Fault Attacks on RSA Signatures with Partially Unknown Messages

Jean-Sébastien Coron[1] Antoine Joux[2] Ilya Kizhvatov[1]
David Naccache[3] Pascal Paillier[4]

[1]Université du Luxembourg

[2]DGA and Université de Versailles

[3]École Normale Supérieure

[4]CryptoExperts

CHES 2009, Lausanne, Switzerland

Outline

Outline

RSA with Chinese Remaindering (RSA-CRT)

Modulus $N = pq$, key pair (e, d), message m, padding function μ

Signing:

1. $\sigma_p = \mu(m)^d \mod p$
2. $\sigma_q = \mu(m)^d \mod q$
3. recombination: $\sigma = CRT(\sigma_p, \sigma_q) = \mu(m)^d \mod N$

Verification: $\sigma^e = \mu(m) \mod N$

CRT gives up to 4x speedup compared to the straightforward RSA implementation

The Bellcore Attack on RSA-CRT [Boneh et al. '96]

Signing

1. $\sigma_p = \mu(m)^d \mod p$
2. $\sigma'_q \neq \mu(m)^d \mod q \longleftarrow$ **fault**
3. $\sigma' = CRT(\sigma_p, \sigma_q)$ faulty signature

Verification: $\sigma'^e = \mu(m) \mod p$, $\sigma'^e \neq \mu(m) \mod q$

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

Applies to

- any deterministic RSA padding
  Example: FDH $\sigma = H(m)^d \mod N, \quad H : \{0,1\}^* \mapsto \mathbb{Z}_N$
- probabilistic signature schemes where the randomizer r is sent along with the signature
  Example: PFDH $\sigma = H(m \| r)^d \bmod N$

The Fault Attacker's Deadlock

Partially-Known Messages

Example: $\sigma = (m\|r)^d \bmod N$

r is a random nonce **not** sent along with σ

Deadlock: given σ', the attacker only gets the **faulty** padded message σ'^e and therefore can neither retrieve r nor infer $(m\|r)$. So he/she cannot compute

$$\gcd(\sigma'^e - (m\|r) \bmod N, N) = p$$

- inducing faults in many signatures does not help since different r values are used in successive signatures
- short r can be guessed by exhaustive search

The New Result

Extension of the Bellcore attack to a large class of partially known message configurations, in particular to ISO/IEC 9796-2

Overcoming the deadlock

- recovering the **u**nknown **m**essage **p**art (UMP) under certain conditions on the size of the unknowns
- extensions to multiple UMP's and multiple faulty signatures

Outline

The ISO/IEC 9796-2 Standard

ISO/IEC 9796-2 encoding of $m = m[1] \parallel m[2]$

$$\mu(m) = \texttt{6A}_{16} \parallel m[1] \parallel H(m) \parallel \texttt{BC}_{16}$$

Variant used in EMV

$$m[1] = \alpha \parallel r \parallel \alpha' \,, \qquad m[2] = \text{DATA}$$

r is unknown to the adversary. The encoded message is

$$\mu(m) = \texttt{6A}_{16} \parallel \alpha \parallel r \parallel \alpha' \parallel H(\alpha \parallel r \parallel \alpha' \parallel \text{DATA}) \parallel \texttt{BC}_{16}$$

The total number of unknown bits in $\mu(m)$ is $k_r + k_h$

Fault Attack on Partially-Known Message ISO/IEC 9796-2

Let's represent the message as

$$\mu(m) = t + r \cdot 2^{n_r} + H(m) \cdot 2^8$$

where t is a known value, both r and $H(m)$ are unknown.
After a fault, we have

$$\sigma'^e = t + r \cdot 2^{n_r} + H(m) \cdot 2^8 \mod p$$

Then $(r, H(m))$ must be a solution of the equation

$$a + b \cdot x + c \cdot y = 0 \mod p$$

where $a = t - \sigma'^e \bmod N$, $b = 2^{n_r}$ and $c = 2^8$ are known.

Fault Attack on Partially-Known Message ISO/IEC 9796-2

Now we are left with solving

$$a + b \cdot x + c \cdot y = 0 \mod p$$

that admits a small root $(x_0, y_0) = (r, H(m))$. However p is unknown.

- apply the method of [Herrmann and May ASIACRYPT'08] (originally for factoring an RSA modulus $N = pq$ when some blocks of p are known)
- the method is based on the Coppersmith's technique for finding small roots of polynomial equations
- in turn, Coppersmith technique uses LLL to obtain (x_0, y_0)
- finally, given (x_0, y_0), recover $\mu(m)$ and factor N by GCD

Bounds on UMP size

For a balanced RSA modulus from [Herrmann and May ASIACRYPT'08] we get

$$\gamma + \delta \leq \frac{\sqrt{2}-1}{2} \cong 0.207$$

where $\gamma = k_r/k$, $\delta = k_h/k$, k being the modulus size

Example: for 1024-bit RSA the total size of the unknowns x_0 and y_0 can be at most 212 bits, so for ISO/IEC 9796-2 with $k_h = 160$ the size of randomizer r can be as large as 52 bits

Outline

Attack Extensions

- several disjoint UMP blocks in the encoding function
- two faults modulo different factors (one modulo p and one modulo q)
- two or more faults modulo the same prime factor

Several Unknown Bits Blocks

Padding scheme

$$\mu(m) = \texttt{6A}_{16} \,\|\, \alpha_1 \,\|\, r_1 \,\|\, \alpha_2 \,\|\, r_2 \,\|\, \cdots \,\|\, \alpha_n \,\|\, r_n \,\|\, \alpha_{n+1} \,\|\, H(m) \,\|\, \texttt{BC}_{16}$$

Bound

Using the extended result of [Herrmann and May '08], we get

$$\sum_{i=1}^{n} \gamma_i \leq \frac{1 - \ln 2}{2} \cong 0.153$$

for a balanced RSA modulus and a large number of blocks n

Limitation

Runtime increases **exponentially** with n

Two Faults Modulo Different Factors

Having one signature incorrect mod p and the other incorrect mod q, we get

$$
\times \quad
\begin{array}{rll}
a_0 + b_0 \cdot x_0 + c_0 \cdot y_0 & = 0 & \bmod p \\
a_1 + b_1 \cdot x_1 + c_1 \cdot y_1 & = 0 & \bmod q \\
\hline
a_0 a_1 + \ldots + c_0 c_1 \cdot y_0 y_1 & = 0 & \bmod N
\end{array}
$$

Can be solved by linearization under the bound

$$\gamma + \delta \leq \frac{1}{6} \cong 0.167$$

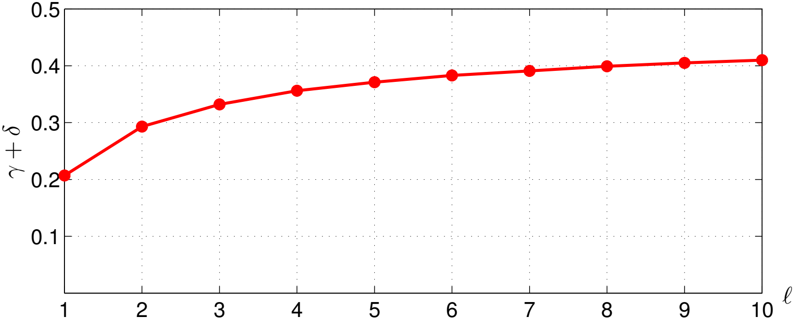
- this attack is significantly faster than the basic one
- the 16.7% bound is likely to lend itself to further improvements using Coppersmith's technique

Several Faults Modulo the Same Factor

Extension of Coppersmith's technique to multiple equations

$$f_u(x_u, y_u) = a_u + x_u + c_u y_u, \quad 1 \le u \le \ell$$

coming from ℓ successive faults



Bound on the size of the unknowns is asymptotically 0.5

Outline

1. Fault attacks on RSA with CRT
2. Our Basic Attack on ISO/IEC 9796-2
3. Attack Extensions
4. Experimental Results

Simulation

Simulation parameters

- $H = \text{SHA-1}$, *i.e.* $k_h = 160$
- 1024-, 1536- and 2048-bit RSA
- LLL implementation: SAGE
- standard 2 GHz Intel laptop

Single-Fault Attack Simulations

modulus size k	**UMP size k_r**	**runtime**
1024	6	4 minutes
1024	13	51 minutes
1536	70	39 seconds
1536	90	9 minutes
2048	158	55 seconds

- exhausting a 13-bit randomizer took 0.13 seconds
- the attack becomes more efficient for larger moduli

Multiple-Fault Simulations

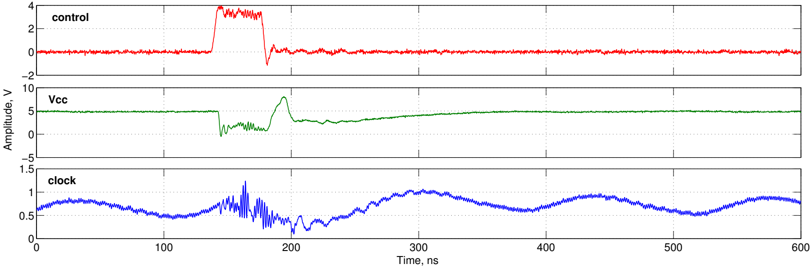
- three faulty signatures
- $\gamma + \delta \leq 0.204$

modulus size k	**UMP size k_r**	**runtime**
1024	40	49 seconds
1536	150	74 seconds
2048	250	111 seconds

- multiple-fault attacks with three faults are more efficient than single-fault attacks
- exhausting a 40-bit randomizer would take about a year on the same PC

Physical Fault Injection

- unprotected 1536-bit RSA-CRT on ATmega128 (running time several minutes at 7.68 MHz)
- spike (**sag**) attack [Schmidt FDTC'08]
- 40 ns cut-off in power supply using FPGA
- recovering factorization of N from the faulty signature with our basic attack



Before Concluding: Another Practical Application

PKCS#1 v1.5

$$\mu(m) = 0001_{16} \parallel \underbrace{\mathtt{FF}_{16} \ldots \mathtt{FF}_{16}}_{k_1 \text{ bytes}} \parallel 00_{16} \parallel T \parallel H(m)$$

- T is a known sequence of bytes
- k_1 adjusted to make $\mu(m)$ have the same size as the modulus

With the single unknown the bound is $\delta < 0.25$, therefore for the 2048-bit modulus and $H =$SHA-512 the modulus can be factored with a single faulty signature even when the signed message is **totally unknown**

Conclusion

- a novel practical attack on RSA-CRT with partially unknown messages
- particularly applicable to EMV and PKCS#1 v1.5 padding schemes
- not applicable to PSS [Coron and Mandal, ASIACRYPT'09]

Extended version of the paper: **ePrint 2009/309**