Further Results and Considerations on Side Channel Attacks on RSA

Vladimír Klima¹ and Tomáš Rosa²

¹ ICZ, Prague, Czech Republic
² Dept. of Computer Science and Eng., FEE, Czech Technical University in Prague

[vladimir.klima, tomas.rosa]@fi.muni.cz

Presented at the workshop CHES 2002, August 13-15, 2002, Redwood Shores, California, USA

Main Topics

- Another possible side channel attack on RSAES-OAEP
- Fault side channel attacks on RSA-KEM
- Note on the conversion from plaintext-oriented attacks to signature-oriented ones

Side Channels

- Definition: Side Channel (SC)
  - An undesirable way, which a cryptographic module exchanges some information with its neighborhood in.
    - Timing
    - Power
    - Electromagnetic
    - Fault
    - Kleptographic

Side Channels

- Analysis of the side channel
  - A process of extracting a useful information from the particular side channel.
- Attack based on the side channel
  - A process of using an analysis of the particular side channel against a given cryptographic module.

PART I

- Another possible side channel attack on RSAES-OAEP

Partial Information Oracle

- Definition: Partial Information Oracle (PIO) is a black-box function of the ciphertext $c$, $c \in C$, such that:
  - PIO: $C \rightarrow \text{Im}(PIO)$,
  - for at least one bit $m$ of the RSA plaintext $m$, $m = c^2 \mod n$, we have:
    - $H(m) \cdot \text{PIO}(c) < H(m)$,
    - information from the PIO gained in this way must be non negligible - must be induced by a non negligible advantage $\text{adv}$.  

Whole Information Oracle

- Definition: *Whole Information Oracle (WIO)* is a known algorithm based on an access to the particular PIO. It takes as an input the value of ciphertext $c$ and it returns the value of the whole plaintext $m$, $m = c^d \mod n$.
- $\text{WIO}_{\text{PIO}}: \text{c} \rightarrow \text{M}; \text{c} \rightarrow m = (c^d \mod n)$

WIO Existence

- Having an access to the particular PIO, it is possible to build up the WIO.
- Given the ciphertext $c$, WIO returns the corresponding plaintext $m$ in the random polynomial time.
- The proof of WIO existence is based on the theorem of *The Security of Individual RSA Bits* [13].

PIO Hunting Problem

- Where and how to find an appropriate PIO...
- Once we are having one, we are also having a chance to develop an efficient chosen ciphertext attack on the whole RSA scheme.
PIO - Examples (1)

- **PIO\text{lsb}**
  - Returns the least significant bit of the plaintext (i.e. \( m_0 \)).

- **PIO\text{half}**
  - Says whether the plaintext \( m \) is less or higher than one half of the modulus \( n \).
  - Note: The PIO\text{lsb} and the PIO\text{half} are polynomially equivalent ([25]).

PIO - Examples (2)

- **PIO\text{MByte}**
  - Indicates whether the most significant byte is zero or not.
  - Used in the Manger's attack on the RSAES-OAEP [15].

- **PIO\text{PKCS1-v1_5}**
  - Indicates whether the plaintext is "PKCS1-v1_5 conforming" or not.
  - Used in the Bleichenbacher's attack on the RSAES-PKCS1-v1_5 [5].

PIO Based on a Power Side Channel

- We assume that Hamming weights of arguments of the particular inner operation in the RSAES-OAEP scheme are leaking out through a power side channel. We have a PIO\text{...}
Place of Our Attack on RSAES-OAEP

Assumptions on PIO\textsubscript{PA}:
- Let us send \(c\) and \(c'\) to the PIO\textsubscript{PA} ... 
  - Let \(m = m[k] \mid ... \mid m[6]\).
  - Let us denote (we use MGF/SHA-1 notation):
    - \(W_0 = m[10] \mid ... \mid m[9]\).
    - \(W_0 = m[6] \mid m[5] \mid m[4] \mid m[3]\).
    - \(W_0 = m[2] \mid m[1] \mid m[0] \mid 00\) (00 appended by MGF).
  - Analogically also for \(m'\) and \(W_0', W_0', W_1', W_1',\).

Main assumption:
- From the PIO\textsubscript{PA}, we get triplets of Hamming weights:
  - \(HW = (w(W_0), w(W_0), w(W_1))\).
  - \(HW' = (w(W_0'), w(W_0'), w(W_1')).\)

Converting PIO\textsubscript{PA} to a PIO\textsubscript{lsb} (1):
- Main aim:
  - Convert the PIO\textsubscript{PA} to the PIO\textsubscript{lsb}.

Main idea:
- Let \(c = c' \mod n\). Let \(m = \bar{m} \mod n\) and \(m' = \bar{m}' \mod n\).
  - If \(lbd(m) = 0\) then \(m' = m + 1\).
  - If \(lbd(m) = 1\) then \(m' = m + n \gg 1\).

Putting it together:
- If \(lbd(m) = 0\) then \(HW\) and \(HW'\) are related linearly.
- If \(lbd(m) = 1\) then the probability of a random linear relationship is very low.
  - From here we get \(lsb(m)\).
Converting PIO\textsubscript{PA} to a PIO\textsubscript{lsb} (2)

<table>
<thead>
<tr>
<th>(W_{lsb}^1)</th>
<th>(W_{lsb}^2)</th>
<th>Possible relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>0 0 1</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>0 1 0</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>0 1 1</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>1 0 0</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>1 0 1</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>1 1 0</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
<tr>
<td>1 1 1</td>
<td></td>
<td>(w(W_{lsb}^1) = w(W_{lsb}^2))</td>
</tr>
</tbody>
</table>

Closure: Building WIO from PIO\textsubscript{lsb}

- PIO\textsubscript{lsb} seems to be one of those best oracles suitable for building up an efficient WIO.
- We suggest the RSA inversion algorithm
  - This algorithm is based on a binary halving technique; errors are corrected through a majority decision.
  - It requires \(O(L(n)/\text{adv})\) oracle calls, where:
    - \(L(n)\) is the length of an RSA modulus.
    - \(\text{adv}\) is the oracle advantage in the lsb prediction.
  - Post-processing complexity is \(O(L(n)^2/\text{adv}^2)\).
  - It doesn't require further oracle access.

---

PART II

- Fault side channel attacks on RSA-KEM
Confirmation Oracle (CO)

- A CO for arbitrary chosen integers \( r, y \) confirms whether \( r = y^d \mod n \).
- It can be generalized for any encryption scheme (the condition tested may be also more general). Here we use an RSA-CO only.
- If there are faults then an RSA-CO may reveal nontrivial information about the private key.
- Depends on what kind of error it is and on a description accuracy.

Example – Bit Errors

- In [3] it was shown that bit errors in a private exponent \( d \) enable us to compute the whole private key efficiently.
- We observed that it is not necessary to have an access to the whole result of the faulty computation.
- Having an access to an RSA-CO is enough.

An RSARSA–CO seems to be an effective generalization of some existing (as well as new ones) attacks developed with the assumption that an attacker could observe the whole output of an RSA computation (i.e., corrupted plaintext or signature).

Bit errors are becoming even more interesting due to Optical Fault Induction Attacks.

Modified Pohlig-Hellman Algorithm

- This well-known algorithm can also be modified to use an access to an RSA-CO only.
- Suppose that we want to use it to compute the private exponent \( d \) from the triplet \((r, g, n')\), such that \( r = g^d \mod n' \).
- Suppose that an attacker is able to force the change of the modulus \( n \) to the modulus \( n' \) inducing weak instances of DLP.
- We show, that it is not necessary to know the value of \( r \) exactly – having an access to the RSA-CO is sufficient here.
RSA-KEM ([23])

20.3 Decryption

Given a ciphertext \( C \), decryption runs as follows.
1. Check that \( |C| = nLm \); if not, then fail.
2. Set \( y = \text{OS2IP}(C_0) \).
3. Check that \( y < n \); if not, then fail.
4. Compute \( r = y^2 \mod n \).
5. Compute \( K = \text{KDF}[\text{HSF}(r, nLm), \text{KeyLen}] \).
6. Output the key \( K \).

...then continue as: \( M = \text{DEM}.\text{Decrypt}(K, L, \text{DEM}.\text{Encrypt}(K, L, M)) \).

\( M \sim \text{message}, L \sim \text{label}, K \sim \text{symmetrical key} \)

---

Building RSA-CO on RSA-KEM

- We use the properties of the whole hybrid scheme H-PKE.
- There is no integrity check for the RSA plaintext. This is obviously a bad property against CCA2, however it also implies that any permuting RSA plaintexts will be used for a symmetric decryption.
- Further integrity controls applied on the message decrypted symmetrically then confirm our guess of \( r \).

Summary: What makes the RSA-KEM stronger in other areas, that makes it very vulnerable to fault attacks.

---

RSA-CO Definition

**Definition.** RSA confirmation oracle RSA-CO(\( r, y \)). Let us have a receiver oracle RO that uses RSA in the hybrid encryption H-PKE cases. We will construct a RSA confirmation oracle RSA-CO(\( r, y \)) \( \rightarrow (\text{ANSWER} = \text{"yes"}) \) as follows:
1. \( K = \text{KDF}(r) \), \( \text{KDF} - \text{Key Derivation Function} \)
2. \( CT = r \); for simplicity we omit the conversion between integers and strings
3. \( CT = \text{DEM}.\text{Encrypt}(K, CT) \), where \( M \) contains an integrity check
4. \( C = \text{C} \parallel CT \)
5. Send the ciphertext \( C \) to the receiver oracle RO. RO then continues:
   a. Compute \( K = \text{KEM}.\text{Decrypt}(L, CT) \) following these steps:
      i. Check if \( y = CT < n \); if not, an error has occurred.
      ii. Compute \( r' = (y^2 \mod n) \)
   b. \( M = \text{DEM}.\text{Decrypt}(K, CT) \)
   c. Check the integrity of \( M \)
   d. If it is correct, the answer of RO is "yes", otherwise it is "no"
6. The answer of RSA-CO(\( r, y \)) is "yes", if RO returned "yes", otherwise it is "no"
Summary

- RSAES-OAEP can be attacked when Hamming weights of data processed leak to an attacker.
- All steps in the OAEP decoding should be resistant to a side-channel leakage.
- RSA-KEM has serious problems with fault attacks.
  - It could lead to a private key disclosure.
  - Integrity of private key must be ensured and the computation must be checked for faults.
- Note that Manger's and Bleichenbacher's attacks can easily be converted to compute signatures instead of plaintexts.
- Server's certificate should not be attributed for a signature and a key exchange purposes at the same time – this helps to reduce potential attack impacts.

Thank You