin{array}{l} t \leftarrow \sigma_p - \sigma_q \\ \text{if } t < 0 \text{ then } t \leftarrow t + p \\ \sigma \leftarrow \sigma_q + (t \cdot \gamma \bmod p) \cdot q \\ \text{return}(\sigma) \end{array}}$$

where we assume that $p > q$, and γ is the usual CRT coefficient $q^{-1} \bmod p$. Note that the evaluation of σ does not require a modular reduction because

$$\sigma = \sigma_q + (t \cdot \gamma \bmod p) \cdot q \leq q - 1 + (p-1)q < N$$

Besides the obvious countermeasures consisting in checking signatures before release, it would be interesting to devise specific countermeasures for protecting Formula (1) (or Garner's formula) taking into account the possible corruption of all data involved.

Finally, in a number of special cases and particular settings (*e.g.* Appendix A) other fault attacks on the CRT recombination phase can be devised. A thorough analysis of such scenarios is also an interesting research direction.

Acknowledgments

We would like to thank the anonymous referees for helpful comments. The work described in this paper has been supported in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II.

References

1. 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., Ç. K. Koç, and Christof Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 260–275. Springer, 2002.
2. H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. The sorcerer's apprentice guide to fault attacks. *Proc. IEEE*, 94(2):370–382, 2006.
3. M. Bellare and P. Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In *EUROCRYPT*, pages 399–416, 1996.
4. A. Berzati, C. Canovas, J.-G. Dumas, and L. Goubin. Fault attacks on RSA public keys: Left-to-right implementations are also vulnerable. In M. Fischlin, editor, *CT-RSA*, volume 5473 of *Lecture Notes in Computer Science*, pages 414–428. Springer, 2009.
5. A. Berzati, C. Canovas, and L. Goubin. Public key perturbation of randomized RSA implementations. In S. Mangard and F.-X. Standaert, editors, *CHES*, volume 6225 of *Lecture Notes in Computer Science*, pages 306–319. Springer, 2010.
6. A. Berzati, C. Canovas, and L. Goubin. Perturbating RSA public keys: An improved attack. In E. Oswald and P. Rohatgi, editors, *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 380–395. Springer, 2008.
7. D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of eliminating errors in cryptographic computations. *J. Cryptology*, 14(2):101–119, 2001.
8. A. Boscher, R. Naciri, and E. Prouff. CRT-RSA algorithm protected against fault attacks. In D. Sauveron, C. Markantonakis, A. Bilas, and J.-J. Quisquater, editors, *WISTP*, volume 4462 of *Lecture Notes in Computer Science*, pages 229–243. Springer, 2007.
9. E. Brier, B. Chevallier-Mames, M. Ciet, and C. Clavier. Why one should also secure RSA public key elements. In L. Goubin and M. Matsui, editors, *CHES*, volume 4249 of *Lecture Notes in Computer Science*, pages 324–338. Springer, 2006.

10. M. Brstle et al. *SOSSE: Simple Operating System for Smartcard Education*, 2003. http://www.mbsks.franken.de/sosse/.

Can09. G. Canivet. *Analyse des effets d'attaques par fautes et conception sécurisée sur plate-forme reconfigurable*. PhD thesis, Institut polytechnique de Grenoble, 2009.

11. 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.

12. D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *J. Cryptology*, 10(4):233–260, 1997.

13. J.-S. Coron, A. Joux, I. Kizhvatov, D. Naccache, and P. Paillier. Fault attacks on RSA signatures with partially unknown messages. In C. Clavier and K. Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 444–456. Springer, 2009.

14. J.-S. Coron. Optimal security proofs for PSS and other signature schemes. In L. R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 272–287. Springer, 2002.

15. F. Darracq, T. Beauchene, V. Pouget, H. Lapuyade, D. Lewis, P. Fouillat, and A. Touboul. Single-event sensitivity of a single SRAM cell. *IEEE Trans. Nuclear Sci.*, 49:1486–1490, 2002.

16. C. Giraud. An RSA implementation resistant to fault attacks and to simple power analysis. *IEEE Trans. Computers*, 55(9):1116–1120, 2006.

17. A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.

18. A.-P. Mirbaha, J. M. Dutertre, A. Tria, M. Agoyan, A.-L. Ribotta, and D. Naccache. Study of single-bit fault injection techniques by laser on an AES cryptosystem. In D. Gizopoulos and A. Chatterjee, editors, *IOLTS*, 2010.

19. J. A. Muir. Seifert's RSA fault attack: Simplified analysis and generalizations. In P. Ning, S. Qing, and N. Li, editors, *ICICS*, volume 4307 of *Lecture Notes in Computer Science*, pages 420–434. Springer, 2006.

20. P. Q. Nguyen. Public-key cryptanalysis. In I. Luengo, editor, *Recent Trends in Cryptography*, volume 477 of *Contemporary Mathematics*. AMS–RSME, 2009.

21. P. Q. Nguyen and J. Stern. Merkle-Hellman revisited: A cryptoanalysis of the Qu-Vanstone cryptosystem based on group factorizations. In B. S. Kaliski Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 198–212. Springer, 1997.

22. P. Q. Nguyen and J. Stern. Cryptanalysis of a fast public key cryptosystem presented at SAC '97. In S. E. Tavares and H. Meijer, editors, *Selected Areas in Cryptography*, volume 1556 of *Lecture Notes in Computer Science*, pages 213–218. Springer, 1998.

23. P. Q. Nguyen and J. Stern. The two faces of lattices in cryptology. In J. H. Silverman, editor, *CaLC*, volume 2146 of *Lecture Notes in Computer Science*, pages 146–180. Springer, 2001.

24. V. Pouget. Test et analyse par faisceau laser: Plateforme et applications. Technical report, Journée thématique du GDR SOC-SIP, 2007. http://www.lirmm.fr/soc_sip/6fev/GCT_R1_Pouget.pdf.

25. M. Rivain. Securing RSA against fault analysis by double addition chain exponentiation. In M. Fischlin, editor, *CT-RSA*, volume 5473 of *Lecture Notes in Computer Science*, pages 459–480. Springer, 2009.

26. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

27. J.-P. Seifert. On authenticated computing and rsa-based authentication. In V. Atluri, C. Meadows, and A. Juels, editors, *ACM Conference on Computer and Communications Security*, pages 122–127. ACM, 2005.

28. S. P. Skorobogatov and R. J. Anderson. Optical fault induction attacks. In B. S. Kaliski Jr., Ç. K. Koç, and Christof Paar, editors, *CHES*, volume 2523 of *Lecture Notes in Computer Science*, pages 2–12. Springer, 2002.

29. W. A. Stein et al. *Sage Mathematics Software (Version 4.4.2)*. The Sage Development Team, 2010. http://www.sagemath.org.

30. D. Vigilant. RSA with CRT: A new cost-effective solution to thwart fault attacks. In E. Oswald and P. Rohatgi, editors, *CHES*, volume 5154 of *Lecture Notes in Computer Science*, pages 130–145. Springer, 2008.

A Using Dichotomy in the Absence of Padding

Consider again the setting of §2.1, in which an adversary is able to obtain both a correct signature σ on a message m, and a signature on the same message m computed with a faulty modulus, allowing him to deduce the non reduced value $v = \sigma_p \cdot \alpha + \sigma_q \cdot \beta \in \mathbb{Z}$. We can write:

$$v = (\sigma \bmod p) \cdot \alpha + (\sigma \bmod q) \cdot \beta = \left(\sigma - p \left\lfloor \frac{\sigma}{p} \right\rfloor\right) \cdot \alpha + \left(\sigma - q \left\lfloor \frac{\sigma}{q} \right\rfloor\right) \cdot \beta$$

Moreover, observe that $\alpha + \beta = N + 1$ (as is easily seen by reducing $\alpha + \beta$ modulo p and q). Therefore, we have:

$$v = \sigma \cdot (N + 1) - p\alpha \left\lfloor \frac{\sigma}{p} \right\rfloor - q\beta \left\lfloor \frac{\sigma}{q} \right\rfloor$$

Hence, if we let $\omega = (\sigma \cdot (N + 1) - v)/N$, we get:

$$\omega = \frac{\sigma \cdot (N + 1) - v}{N} = \frac{\alpha}{q} \left\lfloor \frac{\sigma}{p} \right\rfloor + \frac{\beta}{p} \left\lfloor \frac{\sigma}{q} \right\rfloor \tag{4}$$

and this value ω is an integer since $v \equiv \sigma \pmod{N}$.

Now assume further that the adversary can ask signatures on messages m such that σ is small. This is the case, for example, when signatures are computed without padding and the physical device under consideration will answer arbitrary signature queries: then, the adversary can simply ask signatures on messages of the form σ^e for small values σ of his choice.

In such a setting, the adversary can pick a σ close to $N^{1/2}$, carry out the fault attack and compute the integer ω. By (4), he gets $\omega = 0$ if $\sigma < \min(p, q)$ and $\omega > 0$ otherwise. Trying this process again several times, the smallest prime factor of N can be recovered by dichotomy.

B Laser Fault Injection

Laser (Light Amplification by Stimulated Emission of Radiation) is a stimulated-emission electromagnetic radiation in the visible or the invisible domain. Laser light is monochromatic, unidirectional, coherent and artificial (*i.e.* laser does not spontaneously exist in nature). Laser light can be generated as a beam of very small diameter (a few μm). The beam can pass through various material obstacles before impacting a target during a very short duration.

Laser impacts on electronic circuits are known to alter functioning. Current chip manufacturing technologies are in the nanometers range. This, and the laser's brief and precise reaction time, makes laser a particularly suitable fault injection means.

B.1 Photoelectric Effects of Laser on Silicon.

SRAM (Static Random Access Memory) laser exposure is known to cause bit-flips [28], [15], [2], [Can09], a phenomenon called *Single Event Upset* (SEU). By tuning the beam's energy level below a destructive threshold, the target will not suffer any permanent damage.

A conventional one-bit SRAM cell (Figure 1) is made of two cross-coupled inverters. Every cell has two additional transistors controlling the cell's content access during write and read. As every inverter is made of two transistors, an SRAM cell contains six MOS.

In each cell, the states of four transistors encode the stored value. By design, the cell admits only two stable states: a "0" or a "1". In each stable state, two transistors are at an ON state and two others are OFF.

If a laser beam hits the drain/bulk reversed-biased PN junctions of a blocked transistor, the beam's energy may create pairs of electrons as the beam passes through the silicon. The charge carriers induced

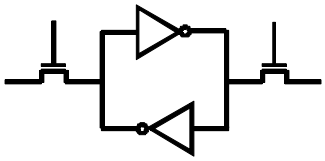

Fig. 1. Architecture of a typical SRAM cell.

in the collection volume of the drain-substrate junction of the blocked transistor are collected and create a transient current that inverts logically the inverter's output voltage. This voltage inversion is in turn applied to the second inverter that switches to its opposite state: all in all, a bit flip happens [15], [2].

From the opponent's perspective, an additional advantage of laser fault injection is *reproducibility*. Identical faults can be repeated by carefully tuning the laser's parameters and the target's operating conditions.

B.2 Different Parameters in a Fault Attack by Laser.

In a laser attack, the opponent usually controls the beam's diameter, wavelength, amount of emitted energy, impact coordinates (attacked circuit part) and the exposure's duration. Sometimes, the opponent may also control the impact's moment[6], the target's clock frequency, V_{cc} and temperature. Finally, laser attacks may indifferently target the chip's front side or back side.

However, the chip's front and back sides have different characteristics when exposed to a laser beam.

Front side attacks are particularly suited to green wavelength ($\sim$ 532nm). The visibility of chips components makes positioning very easy in comparison to backside attacks. But because of the metallic interconnects' reflective effect, it is difficult to target a component with enough accuracy. In addition, progress in manufacturing technologies results in both a proliferation of metal interconnects and much smaller chips. All in all, it becomes increasingly difficult to hit a target area.

Backside attacks are more successful at the infrared wavelength ($\sim$ 1064nm) as the laser needs to deeply enter the silicon. Positioning is more difficult for lack of visibility. Nonetheless, backside attacks allow to circumvent the reflective problem of metallic surfaces.

B.3 Practical CRT Fault Injection

Outline: After chip decapsulation and a mapping of the chip's components, we selected a large target area, given our knowledge of the implementation. Using automated search on the chip's front-side, we modified the impact's coordinates, the beam parameters and timing until a reproducible fault area was obtained.

The target is an 8-bit 0.35 μm 16 MHz RISC microcontroller with an integrated 4KB SRAM and no countermeasures. The device runs SOSSE (Simple Operating System for Smartcard Education [10]) to which we added some commands, most notably for feeding-in the data $\{N, p, q, \sigma_p, \sigma_q, p^{-1} \bmod q, q^{-1} \bmod p\}$. Upon reception all these parameters are stored in SRAM. The laser, shown in Figures 6 and 7, is equipped with a YAG laser emitter in three different wavelengths: green, infrared and ultraviolet.

The spot's diameter can be set between 0 and 2500μm. As the beam passes through a lens, it gets reduced by the lens' zoom factor and loses a big part of its energy. Our experiments were conducted with a 20$\times$ Mitutoyo lens, a green[7] beam of $\varnothing 4\mu$m and $\simeq$ 15pJ per shot[8]. The circuit is installed on a programmable

[6] *i.e.* the impact's synchrony with a given clock cycle of the target.
[7] 532nm wavelength.
[8] At the laser source emitter, before passing through the lens.

Prior Scientific X-Y positioning table[9]. The X-Y table, card reader, laser and an FPGA trigger board, were connected via RS-232 to a control PC. The FPGA trigger board receives an activation signal from the reader and sends a trigger signal to the laser after a delay defined by the control PC.

Experiments were conducted in ambient temperature and at $V_{cc} = 5V$. These parameters are within the device's normal operating conditions $2.7V \leq V_{cc} \leq 5.5V$.

The chip was decapsulated by chemical etching using a Nisene JetEtch automated acid decapsulator. The decapsulator can be programmed for the chemical opening of different chip types using different ratios of nitric acid (HNO_3) and sulfuric acid (H_2SO_4), at a desired temperature and during a specified time. For opening our chip, we used only nitric acid at 80°C for 40 seconds. The etched chip (Figure 2) successfully passed functional tests before and during fault injection.

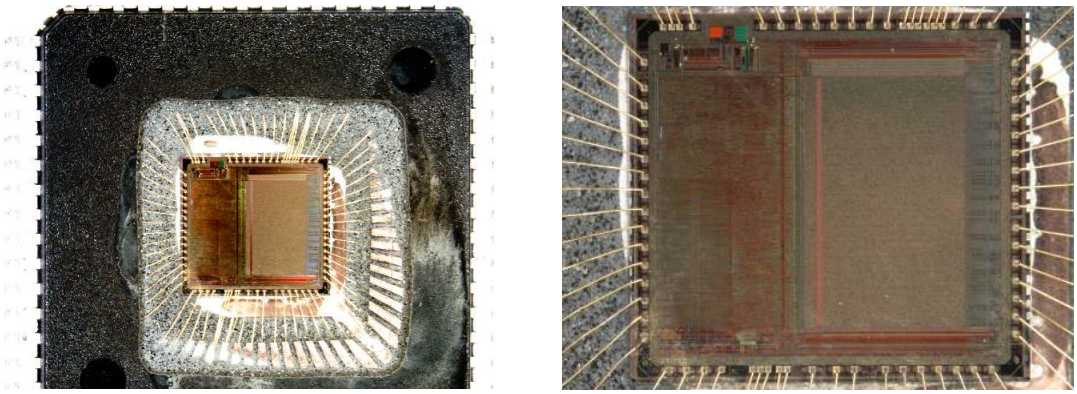

Fig. 2. Decapsulated chip (SRAM is on the left middle and bottom side).

Fig. 3. Decapsulated chip (closeup on SRAM).

As it is very difficult to target the chip's ALU (Arithmetic Logic Unit) during a very specific calculation step, we decided to hit the SRAM to corrupt data rather than modify calculations.

Finding the SRAM area containing each data element and properly tuning the laser's parameters is very time consuming. The number of faults in the read back data, their position and their contents indicate which element has been hit.

[9] Motorized stepper stage for upright microscopes with 0,1 μm steps.

Figure 5 compares a 1μm laser spot and SRAM cells in different technology sizes. As technology advances, transistor density per μm grows. With several transistors are packed into 1μm areas, single-bit fault injection will require much more precise equipment and are likely to become unfeasible using cheap lasers.

Figure 4 shows how we explored the target's SRAM's space. The method consisted in searching N's precise storage area, shooting into it and reading data back. Figure 4 is just a schematic model of the real SRAM (Figure 3) to describe our technique and does not correspond to real address allocation. We could successfully inject multiple byte faults into selected parts of N and iterate the process for different moduli and signatures. This sufficed to implement our scenario.

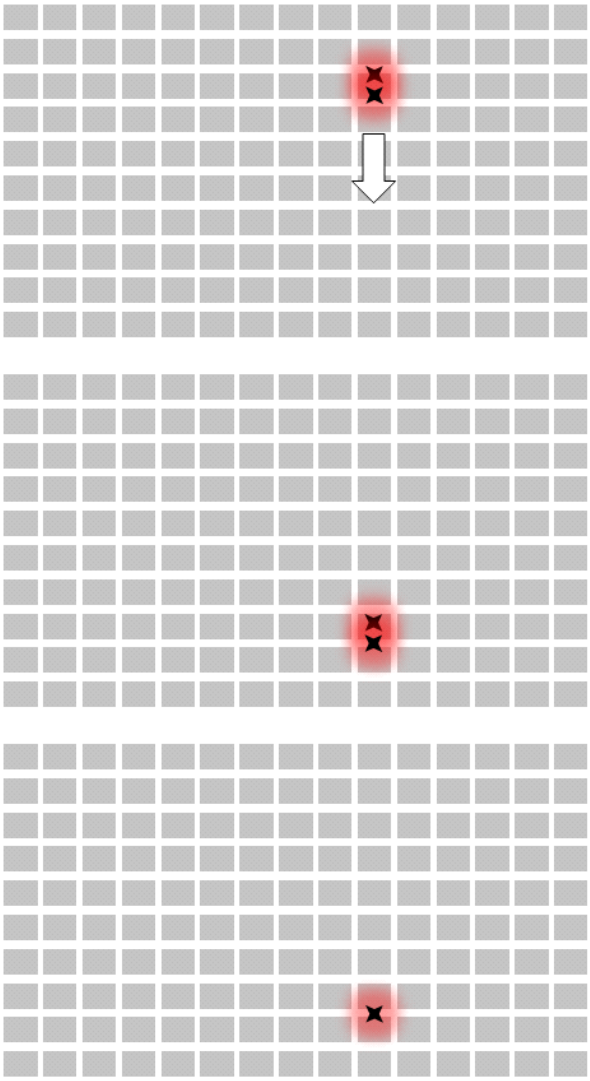

Fig. 4. Exploration process.

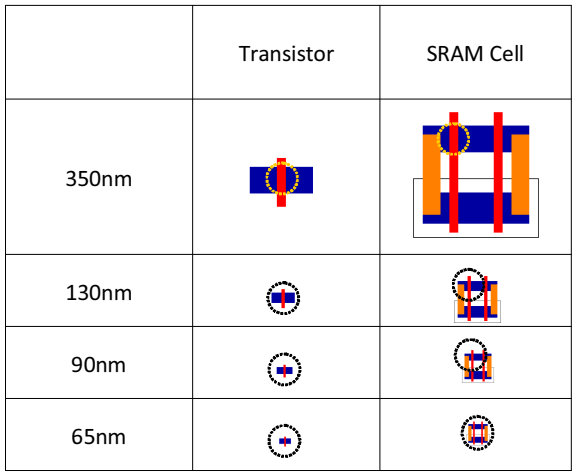

Fig. 5. 1μm laser spot (dotted circle) *vs.* technology sizes [24].

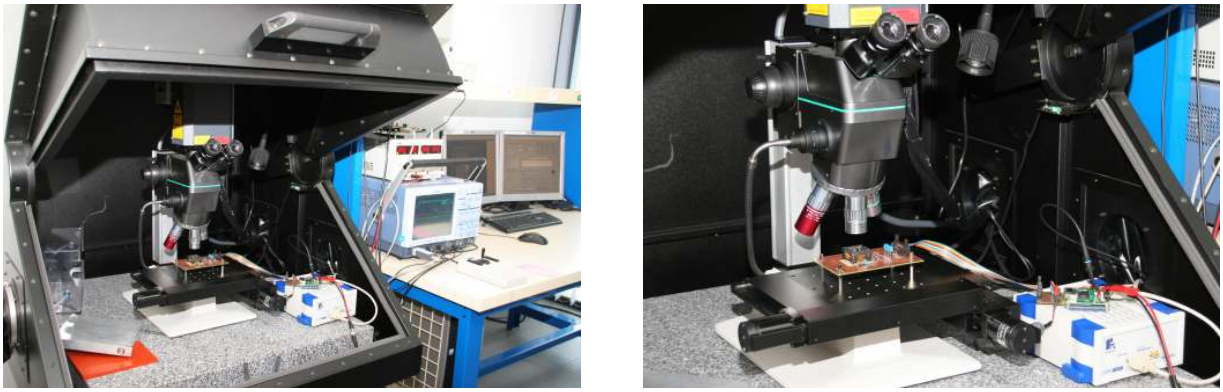

Fig. 6. Laser and target (general overview).

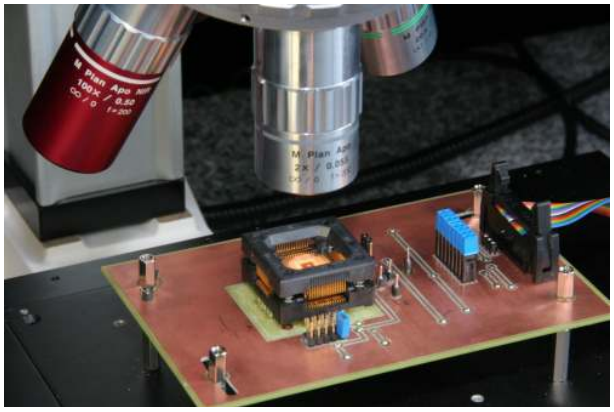

Fig. 7. Laser and target (closeup).