Modulus Fault Attacks Against RSA–CRT Signatures

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

[1]Ingenico

[2]École normale supérieure

[3]INRIA

CHES 2011, Nara, 2011–09–30

Outline

Signing with RSA–CRT

- RSA signatures:
$$\sigma = \mu(m)^d \bmod N$$
For suitable padding functions μ (e.g. FDH, PSS...) this is a provably secure signature scheme.
- Remains the most widely used signature scheme today. Implemented in many embedded applications (esp. smart cards).
- However, modular exponentiation is rather slow.
- Very commonly used improvement: using the Chinese Remainder Theorem.
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma_q = \mu(m)^{d \bmod q-1} \bmod q$
  3. $\sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N$
- Roughly 4-fold speed-up.

Signing with RSA–CRT

- RSA signatures:

$$\sigma = \mu(m)^d \bmod N$$

  For suitable padding functions μ (e.g. FDH, PSS...) this is a provably secure signature scheme.

- Remains the most widely used signature scheme today. Implemented in many embedded applications (esp. smart cards).
- However, modular exponentiation is rather slow.
- Very commonly used improvement: using the Chinese Remainder Theorem.
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma_q = \mu(m)^{d \bmod q-1} \bmod q$
  3. $\sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N$
- Roughly 4-fold speed-up.

Signing with RSA–CRT

- RSA signatures:

$$\sigma = \mu(m)^d \bmod N$$

  For suitable padding functions μ (e.g. FDH, PSS...) this is a provably secure signature scheme.
- Remains the most widely used signature scheme today. Implemented in many embedded applications (esp. smart cards).
- However, modular exponentiation is rather slow.
- Very commonly used improvement: using the Chinese Remainder Theorem.
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma_q = \mu(m)^{d \bmod q-1} \bmod q$
  3. $\sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N$
- Roughly 4-fold speed-up.

Signing with RSA–CRT

- RSA signatures:

$$\sigma = \mu(m)^d \bmod N$$

  For suitable padding functions μ (e.g. FDH, PSS...) this is a provably secure signature scheme.

- Remains the most widely used signature scheme today. Implemented in many embedded applications (esp. smart cards).
- However, modular exponentiation is rather slow.
- Very commonly used improvement: using the Chinese Remainder Theorem.
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma_q = \mu(m)^{d \bmod q-1} \bmod q$
  3. $\sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N$
- Roughly 4-fold speed-up.

Signing with RSA–CRT

- RSA signatures:
$$\sigma = \mu(m)^d \bmod N$$
For suitable padding functions μ (e.g. FDH, PSS...) this is a provably secure signature scheme.
- Remains the most widely used signature scheme today. Implemented in many embedded applications (esp. smart cards).
- However, modular exponentiation is rather slow.
- Very commonly used improvement: using the Chinese Remainder Theorem.
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma_q = \mu(m)^{d \bmod q-1} \bmod q$
  3. $\sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N$
- Roughly 4-fold speed-up.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma'_q \neq \mu(m)^{d \bmod q-1} \bmod q$ ← fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma'_q) \bmod N$ ← faulty signature
- Then ${\sigma'}^e$ is $\mu(m) \bmod p$ but not mod q, so the attacker can then factor N:

$$p = \gcd({\sigma'}^e - \mu(m), N)$$

- This attack applies to any deterministic padding, including "provably secure" ones like FDH.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma_q' \neq \mu(m)^{d \bmod q-1} \bmod q$ $\leftarrow$ fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q') \bmod N$ $\leftarrow$ faulty signature
- Then σ'^e is $\mu(m) \bmod p$ but not mod q, so the attacker can then factor N:

$$p = \gcd(\sigma'^e - \mu(m), N)$$

- This attack applies to any deterministic padding, including “provably secure” ones like FDH.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma'_q \neq \mu(m)^{d \bmod q-1} \bmod q$ $\leftarrow$ fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma'_q) \bmod N$ $\leftarrow$ faulty signature
- Then σ'^e is $\mu(m) \bmod p$ but not mod q, so the attacker can then factor N:

$$p = \gcd(\sigma'^e - \mu(m), N)$$

- This attack applies to any deterministic padding, including "provably secure" ones like FDH.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma'_q \neq \mu(m)^{d \bmod q-1} \bmod q$ ← fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma'_q) \bmod N$ ← faulty signature
- Then σ'^e is $\mu(m) \bmod p$ but not mod q, so the attacker can then factor N:

$$p = \gcd(\sigma'^e - \mu(m), N)$$

- This attack applies to any deterministic padding, including "provably secure" ones like FDH.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma'_q \neq \mu(m)^{d \bmod q-1} \bmod q$ ← fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma'_q) \bmod N$ ← faulty signature
- Then σ'^e is $\mu(m) \bmod p$ but not mod q, so the attacker can then factor N:

$$p = \gcd(\sigma'^e - \mu(m), N)$$

- This attack applies to any deterministic padding, including "provably secure" ones like FDH.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma'_q \neq \mu(m)^{d \bmod q-1} \bmod q \quad \leftarrow$ fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma'_q) \bmod N \quad \leftarrow$ faulty signature
- Then σ'^e is $\mu(m)$ mod p but not mod q, so the attacker can then factor N:

$$p = \gcd(\sigma'^e - \mu(m), N)$$

- This attack applies to any deterministic padding, including "provably secure" ones like FDH.

The Boneh-DeMillo-Lipton fault attack (1997)

- The problem with CRT: fault attacks.
- A fault in signature generation makes it possible to recover the secret key!
  1. $\sigma_p = \mu(m)^{d \bmod p-1} \bmod p$
  2. $\sigma'_q \neq \mu(m)^{d \bmod q-1} \bmod q$ $\leftarrow$ fault
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma'_q) \bmod N$ $\leftarrow$ faulty signature
- Then ${\sigma'}^e$ is $\mu(m)$ mod p but not mod q, so the attacker can then factor N:
$$p = \gcd({\sigma'}^e - \mu(m), N)$$
- This attack applies to any deterministic padding, including "provably secure" ones like FDH.

Shamir's trick

- Faults against RSA–CRT signatures have been an active research subject since then. Many variants and countermeasures have been proposed.
- One simple countermeasure due to Shamir is to compute the signature as follows (r is a small fixed integer like $2^{31} - 1$):
  1. $\sigma_p^+ = \mu(m)^d \bmod r \cdot p$
  2. $\sigma_q^+ = \mu(m)^d \bmod r \cdot q$
  3. if $\sigma_p^+ \not\equiv \sigma_q^+ \pmod{r}$, abort
  4. $\sigma = \mathrm{CRT}(\sigma_p^+, \sigma_q^+) \bmod N$
- If one of the half-exponentiations is perturbed, signature generation is very likely to abort, and hence the fault attacker cannot factor anymore!

Shamir's trick

- Faults against RSA–CRT signatures have been an active research subject since then. Many variants and countermeasures have been proposed.
- One simple countermeasure due to Shamir is to compute the signature as follows (r is a small fixed integer like $2^{31} - 1$):
  1. $\sigma_p^+ = \mu(m)^d \bmod r \cdot p$
  2. $\sigma_q^+ = \mu(m)^d \bmod r \cdot q$
  3. if $\sigma_p^+ \not\equiv \sigma_q^+ \pmod{r}$, abort
  4. $\sigma = \mathrm{CRT}(\sigma_p^+, \sigma_q^+) \bmod N$
- If one of the half-exponentiations is perturbed, signature generation is very likely to abort, and hence the fault attacker cannot factor anymore!

Shamir's trick

- Faults against RSA–CRT signatures have been an active research subject since then. Many variants and countermeasures have been proposed.
- One simple countermeasure due to Shamir is to compute the signature as follows (r is a small fixed integer like $2^{31} - 1$):
  1. $\sigma_p^+ = \mu(m)^d \bmod r \cdot p$
  2. $\sigma_q^+ = \mu(m)^d \bmod r \cdot q$
  3. if $\sigma_p^+ \not\equiv \sigma_q^+ \pmod{r}$, abort
  4. $\sigma = \mathrm{CRT}(\sigma_p^+, \sigma_q^+) \bmod N$
- If one of the half-exponentiations is perturbed, signature generation is very likely to abort, and hence the fault attacker cannot factor anymore!

Outline

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p \quad \leftarrow$ correct
  2. $\sigma_q = \mu(m)^d \bmod q \quad \leftarrow$ correct
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N' \quad \leftarrow$ faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p \quad \leftarrow$ correct
  2. $\sigma_q = \mu(m)^d \bmod q \quad \leftarrow$ correct
  3. $\sigma' = \text{CRT}(\sigma_p, \sigma_q) \bmod N' \quad \leftarrow$ faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p$ ← correct
  2. $\sigma_q = \mu(m)^d \bmod q$ ← correct
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N'$ ← faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p$ ← correct
  2. $\sigma_q = \mu(m)^d \bmod q$ ← correct
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N'$ ← faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p$ ← correct
  2. $\sigma_q = \mu(m)^d \bmod q$ ← correct
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N'$ ← faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p$ ← correct
  2. $\sigma_q = \mu(m)^d \bmod q$ ← correct
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N'$ ← faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Attacking the modulus

- A lot of work has been invested into protecting the exponentiations in RSA–CRT signature generation.
- So what about attacking another part of the algorithm?
- Idea: attack the modular reduction instead!
  1. $\sigma_p = \mu(m)^d \bmod p$ ← correct
  2. $\sigma_q = \mu(m)^d \bmod q$ ← correct
  3. $\sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N'$ ← faulty signature: wrong modular reduction!
- This new, strange type of faults can also be used to factor N.

Using the fault (I)

- More precisely, suppose we can obtain the same signature on a certain message twice, once correctly and once with a fault. Then we get:

$$\begin{cases} \sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N & \leftarrow \text{correct} \\ \sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N' & \leftarrow \text{faulty} \end{cases}$$

- Applying the CRT to these two relations, we obtain the value $\mathrm{CRT}(\sigma_p, \sigma_q) \bmod NN'$.
- Now recall that:

$$\mathrm{CRT}(\sigma_p, \sigma_q) = \alpha \cdot \sigma_p + \beta \cdot \sigma_q$$

where

$$\alpha = q \cdot (q^{-1} \bmod p) \quad \beta = p \cdot (p^{-1} \bmod q)$$

- In particular, $\mathrm{CRT}(\sigma_p, \sigma_q)$ is an integer of size $\approx N^{3/2}$, so if we know it modulo $NN' \approx N^2$, we actually know its value in $\mathbb{Z}$.

Using the fault (I)

- More precisely, suppose we can obtain the same signature on a certain message twice, once correctly and once with a fault. Then we get:

$$\begin{cases} \sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N & \leftarrow \text{correct} \\ \sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N' & \leftarrow \text{faulty} \end{cases}$$

- Applying the CRT to these two relations, we obtain the value $\mathrm{CRT}(\sigma_p, \sigma_q) \bmod NN'$.
- Now recall that:

$$\mathrm{CRT}(\sigma_p, \sigma_q) = \alpha \cdot \sigma_p + \beta \cdot \sigma_q$$

where

$$\alpha = q \cdot (q^{-1} \bmod p) \quad \beta = p \cdot (p^{-1} \bmod q)$$

- In particular, $\mathrm{CRT}(\sigma_p, \sigma_q)$ is an integer of size $\approx N^{3/2}$, so if we know it modulo $NN' \approx N^2$, we actually know its value in $\mathbb{Z}$.

Using the fault (I)

- More precisely, suppose we can obtain the same signature on a certain message twice, once correctly and once with a fault. Then we get:

$$\begin{cases} \sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N & \leftarrow \text{correct} \\ \sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N' & \leftarrow \text{faulty} \end{cases}$$

- Applying the CRT to these two relations, we obtain the value $\mathrm{CRT}(\sigma_p, \sigma_q) \bmod NN'$.
- Now recall that:

$$\mathrm{CRT}(\sigma_p, \sigma_q) = \alpha \cdot \sigma_p + \beta \cdot \sigma_q$$

where

$$\alpha = q \cdot (q^{-1} \bmod p) \quad \beta = p \cdot (p^{-1} \bmod q)$$

- In particular, $\mathrm{CRT}(\sigma_p, \sigma_q)$ is an integer of size $\approx N^{3/2}$, so if we know it modulo $NN' \approx N^2$, we actually know its value in $\mathbb{Z}$.

Using the fault (I)

- More precisely, suppose we can obtain the same signature on a certain message twice, once correctly and once with a fault. Then we get:

$$\begin{cases} \sigma = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N & \leftarrow \text{correct} \\ \sigma' = \mathrm{CRT}(\sigma_p, \sigma_q) \bmod N' & \leftarrow \text{faulty} \end{cases}$$

- Applying the CRT to these two relations, we obtain the value $\mathrm{CRT}(\sigma_p, \sigma_q) \bmod NN'$.
- Now recall that:

$$\mathrm{CRT}(\sigma_p, \sigma_q) = \alpha \cdot \sigma_p + \beta \cdot \sigma_q$$

where

$$\alpha = q \cdot (q^{-1} \bmod p) \quad \beta = p \cdot (p^{-1} \bmod q)$$

- In particular, $\mathrm{CRT}(\sigma_p, \sigma_q)$ is an integer of size $\approx N^{3/2}$, so if we know it modulo $NN' \approx N^2$, we actually know its value in $\mathbb{Z}$.

Using the fault (II)

Each pair formed of a correct and of a faulty signature gives us an equation of the form:

$$v = \alpha \cdot x + \beta \cdot y$$

where v is known, α, β are unknown, fixed and of size N, and x, y are unknown, of size $N^{1/2}$, and depend on the signature.

One such relation doesn't get us far, but since (x, y) is small compared to (α, β), we expect multiple relations of this form to allow us to recover the x's and y's, and hence factor N.

So suppose we can obtain a vector $\mathbf{v}$ of ℓ CRT values, so that we have an equation:

$$\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$$

The goal is to recover $\mathbf{x}$ and $\mathbf{y}$ from $\mathbf{v}$. To do so, we can used a cryptanlytic technique introduced by Nguyen and Stern in the 1990s: orthogonal lattices.

Using the fault (II)

Each pair formed of a correct and of a faulty signature gives us an equation of the form:

$$v = \alpha \cdot x + \beta \cdot y$$

where v is known, α, β are unknown, fixed and of size N, and x, y are unknown, of size $N^{1/2}$, and depend on the signature.

One such relation doesn't get us far, but since (x, y) is small compared to (α, β), we expect multiple relations of this form to allow us to recover the x's and y's, and hence factor N.

So suppose we can obtain a vector $\mathbf{v}$ of ℓ CRT values, so that we have an equation:

$$\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$$

The goal is to recover $\mathbf{x}$ and $\mathbf{y}$ from $\mathbf{v}$. To do so, we can used a cryptanlytic technique introduced by Nguyen and Stern in the 1990s: orthogonal lattices.

Using the fault (II)

Each pair formed of a correct and of a faulty signature gives us an equation of the form:

$$v = \alpha \cdot x + \beta \cdot y$$

where v is known, α, β are unknown, fixed and of size N, and x, y are unknown, of size $N^{1/2}$, and depend on the signature.

One such relation doesn't get us far, but since (x, y) is small compared to (α, β), we expect multiple relations of this form to allow us to recover the x's and y's, and hence factor N.

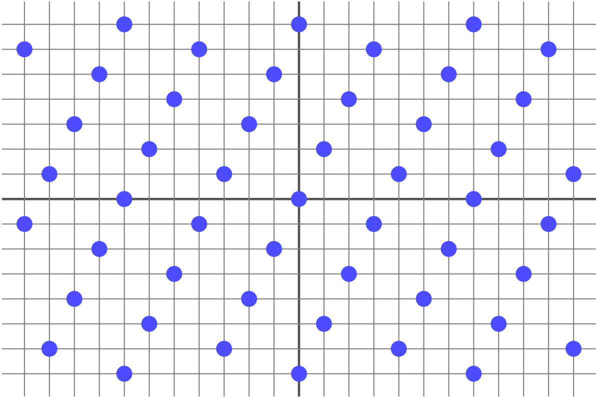
So suppose we can obtain a vector $\mathbf{v}$ of ℓ CRT values, so that we have an equation:

$$\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$$

The goal is to recover $\mathbf{x}$ and $\mathbf{y}$ from $\mathbf{v}$. To do so, we can used a cryptanlytic technique introduced by Nguyen and Stern in the 1990s: orthogonal lattices.

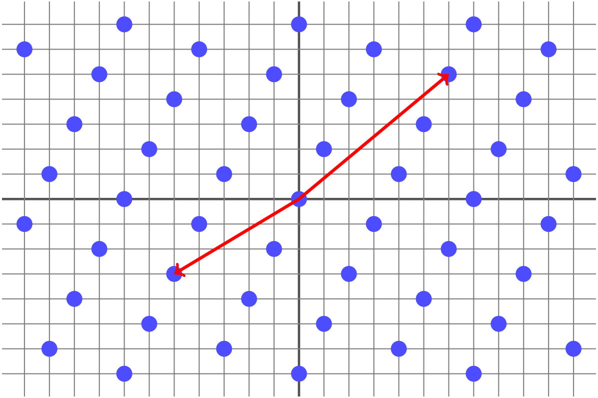
Outline

A primer on lattices



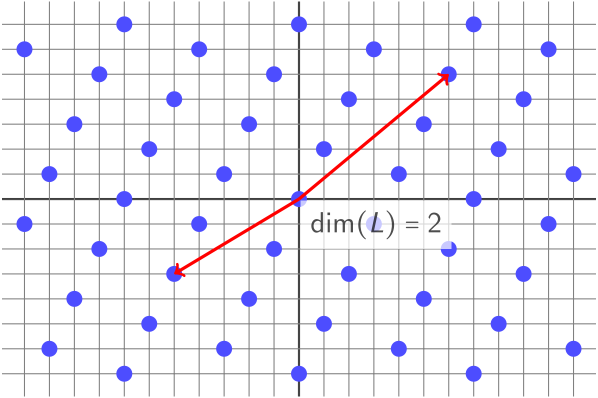
A lattice L is a subgroup of $\mathbb{Z}^n$ for some n:
a regular arrangement of points in $\mathbb{R}^n$.

A primer on lattices



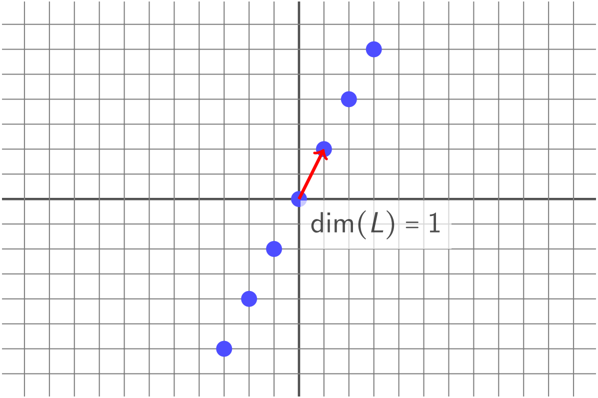
Often represented by a basis
(minimal generating set of vectors in L).

A primer on lattices



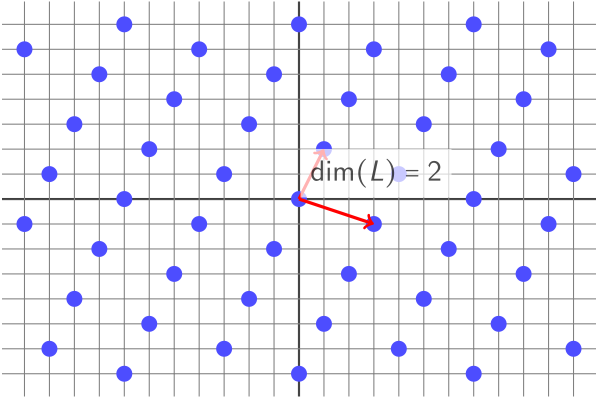
The number of vectors in a basis is called the rank or dimension $\dim(L)$. It is well-defined.

A primer on lattices



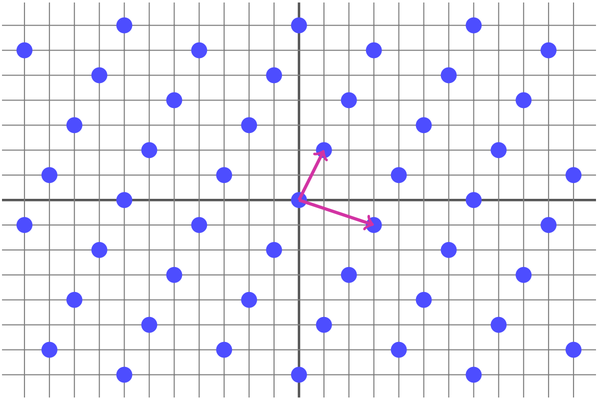
The number of vectors in a basis is called the rank or dimension $\dim(L)$. It is well-defined.

A primer on lattices



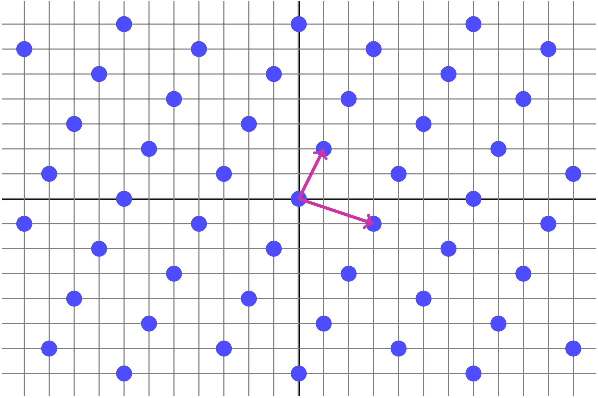
The number of vectors in a basis is called the rank or dimension $\dim(L)$. It is well-defined.

A primer on lattices



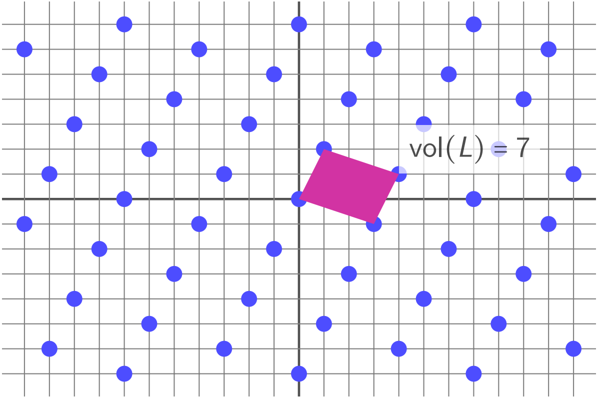
Some bases are better than others: with shorter, almost orthogonal vectors. We call them reduced basis.

A primer on lattices



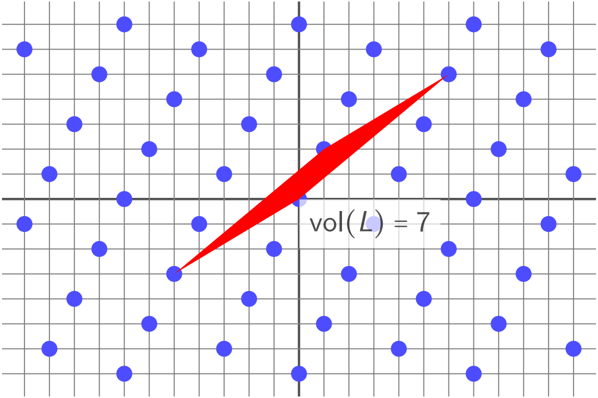
We have algorithms, such as LLL, to compute reduced bases. In low dimension (say $\lesssim 100$), we can obtain "optimal" lattice reduction in practice.

A primer on lattices



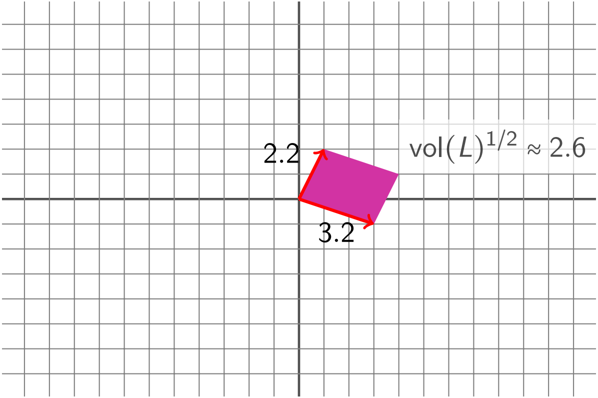
Another important invariant: lattice volume; d-dimensional volume of the parallelipiped defined by a basis. Independent of the basis.

A primer on lattices



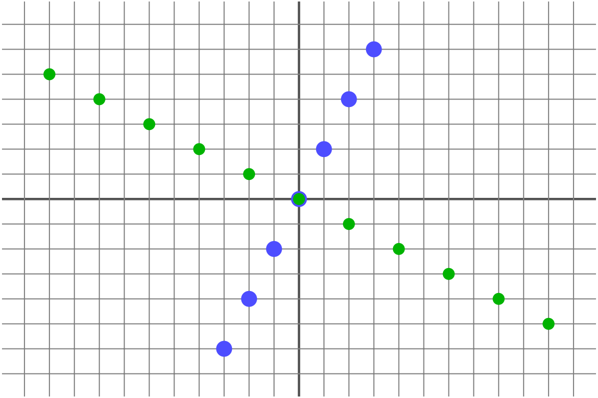
Another important invariant: lattice volume; d-dimensional volume of the parallelipiped defined by a basis. Independent of the basis.

A primer on lattices



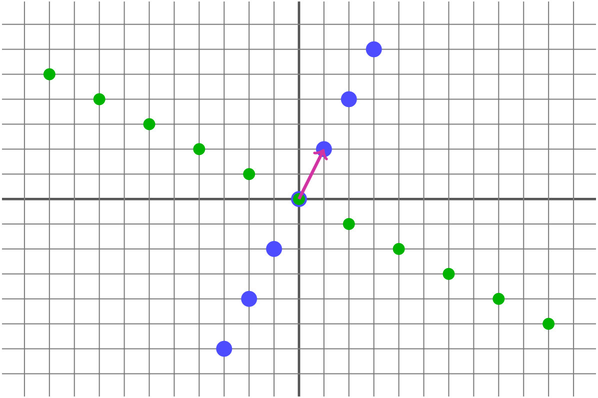
For "typical" (e.g. random) lattices, vectors in a short basis are all roughly the same length $\approx \mathrm{vol}(L)^{1/\dim(L)}$.

A primer on lattices



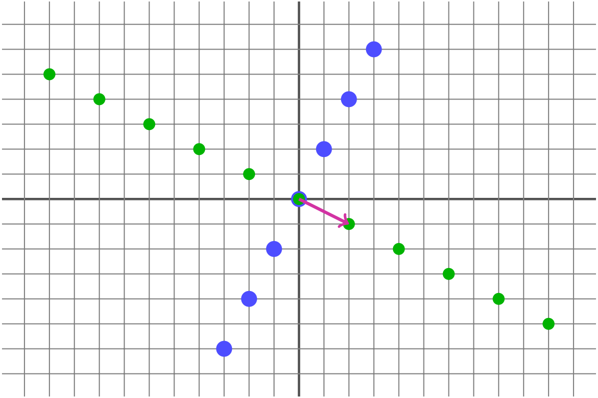
Given a lattice L of dimension d in $\mathbb{Z}^n$, the set of vectors in $\mathbb{Z}^n$ orthogonal to all of the vectors in L is also a lattice $L^\perp$, of dimension $n - d$ and volume $\mathrm{vol}(L^\perp) = \mathrm{vol}(L)$.

A primer on lattices



Given a basis of L, we can compute a reduced basis of $L^{\perp}$ using an algorithm due to Nguyen and Stern (LLL in dimension $n + d$).

A primer on lattices



Given a basis of L, we can compute a reduced basis of $L^{\perp}$ using an algorithm due to Nguyen and Stern (LLL in dimension $n + d$).

Lattice attack overview (I)

- Recall that we have a vector $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$ in $\mathbb{Z}^\ell$ with $\mathbf{x}, \mathbf{y}$ unknown. We want to recover these hidden vectors. Let $L = \mathbb{Z}\mathbf{v} \subset \mathbb{Z}^\ell$.
- Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the lattice $L^\perp$ of vectors in $\mathbb{Z}^\ell$ orthogonal to $\mathbf{v}$. The volume of this lattice is

$$\mathrm{vol}(L^\perp) = \mathrm{vol}(L) = \|\mathbf{v}\| \approx N^{3/2}$$

- Since $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$, the $\mathbf{b_i}$'s satisfy:

$$\alpha\langle \mathbf{b}_i, \mathbf{x}\rangle + \beta\langle \mathbf{b}_i, \mathbf{y}\rangle = 0$$

- But the smallest nonzero solution (s, t) to $\alpha s + \beta t = 0$ is of size $\approx N$, so a given $\mathbf{b}_i$ is either orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, or it is of norm $> \sqrt{N}$.
- Only $\ell - 2$ independent vectors orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, so $\mathbf{b}_{\ell-1}$ must be of length $> \sqrt{N}$. The remaining vectors $\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ generate a lattice L' of volume $\approx \mathrm{vol}(L)/\|\mathbf{b}_{\ell-1}\| \approx N$.

Lattice attack overview (I)

- Recall that we have a vector $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$ in $\mathbb{Z}^\ell$ with $\mathbf{x}, \mathbf{y}$ unknown. We want to recover these hidden vectors. Let $L = \mathbb{Z}\mathbf{v} \subset \mathbb{Z}^\ell$.
- Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the lattice $L^\perp$ of vectors in $\mathbb{Z}^\ell$ orthogonal to $\mathbf{v}$. The volume of this lattice is

$$\mathrm{vol}(L^\perp) = \mathrm{vol}(L) = \|\mathbf{v}\| \approx N^{3/2}$$

- Since $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$, the $\mathbf{b_i}$'s satisfy:

$$\alpha\langle \mathbf{b}_i, \mathbf{x}\rangle + \beta\langle \mathbf{b}_i, \mathbf{y}\rangle = 0$$

- But the smallest nonzero solution (s, t) to $\alpha s + \beta t = 0$ is of size $\approx N$, so a given $\mathbf{b}_i$ is either orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, or it is of norm $> \sqrt{N}$.
- Only $\ell - 2$ independent vectors orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, so $\mathbf{b}_{\ell-1}$ must be of length $> \sqrt{N}$. The remaining vectors $\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ generate a lattice L' of volume $\approx \mathrm{vol}(L)/\|\mathbf{b}_{\ell-1}\| \approx N$.

Lattice attack overview (I)

- Recall that we have a vector $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$ in $\mathbb{Z}^\ell$ with $\mathbf{x}, \mathbf{y}$ unknown. We want to recover these hidden vectors. Let $L = \mathbb{Z}\mathbf{v} \subset \mathbb{Z}^\ell$.
- Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the lattice $L^\perp$ of vectors in $\mathbb{Z}^\ell$ orthogonal to $\mathbf{v}$. The volume of this lattice is

$$\mathrm{vol}(L^\perp) = \mathrm{vol}(L) = \|\mathbf{v}\| \approx N^{3/2}$$

- Since $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$, the $\mathbf{b_i}$'s satisfy:

$$\alpha\langle \mathbf{b}_i, \mathbf{x}\rangle + \beta\langle \mathbf{b}_i, \mathbf{y}\rangle = 0$$

- But the smallest nonzero solution (s, t) to $\alpha s + \beta t = 0$ is of size $\approx N$, so a given $\mathbf{b}_i$ is either orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, or it is of norm $> \sqrt{N}$.
- Only $\ell - 2$ independent vectors orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, so $\mathbf{b}_{\ell-1}$ must be of length $> \sqrt{N}$. The remaining vectors $\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ generate a lattice L' of volume $\approx \mathrm{vol}(L)/\|\mathbf{b}_{\ell-1}\| \approx N$.

Lattice attack overview (I)

- Recall that we have a vector $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$ in $\mathbb{Z}^\ell$ with $\mathbf{x}, \mathbf{y}$ unknown. We want to recover these hidden vectors. Let $L = \mathbb{Z}\mathbf{v} \subset \mathbb{Z}^\ell$.
- Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the lattice $L^\perp$ of vectors in $\mathbb{Z}^\ell$ orthogonal to $\mathbf{v}$. The volume of this lattice is

$$\mathrm{vol}(L^\perp) = \mathrm{vol}(L) = \|\mathbf{v}\| \approx N^{3/2}$$

- Since $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$, the $\mathbf{b_i}$'s satisfy:

$$\alpha\langle\mathbf{b}_i, \mathbf{x}\rangle + \beta\langle\mathbf{b}_i, \mathbf{y}\rangle = 0$$

- But the smallest nonzero solution (s, t) to $\alpha s + \beta t = 0$ is of size $\approx N$, so a given $\mathbf{b}_i$ is either orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, or it is of norm $> \sqrt{N}$.
- Only $\ell - 2$ independent vectors orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, so $\mathbf{b}_{\ell-1}$ must be of length $> \sqrt{N}$. The remaining vectors $\mathbf{b}_1, \dots,$ $\mathbf{b}_{\ell-2}$ generate a lattice L' of volume $\approx \mathrm{vol}(L)/\|\mathbf{b}_{\ell-1}\| \approx N$.

Lattice attack overview (I)

- Recall that we have a vector $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$ in $\mathbb{Z}^\ell$ with $\mathbf{x}, \mathbf{y}$ unknown. We want to recover these hidden vectors. Let $L = \mathbb{Z}\mathbf{v} \subset \mathbb{Z}^\ell$.
- Compute a reduced basis $(\mathbf{b}_1, \ldots, \mathbf{b}_{\ell-1})$ of the lattice $L^\perp$ of vectors in $\mathbb{Z}^\ell$ orthogonal to $\mathbf{v}$. The volume of this lattice is

$$\mathrm{vol}(L^\perp) = \mathrm{vol}(L) = \|\mathbf{v}\| \approx N^{3/2}$$

- Since $\mathbf{v} = \alpha\mathbf{x} + \beta\mathbf{y}$, the $\mathbf{b_i}$'s satisfy:

$$\alpha\langle \mathbf{b}_i, \mathbf{x}\rangle + \beta\langle \mathbf{b}_i, \mathbf{y}\rangle = 0$$

- But the smallest nonzero solution (s, t) to $\alpha s + \beta t = 0$ is of size $\approx N$, so a given $\mathbf{b}_i$ is either orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, or it is of norm $> \sqrt{N}$.
- Only $\ell - 2$ independent vectors orthogonal to both $\mathbf{x}$ and $\mathbf{y}$, so $\mathbf{b}_{\ell-1}$ must be of length $> \sqrt{N}$. The remaining vectors $\mathbf{b}_1, \ldots,$ $\mathbf{b}_{\ell-2}$ generate a lattice L' of volume $\approx \mathrm{vol}(L)/\|\mathbf{b}_{\ell-1}\| \approx N$.

Lattice attack overview (II)

- Since L' has no reason to be special, assume heuristically that it behaves like a random lattice. In particular, we expect all of the vectors in the reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ to be roughly of length $\mathrm{vol}(L')^{1/(\ell-2)} \approx N^{1/(\ell-2)}$.
- In particular, if $\ell \geq 5$, they are all of length $\ll \sqrt{N}$. Therefore, they are orthogonal to $\mathbf{x}, \mathbf{y}$.
- Then, compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice $(L')^\perp$. This lattice is of volume $\mathrm{vol}(L') \approx N$, and in particular doesn't contain many vectors of length $\leq \sqrt{\ell N}$ (we can enumerate them easily). But $\mathbf{x}$ is one of them!
- For each pair (s, t) such that $\mathbf{z} = s\mathbf{x}' + t\mathbf{y}'$ is of length $\leq \sqrt{\ell N}$, compute $\gcd(\mathbf{v} - \mathbf{z}, N)$. When we reach $\mathbf{z} = \mathbf{x}$, this GCD is p, because $\mathbf{v}$ is equal to $\mathbf{x}$ mod p but not mod q.
- Hence, we have factored N (provided that $\ell \geq 5$)!

Lattice attack overview (II)

- Since L' has no reason to be special, assume heuristically that it behaves like a random lattice. In particular, we expect all of the vectors in the reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ to be roughly of length $\mathrm{vol}(L')^{1/(\ell-2)} \approx N^{1/(\ell-2)}$.
- In particular, if $\ell \geq 5$, they are all of length $\ll \sqrt{N}$. Therefore, they are orthogonal to $\mathbf{x}, \mathbf{y}$.
- Then, compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice $(L')^\perp$. This lattice is of volume $\mathrm{vol}(L') \approx N$, and in particular doesn't contain many vectors of length $\leq \sqrt{\ell N}$ (we can enumerate them easily). But $\mathbf{x}$ is one of them!
- For each pair (s, t) such that $\mathbf{z} = s\mathbf{x}' + t\mathbf{y}'$ is of length $\leq \sqrt{\ell N}$, compute $\gcd(\mathbf{v} - \mathbf{z}, N)$. When we reach $\mathbf{z} = \mathbf{x}$, this GCD is p, because $\mathbf{v}$ is equal to $\mathbf{x}$ mod p but not mod q.
- Hence, we have factored N (provided that $\ell \geq 5$)!

Lattice attack overview (II)

- Since L' has no reason to be special, assume heuristically that it behaves like a random lattice. In particular, we expect all of the vectors in the reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ to be roughly of length $\mathrm{vol}(L')^{1/(\ell-2)} \approx N^{1/(\ell-2)}$.
- In particular, if $\ell \geq 5$, they are all of length $\ll \sqrt{N}$. Therefore, they are orthogonal to $\mathbf{x}, \mathbf{y}$.
- Then, compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice $(L')^{\perp}$. This lattice is of volume $\mathrm{vol}(L') \approx N$, and in particular doesn't contain many vectors of length $\leq \sqrt{\ell N}$ (we can enumerate them easily). But $\mathbf{x}$ is one of them!
- For each pair (s, t) such that $\mathbf{z} = s\mathbf{x}' + t\mathbf{y}'$ is of length $\leq \sqrt{\ell N}$, compute $\gcd(\mathbf{v} - \mathbf{z}, N)$. When we reach $\mathbf{z} = \mathbf{x}$, this GCD is p, because $\mathbf{v}$ is equal to $\mathbf{x}$ mod p but not mod q.
- Hence, we have factored N (provided that $\ell \geq 5$)!

Lattice attack overview (II)

- Since L' has no reason to be special, assume heuristically that it behaves like a random lattice. In particular, we expect all of the vectors in the reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ to be roughly of length $\mathrm{vol}(L')^{1/(\ell-2)} \approx N^{1/(\ell-2)}$.
- In particular, if $\ell \geq 5$, they are all of length $\ll \sqrt{N}$. Therefore, they are orthogonal to $\mathbf{x}, \mathbf{y}$.
- Then, compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice $(L')^{\perp}$. This lattice is of volume $\mathrm{vol}(L') \approx N$, and in particular doesn't contain many vectors of length $\leq \sqrt{\ell N}$ (we can enumerate them easily). But $\mathbf{x}$ is one of them!
- For each pair (s, t) such that $\mathbf{z} = s\mathbf{x}' + t\mathbf{y}'$ is of length $\leq \sqrt{\ell N}$, compute $\gcd(\mathbf{v} - \mathbf{z}, N)$. When we reach $\mathbf{z} = \mathbf{x}$, this GCD is p, because $\mathbf{v}$ is equal to $\mathbf{x}$ mod p but not mod q.
- Hence, we have factored N (provided that $\ell \geq 5$)!

Lattice attack overview (II)

- Since L' has no reason to be special, assume heuristically that it behaves like a random lattice. In particular, we expect all of the vectors in the reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ to be roughly of length $\mathrm{vol}(L')^{1/(\ell-2)} \approx N^{1/(\ell-2)}$.
- In particular, if $\ell \geq 5$, they are all of length $\ll \sqrt{N}$. Therefore, they are orthogonal to $\mathbf{x}, \mathbf{y}$.
- Then, compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice $(L')^{\perp}$. This lattice is of volume $\mathrm{vol}(L') \approx N$, and in particular doesn't contain many vectors of length $\leq \sqrt{\ell N}$ (we can enumerate them easily). But $\mathbf{x}$ is one of them!
- For each pair (s, t) such that $\mathbf{z} = s\mathbf{x}' + t\mathbf{y}'$ is of length $\leq \sqrt{\ell N}$, compute $\gcd(\mathbf{v} - \mathbf{z}, N)$. When we reach $\mathbf{z} = \mathbf{x}$, this GCD is p, because $\mathbf{v}$ is equal to $\mathbf{x}$ mod p but not mod q.
- Hence, we have factored N (provided that $\ell \geq 5$)!

Lattice attack overview (II)

- Since L' has no reason to be special, assume heuristically that it behaves like a random lattice. In particular, we expect all of the vectors in the reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-2}$ to be roughly of length $\mathrm{vol}(L')^{1/(\ell-2)} \approx N^{1/(\ell-2)}$.
- In particular, if $\ell \geq 5$, they are all of length $\ll \sqrt{N}$. Therefore, they are orthogonal to $\mathbf{x}, \mathbf{y}$.
- Then, compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice $(L')^\perp$. This lattice is of volume $\mathrm{vol}(L') \approx N$, and in particular doesn't contain many vectors of length $\leq \sqrt{\ell N}$ (we can enumerate them easily). But $\mathbf{x}$ is one of them!
- For each pair (s, t) such that $\mathbf{z} = s\mathbf{x}' + t\mathbf{y}'$ is of length $\leq \sqrt{\ell N}$, compute $\gcd(\mathbf{v} - \mathbf{z}, N)$. When we reach $\mathbf{z} = \mathbf{x}$, this GCD is p, because $\mathbf{v}$ is equal to $\mathbf{x}$ mod p but not mod q.
- Hence, we have factored N (provided that $\ell \geq 5$)! (At least heuristically).

Outline

Simulation of the attack

Since the attack is heuristic, validation is in order.

Simulate the attack as follows:

- Pick random p, q-parts (x_i, y_i).
- Compute the corresponding CRT values v_i in $\mathbb{Z}$.
- Try to factor N using the orthogonal lattice attack. Namely:
  1. Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the orthogonal lattice of $\mathbb{Z}\mathbf{v}$) with LLL.
  2. Compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice of $\mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbf{Z}\mathbf{b}_{\ell-2}$.
  3. Enumerate the vectors $\mathbf{z}$ of $\mathbb{Z}\mathbf{x}' \oplus \mathbb{Z}\mathbf{y}'$ of length at most $\sqrt{\ell N}$ and compute the GCDs $\gcd(\mathbf{v} - \mathbf{z}, N)$ until a factor is found.

Simulation of the attack

Since the attack is heuristic, validation is in order.

Simulate the attack as follows:

- Pick random p, q-parts (x_i, y_i).
- Compute the corresponding CRT values v_i in $\mathbb{Z}$.
- Try to factor N using the orthogonal lattice attack. Namely:
    1. Compute a reduced basis $(\mathbf{b}_1, \ldots, \mathbf{b}_{\ell-1})$ of the orthogonal lattice of $\mathbb{Z}\mathbf{v}$) with LLL.
    2. Compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice of $\mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbb{Z}\mathbf{b}_{\ell-2}$.
    3. Enumerate the vectors $\mathbf{z}$ of $\mathbb{Z}\mathbf{x}' \oplus \mathbb{Z}\mathbf{y}'$ of length at most $\sqrt{2N}$ and compute the GCDs $\gcd(v - z, N)$ until a factor is found.

Simulation of the attack

Since the attack is heuristic, validation is in order.

Simulate the attack as follows:

- Pick random p, q-parts (x_i, y_i).
- Compute the corresponding CRT values v_i in $\mathbb{Z}$.
- Try to factor N using the orthogonal lattice attack. Namely:

  1. Compute a reduced basis $(\mathbf{b}_1, \ldots, \mathbf{b}_{\ell-1})$ of the orthogonal lattice of $\mathbb{Z}\mathbf{v}$) with LLL.
  2. Compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice of $\mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbb{Z}\mathbf{b}_{\ell-2}$.
  3. Enumerate the vectors $\mathbf{z}$ of $\mathbb{Z}\mathbf{x}' \oplus \mathbb{Z}\mathbf{y}'$ of length at most $\sqrt{\ell N}$ and compute the GCDs $\gcd(\mathbf{v} - \mathbf{z}, N)$ until a factor is found.

Simulation of the attack

Since the attack is heuristic, validation is in order.

Simulate the attack as follows:

- Pick random p, q-parts (x_i, y_i).
- Compute the corresponding CRT values v_i in $\mathbb{Z}$.
- Try to factor N using the orthogonal lattice attack. Namely:
  1. Compute a reduced basis $(\mathbf{b}_1, \ldots, \mathbf{b}_{\ell-1})$ of the orthogonal lattice of $\mathbb{Z}\mathbf{v}$) with LLL.
  2. Compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice of $\mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbb{Z}\mathbf{b}_{\ell-2}$.
  3. Enumerate the vectors $\mathbf{z}$ of $\mathbb{Z}\mathbf{x}' \oplus \mathbb{Z}\mathbf{y}'$ of length at most $\sqrt{\ell N}$ and compute the GCDs $\gcd(\mathbf{v} - \mathbf{z}, N)$ until a factor is found.

Simulation of the attack

Since the attack is heuristic, validation is in order.

Simulate the attack as follows:

- Pick random p, q-parts (x_i, y_i).
- Compute the corresponding CRT values v_i in $\mathbb{Z}$.
- Try to factor N using the orthogonal lattice attack. Namely:
  1. Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the orthogonal lattice of $\mathbb{Z}\mathbf{v}$) with LLL.
  2. Compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice of $\mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbb{Z}\mathbf{b}_{\ell-2}$.
  3. Enumerate the vectors $\mathbf{z}$ of $\mathbb{Z}\mathbf{x}' \oplus \mathbb{Z}\mathbf{y}'$ of length at most $\sqrt{\ell N}$ and compute the GCDs $\gcd(\mathbf{v} - \mathbf{z}, N)$ until a factor is found.

Simulation of the attack

Since the attack is heuristic, validation is in order.

Simulate the attack as follows:

- Pick random p, q-parts (x_i, y_i).
- Compute the corresponding CRT values v_i in $\mathbb{Z}$.
- Try to factor N using the orthogonal lattice attack. Namely:
    1. Compute a reduced basis $(\mathbf{b}_1, \dots, \mathbf{b}_{\ell-1})$ of the orthogonal lattice of $\mathbb{Z}\mathbf{v}$) with LLL.
    2. Compute a reduced basis $(\mathbf{x}', \mathbf{y}')$ of the orthogonal lattice of $\mathbb{Z}\mathbf{b}_1 \oplus \cdots \oplus \mathbb{Z}\mathbf{b}_{\ell-2}$.
    3. Enumerate the vectors $\mathbf{z}$ of $\mathbb{Z}\mathbf{x}' \oplus \mathbb{Z}\mathbf{y}'$ of length at most $\sqrt{\ell N}$ and compute the GCDs $\gcd(\mathbf{v} - \mathbf{z}, N)$ until a factor is found.

Simulation results

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%

Success probability of the attack with various parameters.

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

Efficiency of the attack with $\ell = 5$.

The attack in practice

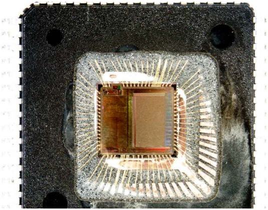
We carried out the attack against an implementation of RSA–CRT signatures on an unprotected 8-bit microcontroller.

1. Decapsulate the chip.
2. Target the SRAM and find the location of the modulus N.
3. Strike with
4. After obtaining 5 pairs of correct and faulty signatures, factor N in a fraction of a second as expected.

The attack in practice

We carried out the attack against an implementation of RSA–CRT signatures on an unprotected 8-bit microcontroller.

1. Decapsulate the chip.

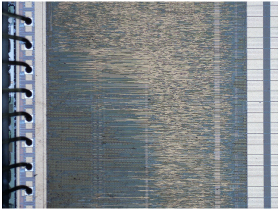


2. Target the SRAM and find the location of the modulus N.
3. Strike with
4. After obtaining 5 pairs of correct and faulty signatures, factor N in a fraction of a second as expected.

The attack in practice

We carried out the attack against an implementation of RSA–CRT signatures on an unprotected 8-bit microcontroller.

1. Decapsulate the chip.
2. Target the SRAM and find the location of the modulus N.



3. Strike with
4. After obtaining 5 pairs of correct and faulty signatures, factor N in a fraction of a second as expected.

The attack in practice

We carried out the attack against an implementation of RSA–CRT signatures on an unprotected 8-bit microcontroller.

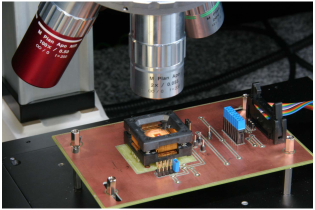
1. Decapsulate the chip.
2. Target the SRAM and find the location of the modulus N.
3. Strike with lasers!

4. After obtaining 5 pairs of correct and faulty signatures, factor N in a fraction of a second as expected.

The attack in practice

We carried out the attack against an implementation of RSA–CRT signatures on an unprotected 8-bit microcontroller.

1. Decapsulate the chip.
2. Target the SRAM and find the location of the modulus N.
3. Strike with a focused laser beam.



4. After obtaining 5 pairs of correct and faulty signatures, factor N in a fraction of a second as expected.

The attack in practice

We carried out the attack against an implementation of RSA–CRT signatures on an unprotected 8-bit microcontroller.

1. Decapsulate the chip.
2. Target the SRAM and find the location of the modulus N.
3. Strike with
4. After obtaining 5 pairs of correct and faulty signatures, factor N in a fraction of a second as expected.

Outline

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ'_i) formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \mathrm{CRT}(\sigma_i, \sigma'_i)$, one needs to know the faulty modulus N'_i.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:
  1. Single-byte faults: the faulty moduli N'_i only differ from N on 8 consecutive bits (e.g. glitch attack when copying the modulus from memory on an 8-bit architecture).
  2. LSB faults: the faulty moduli N'_i only differ from N on the least significant half of all bits (e.g. laser beam targeted at the LSBs of N in memory).

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ_i') formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \mathrm{CRT}(\sigma_i, \sigma_i')$, one needs to know the faulty modulus N_i'.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ_i') formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \mathrm{CRT}(\sigma_i, \sigma_i')$, one needs to know the faulty modulus N_i'.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:
  1. Single-byte faults: the faulty moduli N_i' only differ from N on 8 consecutive bits (e.g. glitch attack when copying the modulus from memory on an 8-bit architecture).
  2. LSB faults: the faulty moduli N_i' only differ from N on the least significant half of all bits (e.g. laser beam targeted at the LSBs of N in memory).

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ_i') formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \mathrm{CRT}(\sigma_i, \sigma_i')$, one needs to know the faulty modulus N_i'.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:
    1. Single-byte faults: the faulty moduli N_i' only differ from N on 8 consecutive bits (e.g. glitch attack when copying the modulus from memory on an 8-bit architecture).
    2. LSB faults: the faulty moduli N_i' only differ from N on the least significant half of all bits (e.g. [illegible] LSBs of N in memory).

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ_i') formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \mathrm{CRT}(\sigma_i, \sigma_i')$, one needs to know the faulty modulus N_i'.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:
  1. Single-byte faults: the faulty moduli N_i' only differ from N on 8 consecutive bits (e.g. glitch attack when copying the modulus from memory on an 8-bit architecture).
  2. LSB faults: the faulty moduli N_i' only differ from N on the least significant half of all bits (e.g. laser beam targeted at the LSBs of N in memory).

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ_i') formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \text{CRT}(\sigma_i, \sigma_i')$, one needs to know the faulty modulus N_i'.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:
  1. Single-byte faults: the faulty moduli N_i' only differ from N on 8 consecutive bits (e.g. glitch attack when copying the modulus from memory on an 8-bit architecture).
  2. LSB faults: the faulty moduli N_i' only differ from N on the least significant half of all bits (e.g. laser beam targeted at the LSBs of N in memory).

Problem with the faulty moduli

- Earlier, I claimed that to obtain the CRT values v_i in $\mathbb{Z}$, we needed pairs (σ_i, σ'_i) formed of a correct and a faulty signature on the same message.
- But this is not enough: to compute $v_i = \mathrm{CRT}(\sigma_i, \sigma'_i)$, one needs to know the faulty modulus N'_i.
- Not very realistic: the signing device is unlikely to output its public modulus together with a signature.
- Fortunately, with a few more faulty of a certain reasonable shape, we can find the v_i's without knowing the faulty moduli.
- We give solutions under the following two fault models:
  1. Single-byte faults: the faulty moduli N'_i only differ from N on 8 consecutive bits (e.g. glitch attack when copying the modulus from memory on an 8-bit architecture).
  2. LSB faults: the faulty moduli N'_i only differ from N on the least significant half of all bits (e.g. laser beam targeted at the LSBs of N in memory).

Solution for LSB faults (I)

- Suppose that, on a given message m, we can obtain not a correct-faulty signature pair (σ, σ'), but several faulty signatures σ'_j, $1 \leq j \leq k$ corresponding to unknown faulty moduli $N'_j = N + \varepsilon_j$ $(|\varepsilon_j| \ll \sqrt{N})$.
- Given this data, we want to recover the CRT value v in $\mathbb{Z}$.
- We can write:

$$v = \sigma + t_0 \cdot N = \sigma'_j + t_j \cdot (N + \varepsilon_j)$$

  for some integers t_j of size $\sqrt{N}$.
- Hence, for $1 \leq j \leq k$, $\sigma - \sigma'_j \equiv t_j \varepsilon_j \pmod{N}$, and since $|t_j \varepsilon_j| \ll N$, the equality holds in $\mathbb{Z}$.
- As a result, we get $t_j = t_0$ for all j, and hence:

$$\sigma - \sigma'_j = t_0 \cdot \varepsilon_j$$

- If $\gcd(\varepsilon_1, \ldots, \varepsilon_k) = 1$, we can compute t_0 as $\gcd(\sigma - \sigma'_1, \ldots, \sigma - \sigma'_j)$, and deduce v accordingly.

Solution for LSB faults (I)

- Suppose that, on a given message m, we can obtain not a correct-faulty signature pair (σ, σ'), but several faulty signatures σ'_j, $1 \le j \le k$ corresponding to unknown faulty moduli $N'_j = N + \varepsilon_j$ ($|\varepsilon_j| \ll \sqrt{N}$).
- Given this data, we want to recover the CRT value v in $\mathbb{Z}$.
- We can write:

$$v = \sigma + t_0 \cdot N = \sigma'_j + t_j \cdot (N + \varepsilon_j)$$

  for some integers t_j of size $\sqrt{N}$.
- Hence, for $1 \le j \le k$, $\sigma - \sigma'_j \equiv t_j \varepsilon_j \pmod{N}$, and since $|t_j \varepsilon_j| \ll N$, the equality holds in $\mathbb{Z}$.
- As a result, we get $t_j = t_0$ for all j, and hence:

$$\sigma - \sigma'_j = t_0 \cdot \varepsilon_j$$

- If $\gcd(\varepsilon_1, \dots, \varepsilon_k) = 1$, we can compute t_0 as $\gcd(\sigma - \sigma'_1, \dots, \sigma - \sigma'_j)$, and deduce v accordingly.

Solution for LSB faults (I)

- Suppose that, on a given message m, we can obtain not a correct-faulty signature pair (σ, σ'), but several faulty signatures σ'_j, $1 \le j \le k$ corresponding to unknown faulty moduli $N'_j = N + \varepsilon_j$ ($|\varepsilon_j| \ll \sqrt{N}$).
- Given this data, we want to recover the CRT value v in $\mathbb{Z}$.
- We can write:

$$v = \sigma + t_0 \cdot N = \sigma'_j + t_j \cdot (N + \varepsilon_j)$$

for some integers t_j of size $\sqrt{N}$.

- Hence, for $1 \le j \le k$, $\sigma - \sigma'_j \equiv t_j \varepsilon_j \pmod{N}$, and since $|t_j \varepsilon_j| \ll N$, the equality holds in $\mathbb{Z}$.
- As a result, we get $t_j = t_0$ for all j, and hence:

$$\sigma - \sigma'_j = t_0 \cdot \varepsilon_j$$

- If $\gcd(\varepsilon_1, \ldots, \varepsilon_k) = 1$, we can compute t_0 as $\gcd(\sigma - \sigma'_1, \ldots, \sigma - \sigma'_j)$, and deduce v accordingly.

Solution for LSB faults (I)

- Suppose that, on a given message m, we can obtain not a correct-faulty signature pair (σ, σ'), but several faulty signatures σ'_j, $1 \leq j \leq k$ corresponding to unknown faulty moduli $N'_j = N + \varepsilon_j$ ($|\varepsilon_j| \ll \sqrt{N}$).
- Given this data, we want to recover the CRT value v in $\mathbb{Z}$.
- We can write:

$$v = \sigma + t_0 \cdot N = \sigma'_j + t_j \cdot (N + \varepsilon_j)$$

  for some integers t_j of size $\sqrt{N}$.
- Hence, for $1 \leq j \leq k$, $\sigma - \sigma'_j \equiv t_j \varepsilon_j \pmod{N}$, and since $|t_j \varepsilon_j| \ll N$, the equality holds in $\mathbb{Z}$.
- As a result, we get $t_j = t_0$ for all j, and hence:

$$\sigma - \sigma'_j = t_0 \cdot \varepsilon_j$$

- If $\gcd(\varepsilon_1, \ldots, \varepsilon_k) = 1$, we can compute t_0 as $\gcd(\sigma - \sigma'_1, \ldots, \sigma - \sigma'_j)$, and deduce v accordingly.

Solution for LSB faults (I)

- Suppose that, on a given message m, we can obtain not a correct-faulty signature pair (σ, σ'), but several faulty signatures σ'_j, $1 \leq j \leq k$ corresponding to unknown faulty moduli $N'_j = N + \varepsilon_j$ ($|\varepsilon_j| \ll \sqrt{N}$).
- Given this data, we want to recover the CRT value v in $\mathbb{Z}$.
- We can write:

$$v = \sigma + t_0 \cdot N = \sigma'_j + t_j \cdot (N + \varepsilon_j)$$

  for some integers t_j of size $\sqrt{N}$.
- Hence, for $1 \leq j \leq k$, $\sigma - \sigma'_j \equiv t_j \varepsilon_j \pmod{N}$, and since $|t_j \varepsilon_j| \ll N$, the equality holds in $\mathbb{Z}$.
- As a result, we get $t_j = t_0$ for all j, and hence:

$$\sigma - \sigma'_j = t_0 \cdot \varepsilon_j$$

- If $\gcd(\varepsilon_1, \ldots, \varepsilon_k) = 1$, we can compute t_0 as $\gcd(\sigma - \sigma'_1, \ldots, \sigma - \sigma'_j)$, and deduce v accordingly.

Solution for LSB faults (I)

- Suppose that, on a given message m, we can obtain not a correct-faulty signature pair (σ, σ'), but several faulty signatures σ'_j, $1 \leq j \leq k$ corresponding to unknown faulty moduli $N'_j = N + \varepsilon_j$ ($|\varepsilon_j| \ll \sqrt{N}$).
- Given this data, we want to recover the CRT value v in $\mathbb{Z}$.
- We can write:

$$v = \sigma + t_0 \cdot N = \sigma'_j + t_j \cdot (N + \varepsilon_j)$$

  for some integers t_j of size $\sqrt{N}$.
- Hence, for $1 \leq j \leq k$, $\sigma - \sigma'_j \equiv t_j \varepsilon_j \pmod{N}$, and since $|t_j \varepsilon_j| \ll N$, the equality holds in $\mathbb{Z}$.
- As a result, we get $t_j = t_0$ for all j, and hence:

$$\sigma - \sigma'_j = t_0 \cdot \varepsilon_j$$

- If $\gcd(\varepsilon_1, \ldots, \varepsilon_k) = 1$, we can compute t_0 as $\gcd(\sigma - \sigma'_1, \ldots, \sigma - \sigma'_j)$, and deduce v accordingly.

Solution for LSB faults (II)

- The probability that this method works is the probability that $\varepsilon_1, \dots, \varepsilon_k$ are coprime, namely $1/\zeta(k)$. This converges quickly to 1 as k grows, and this theoretical value is verified very well in simulation.
- Since we need $\ell = 5$ CRT values to carry out the lattice attack, this method requires $k \cdot \ell$ faulty signatures overall and has a success probability of $\zeta(k)^{-\ell}$.
- Taking $\ell = 5, k = 9$ (45 faults in total) gives a success probability $> 99\%$.
- Validated experimentally using laser fault injection, even with k as low as 4 (theoretical success probability of 67%)!

Solution for LSB faults (II)

- The probability that this method works is the probability that $\varepsilon_1, \dots, \varepsilon_k$ are coprime, namely $1/\zeta(k)$. This converges quickly to 1 as k grows, and this theoretical value is verified very well in simulation.
- Since we need $\ell = 5$ CRT values to carry out the lattice attack, this method requires $k \cdot \ell$ faulty signatures overall and has a success probability of $\zeta(k)^{-\ell}$.
- Taking $\ell = 5, k = 9$ (45 faults in total) gives a success probability $> 99\%$.
- Validated experimentally using laser fault injection, even with k as low as 4 (theoretical success probability of 67%)!

Solution for LSB faults (II)

- The probability that this method works is the probability that $\varepsilon_1, \ldots, \varepsilon_k$ are coprime, namely $1/\zeta(k)$. This converges quickly to 1 as k grows, and this theoretical value is verified very well in simulation.
- Since we need $\ell = 5$ CRT values to carry out the lattice attack, this method requires $k \cdot \ell$ faulty signatures overall and has a success probability of $\zeta(k)^{-\ell}$.
- Taking $\ell = 5, k = 9$ (45 faults in total) gives a success probability $> 99\%$.
- Validated experimentally using laser fault injection, even with k as low as 4 (theoretical success probability of 67%)!

Solution for LSB faults (II)

- The probability that this method works is the probability that $\varepsilon_1, \ldots, \varepsilon_k$ are coprime, namely $1/\zeta(k)$. This converges quickly to 1 as k grows, and this theoretical value is verified very well in simulation.
- Since we need $\ell = 5$ CRT values to carry out the lattice attack, this method requires $k \cdot \ell$ faulty signatures overall and has a success probability of $\zeta(k)^{-\ell}$.
- Taking $\ell = 5, k = 9$ (45 faults in total) gives a success probability $> 99\%$.
- Validated experimentally using laser fault injection, even with k as low as 4 (theoretical success probability of 67%)!

Conclusion

- This new attack presents a number of nice features:
  - Quite efficient and doesn't require too many faults (45 faulty signatures enough in a typical setting for $> 99\%$ success rate).
  - Not thwarted by e.g. Shamir's trick.
- However, it does have some limitations:
  - Must be able to obtain a correct and a faulty signature with the same CRT value: not possible with probabilistic paddings like PSS.
  - Most seriously: a faster, frequently used technique for CRT interpolation (Garner's formula) avoids reducing mod N altogether, and hence defeats this attack.
- Possible extension to protected RSA–CRT implementations that *do* a final modular reduction? To discrete log settings?

Conclusion

- This new attack presents a number of nice features:
  - Quite efficient and doesn't require too many faults (45 faulty signatures enough in a typical setting for $> 99\%$ success rate).
  - Not thwarted by e.g. Shamir's trick.
- However, it does have some limitations:
  - Must be able to obtain a correct and a faulty signature with the same CRT value: not possible with probabilistic paddings like PSS.
  - Most seriously: a faster, frequently used technique for CRT interpolation (Garner's formula) avoids reducing mod N altogether, and hence defeats this attack.
- Possible extension to protected RSA–CRT implementations that *do* a final modular reduction? To discrete log settings?

Conclusion

- This new attack presents a number of nice features:
  - Quite efficient and doesn't require too many faults (45 faulty signatures enough in a typical setting for $> 99\%$ success rate).
  - Not thwarted by e.g. Shamir's trick.
- However, it does have some limitations:
  - Must be able to obtain a correct and a faulty signature with the same CRT value: not possible with probabilistic paddings like PSS.
  - Most seriously: a faster, frequently used technique for CRT interpolation (Garner's formula) avoids reducing mod N altogether, and hence defeats this attack.
- Possible extension to protected RSA–CRT implementations that *do* a final modular reduction? To discrete log settings?

Thank you!