Making RSA-PSS Provably Secure Against Non-Random Faults

Gilles Barthe (IMDEA)
François Dupressoir (IMDEA)
Pierre-Alain Fouque (Univ. Rennes 1)
Benjamin Grégoire (Inria)
Mehdi Tibouchi (NTT)
Jean-Christophe Zapalowicz (Inria)

CHES 2014

RSA signatures

RSA signatures still widely used (Software & Embedded devices)

- Chinese Remainder Theorem (CRT) technique for efficiency
- Bellcore attack to avoid
  - Fault attack
  - Inject a fault during one of the half exponentiations
  - Message + faulted signature $\Rightarrow$ Secret key

RSA signatures

RSA signatures still widely used (Software & Embedded devices)

- Chinese Remainder Theorem (CRT) technique for efficiency

- Bellcore attack to avoid
  - Fault attack
  - Inject a fault during one of the half exponentiations
  - Message + faulted signature $\Rightarrow$ Secret key

RSA signatures

RSA signatures still widely used (Software & Embedded devices)

- Chinese Remainder Theorem (CRT) technique for efficiency

- Bellcore attack to avoid
  - Fault attack
  - Inject a fault during one of the half exponentiations
  - Message + faulted signature $\Rightarrow$ Secret key

RSA signatures and fault attacks

Bellcore attack (or some extensions) efficient against

- no encoding
- deterministic encodings
- some randomized padding schemes (ISO 9796-2, EMV, ...)

RSA signatures and fault attacks

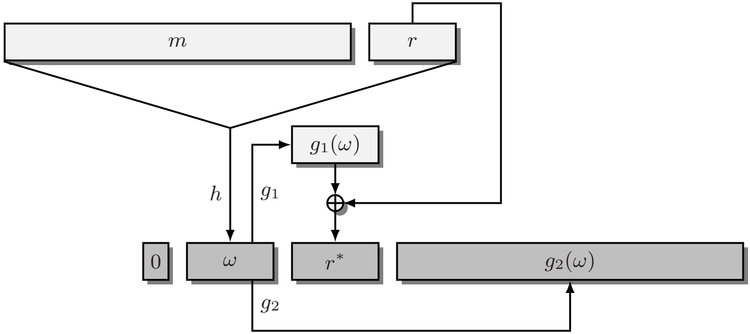
Bellcore attack (or some extensions) efficient against

- no encoding
- deterministic encodings
- some randomized padding schemes (ISO 9796-2, EMV, ...)

But RSA-PSS (randomized padding scheme proposed by Bellare and Rogaway, Eurocrypt 96) is secure against random faults
(proven by Coron and Mandal, Asiacrypt 2009)

RSA-PSS

h outputs bitstrings of length k_h ($\{0,1\}^* \times \{0,1\}^{k_0} \rightarrow \{0,1\}^{k_h}$)
g (mixing g_1 and g_2) outputs bitstrings of length k_g ($\{0,1\}^{k_h} \rightarrow \{0,1\}^{k_g}$)
$k_h + k_g + 1 = n$ and $k_0 < k_g$



RSA signatures and fault attacks

Different non-random fault models proposed in 2012
(Fouque, Guillermin, Leresteux, Tibouchi, Zapalowicz, CHES 2012)

- Require Montgomery multiplication to be used
- Apply to any padding function

RSA signatures and fault attacks

Different non-random fault models proposed in 2012
(Fouque, Guillermin, Leresteux, Tibouchi, Zapalowicz, CHES 2012)

- Require Montgomery multiplication to be used
- Apply to any padding function

Our Goal:

- To propose a countermeasure for RSA-PSS against a large class of non-random faults
  $\Rightarrow$ Extend Coron and Mandal's result to a stronger model
- To prove this countermeasure
- To formally verify the proof with the tool EasyCrypt

Fault model

Coron and Mandal's fault model:

- Correct value modulo p
- Random value modulo q

Our fault model:

- Correct value modulo p
- Precise value modulo q fixed by the adversary

Countermeasure

Simplest protection against fault attacks: to verify the signature

- $y' = S^e \bmod N$
- **If** $y' = y$ **then Return** S **Else Return** Error

Countermeasure

Simplest protection against fault attacks: to verify the signature

- $y' = S^e \bmod N$
- **If** $y' = y$ **then Return** S **Else Return** Error

But a test can be bypassed $\Rightarrow$ Infective countermeasure

- A result released all the time
- Gibberish when faulty computations occur

Protected signing algorithm

1: **function** Sign(sk, pk, m)
2: $r \leftarrow \{0,1\}^{k_0}$ ▷ Start of PSS padding
3: $y \leftarrow PSS(m, r)$

Protected signing algorithm

function Sign(sk, pk, m)
    $r \leftarrow \{0,1\}^{k_0}$ ▷ Start of PSS padding
    $y \leftarrow PSS(m, r)$
    $S_p \leftarrow y^{d_p} \bmod p$ ▷ Signature computation
    $S_q \leftarrow y^{d_q} \bmod q$
    $S \leftarrow (\alpha_p \cdot S_p + \alpha_q \cdot S_q) \bmod N$ ▷ $\alpha_p = q \cdot (q^{-1} \bmod p)$

Protected signing algorithm

function Sign(sk, pk, m)
    $r \leftarrow \{0,1\}^{k_0}$ $\triangleright$ Start of PSS padding
    $y \leftarrow PSS(m, r)$
    $S_p \leftarrow y^{d_p} \bmod p$ $\triangleright$ Signature computation
    $S_q \leftarrow y^{d_q} \bmod q$
    $S \leftarrow (\alpha_p \cdot S_p + \alpha_q \cdot S_q) \bmod N$ $\triangleright$ $\alpha_p = q \cdot (q^{-1} \bmod p)$
    $y' \leftarrow S^e \bmod N$

Protected signing algorithm

function Sign(sk, pk, m)
$r \leftarrow \{0,1\}^{k_0}$ ▷ Start of PSS padding
$y \leftarrow PSS(m, r)$
$S_p \leftarrow y^{d_p} \bmod p$ ▷ Signature computation
$S_q \leftarrow y^{d_q} \bmod q$
$S \leftarrow (\alpha_p \cdot S_p + \alpha_q \cdot S_q) \bmod N$ ▷ $\alpha_p = q \cdot (q^{-1} \bmod p)$
$y' \leftarrow S^e \bmod N$
$r' \leftarrow \{0,1\}^{\rho}$ ▷ Infective countermeasure
$S' \leftarrow S + r' \cdot (y - y') \bmod N$
return S'

Protected signing algorithm

function Sign(sk, pk, m)

$r \leftarrow \{0,1\}^{k_0}$ $\triangleright$ Start of PSS padding

$y \leftarrow PSS(m, r)$

$S_p \leftarrow y^{d_p} \bmod p$ $\triangleright$ Signature computation

$S_q \leftarrow y^{d_q} \bmod q$

$S \leftarrow (\alpha_p \cdot S_p + \alpha_q \cdot S_q) \bmod N$ $\triangleright$ $\alpha_p = q \cdot (q^{-1} \bmod p)$

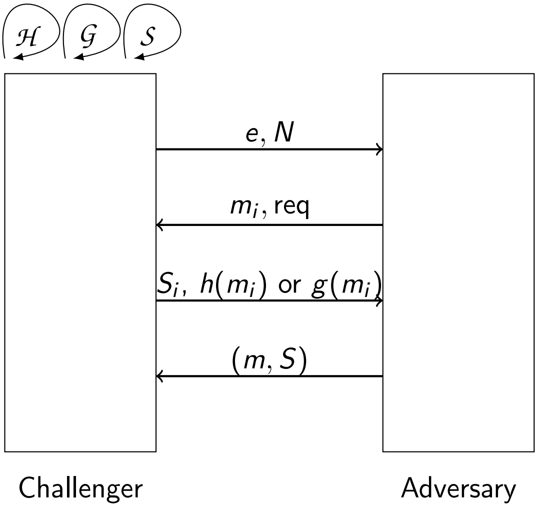
$y' \leftarrow S^e \bmod N$

$r' \leftarrow \{0,1\}^{\rho}$ $\triangleright$ Infective countermeasure

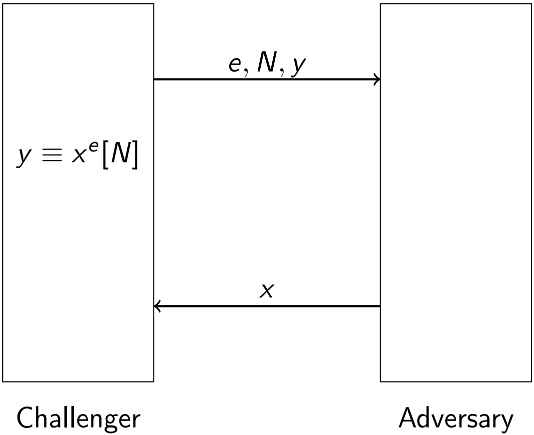
$S' \leftarrow S + r' \cdot (y - y') \bmod N$

return S'

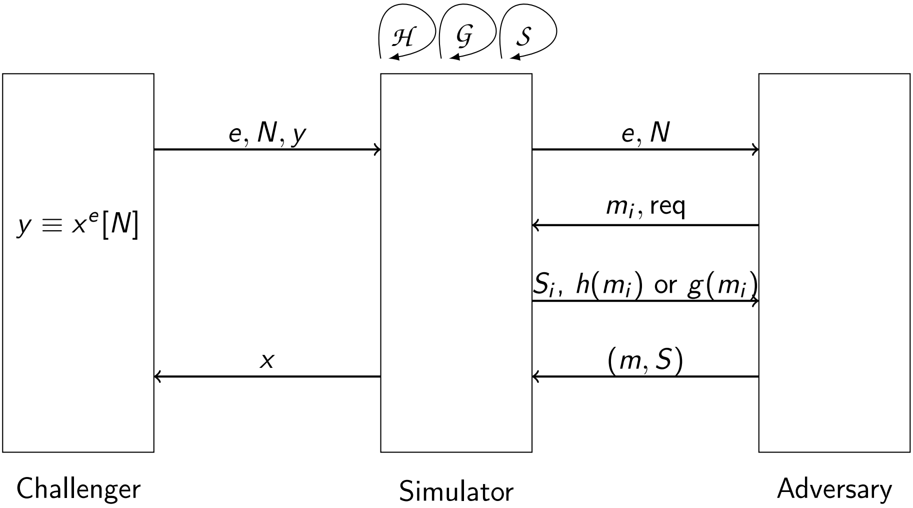
Size of r' (noted ρ) to define



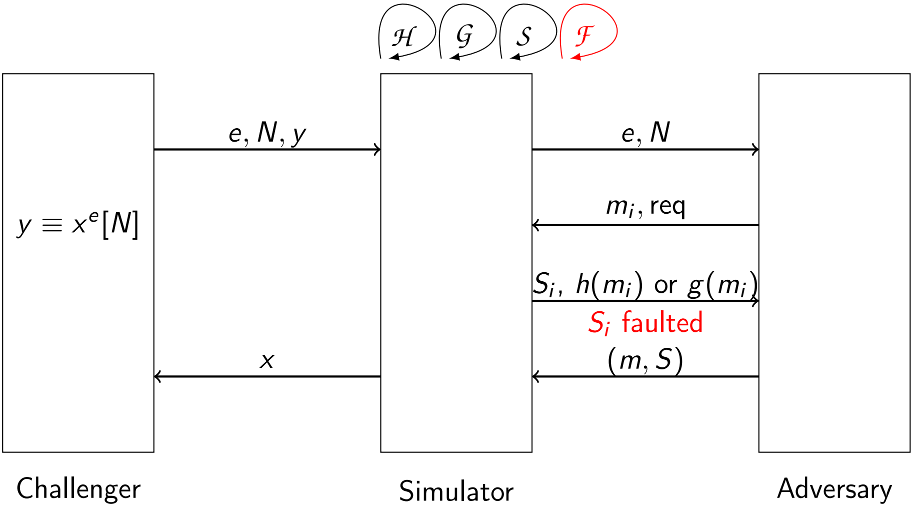
UF-CMA Challenge



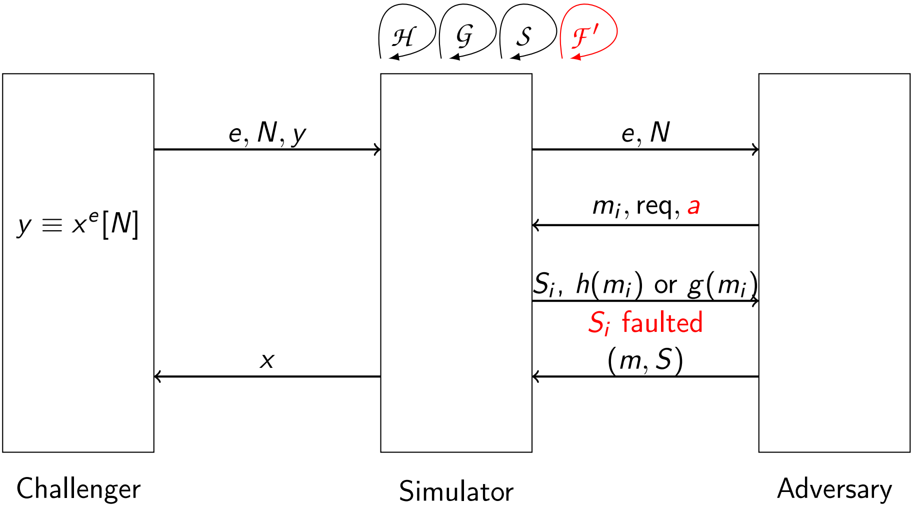
OW-RSA Challenge



RSA-PSS reduction (Bellare and Rogaway)



RSA-PSS reduction with random faults (Coron and Mandal)



RSA-PSS reduction with non-random faults

Main Theorem

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

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

$$\xrightarrow{\text{7 games later}}$$

Result

Given a CMA adversary $\mathcal{A}$ against the faulty signature scheme $(\mathcal{K}, \mathcal{S}, \mathcal{F}, \mathcal{V})$, we build a one-way inverter $\mathcal{I}$ such that

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

Behind this theorem

Our proof follows the same path as Coron and Mandal's proof

Behind this theorem

Our proof follows the same path as Coron and Mandal's proof

1 lemma for the distribution of the faulted signatures

- Prove that the faulted signatures provide no information
- We want to remove the computation of S' (which uses the secret key)

Behind this theorem

Our proof follows the same path as Coron and Mandal's proof

1 lemma for the distribution of the faulted signatures

- Prove that the faulted signatures provide no information
- We want to remove the computation of S' (which uses the secret key)

First lemma

$\{S' = y^d \cdot \alpha_p + \big(a + r'(y - a^e)\big) \cdot \alpha_q \bmod N \,|\, (y, r') \in [0, 2^{n-1}) \times [1, 2^\rho)\}$
$\{S' = y^d + r'y \bmod N \,|\, (y, r') \in [0, 2^{n-1}) \times [1, 2^\rho)\} \approx_s \mathcal{U}_N$
$\Rightarrow$For large enough N, it suffices to take ρ slightly larger than a given ε to obtain a statistical distance of $2^{-\varepsilon}$

Behind this theorem

1 lemma for the probability of guessing ω given S'
$\Rightarrow$ represents a bad event in the initial proof of RSA-PSS

Second lemma

$\Pr\left[\omega = \omega' | S'\right] \leq 3/2^{k_h}$
$\Rightarrow$ Requires ρ larger than k_h

Behind this theorem

1 lemma for the probability of guessing ω given S'
$\Rightarrow$ represents a bad event in the initial proof of RSA-PSS

Second lemma

$\Pr\left[\omega = \omega' | S'\right] \leq 3/2^{k_h}$
$\Rightarrow$ Requires ρ larger than k_h

However counts are more difficult to perform because of our stronger fault model
$\Rightarrow$ Use of more complex mathematic tools

- Dirichlet characters sum
- Generalized Riemann Hypothesis for best result on ρ
  (can be replaced by Polya-Vinogradov inequality or Burgess bound, but ρ increases)

Formal verification

Our proof = Game-based proof
⇒ can be handled by a computer-aided tool
⇒ EasyCrypt

Game = Program & Games transition = Approximate program equivalence

EasyCrypt: SMT based interactive proof engine
Already done:

- proof of standard crypto schemes (FDH, OAEP, ...)
- proof of a OAEP implementation with side channel
- Framework for building higher level tools
  - ZooCrypt: Automatic synthesis of padding based encryption schemes
  - Automatic verification of batch signature algorithms
  - Automatic verification of assumptions in the generic group model
  - Automatic synthesis of fault attacks

Without computer-aided tool:

- Lot of work for writing a proof
- Lot of work for trusting it

Without computer-aided tool:

- Lot of work for writing a proof
- Lot of work for trusting it

With computer-aided tool:

- Lot of work for writing a proof
- Easy to trust it

Without computer-aided tool:

- Lot of work for writing a proof
- Lot of work for trusting it

With computer-aided tool:

- Lot of work for writing a proof
- Easy to trust it

As an example, Coron and Mandal's proof

- Correct result
- **But** glitch in the proof

Conclusion

Proven infective countermeasure for protecting RSA-PSS against a stronger fault model

Proof verified with EasyCrypt

Another step towards the provable security in the context of fault attacks

Thank you for your attention