Efficient 2nd-order Power Analysis on Masked Devices Utilizing Multiple Leakage

Liwei Zhang, A. Adam Ding, Yunsi Fei, Pei Luo

Northeastern University

IEEE Int. Symposium on Hardware-Oriented Security and Trust 2015, McLean, VA
Preliminaries

Advantage

Proposed Attack
- Step 1: Reduce the candidate set of \((t^1_0, t^2_0)\).
- Step 2: Combine 2nd-order CPA attacks on the candidate set into one single attack.

Statistical Analysis

Numerical Results

Conclusion
Traditional 2O-CPA–With Profiling

Two most leakage points could be found by Pearson’s correlation: one related with the mask \( V_1(M) \), and the other related to the key, plaintext, and mask \( V_2(k, M, P) \). Then 2O-CPA attack can be used.
For attacker: No Profiling

Without profiling, the exact temporal locations of $t_0^1, t_0^2$ are unknown to attackers. However, two windows $\mathcal{L}_1$ and $\mathcal{L}_2$ which contain $t_0^1, t_0^2$ are easy to guess.

Apply 2nd-order CPA over all possible pairs $(l_{t_1}, l_{t_2}) \in (\mathcal{L}_1, \mathcal{L}_2)$, the complexity is $O(n_w^2)$. 
Preliminaries

Advantage

Proposed Attack

- Step 1: Reduce the candidate set of \((t^1_0, t^2_0)\).
- Step 2: Combine 2nd-order CPA attacks on the candidate set into one single attack.

Statistical Analysis

Numerical Results

Conclusion
Efficient 2nd-order Power Analysis on Masked Devices Utilizing Multiple Leakage

- Propose new attacks: combine key-decisions at multiple leakage without profiling.
  - Just need to know the time windows that contain $t_0^1 / t_0^2$.

- Keep the computational complexity at $O(n_L \log_2(n_L))$.
  - Applying exhaustive 2O-CPAs, the computational complexity is $O(n_L^2)$.

- Improve the power of attacks.
3 Proposed Attack

- Step 1: Reduce the candidate set of \((t_0^1, t_0^2)\).
- Step 2: Combine 2nd-order CPA attacks on the candidate set into one single attack.
Notation

- $\mathcal{L}_1, \mathcal{L}_2$ : Leakages of two distinct windows with $|\mathcal{L}_1| = |\mathcal{L}_2| = n_w$.
- $\mathcal{T}_1 = \{t^1_1, ..., t^1_{n_w}\}$ : Leaking time points in Window 1.
- $\mathcal{T}_2 = \{t^2_1, ..., t^2_{n_w}\}$ : Leaking time points in Window 2.
- $\mathcal{K} = \{0, ..., n_{\mathcal{K}} - 1\}$ : Candidate keys.
- $V_k$ : Intermediate variable, $k \in \mathcal{K}$.
- $k_0$ : Correct key.
- $t^1_0$ and $t^2_0$ : Time points with the strongest leakage at each window.
Step 1: Preliminary

Belgarric et al. proposed five preprocessing techniques using time-frequency conversion tools basing on FFT in 2012.

Definition (Cross Correlation)

Let $L_0$ and $L_1$ be two discrete sequences of $n_w$ leakage points,

\[
x{\text{-}}\text{corr}(\Delta t) = \sum_{t'=0}^{n-1} L_0(t').L_1(\Delta t + t' \mod n_w)
\]

\[
= \sqrt{n_w}.IDFT[DFT[L_0].DFT[L_1]],
\]

where $\overline{\_}$ denotes complex conjugation. DFT and IDFT denote the discrete Fourier transform and Inverse Fourier transform respectively.

\[
(\hat{k}, \hat{\Delta t}) = \arg \max_{k \in \mathcal{K}, \Delta t} |corr(x{\text{-}}\text{corr}(\Delta t), V_k)|
\]

The preprocessing time using DFT and IDFT has computational complexity of $O(n_w \log_2(n_w))$. 
Idea of Step 1: Candidate set of \((t_0^1, t_0^2)\)

- Time-lag \(\Delta t_0\): the difference of \(t_0^2 - t_1^2\) and \(t_0^1 - t_1^1\).
- \(t_0^2 = t_1^2 + \Delta t_0 + (t_0^1 - t_1^1)\).

Search of \((t_0^1, t_0^2)\) is **equivalent** to search of \((t_0^1, \Delta t_0)\).

- The possible values of \(t_0^1\) are all values in \(T_1\).
- The possible values of \(\Delta t_0\) are 0, ..., \(n_w - 1\).

Here, we focus on the reduction of \(\Delta t_0\).
Details of Step 1: Screen for the candidate set $S$ of $\Delta t_0$
with $|S| = \lfloor 2\log(n_w) \rfloor$

- Compute the $x$-corr($\Delta t$), $\Delta t = 0, \ldots, n_w - 1$. Where $x$-corr($\Delta t$) is the sum of center product corresponding to time-lag $\Delta t$, i.e.

$$x\text{-corr}(\Delta t) = \sum_{t=1}^{n_w} \mathcal{L}_1(t + \Delta t) \cdot \mathcal{L}_2(t) = IDFT[\overline{DFT[\mathcal{L}_1] \cdot DFT[\mathcal{L}_2]}].$$

- At each $\Delta t$, find the maximum of absolute value of correlation between $x$-corr($\Delta t$) and $V_k$ for all $N$ key candidates $k \in \mathcal{K}$.

$$\rho^*(\Delta t) = \max_{k \in \mathcal{K}} |CORR(x\text{-corr}(\Delta t), V_k)|.$$ 

- Rank $\Delta t$ by the $\rho^*(\Delta t)$ values, the corresponding $\Delta t$ to the top $\lfloor 2\log(n_w) \rfloor$ values are retained.

$$S = \{\Delta t_1, \Delta t_2, \ldots, \Delta t_{\lfloor 2\log(n_w) \rfloor}\}.$$
Step 1: Candidate set of $(t_0^1, \Delta t_0)$

Now, through the screening process,

there are $n_w$ possible values of $t_0^1$,

$\lfloor 2\log(n_w) \rfloor$ possible values of $\Delta t_0$,

yields $n_L = n_w \lfloor 2\log(n_w) \rfloor$ center products,

Keep the computational complexity at $O(n_w \log_2 n_w)$. 
Idea of Step 2: Combine Key-decisions

- With $n_L$ number of candidate pairs $(t_0^1, \Delta t_0)$, each pair makes a decision on the most likely key.
- Combine the decisions together to find the right key.
Details of Step 2: Combine Key-decisions

- Compute $n_L$ center products, denoting by $CP_i$, $i = 1, 2, ..., n_L$.
- Compute Pearson’s correlation between $CP_i$ and $V_k$.

$$\rho_{i,k} = \text{corr}(CP_i, V_k), \quad k \in \mathcal{K} \text{ and } i = 1, ..., n_L.$$

- For each $i$, select the key which maximizes $|\rho_{i,k}|$ as the key-decision, denoted as $k_{i,\text{max}}$, the corresponding absolute value of correlation denoted as $\rho_i$.

$$k_{i,\text{max}} = \arg \max_{k \in \mathcal{K}} |\rho_{i,k}|, \quad i = 1, ..., n_L.$$

$$\rho_i = |\rho_{i,k_{i,\text{max}}}|, \quad i = 1, ..., n_L.$$
• Maximum Attack: select the key that corresponds to the maximum attack statistic $\rho_{\text{max}} = \max_i \rho_i$.

$$\hat{k} = \{ k : |\rho_{i,k}| = \rho_{\text{max}} \}$$

• Majority Vote Attack: select the key which is selected most often among all leakage candidates. Let $N_k = \sum_{i=1}^{n_L} \mathbb{I}\{ k = k_{i,\text{max}} \}$.

$$\hat{k} = \arg\max_{k \in \mathcal{K}} N_k$$
1 Preliminaries

2 Advantage

3 Proposed Attack
   - Step 1: Reduce the candidate set of \((t_0^1, t_0^2)\).
   - Step 2: Combine 2nd-order CPA attacks on the candidate set into one single attack.

4 Statistical Analysis

5 Numerical Results

6 Conclusion
Statistical Analysis

Assumption:
- The first $n_0$ candidate points are real leakage.
- The rest $n_L - n_0$ candidate points are false leakage.

Model:
- at the real leakage
  \[ \rho_{i,k_c} \sim N(\text{mean} = \rho_0, \text{var} = \frac{(\rho\sigma_0)^2}{n_{tr}}), \quad i = 1, \ldots, n_0. \]
  \[ \rho_{i,k} \sim N(\text{mean} = 0, \text{var} = \frac{1}{n_{tr}}), \quad k \in K \setminus k_c, \quad i = 1, \ldots, n_0. \]
- at the false leakage
  \[ \rho_{i,k} \sim N(\text{mean} = 0, \text{var} = \frac{1}{n_{tr}}), \quad k \in K, \quad i = n_0 + 1, \ldots, n_L. \]
Statistical Analysis

- CPA successes at a **single real leakage**, if $|\rho_{i,k_c}| > \max_{k_g \in \mathcal{K}\setminus k_c} |\rho_{i,k_g}|$.

  The minimum condition for a CPA to succeed is

  $$\tau_1 = \frac{\sqrt{\log n_K}}{\rho_0 \sqrt{n_{tr}}} \to 0.$$ 

- **Maximum attack** succeeds if $\max_{1 \leq i \leq n_0} |\rho_{i,k_c}| > \max_{1 \leq i \leq n_L, k_g \in \mathcal{K}\setminus k_c} |\rho_{i,k_g}|$.

  The maximum attack succeeds asymptotically with probability one if

  $$\tau_0 = \frac{\sqrt{\log(n_K) + \log(n_L)}}{\rho_0 \sqrt{n_{tr}}} \to 0.$$ 

- **Majority vote attack** succeeds if $N_{k_c} > \max_{k_g \in \mathcal{K}\setminus k_c} N_{k_g}$.

  The condition of majority vote attack succeeding will become

  $$\tau_2 = O\left(\frac{\sqrt{n_L \log n_K}}{n_0 \sqrt{n_K}}\right) \to 0.$$
Figure: \(\rho_0\) on hardware \((n_w = 210, n_L = 3150)\) and software \((n_w = 200, n_L = 3000)\) implementations data for one trail. Red lines denote the position with one fifth of \(\max\{\rho_0\}\).

Table: Comparison of required conditions on two data sets

<table>
<thead>
<tr>
<th>Attack</th>
<th>Majority vote attack</th>
<th>Maximum attack</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hardware</td>
<td>Software</td>
</tr>
<tr>
<td>Data</td>
<td>n_w</td>
<td>n_0</td>
</tr>
<tr>
<td>Hardware</td>
<td>210</td>
<td>1119</td>
</tr>
<tr>
<td>Software</td>
<td>50</td>
<td>226</td>
</tr>
<tr>
<td>Software</td>
<td>200</td>
<td>147</td>
</tr>
<tr>
<td>Software</td>
<td>2000</td>
<td>239</td>
</tr>
<tr>
<td>Software</td>
<td>6000</td>
<td>371</td>
</tr>
</tbody>
</table>
3 Proposed Attack

- Step 1: Reduce the candidate set of \((t_0^1, t_0^2)\).
- Step 2: Combine 2nd-order CPA attacks on the candidate set into one single attack.
Numerical Results

(a) Empirical success rates on hardware ($n_w = 210$).

(b) Sample size for achieving SR $\geq 0.8$ for different window sizes. The arrows mean more traces needed.
## Numerical Results

Table: Computing time (s) of different attacks

<table>
<thead>
<tr>
<th>$n_w$</th>
<th>50</th>
<th>200</th>
<th>2000</th>
<th>6000</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-corr attack</td>
<td>0.27</td>
<td>0.63</td>
<td>1.62</td>
<td>7.76</td>
</tr>
<tr>
<td>maximum attack</td>
<td>0.35</td>
<td>0.88</td>
<td>3.93</td>
<td>23.39</td>
</tr>
<tr>
<td>$n_w^2$-2O-CPA</td>
<td>1.62</td>
<td>6.19</td>
<td>501.82</td>
<td>Unknown</td>
</tr>
</tbody>
</table>
1 Preliminaries

2 Advantage

3 Proposed Attack
   - Step 1: Reduce the candidate set of \((t_0^1, t_0^2)\).
   - Step 2: Combine 2nd-order CPA attacks on the candidate set into one single attack.

4 Statistical Analysis

5 Numerical Results

6 Conclusion
Conclusion

- Methodology of attack
  - Maximum Attack
  - Majority Vote Attack

- Theoretical Study
  - $\tau_1$: Condition of succeeding at one real leakage
  - $\tau_0$: Maximum Attack succeed
  - $\tau_2$: Majority Vote Attack succeed

- The new attacks improve the power

- Keep the computational complexity at $O(n_L \log(n_L))$.

- The theoretical conditions agree with the patterns observed in two real data.
Thank You