Attacking RSA–CRT Signatures with Faults on Montgomery Multiplication

Pierre-Alain Fouque (INRIA)
Nicolas Guillermin (DGA)
Delphine Lerestreux (DGA)
Mehdi Tibouchi (NTT)
Jean-Christophe Zapalowicz (INRIA)

CHES 2012

About this talk

Cryptanalysis of RSA–CRT signatures

which use of the efficient **Montgomery Multiplication**

whatever the encoding function

Fault attacks

Montgomery multiplication algorithm

- **Classical modular multiplication** uses: multiplications, additions and divisions
- **Montgomery multiplication (CIOS)** uses... **shifts instead of divisions!**
  $\Rightarrow$ cost only twice that of a non modular multiplication

$\bar{x} = xR \bmod q$ is the Montgomery representation of x (R constant)

- $\text{CIOS}(\bar{x}, \bar{y}) = \bar{x}\bar{y} \cdot R^{-1} \bmod q = xy \cdot R \bmod q$
- Classical representation $\longrightarrow$ Montgomery representation:

$$\text{CIOS}(x, R^2 \bmod q) = xR = \bar{x}$$

- Montgomery representation $\longrightarrow$ Classical representation:

$$\text{CIOS}(\bar{x}, 1) = xR = x$$

Montgomery multiplication algorithm

- **Classical modular multiplication** uses:
  multiplications, additions and divisions
- **Montgomery multiplication (CIOS)** uses...
  shifts instead of divisions!
  $\Rightarrow$ cost only twice that of a non modular multiplication

$\bar{x} = xR \bmod q$ is the Montgomery representation of x (R constant)

- $\text{CIOS}(\bar{x}, \bar{y}) = \bar{x}\bar{y} \cdot R^{-1} \bmod q = xy \cdot R \bmod q$
- Classical representation $\longrightarrow$ Montgomery representation:

$$\text{CIOS}(x, R^2 \bmod q) = xR = \bar{x}$$

- Montgomery representation $\longrightarrow$ Classical representation:

$$\text{CIOS}(\bar{x}, 1) = xR = x$$

Montgomery multiplication algorithm

- **Classical modular multiplication** uses:
  multiplications, additions and divisions
- **Montgomery multiplication (CIOS)** uses...
  shifts instead of divisions!
  $\Rightarrow$ cost only twice that of a non modular multiplication

$\bar{x} = xR \bmod q$ is the Montgomery representation of x (R constant)

- $\text{CIOS}(\bar{x}, \bar{y}) = \bar{x}\bar{y} \cdot R^{-1} \bmod q = xy \cdot R \bmod q$
- Classical representation $\longrightarrow$ Montgomery representation:

$$\text{CIOS}(x, R^2 \bmod q) = xR = \bar{x}$$

- Montgomery representation $\longrightarrow$ Classical representation:

$$\text{CIOS}(\bar{x}, 1) = xR = x$$

Exponentiation algorithms

Square-and-Multiply MSB	Montgomery Ladder
function $\text{Exp}_{\text{MSB}}(x, d, q)$	**function** $\text{Exp}_{\text{LADDER}}(x, d, q)$
$\bar{x} \leftarrow \text{CIOS}(x, R^2 \bmod q)$	$\bar{x} \leftarrow \text{CIOS}(x, R^2 \bmod q)$
$\overline{A} \leftarrow R \bmod q$	$\overline{A} \leftarrow R \bmod q$
for $i = t$ down to 0 **do**	**for** $i = t$ down to 0 **do**
$\quad\overline{A} \leftarrow \text{CIOS}(\overline{A}, \overline{A})$	$\quad$**if** $d_i = 0$ **then**
$\quad$**if** $d_i = 1$ **then**	$\quad\quad\bar{x} \leftarrow \text{CIOS}(\overline{A}, \bar{x})$
$\quad\quad\overline{A} \leftarrow \text{CIOS}(\overline{A}, \bar{x})$	$\quad\quad\overline{A} \leftarrow \text{CIOS}(\overline{A}, \overline{A})$
return $\text{CIOS}(\overline{A}, 1) = x^d \bmod q$	$\quad$**else if** $e_i = 1$ **then**
	$\quad\quad\overline{A} \leftarrow \text{CIOS}(\overline{A}, \bar{x})$
	$\quad\quad\bar{x} \leftarrow \text{CIOS}(\bar{x}, \bar{x})$
	return $\text{CIOS}(\overline{A}, 1) = x^d \bmod q$

RSA–CRT signature

p, q: two secret primes — $N = pq$: public modulus
e: public exponent — d: secret exponent

$$ed \equiv 1 \bmod ((p-1)(q-1))$$

- **RSA signature**: $S \equiv M^d \bmod N$
- **RSA–CRT signature:**

1: **function** $\textsc{Sign}_{\text{RSA–CRT}}(M)$
2: $S_p \leftarrow M^{d \bmod p-1} \bmod p$
3: $S_q \leftarrow M^{d \bmod q-1} \bmod q$
4: $\begin{cases} S = \text{CRT}(S_p, S_q) \bmod N \\ \quad\quad or \\ S = \text{Garner}(S_p, S_q) \bmod N \end{cases}$
5: **return** S

RSA–CRT signature

p, q: two secret primes — $N = pq$: public modulus
e: public exponent — d: secret exponent

$$ed \equiv 1 \bmod ((p-1)(q-1))$$

- **RSA signature**: $S \equiv M^d \bmod N$
- **RSA–CRT signature**:

1: **function** $\textsc{Sign}_{\text{RSA–CRT}}(M)$
2: $S_p \leftarrow M^{d \bmod p-1} \bmod p$
3: $S_q \leftarrow M^{d \bmod q-1} \bmod q$
4: $\begin{cases} S = \text{CRT}(S_p, S_q) \bmod N \\ \quad \textit{or} \\ S = \text{Garner}(S_p, S_q) \bmod N \end{cases}$
5: **return** S

RSA–CRT signature

p, q: two secret primes $\qquad N = pq$: public modulus

e: public exponent $\qquad d$: secret exponent

$$ed \equiv 1 \bmod ((p-1)(q-1))$$

- **RSA signature**: $S \equiv M^d \bmod N$

- **RSA–CRT signature**:

1: **function** $\textsc{Sign}_{\text{RSA–CRT}}(M)$
2: $\quad S_p \leftarrow M^{d \bmod p-1} \bmod p$
3: $\quad S_q \leftarrow M^{d \bmod q-1} \bmod q$
4: $\quad \begin{cases} S = \text{CRT}(S_p, S_q) \bmod N \\ \textit{or} \\ S = \text{Garner}(S_p, S_q) \bmod N \end{cases}$
5: $\quad$ **return** S

4× faster!!

Bellcore attack

function $\textsc{Sign}_{\text{RSA-CRT}}(M)$
    $S_p \leftarrow M^{d \bmod p-1} \bmod p$
    $S_q \leftarrow M^{d \bmod q-1} \bmod q$

$$\begin{cases} S = \mathrm{CRT}(S_p, S_q) \bmod N \\ \textit{or} \\ S = \mathrm{Garner}(S_p, S_q) \bmod N \end{cases}$$

    return S

Bellcore attack

function $\textsc{Sign}_{\text{RSA-CRT}}(M)$

$\quad \mathbf{S_p \leftarrow M^{d \bmod p-1} \bmod p}$

$\quad S_q \leftarrow M^{d \bmod q-1} \bmod q$

$$\left\{ \begin{array}{c} S = \text{CRT}(S_p, S_q) \bmod N \\ or \\ S = \text{Garner}(S_p, S_q) \bmod N \end{array} \right.$$

$\quad$ **return** S

Bellcore attack

1: **function** $\textsc{Sign}_{\text{RSA-CRT}}(M)$
2: $\boldsymbol{S_p \leftarrow M^{d \bmod p-1} \bmod p}$
3: $S_q \leftarrow M^{d \bmod q-1} \bmod q$
4:
$$\begin{cases} S = \text{CRT}(S_p, S_q) \bmod N \\ \text{or} \\ S = \text{Garner}(S_p, S_q) \bmod N \end{cases}$$
5: **return** S

Attack

$S_p \rightarrow \widetilde{S}_p \qquad S \rightarrow \widetilde{S}$

$\widetilde{S}_p \neq M^d \bmod p \qquad S_q = M^d \bmod q$

$\widetilde{S}^e \neq M \bmod p \qquad \widetilde{S}^e = M \bmod q$

$\boldsymbol{\gcd(\widetilde{S}^e - M \bmod N, N) = q}$

Bellcore attack

function $\textsc{Sign}_{\textsc{RSA-CRT}}(M)$
    $M \leftarrow \mu(m) \in \mathbb{Z}_N$
    $\boldsymbol{S_p \leftarrow M^{d \bmod p-1} \bmod p}$
    $S_q \leftarrow M^{d \bmod q-1} \bmod q$
    $$\begin{cases} S = \mathrm{CRT}(S_p, S_q) \bmod N \\ \quad\quad\quad\quad \textit{or} \\ S = \mathrm{Garner}(S_p, S_q) \bmod N \end{cases}$$
    return S

$\mu =$ deterministic encoding function $\Rightarrow$ Attack works!

Bellcore attack

function $\text{SIGN}_{\text{RSA-CRT}}(M)$

$M \leftarrow \mu(m) \in \mathbb{Z}_N$

$\mathbf{S_p \leftarrow M^{d \bmod p-1} \bmod p}$

$S_q \leftarrow M^{d \bmod q-1} \bmod q$

$$\begin{cases} S = \text{CRT}(S_p, S_q) \bmod N \\ \quad \textit{or} \\ S = \text{Garner}(S_p, S_q) \bmod N \end{cases}$$

return S

$\mu =$ deterministic encoding function $\Rightarrow$ Attack works!

$\mu =$ probabilistic encoding function:

Bellcore attack

1: **function** $\text{SIGN}_{\text{RSA-CRT}}(M)$
2: $M \leftarrow \mu(m) \in \mathbb{Z}_N$
3: $\mathbf{S_p \leftarrow M^{d \bmod p-1} \bmod p}$
4: $S_q \leftarrow M^{d \bmod q-1} \bmod q$
5: $\begin{cases} S = \text{CRT}(S_p, S_q) \bmod N \\ \quad \text{or} \\ S = \text{Garner}(S_p, S_q) \bmod N \end{cases}$
6: **return** S

$\mu =$ deterministic encoding function $\Rightarrow$ Attack works!

$\mu =$ probabilistic encoding function:

random sent with signature $\Rightarrow$ Attack works!

Bellcore attack

function $\textsc{Sign}_{\textrm{RSA–CRT}}(M)$

$M \leftarrow \mu(m) \in \mathbb{Z}_N$

$\boldsymbol{S_p \leftarrow M^{d \bmod p-1} \bmod p}$

$S_q \leftarrow M^{d \bmod q-1} \bmod q$

$$\begin{cases} S = \mathrm{CRT}(S_p, S_q) \bmod N \\ \text{or} \\ S = \mathrm{Garner}(S_p, S_q) \bmod N \end{cases}$$

return S

$\mu =$ deterministic encoding function $\Rightarrow$ Attack works!

$\mu =$ probabilistic encoding function:

random sent with signature $\Rightarrow$ Attack works!

otherwise, in general (RSA–PSS ...) $\Rightarrow$ No attack!

For now:

We focus on **hardware designs for RSA signatures** using:

- **RSA–CRT**
- **Montgomery multiplication**
- regardless of the encoding function

Null faults: Presentation

Fault model: force a small precomputed value to zero

Null faults: Presentation

Fault model: force a small precomputed value to zero

function CIOS($\bar{x}, \bar{y}$)
  $a \leftarrow 0$
  $\bar{y}_0 \leftarrow \bar{y} \bmod 2^r$
  for $j = 0$ to $k - 1$ **do**
    $a_0 \leftarrow a \bmod 2^r$
    $u_j \leftarrow (a_0 + \bar{x}_j \cdot \bar{y}_0) \cdot q' \bmod 2^r$
    $a \leftarrow \left\lfloor \frac{a + \bar{x}_j \cdot \bar{y} + u_j \cdot q}{2^r} \right\rfloor$
  if $a \geq q$ **then** $a \leftarrow a - q$
  return $a = \bar{x}\bar{y}R^{-1} \bmod q$

r: size of the registers
k s.t $R = 2^{rk}$ ($R > q$, $\gcd(q, R) = 1$)

$q' = -q^{-1} \bmod 2^r$ precomputed

division implemented as right shift

Null faults: Presentation

Fault model: force a small precomputed value to zero

```
function CIOS(x̄, ȳ)
    a ← 0
    ȳ_0 ← ȳ mod 2^r
    for j = 0 to k − 1 do
        a_0 ← a mod 2^r
        u_j ← (a_0 + x̄_j · ȳ_0) · q' mod 2^r
        a ← ⌊(a + x̄_j · ȳ + u_j · q) / 2^r⌋
    if a ≥ q then a ← a − q
    return a = x̄ȳR^{-1} mod q
```

r: size of the registers

k s.t $R = 2^{rk}$ ($R > q$, $\gcd(q, R) = 1$)

$q' = -q^{-1} \bmod 2^r$ precomputed

division implemented as right shift

Objective: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$ (Garner)

If $\widetilde{S}_q = 0$ then $\gcd(\widetilde{S}, N) = q$ with a single faulted signature

Null faults: Attacks

2 possible recombinations:

- Garner: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$
  $\Rightarrow S_q$ in classical representation required

2 attacks:

- Attacking CIOS($\overline{A}$, 1) ($\overline{A} = \overline{S}_q$)
- Attacking consecutive CIOS steps

Null faults: Attacks

2 possible recombinations:

- Garner: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$

2 attacks:

- Attacking CIOS($\overline{A}$, 1)
- Attacking consecutive CIOS steps

```
function CIOS(Ā, 1)
    a ← 0
    y_0 ← 1
    for j = 0 to k − 1 do
        a_0 ← a mod 2^r
        u_j ← (a_0 + Ā_j) · q′ mod 2^r
        a ← ⌊(a + x_j · 1 + u_j · q) / 2^r⌋
    if a ≥ q then a ← a − q
    return a = S_q
```

```
function Exp_LSB(x, d, q)
    x̄ ← CIOS(x, R^2 mod q)
    Ā ← R mod q
    for i = 0 to t do
        if d_i = 1 then
            Ā ← CIOS(Ā, x̄)
        x̄ ← CIOS(x̄, x̄)
    return CIOS(Ā, 1) = S_q
```

Null faults: Attacks

2 possible recombinations:

- Garner: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$

2 attacks:

- Attacking $\mathrm{CIOS}(\overline{A}, 1)$
- Attacking consecutive CIOS steps

function $\mathrm{CIOS}(\overline{A}, 1)$
- $a \leftarrow 0$
- $y_0 \leftarrow 1$
- **for** $j = 0$ to $k - 1$ **do**
  - $a_0 \leftarrow a \bmod 2^r$
  - $0 = u_j \leftarrow (a_0 + \overline{A}_j) \cdot q' \bmod 2^r$
  - $0 = a \leftarrow \left\lfloor \frac{a + x_j \cdot 1 + \cancel{u_j \cdot q}}{2^r} \right\rfloor$
- **if** $a \geq q$ **then** $a \leftarrow a - q$
- **return** $a = S_q = 0$

function $\mathrm{Exp_{LSB}}(x, d, q)$
- $\bar{x} \leftarrow \mathrm{CIOS}(x, R^2 \bmod q)$
- $\overline{A} \leftarrow R \bmod q$
- **for** $i = 0$ to t **do**
  - **if** $d_i = 1$ **then**
    - $\overline{A} \leftarrow \mathrm{CIOS}(\overline{A}, \bar{x})$
  - $\bar{x} \leftarrow \mathrm{CIOS}(\bar{x}, \bar{x})$
- **return** $\mathrm{CIOS}(\overline{A}, 1) = S_q$

Null faults: Attacks

2 possible recombinations:

- Garner: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$
- CRT: $S = (S_q \cdot \underbrace{p^{-1} \bmod q}_{V}) \cdot p + (S_p \cdot q^{-1} \bmod p) \cdot q \bmod N$

  $S_q \cdot V$ in classical representation required: CIOS$(V, S_q \cdot R)$

2 attacks:

- Attacking CIOS$(\overline{A}, 1)$
- Attacking consecutive CIOS steps

Null faults: Attacks

2 possible recombinations:

- Garner: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$
- CRT: $S = (S_q \cdot p^{-1} \bmod q) \cdot p + (S_p \cdot q^{-1} \bmod p) \cdot q \bmod N$

2 attacks:

- Attacking $\mathrm{CIOS}(\overline{A}, 1)$
- Attacking consecutive CIOS steps

```
function CIOS(x̄, x̄)
    a ← 0
    x̄_0 ← x̄ mod 2^r
    for j = 0 to k − 1 do
        a_0 ← a mod 2^r
        u_j ← (a_0 + x̄_j · x̄_0) · q' mod 2^r
        a ← ⌊(a + x̄_j · x̄ + u_j · q) / 2^r⌋
    if a ≥ q then a ← a − q
    return a
```

```
function EXP_LSB(x, d, q)
    x̄ ← CIOS(x, R^2 mod q)
    Ā ← R mod q
    for i = 0 to t do
        if d_i = 1 then
            Ā ← CIOS(Ā, x̄)
        x̄ ← CIOS(x̄, x̄)
    return CIOS(Ā, 1)
```

Null faults: Attacks

2 possible recombinations:

- Garner: $S = S_q + q \cdot (q^{-1} \cdot (S_p - S_q) \bmod p)$
- CRT: $S = (S_q \cdot p^{-1} \bmod q) \cdot p + (S_p \cdot q^{-1} \bmod p) \cdot q \bmod N$

2 attacks:

- Attacking $\mathrm{CIOS}(\overline{A}, 1)$
- Attacking consecutive CIOS steps

function $\mathrm{CIOS}(\bar{x}, \bar{x})$
  $a \leftarrow 0$
  $\bar{x}_0 \leftarrow \bar{x} \bmod 2^r$
  for $j = 0$ to $k - 1$ **do**
    $a_0 \leftarrow a \bmod 2^r$
    $u_j \leftarrow (a_0 + \bar{x}_j \cdot \bar{x}_0) \cdot 0 \bmod 2^r$
    $a \leftarrow \left\lfloor \frac{a + \bar{x}_j \cdot \bar{x} + \cancel{u_j \cdot q}}{2^r} \right\rfloor$
  if $a \geq q$ **then** $a \leftarrow a - q$
  return $a = \left\lfloor \frac{\bar{x}_{k-1} \bar{x}}{2^r} \right\rfloor + o(2^{r(k-1)})$

function $\mathrm{Exp}_{\mathrm{LSB}}(x, d, q)$
  $\bar{x} \leftarrow \mathrm{CIOS}(x, R^2 \bmod q)$
  $\overline{A} \leftarrow R \bmod q$
  for $i = 0$ to t **do**
    if $e_i = 1$ **then**
      $\overline{A} \leftarrow \mathrm{CIOS}(\overline{A}, \bar{x})$
    $\bar{x} \leftarrow \mathrm{CIOS}(\bar{x}, \bar{x})$
  return $\mathrm{CIOS}(\overline{A}, 1)$

Null faults: Attacks

function $\mathrm{CIOS}(\bar{x}, \bar{x})$
$a \leftarrow 0$
$\bar{x}_0 \leftarrow \bar{x} \bmod 2^r$
for $j = 0$ to $k-1$ **do**
$a_0 \leftarrow a \bmod 2^r$
$u_j \leftarrow (a_0 + \bar{x}_j \cdot \bar{x}_0) \cdot 0 \bmod 2^r$
$a \leftarrow \left\lfloor \frac{a + \bar{x}_j \cdot \bar{x} + \cancel{u_j \cdot q}}{2^r} \right\rfloor$
if $a \geq q$ **then** $a \leftarrow a - q$
return $a = \left\lfloor \frac{\bar{x}_{k-1}\bar{x}}{2^r} \right\rfloor + o(2^{r(k-1)})$

function $\mathrm{Exp}_{\mathrm{LSB}}(x, d, q)$
$\bar{x} \leftarrow \mathrm{CIOS}(x, R^2 \bmod q)$
$\overline{A} \leftarrow R \bmod q$
for $i = 0$ to t **do**
if $e_i = 1$ **then**
$\overline{A} \leftarrow \mathrm{CIOS}(\overline{A}, \bar{x})$
$\bar{x} \leftarrow \mathrm{CIOS}(\bar{x}, \bar{x})$
return $\mathrm{CIOS}(\overline{A}, 1)$

$$\bar{x} \leftarrow \left\lfloor \frac{\bar{x}_{k-1}\bar{x}}{2^r} \right\rfloor + o(2^{r(k-1)})$$

$$\underbrace{|\bar{x}| \leq \lceil \log_2 q \rceil - 1}_{\text{true with probability } 1/2} \overset{CIOS_f}{\Rightarrow} |\bar{x}| \leq \lceil \log_2 q \rceil - 2 \overset{CIOS_f}{\Rightarrow} |\bar{x}| \leq \lceil \log_2 q \rceil - 4 \ldots$$

$\lceil log_2 \lceil \log_2 q \rceil \rceil$ consecutive faulted iterations $\Rightarrow \widetilde{S}_q = 0$

Null faults: Attacks

	S&M LSB	S&M MSB		Montgomery Ladder	
Faulty iterations	(%)	Start (%)	Anywhere (%)	Start (%)	Anywhere (%)
8	31	93	62	45	30
9	65	100	93	87	76
10	89	100	100	99	93

Table: 100 faulty signatures computed with SAGE for a 512-bit prime q and $r = 16$.

Null faults: Conclusion

2 attacks:

- Attacking CIOS($\overline{A}$, 1) $\Rightarrow$ 1 signature and 1 fault
- Attacking consecutive CIOS steps $\Rightarrow$ 1 signature and a few faulty iterations (sometimes even a single fault if q' is not recomputed!)

Realistic attacks:

- A single signature
- Targeting small registers
- Targeting a precomputed value
- Few faulty iterations required

Constant faults: Presentation

Fault model: force a small value to some (possibly unknown) constant value

Constant faults: Presentation

Fault model: force a <u>small</u> value to some (possibly unknown) constant value

function CIOS$(\bar{x}, \bar{y})$

  $a \leftarrow 0$

  $\bar{y}_0 \leftarrow \bar{y} \bmod 2^r$

  for $j = 0$ to $k - 1$ **do**

    $a_0 \leftarrow a \bmod 2^r$

    $u_j \leftarrow (a_0 + \bar{x}_j \cdot \bar{y}_0) \cdot q' \bmod 2^r$

    $a \leftarrow \left\lfloor \dfrac{a + \bar{x}_j \cdot \bar{y} + u_j \cdot q}{2^r} \right\rfloor$

  if $a \geq q$ **then** $a \leftarrow a - q$

  return a

Constant faults: Presentation

Fault model: force a small value to some (possibly unknown) constant value

function CIOS($\bar{x}, \bar{y}$)
  $a \leftarrow 0$
  $\bar{y}_0 \leftarrow \bar{y} \bmod 2^r$
  for $j = 0$ to $k - 1$ **do**
    $a_0 \leftarrow a \bmod 2^r$
    $u_j \leftarrow (a_0 + \bar{x}_j \cdot \bar{y}_0) \cdot q' \bmod 2^r$
    $a \leftarrow \left\lfloor \frac{a + \bar{x}_j \cdot \bar{y} + u_j \cdot q}{2^r} \right\rfloor$
  if $a \geq q$ **then** $a \leftarrow a - q$
  return a

Objective: Having $\widetilde{S}$ a close multiple of q

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  - $\widetilde{S}_q$ is close to the real number $\widetilde{u} \cdot q/(2^r - 1)$.
  - $\left|(2^r - 1) \cdot (\widetilde{S} + 1) - qT\right| \leq 2^{r+1}$ with T an integer.
  - A single faulty signature yields $V = (2^r - 1) \cdot (\widetilde{S} + 1) \bmod N$.
  - $(r = 8, 16)$: $\gcd(V + X, N)$ for $|X| \leq 2^{r+1}$
  - $(r = 32)$: Baby step, giant step-like algorithm by Chen and Nguyen
  - $(r < \lceil \log_2 q/2 \rceil)$: Howgrave-Graham's algorithm
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  - $\widetilde{S}_q$ is close to the real number $\widetilde{u} \cdot q/(2^r - 1)$.
  - $\left|(2^r - 1) \cdot (\widetilde{S} + 1) - qT\right| \leq 2^{r+1}$ with T an integer.
  - A single faulty signature yields $V = (2^r - 1) \cdot (\widetilde{S} + 1) \bmod N$.
  - $(r = 8, 16)$: $\gcd(V + X, N)$ for $|X| \leq 2^{r+1}$
  - $(r = 32)$: Baby step, giant step-like algorithm by Chen and Nguyen
  - $(r < \lceil \log_2 q/2 \rceil)$: Howgrave-Graham's algorithm
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  - $\widetilde{S}_q$ is close to the real number $\widetilde{u} \cdot q/(2^r - 1)$.
  - $\left|(2^r - 1) \cdot (\widetilde{S} + 1) - qT\right| \leq 2^{r+1}$ with T an integer.
  - A single faulty signature yields $V = (2^r - 1) \cdot (\widetilde{S} + 1) \bmod N$.

  - $(r = 8, 16)$: $\gcd(V + X, N)$ for $|X| \leq 2^{r+1}$
  - $(r = 32)$: Baby step, giant step-like algorithm by Chen and Nguyen
  - $(r < \lceil \log_2 q/2 \rceil)$: Howgrave-Graham's algorithm
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  - $\widetilde{S}_q$ is close to the real number $\widetilde{u} \cdot q/(2^r - 1)$.
  - $\left|(2^r - 1) \cdot (\widetilde{S} + 1) - qT\right| \leq 2^{r+1}$ with T an integer.
  - A single faulty signature yields $V = (2^r - 1) \cdot (\widetilde{S} + 1) \bmod N$.
  - $(r = 8, 16)$: $\gcd(V + X, N)$ for $|X| \leq 2^{r+1}$
  - $(r = 32)$: Baby step, giant step-like algorithm by Chen and Nguyen
  - $(r < \lceil \log_2 q/2 \rceil)$: Howgrave-Graham's algorithm
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
  $\Rightarrow$ some signatures and some faults
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
  $\Rightarrow$ some signatures and some faults
- Attacking consecutive CIOS steps

Constant faults: Attacks

4 attacks:

- Attacking CIOS($\overline{A}$, 1) without miss ($u_0 = \cdots = u_{k-1} = \widetilde{u}$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with some misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j < k/2$)
  $\Rightarrow$ 1 signature and a few faults
- Attacking CIOS($\overline{A}$, 1) with more misses ($u_j = \cdots = u_{k-1} = \widetilde{u}, j > k/2$)
  $\Rightarrow$ some signatures and some faults
- Attacking consecutive CIOS steps
  $\Rightarrow$ 2 signatures and some faults

Are the models realistic?

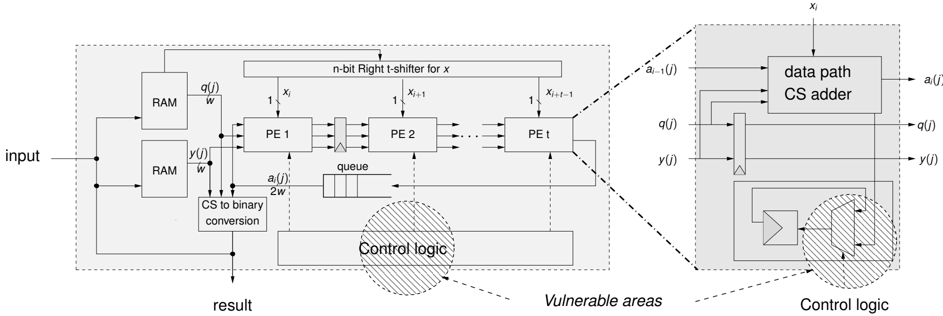
Fault injection has to be made on several rounds of the CIOS algorithm

q' or u_j must be isolated in the architecture:

Are the models realistic?

Fault injection has to be made on several rounds of the CIOS algorithm

q' or u_j must be isolated in the architecture:

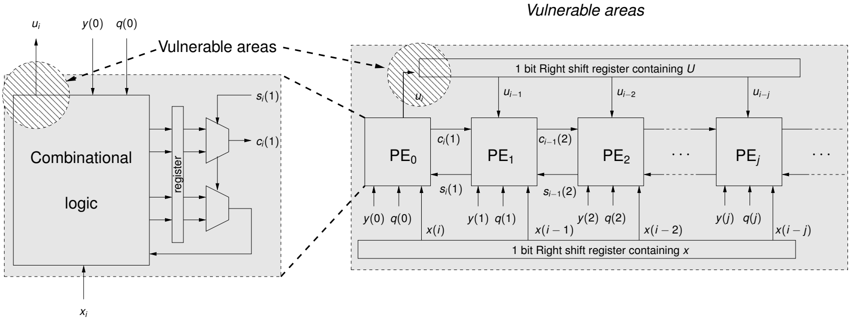


Tenca and Koç

Are the models realistic?

Fault injection has to be made on several rounds of the CIOS algorithm

q' or u_j must be isolated in the architecture:

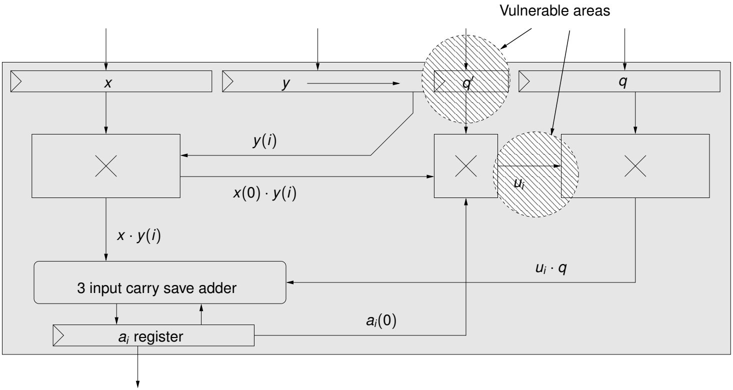


Huang et al.

Are the models realistic?

Fault injection has to be made on several rounds of the CIOS algorithm

q' or u_j must be isolated in the architecture:



Mentens et al.

Conclusion

Fault model: force the highest-order bits of a small value to zero
$\Rightarrow$ **Another attack!**

Summary

- RSA–CRT and Montgomery multiplication are widespread
- Attacks defeat unprotected RSA–CRT signatures **with any padding scheme**
- No need to know the message (works with **message recovery**)
- **First fault attacks effective** against the widespread RSA–PSS scheme (proven secure against random faults)
- **Realistic faults** (yet to be implemented)
- **Countermeasure:** verifying the signature

Thank you!