Side Channels in the McEliece PKC

F. Strenzke E. Tews H. G. Molter R. Overbeck A. Shoufan

November 6, 2008

Author Information

- F. Strenzke: strenzke@flexsecure.de,
  FlexSecure GmbH, Germany
- E. Tews: e_tews@cdc.informatik.tu-darmstadt.de,
  Cryptography and Computeralgebra, Department of Computer Science, Technische Universität Darmstadt, Germany
- H. G. Molter and A. Shoufan:
  molter@iss.tu-darmstadt.de,
  shoufan@iss.tu-darmstadt.de, Integrated Circuits and Systems Lab, Department of Computer Science, Technische Universität Darmstadt, Germany
- R. Overbeck: raphael.overbeck@epfl.ch
  Ecole Polytechnique Federale de Lausanne, Switzerland

Side Channel Attacks

- Cryptographic algorithms are executed by devices.
- These devices reveal certain physical properties to the environment:
  - power consumption
  - running time
  - electromagnetic radiation
- These quantities might be related to secrets (secret key) that are input to the algorithms.

Measurement and evaluation of these quantities to reveal the secret $\longrightarrow$ side channel attack

Side Channel Attacks

Some facts about side channel attacks

- Known since 1996
- Most explored variants:
  - Power Analyis Attacks
  - Timing Attacks (affects also general purpose computers)
- new variants on general purpose computers: microarchitectural attacks
  - branch prediction attacks
  - cache attacks

Error Correction with the Error Locator Polynomial

1. definition of the error locator polynomial:

$$\sigma_{\vec{e}}(X) = \prod_{j \in \mathcal{T}_e} (X - \gamma_j) \in \mathbb{F}_{2^m}[X], \tag{1}$$

   where $\mathcal{T}_{\vec{e}} = \{i | e_i = 1\}$ and $\vec{e}$ is the error vector of the distorted code word to be decoded.

2. Once the error locator polynomial is known, the error vector $\vec{e}$ is determined as

$$\vec{e} = (\sigma_{\vec{e}}(\gamma_0), \sigma_{\vec{e}}(\gamma_1), \cdots, \sigma_{\vec{e}}(\gamma_{n-1})) \oplus (1, 1, \cdots, 1). \tag{2}$$

The Patterson Algorithm - 1

- The Patterson Algorithm actually does not determine $\sigma_{\vec{e}}(X)$ as defined above, but computes $\bar{\sigma}_{\vec{e}}(X)$ where

$$\bar{\sigma}_{\vec{e}}(X) = \sigma_{\vec{e}}(X) \text{ if } \mathrm{wt}\,(\vec{e}) \leq t.$$

- Without derivation: For $\mathrm{wt}\,(\vec{e}) > t$, the degree of $\bar{\sigma}_{\vec{e}}(X)$ will be t with very high probability.
- From the definition of the error locator polynomial $\sigma_{\vec{e}}(X) = \prod_{j \in \mathcal{T}_e}(X - \gamma_j) \in \mathbb{F}_{2^m}[X]$, it follows that for $\mathrm{wt}\,(\vec{e}) \leq t$ its degree is equal to $\mathrm{wt}\,(\vec{e})$

The Patterson Algorithm - 2

Relation of the degree of the error locator polynomial in the decryption operation to the number of "errors in the ciphertext"

- $\mathrm{wt}\,(\vec{e}) = t \longrightarrow \deg\,(\bar{\sigma}_{\vec{e}}(X)) = t$ (proper ciphertext)
- $\mathrm{wt}\,(\vec{e}) > t \longrightarrow \deg\,(\bar{\sigma}_{\vec{e}}(X)) = t$ (with very high probability)
- $\mathrm{wt}\,(\vec{e}) < t \longrightarrow \deg\,(\bar{\sigma}_{\vec{e}}(X)) = \mathrm{wt}\,(\vec{e})$

The Properties exploited by the Attack

- The larger the degree of error locator polynomial, the longer the running time of the decryption operation: it is evaluated $n = 2^m$ times in the final step

  $$\vec{e} = (\sigma_{\vec{e}}(\gamma_0), \sigma_{\vec{e}}(\gamma_1), \cdots, \sigma_{\vec{e}}(\gamma_{n-1})) \oplus (1, 1, \cdots, 1).$$

  $(m = 11 \rightarrow n = 2048)$
- The attacker can flip bits of an intercepted ciphertext and influence the actual number of "errors in the ciphertext"
- The goal of the attacker is to determine the secret $\vec{e}$ used during encryption. (Allows to recover the message.)

The Concept of the Attack

Alice		Eve		Bob
encypts m to z				
sends z	$\longrightarrow$	intercepts z, flips bit at position i in z resulting in z'		
		sends z'	$\longrightarrow$	decrypts z'
		measures decryption time	$\longleftarrow$	

Eves decision strategy:

"long" decryption time	$\longrightarrow$	$\deg(\bar{\sigma}_{\vec{e}}) = t$	$\longrightarrow$	$e_i = 0$
"short" decryption time	$\longrightarrow$	$\deg(\bar{\sigma}_{\vec{e}}) = t - 1$	$\longrightarrow$	$e_i = 1$

The Attack Algorithm

Require: ciphertext $\vec{z}$, and the parameter t, of the McEliece PKC.
Ensure: a guess $\vec{e}'$ of the error vector $\vec{e}$ used by Alice to encrypt $\vec{z}$.

1: **for** $i = 0$ to $n - 1$ **do**
2: Compute $\vec{z}_i = \vec{z} \oplus \mathrm{sparse\_vec}\,(i)$.
3: Take the time u_i as the mean of N measured decryption times where $\vec{z}_i$ is used as the input to the decryption device.
4: **end for**
5: Put the t smallest timings u_i into the set M.
6: **return** the vector $\vec{e}'$ with entries $e'_i = 1$ when $u_i \in M$ and all other entries as zeros.

Experimental Results

The Attack was executed against a Java Implementation on a PC:
$N = 2 \longrightarrow 48\%$ of the executed attacks recovered all positions of $\vec{e}$ correctly

Countermeasure

- A straightforward countermeasure: artificially increase the degree of the error locator polynomial to t before evaluating it.
- Remaining research problem: Still a detectable difference?

CCA2 Conversion

Using the McEliece PKC with a CCA2-Conversion does not affect the attack:
Still a substring corresponding to $\vec{m}\mathbf{G}^{\mathrm{pub}} \oplus \vec{e}$ will appear in the ciphertext.

Power Analysis Attacks

A *Power Analysis Attack* is a side channel attack in which

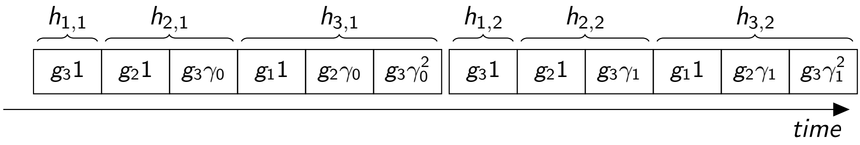
- the power consumption during the execution of the secret operation is measured,
- and the attacker tries to extract information about the secret on the basis of the measured data.

Example: RSA (square and multiply)
If the power traces of a multiplication and a squaring can be dinstinguished, the key can be extracted.

Generation of Parity Check Matrix

$$h_{i,j} = g(\gamma_{j-1})^{-1} \sum_{s=t-i+1}^{t} g_s \gamma_{j-1}^{s-t+i-1}, \tag{3}$$

where $i = 1, \ldots, t$ and $j = 1, \ldots, n$.



Countermeasures

Countermeasures to protect against Power Analysis Attacks:

- blinding:
  random non-zero value $r_i \in \mathbb{F}_{2^m}$

$$h_{i,j} = g(\gamma_{j-1})^{-1} r_i^{-1} \left(\sum_{s=t-i+1}^{t} (r_i g_s) \gamma_{j-1}^{s-t+i-1} \right). \qquad (4)$$

- randomize the order of evaluation of the matrix elements

- we have seen examples for possible side channel attacks and countermeasures
- much more research has to be done

Thank you!