Efficient Leak Resistant Modular Exponentiation in RNS

Andrea Lesavourey, Christophe Negre, Thomas Plantard

▶ To cite this version:

Andrea Lesavourey, Christophe Negre, Thomas Plantard. Efficient Leak Resistant Modular Exponentiation in RNS. ARITH: Computer Arithmetic, Jul 2017, London, United Kingdom. pp.156-163, 10.1109/ARITH.2017.39 . lirmm-01925642

HAL Id: lirmm-01925642
https://hal-lirmm.ccsd.cnrs.fr/lirmm-01925642

Submitted on 16 Nov 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Efficient Leak Resistant Modular Exponentiation in RNS

Andrea Lesavourey[(1)], **Christophe Negre**[(1)] and Thomas Plantard[(2)]

(1) DALI (UPVD) and LIRMM (Univ. of Montpellier, CNRS), Perpignan, France

(2) CCISR, SCIT, University of Wollongong, Wollongong, Australia

24-th Symposium on Computer Arithmetic,
London, July 26, 2017

Outline

Outline

1. Cryptography
   - RSA cryptosystem
   - Power analysis
   - Montgomery multiplication in RNS
2. Randomized modular exponentiation in RNS
   - Randomized Montgomery multiplication
   - Proposed approach
   - Level of randomization
3. Conclusion

RSA encryption (Rivest, Shamir and Adleman)

Bob chooses p and q two large prime numbers and computes $N = pq$. He generates E and D two integers such that $ED = 1 \pmod{(p-1)(q-1)}$.

- Public Key: N, D.
- Private Key: E, p, q.
- Alice encrypts a message m by: $c = m^D \mod N$.
- Bob decrypts c by doing: $c^E = m^{ED} \mod N = m$.

An algorithm for modular exponentiation : Right-to-left Square-and-multiply

Require: A modulus N, an integer $X \in [0, N[$ and an exponent $E = (e_{\ell-1}, \ldots, e_0)_2$
Ensure: $R = X^E \pmod{N}$

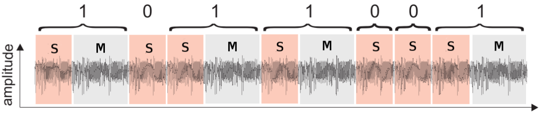
```
1: R ← 1
2: Z ← X
3: for i from 0 to ℓ − 1 do
4:     if e_i = 1 then
5:         R ← R × Z (mod N)
6:     end if
7:     Z ← Z^2 (mod N)
8: end for
9: return R
```

$$X^E = X^{\sum_{i=0}^{\ell-1} e_i 2^i}$$

$$X^E = X^{e_{\ell-1}2^{\ell-1}} \times \cdots \times X^{e_1 2^1} \times X^{e_0 2^0}$$

Simple power analysis

$E = (e_\ell, \dots, e_0)_2$ and $X \in [0, N[$



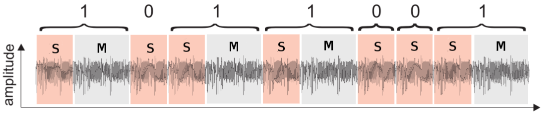
$\uparrow$

Square-and-multiply

$R \leftarrow 1$
$Z \leftarrow X$
for $i = 0$ **to** $\ell - 1$ **do**
  if $e_i = 1$ **then**
    $R \leftarrow R \cdot Z \mod N$
  endif
  $Z \leftarrow Z^2 \mod N$
endfor
return(R)

Simple power analysis

$E = (e_\ell, \ldots, e_0)_2$ and $X \in [0, N[$



Square-and-multiply

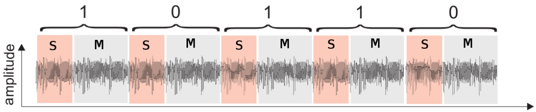
$R \leftarrow 1$
$Z \leftarrow X$
for $i = 0$ **to** $\ell - 1$ **do**
  if $e_i = 1$ **then**
    $R \leftarrow R \cdot Z \mod N$
  endif
  $Z \leftarrow Z^2 \mod N$
endfor
return(R)

Square-and-multiply-always

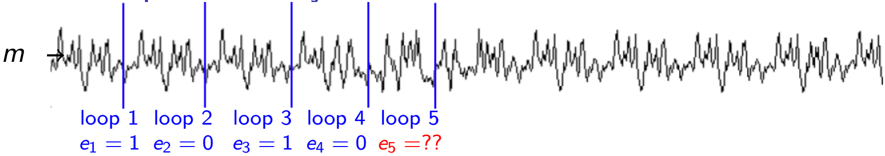
$R_0 \leftarrow 1$
$R_1 \leftarrow 1$
$Z \leftarrow X$
for $i = 0$ **to** $\ell - 1$ **do**
  if $e_i = 0$ **then**
    $R_0 \leftarrow R_0 \cdot Z \mod N$
  else
    $R_1 \leftarrow R_1 \cdot Z \mod N$
  endif
endfor
  $Z \leftarrow Z^2 \mod N$
return(R_1)

Montgomery-ladder

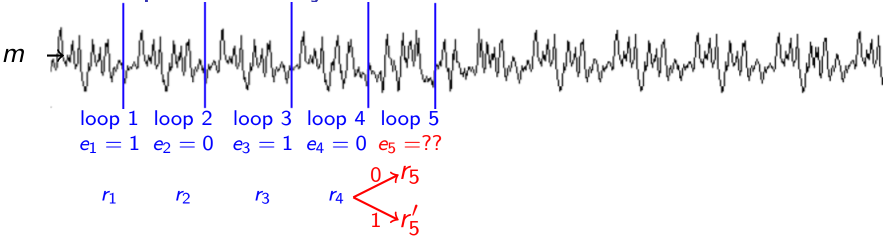
$R \leftarrow 1$
$R' \leftarrow X$
for $i = \ell$ **to** 1 **do**
  if $k_i = 1$ **then**
    $R \leftarrow R \cdot R' \mod N$
    $R' \leftarrow R'^2 \mod N$
  else
    $R' \leftarrow R \cdot R' \mod N$
    $R \leftarrow R^2$
  endif
endfor
return(R)



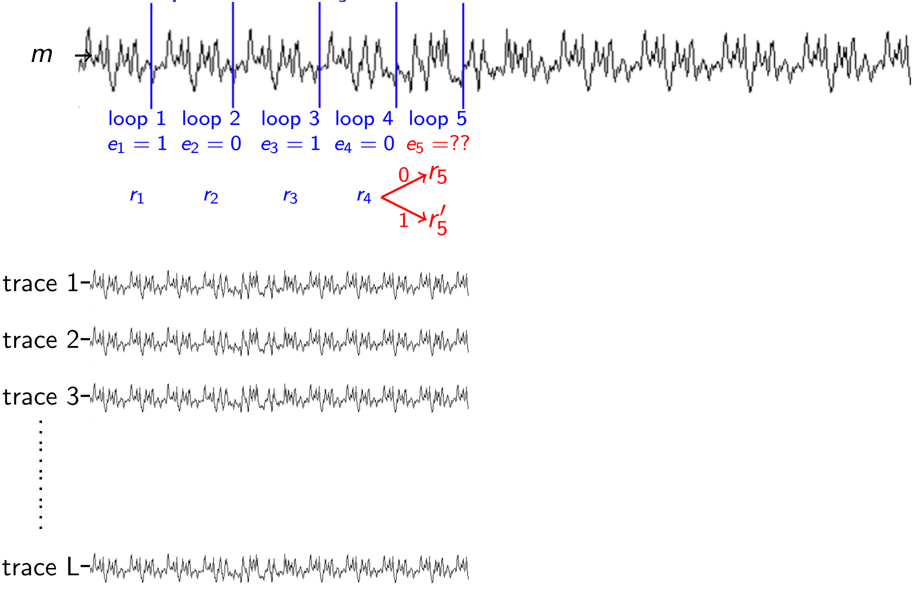
Differential power analysis



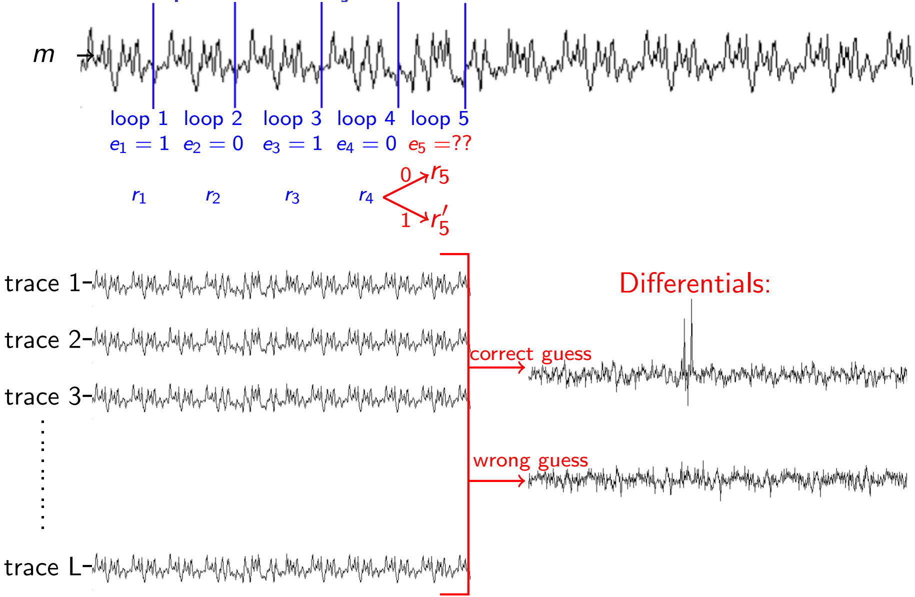
Differential power analysis



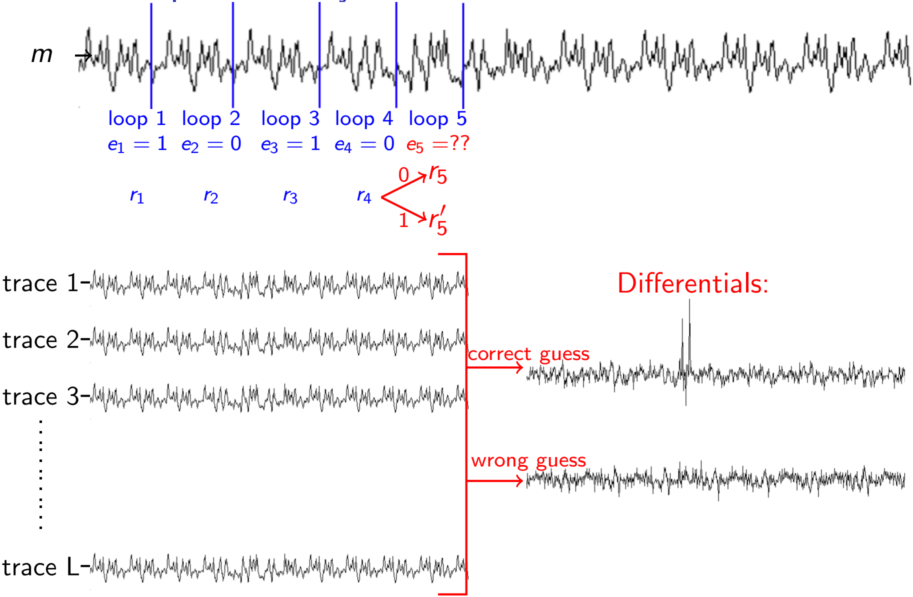
Differential power analysis



Differential power analysis



Differential power analysis



Counter-measure: Randomization of the exponent and data.

Montgomery multiplication

Basic modular multiplication. For $X, Y \in [0, N[$

1. Product. $Z \leftarrow X \times Y$
2. Reduction. $Q \leftarrow \lfloor Z/N \rfloor$ and $R \leftarrow Z - Q \times N$

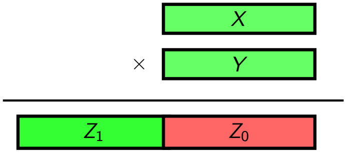
Montgomery multiplication

Basic modular multiplication. For $X, Y \in [0, N[$

1. Product. $Z \leftarrow X \times Y$
2. Reduction. $Q \leftarrow \lfloor Z/N \rfloor$ and $R \leftarrow Z - Q \times N$

Montgomery Multiplication

Require: $X, Y \in [0, N[$ and
$A = 2^n > N$
Ensure: $R = X \times Y \times A^{-1} \pmod{N}$
1: $Z \leftarrow X \times Y$
2: $Q \leftarrow N^{-1} \times Z \pmod{A}$
3: $R \leftarrow (Z - Q \times N)/A$



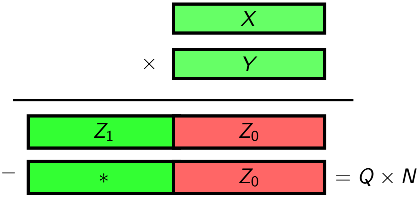
Montgomery multiplication

Basic modular multiplication. For $X, Y \in [0, N[$

1. Product. $Z \leftarrow X \times Y$
2. Reduction. $Q \leftarrow \lfloor Z/N \rfloor$ and $R \leftarrow Z - Q \times N$

Montgomery Multiplication

Require: $X, Y \in [0, N[$ and
$A = 2^n > N$
Ensure: $R = X \times Y \times A^{-1} \pmod N$
1: $Z \leftarrow X \times Y$
2: $Q \leftarrow N^{-1} \times Z \pmod A$
3: $R \leftarrow (Z - Q \times N)/A$



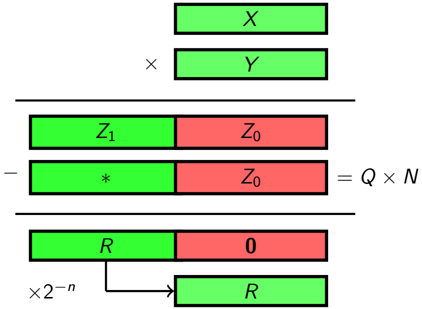
Montgomery multiplication

Basic modular multiplication. For $X, Y \in [0, N[$

1. Product. $Z \leftarrow X \times Y$
2. Reduction. $Q \leftarrow \lfloor Z/N \rfloor$ and $R \leftarrow Z - Q \times N$

Montgomery Multiplication

Require: $X, Y \in [0, N[$ and
$A = 2^n > N$
Ensure: $R = X \times Y \times A^{-1} \pmod N$
1: $Z \leftarrow X \times Y$
2: $Q \leftarrow N^{-1} \times Z \pmod A$
3: $R \leftarrow (Z - Q \times N)/A$



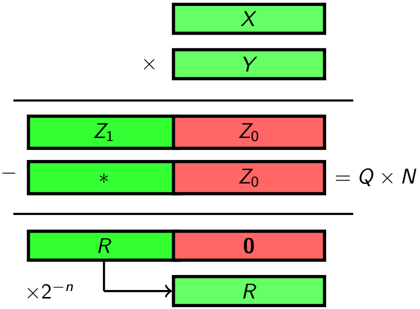
Montgomery multiplication

Basic modular multiplication. For $X, Y \in [0, N[$

1. Product. $Z \leftarrow X \times Y$
2. Reduction. $Q \leftarrow \lfloor Z/N \rfloor$ and $R \leftarrow Z - Q \times N$

Montgomery Multiplication

Require: $X, Y \in [0, N[$ and
$A = 2^n > N$
Ensure: $R = X \times Y \times A^{-1} \pmod N$
1: $Z \leftarrow X \times Y$
2: $Q \leftarrow N^{-1} \times Z \pmod A$
3: $R \leftarrow (Z - Q \times N)/A$



Montgomery representation.

1. $\widetilde{X} = XA \mod N$ provides
2. $MontMul(\widetilde{X}, \widetilde{Y}) = (XA) \times (YA) \times A^{-1} \mod N = XYA \mod N$

Montgomery multiplication in residue number system

- Let $\mathcal{A} = \{a_1, \ldots, a_t\}$ be a set t co-prime integers.

Montgomery multiplication in residue number system

- Let $\mathcal{A} = \{a_1, \ldots, a_t\}$ be a set t co-prime integers.
- An integer X such that $0 \leq X < A = \prod_{i=1}^{t} a_i$ is represented by

$$[X]_{\mathcal{A}} = (x_1 = X \mod a_1, \ldots, x_t = X \mod a_t).$$

Montgomery multiplication in residue number system

- Let $\mathcal{A} = \{a_1, \ldots, a_t\}$ be a set t co-prime integers.
- An integer X such that $0 \leq X < A = \prod_{i=1}^{t} a_i$ is represented by

$$[X]_{\mathcal{A}} = (x_1 = X \mod a_1, \ldots, x_t = X \mod a_t).$$

- The Chinese remainder theorem tell us that for op $\in \{+, \times\}$

$$[X]_{\mathcal{A}} \text{ op } [Y]_{\mathcal{A}} = ([x_1 \text{ op } y_1]_{a_1}, \ldots, [x_t \text{ op } y_t]_{a_t}) \Leftrightarrow X \text{ op } Y \mod A$$

Montgomery multiplication in residue number system

- Let $\mathcal{A} = \{a_1, \ldots, a_t\}$ be a set t co-prime integers.
- An integer X such that $0 \leq X < A = \prod_{i=1}^{t} a_i$ is represented by

$$[X]_{\mathcal{A}} = (x_1 = X \mod a_1, \ldots, x_t = X \mod a_t).$$

- The Chinese remainder theorem tell us that for op $\in \{+, \times\}$

$$[X]_{\mathcal{A}} \text{ op } [Y]_{\mathcal{A}} = ([x_1 \text{ op } y_1]_{a_1}, \ldots, [x_t \text{ op } y_t]_{a_t}) \Leftrightarrow X \text{ op } Y \mod A$$

Montgomery Multiplication in RNS

Require: X, Y in $\mathcal{A} \cup \mathcal{B}$
Ensure: $XYA^{-1} \mod N$ in $\mathcal{A} \cup \mathcal{B}$

1: $[Q]_{\mathcal{A}} \leftarrow [XYN^{-1}]_{\mathcal{A}}$

3: $[Z]_{\mathcal{B}} \leftarrow [(XY - QN)A^{-1}]_{\mathcal{B}}$

5: **return** $(Z_{\mathcal{A} \cup \mathcal{B}})$

Montgomery multiplication in residue number system

- Let $\mathcal{A} = \{a_1, \ldots, a_t\}$ be a set t co-prime integers.
- An integer X such that $0 \leq X < A = \prod_{i=1}^{t} a_i$ is represented by

$$[X]_{\mathcal{A}} = (x_1 = X \mod a_1, \ldots, x_t = X \mod a_t).$$

- The Chinese remainder theorem tell us that for op $\in \{+, \times\}$

$$[X]_{\mathcal{A}} \text{ op } [Y]_{\mathcal{A}} = ([x_1 \text{ op } y_1]_{a_1}, \ldots, [x_t \text{ op } y_t]_{a_t}) \Leftrightarrow X \text{ op } Y \mod A$$

Montgomery Multiplication in RNS

Require: X, Y in $\mathcal{A} \cup \mathcal{B}$

Ensure: $XYA^{-1} \mod N$ in $\mathcal{A} \cup \mathcal{B}$

$[Q]_{\mathcal{A}} \leftarrow [XYN^{-1}]_{\mathcal{A}}$

$[Q]_{\mathcal{B}} \leftarrow BE_{\mathcal{A} \to \mathcal{B}}([Q]_{\mathcal{A}})$

$[Z]_{\mathcal{B}} \leftarrow [(XY - QN)A^{-1}]_{\mathcal{B}}$

$[Z]_{\mathcal{A}} \leftarrow BE_{\mathcal{B} \to \mathcal{A}}([Z]_{\mathcal{B}})$

return $(Z_{\mathcal{A} \cup \mathcal{B}})$

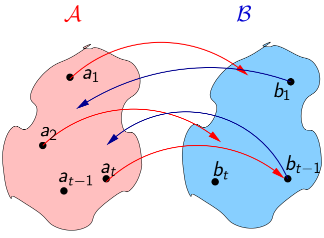
Outline

Randomization in RNS (LRA CHES 2004)

We have

$$\widetilde{X}_{old} = [XA_{old}]_{\mathcal{A}_{old} \cup \mathcal{B}_{old}}$$

we permute the basis elements $\mathcal{A}_{old} \cup \mathcal{B}_{old} \rightarrow \mathcal{A}_{new} \cup \mathcal{B}_{new}$



this leads to a new representation of X

$$\widetilde{X}_{new} = [XA_{new}]_{\mathcal{A}_{new} \cup \mathcal{B}_{new}}$$

Cost

Two Montgomery multiplications :

$$XA_{old} \mod N \rightarrow XA_{old}A_{new} \mod N \rightarrow XA_{new} \mod N.$$

Randomized square-and-multiply-always

- Input: N, $X \in [0, N[$, $E = (e_{\ell-1}, \ldots, e_0)_2$ and $\mathcal{M} = \{m_1, \ldots, m_{2t}\}$.
- Output: $X^E \mod N$

Square-and-mult-always

$\mathcal{A}, \mathcal{B} \xleftarrow{\text{random split}} \mathcal{M}$
$\widetilde{Z} \leftarrow [\widetilde{X}]_{\mathcal{A}\cup\mathcal{B}}$,
$\widetilde{R}_0 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$, $\widetilde{R}_1 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$
for i **from** 0 **to** $\ell - 1$ **do**
  $\widetilde{R}_{e_i} \leftarrow \text{MM\_RNS}(\widetilde{R}_{e_i}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
  $\widetilde{Z} \leftarrow \text{MM\_RNS}(\widetilde{Z}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
end for
return $\widetilde{R}_1$

Randomized square-and-multiply-always

- Input: N, $X \in [0, N[$, $E = (e_{\ell-1}, \ldots, e_0)_2$ and $\mathcal{M} = \{m_1, \ldots, m_{2t}\}$.
- Output: $X^E \mod N$

Randomized Square-and-mult-always

$\mathcal{A}, \mathcal{B} \xleftarrow{\text{random split}} \mathcal{M}$
$\widetilde{Z} \leftarrow [\widetilde{X}]_{\mathcal{A}\cup\mathcal{B}}$,
$\widetilde{R}_0 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$, $\widetilde{R}_1 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$
for i **from** 0 **to** $\ell - 1$ **do**
  $\widetilde{R}_{e_i} \leftarrow \text{MM\_RNS}(\widetilde{R}_{e_i}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
  $\widetilde{Z} \leftarrow \text{MM\_RNS}(\widetilde{Z}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
  $\text{Randomise}(\mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
  $\widetilde{Z} \leftarrow \text{Update}(\widetilde{Z}, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
  $\widetilde{R}_0 \leftarrow \text{Update}(\widetilde{R}_0, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
  $\widetilde{R}_1 \leftarrow \text{Update}(\widetilde{R}_1, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
end for
return $\widetilde{R}_1$

Randomized square-and-multiply-always

- Input: N, $X \in [0, N[$, $E = (e_{\ell-1}, \ldots, e_0)_2$ and $\mathcal{M} = \{m_1, \ldots, m_{2t}\}$.
- Output: $X^E \mod N$

Randomized Square-and-mult-always

$\mathcal{A}, \mathcal{B} \xleftarrow{\text{random split}} \mathcal{M}$
$\widetilde{Z} \leftarrow [\widetilde{X}]_{\mathcal{A}\cup\mathcal{B}}$,
$\widetilde{R}_0 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}, \widetilde{R}_1 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$
for i **from** 0 **to** $\ell - 1$ **do**
  $\widetilde{R}_{e_i} \leftarrow \text{MM\_RNS}(\widetilde{R}_{e_i}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
  $\widetilde{Z} \leftarrow \text{MM\_RNS}(\widetilde{Z}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
  $\text{Randomise}(\mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
  $\widetilde{Z} \leftarrow \text{Update}(\widetilde{Z}, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
  $\cancel{\widetilde{R}_0 \leftarrow \text{Update}(\widetilde{R}_0, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})}$
  $\cancel{\widetilde{R}_1 \leftarrow \text{Update}(\widetilde{R}_1, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})}$
end for
return $\widetilde{R}_1$

Randomized square-and-multiply-always

- Input: N, $X \in [0, N[$, $E = (e_{\ell-1}, \ldots, e_0)_2$ and $\mathcal{M} = \{m_1, \ldots, m_{2t}\}$.
- Output: $X^E \mod N$

Randomized Square-and-mult-always

$\mathcal{A}, \mathcal{B} \xleftarrow{\text{random split}} \mathcal{M}$
$\widetilde{Z} \leftarrow [\widetilde{X}]_{\mathcal{A}\cup\mathcal{B}}$,
$\widetilde{R}_0 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$, $\widetilde{R}_1 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$
for i **from** 0 **to** $\ell - 1$ **do**
    $\widetilde{R}_{e_i} \leftarrow \mathrm{MM\_RNS}(\widetilde{R}_{e_i}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
    $\widetilde{Z} \leftarrow \mathrm{MM\_RNS}(\widetilde{Z}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
    $\mathrm{Randomise}(\mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
    $\widetilde{Z} \leftarrow \mathrm{Update}(\widetilde{Z}, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
    $\cancel{\widetilde{R}_0 \leftarrow \mathrm{Update}(\widetilde{R}_0, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})}$
    $\cancel{\widetilde{R}_1 \leftarrow \mathrm{Update}(\widetilde{R}_1, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})}$
end for
return $\widetilde{R}_1$

Proposed

$\mathcal{A}, \mathcal{B} \xleftarrow{\text{random split}} \mathcal{M}$
$\widetilde{Z} \leftarrow [\widetilde{X}]_{\mathcal{A}\cup\mathcal{B}}$,
$\widetilde{R}_0 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$, $\widetilde{R}_1 \leftarrow [\widetilde{1}]_{\mathcal{A}\cup\mathcal{B}}$
for i **from** 0 **to** $\ell - 1$ **do**
    $\mathcal{A}'_{e_i}, \mathcal{B}'_{e_i} \xleftarrow{\text{random split}} \mathcal{M}$
    $\widetilde{R}_{e_i} \leftarrow \mathrm{MM\_RNS}(\widetilde{R}_{e_i}, \widetilde{Z}, \mathcal{A}'_{e_i}, \mathcal{B}'_{e_i})$
    $\widetilde{Z} \leftarrow \mathrm{MM\_RNS}(\widetilde{Z}, \widetilde{Z}, \mathcal{A}, \mathcal{B})$
    $\mathrm{Randomise}(\mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
    $\widetilde{Z} \leftarrow \mathrm{Update}(\widetilde{Z}, \mathcal{A}_{old}, \mathcal{B}_{old}, \mathcal{A}, \mathcal{B})$
end for
return $\widetilde{R}_1$

Example

For $E = 7 = (111)_2$ and $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$

Example

For $E = 7 = (111)_2$ and $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$

- Initialization: $\mathcal{A} = \{m_1, m_2\}, \mathcal{B} = \{m_3, m_4\}$ leads to

$$\begin{aligned} R_1 &= m_1 m_2 \mod N \\ Z &= X m_1 m_2 \mod N \end{aligned}$$

Example

For $E = 7 = (111)_2$ and $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$

- Initialization: $\mathcal{A} = \{m_1, m_2\}, \mathcal{B} = \{m_3, m_4\}$ leads to

$$\begin{aligned} R_1 &= m_1 m_2 \mod N \\ Z &= X m_1 m_2 \mod N \end{aligned}$$

- Loop 1: $\mathcal{A}_1 = \{m_2, m_4\}, \mathcal{B}_1 = \{m_1, m_3\}$ we get

$$R_1 = (m_1 m_2) \times \underbrace{(X m_1 m_2)}_{Z} \times \underbrace{(m_2^{-1} m_4^{-1})}_{\text{Mont. factor}} = X m_1^2 m_2 m_4^{-1}$$

Example

For $E = 7 = (111)_2$ and $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$

- Initialization: $\mathcal{A} = \{m_1, m_2\}, \mathcal{B} = \{m_3, m_4\}$ leads to

$$\begin{aligned} R_1 &= m_1 m_2 \mod N \\ Z &= X m_1 m_2 \mod N \end{aligned}$$

- Loop 1: $\mathcal{A}_1 = \{m_2, m_4\}, \mathcal{B}_1 = \{m_1, m_3\}$ we get

$$R_1 = (m_1 m_2) \times \underbrace{(X m_1 m_2)}_{Z} \times \underbrace{(m_2^{-1} m_4^{-1})}_{\text{Mont. factor}} = X m_1^2 m_2 m_4^{-1}$$

$\mathcal{A} = \{m_1, m_3\}, \mathcal{B} = \{m_2, m_4\}$ leads to

$$Z = X^2 m_1 m_3$$

Example

For $E = 7 = (111)_2$ and $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$

- Initialization: $\mathcal{A} = \{m_1, m_2\}, \mathcal{B} = \{m_3, m_4\}$ leads to

$$\begin{aligned} R_1 &= m_1 m_2 \mod N \\ Z &= X m_1 m_2 \mod N \end{aligned}$$

- Loop 1: $\mathcal{A}_1 = \{m_2, m_4\}, \mathcal{B}_1 = \{m_1, m_3\}$ we get

$$R_1 = (m_1 m_2) \times \underbrace{(X m_1 m_2)}_{Z} \times \underbrace{(m_2^{-1} m_4^{-1})}_{\text{Mont. factor}} = X m_1^2 m_2 m_4^{-1}$$

  $\mathcal{A} = \{m_1, m_3\}, \mathcal{B} = \{m_2, m_4\}$ leads to

$$Z = X^2 m_1 m_3$$

- Loop 2: $\mathcal{A}_1 = \{m_1, m_4\}, \mathcal{B}_1 = \{m_2, m_3\}$ we get

$$R_1 = X m_1^2 m_2 m_4^{-1} \times (X^2 m_1 m_3) \times (m_1^{-1} m_4^{-1}) = X^3 m_1^2 m_2 m_3 m_4^{-2}$$

Example

For $E = 7 = (111)_2$ and $\mathcal{M} = \{m_1, m_2, m_3, m_4\}$

- Initialization: $\mathcal{A} = \{m_1, m_2\}, \mathcal{B} = \{m_3, m_4\}$ leads to

$$\begin{aligned} R_1 &= m_1 m_2 \mod N \\ Z &= X m_1 m_2 \mod N \end{aligned}$$

- Loop 1: $\mathcal{A}_1 = \{m_2, m_4\}, \mathcal{B}_1 = \{m_1, m_3\}$ we get

$$R_1 = (m_1 m_2) \times \underbrace{(X m_1 m_2)}_{Z} \times \underbrace{(m_2^{-1} m_4^{-1})}_{\text{Mont. factor}} = X m_1^2 m_2 m_4^{-1}$$

  $\mathcal{A} = \{m_1, m_3\}, \mathcal{B} = \{m_2, m_4\}$ leads to

$$Z = X^2 m_1 m_3$$

- Loop 2: $\mathcal{A}_1 = \{m_1, m_4\}, \mathcal{B}_1 = \{m_2, m_3\}$ we get

$$R_1 = X m_1^2 m_2 m_4^{-1} \times (X^2 m_1 m_3) \times (m_1^{-1} m_4^{-1}) = X^3 m_1^2 m_2 m_3 m_4^{-2}$$

- Etc.

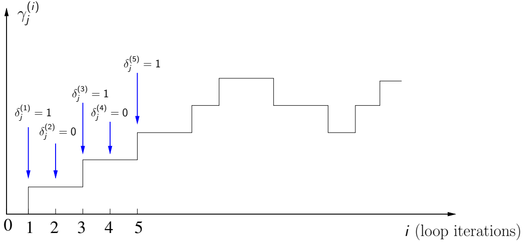
Random evolution of the mask

After i loop iterations we have

$$\widetilde{R}_1^{(i)} = X^{\sum_{j=0}^{i-1} e_j 2^j} \times \prod_{j=0}^{2t} m_j^{\gamma_j^{(i)}} \mod N$$

and each $\gamma_j^{(i)}$ evolves randomly as

$$\gamma_j^{(i+1)} = \gamma_j^{(i)} + \delta_j^{(i)} \text{ with } \delta_j^{(i)} \in \{-1, 0, 1\} \text{ and } \begin{cases} \mathbb{P}(\delta_j^{(i)} = 1) = 1/8, \\ \mathbb{P}(\delta_j^{(i)} = -1) = 1/8, \\ \mathbb{P}(\delta_j^{(i)} = 0) = 3/4. \end{cases}$$



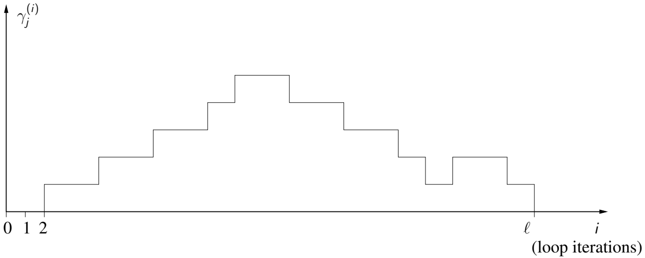
Removing the final mask

Problem: at the end we have to remove the final mask $\prod_{j=1}^{2t} m_j^{\gamma_j^{(\ell)}}$ from

$$\widetilde{X} = X^E \cdot \prod_{j=1}^{2t} m_j^{\gamma_j^{(\ell)}} \mod N.$$

Strategy: we force $\gamma_j^{(\ell)}$ to be equal 0 as follows

- During the first half of the iterations each $\gamma_j^{(i)}$ evolves freely.
- During the second half we constrain each $|\gamma_j^{(i)}|$ to decrease toward 0.



Level of randomization

- The probabilities of the mask exponents satisfy

$$
\begin{aligned}
\mathbb{P}(\gamma_j^{(i)} = d) &= \sum_{k=d}^{d+\lfloor (i-d)/2 \rfloor} \binom{i}{k} \binom{i-k}{k-d} \left(\frac{1}{8}\right)^{2k-d} \left(\frac{3}{4}\right)^{i-2k+d} \\
\mathbb{P}(\Gamma^{(i)} = \Gamma) &\leq \prod_{j=1}^{t} \mathbb{P}(\gamma_j^{(i)} = \gamma_j) \leq \prod_{j=1}^{t} \mathbb{P}(\gamma_j^{(i)} = 0)
\end{aligned}
$$

Level of randomization

- The probabilities of the mask exponents satisfy

$$\begin{array}{rcl} \mathbb{P}(\gamma_j^{(i)} = d) & = & \sum_{k=d}^{d+\lfloor (i-d)/2 \rfloor} \binom{i}{k}\binom{i-k}{k-d}\left(\frac{1}{8}\right)^{2k-d}\left(\frac{3}{4}\right)^{i-2k+d} \\ \mathbb{P}(\Gamma^{(i)} = \Gamma) & \leq & \prod_{j=1}^{t} \mathbb{P}(\gamma_j^{(i)} = \gamma_j) \leq \prod_{j=1}^{t} \mathbb{P}(\gamma_j^{(i)} = 0) \end{array}$$

- Comparison: for a 2048-bit RSA modulus and $t = 32$:
  - CHES 04:
    - Montgomery-ladder,
    - 4MM_RNS per randomization,
    - all masks are controled.
  - Proposed:
    - right-left square-and-multiply-always,
    - 2MM_RNS per randomization
    - the masks for R_0 and R_1 are not controled.

Approach	loop 1	loop 5	loop 10	loop 50	loop 100
CHES 04	$4.17 \cdot 10^{-38}$	$4.17 \cdot 10^{-38}$	$4.17 \cdot 10^{-38}$	$4.17 \cdot 10^{-38}$	$4.17 \cdot 10^{-38}$
Proposed	10^{-8}	$5 \cdot 10^{-28}$	$1.7 \cdot 10^{-38}$	$2.69 \cdot 10^{-61}$	$5.75 \cdot 10^{-71}$

Outline

Conclusion

Secure embedded implementation of RSA:

- Randomized modular exponentiation
- But leak resistant arithmetic (CHES 04) is costly: 4 MM_RNS per randomization

Conclusion

Secure embedded implementation of RSA:

- Randomized modular exponentiation
- But leak resistant arithmetic (CHES 04) is costly: 4 MM_RNS per randomization

We proposed:

- To apply LRA to right-to-left exponentiation.
- Avoid some correction of Montgomery Factor.
- This decreases the computational cost: 2 MM_RNS per randomization.
- Increases the level of randomization after a small number of loop.

Conclusion

Secure embedded implementation of RSA:

- Randomized modular exponentiation
- But leak resistant arithmetic (CHES 04) is costly: 4 MM_RNS per randomization

We proposed:

- To apply LRA to right-to-left exponentiation.
- Avoid some correction of Montgomery Factor.
- This decreases the computational cost: 2 MM_RNS per randomization.
- Increases the level of randomization after a small number of loop.

Perspectives:

- A better estimation of the level of randomization.
- Is it a good counter-measure against horizontal power analysis ?

Thank you for your attention!