Cryptography and Embedded System Security CRAESS_I

Xiaolu Hou

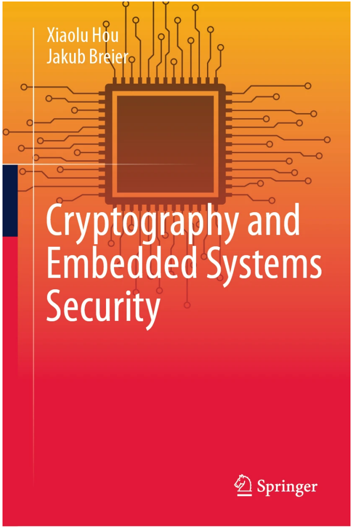
FIIT, STU
xiaolu.hou @ stuba.sk

Course Outline

- Abstract algebra and number theory
- Introduction to cryptography
- Symmetric block ciphers and their implementations
- RSA, RSA signatures, and their implementations
- Probability theory and introduction to SCA
- SPA and non-profiled DPA
- Profiled DPA
- SCA countermeasures
- FA on RSA and countermeasures
- FA on symmetric block ciphers
- FA countermeasures for symmetric block cipher
- Practical aspects of physical attacks
  - Invited speaker: Dr. Jakub Breier, Senior security manager, TTControl GmbH

Recommended reading

- Textbook
  - Sections
    - 4.3.2;
    - 4.6.1.



Lecture Outline

- Profiled DPA Attack Steps
- Stochastic Leakage Model
- Template-based DPA
- Success Rate and Guessing Entropy
- Further Reading

Non-profiled SCA

- If the attacker does not have access to a similar device, just the target device or just the measurements coming from the target device, we talk about a *non-profiled SCA*.
- In a general scenario, this attack utilizes a set of traces where a fixed secret key is used to encrypt multiple (random) plaintexts.

Profiled SCA

- If we assume the attacker has access to a clone device of the target device, then the attacker can carry out a *profiled SCA*.
- The attack operates in two phases.
- In the profiling phase, the attacker acquires side-channel measurements for known plaintext/ciphertext and known key pairs.
- This set of data is used to characterize or model the device.
- Then the attacker acquires a few measurements from the target device, usually identical to the clone device, with known plaintext/ciphertext but the key is secret.
- These measurements from the target device are then tested against the characterized model from the clone device.

Datasets

There are two datasets that will be analyzed in more detail in this lecture.
Both datasets

- Capture one round of software implementation of PRESENT.
- `nop` operations before and after PRESENT computation
- Each trace has 3600 time samples

More details on individual datasets:

- *Random plaintext dataset*: This dataset contains 5000 traces with a fixed round key `0xFEDCBA0123456789` and a random plaintext for each trace – **attack dataset**
- *Random dataset*: This dataset contains 10000 traces with a random round key and a random plaintext for each trace – **profiling dataset**

Profiled DPA

- Profiled DPA Attack Steps
- Stochastic Leakage Model
- Template-based DPA
- Success Rate and Guessing Entropy
- Further Reading

Profiling phase

- Suppose the *Random dataset* is obtained from a clone device – *profiling traces*
- The *Random plaintext dataset* is from the target device – *attack traces*
- Before the attack, we can analyze *Random dataset* to obtain more information about the leakage behavior of the devices – *profiling phase*.
- The first major step in the profiling phase is to find the POIs – time samples that will give us more information, or with better signal.
- Instead of computing the sample correlation coefficients for all time samples, we can just focus on the POIs.

Attacker assumption

- The attacker has the knowledge of the plaintext and the goal is to recover the very first round key of a symmetric block cipher – for some ciphers, e.g. PRESENT, this is the first round key; for some ciphers, e.g. AES, this is the whitening key, which is equal to the master key.
- Similar attack strategies apply if we assume the attacker has the knowledge of the ciphertext and aims to recover the last round key.
- We also assume the attacker has the knowledge of the detailed implementation so that the same program can be implemented by the attacker on the clone device.
  - This is different from the non-profiled setting where only certain basic knowledge of the implementation is required – For example, how to interface with the encryption routine, whether the computation is executed serially or in parallel, or whether some types of countermeasures are present.

Profiled DPA – step 1

Identify the target cryptographic implementation

- Profiled DPA attacks can be applied to unprotected implementations of any symmetric block cipher that has been proposed so far.

Example

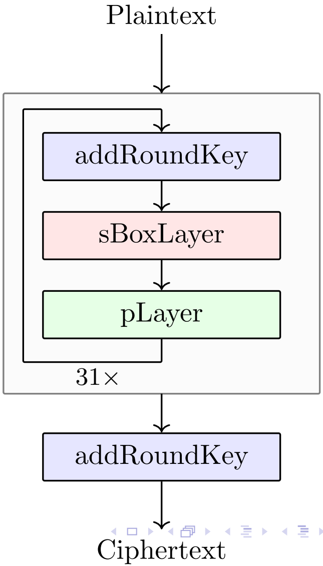
As a running example, we will look at the computation of PRESENT.

PRESENT – encryption

- Round function: addRoundKey, sBoxLayer, and pLayer.
- After 31 rounds, addRoundKey is applied again before the ciphertext output

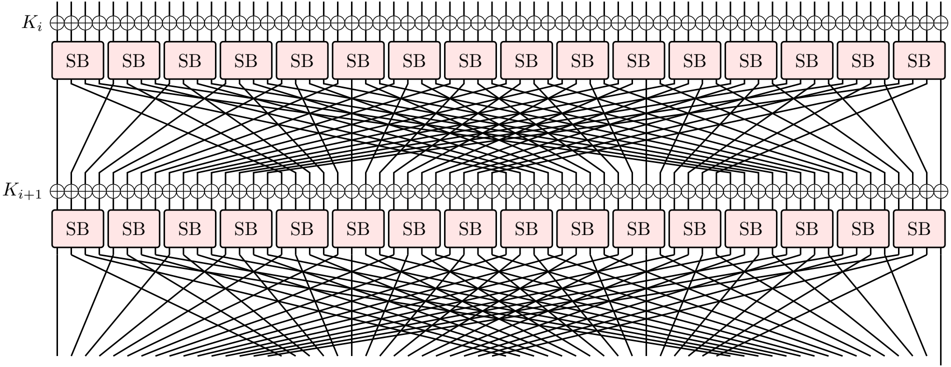
Remark

For PRESENT specification, we consider the 0th bit of a value as the rightmost bit in its binary representation. For example, the 0th bit of $3 = 011_2$ is 1, the 1st bit is 1 and the 2nd bit is 0.



Two rounds of PRESENT

- For our DPA attacks, we will attack the 0th Sbox and try to recover one nibble of the first round key


K_i
SB SB SB SB SB SB SB SB SB SB SB SB SB SB SB SB
K_{i+1}
SB SB SB SB SB SB SB SB SB SB SB SB SB SB SB SB

Profiled DPA – step 2

Measurement of profiling traces

- We first collect a set of traces for profiling using the clone device with random plaintexts and random keys.
- Suppose there are in total M_{pf} profiling traces and each trace contains q time samples.

Example

- For our illustrations, we will use the *Random dataset* as profiling traces
- This dataset contains 10000 traces with a random round key and a random plaintext for each trace.
- Each trace has 3600 time samples
- Then $M_{pf} = 10000$, and $q = 3600$.

Profiled DPA – step 3

Choose the part of the key to recover

- DPA attack is normally carried out in a divide-and-conquer manner.
- We focus on a small part (e.g. a nibble, a byte) of a round key in each attack and each part of the round key can be recovered independently.
- With the inverse key schedule, one (e.g. AES) or two round keys (e.g. PRESENT, DES) will reveal the master key
- Let k denote the target part of the key and let M_k denote the number of possible values of k.

Example

For our attack example, we will focus on the 0th nibble of the first round key for PRESENT and $M_k = 16$.

Profiled DPA – step 4

Choose the target intermediate value

- To recover the key, we exploit relationships between leakages and a certain intermediate value being processed in the DUT.
- The goal is to gain information about this intermediate value, which reveals information about our chosen part of the key.
- Let $\boldsymbol{v}$ denote the target intermediate value.
- We require that there is a function φ, such that

$$\boldsymbol{v} = \varphi(k, p),$$

  where p denotes (part of) the plaintext.

Example

- k: 0th nibble of the first round key
- $\boldsymbol{v}$: 0th Sbox output of the first round
- p: 0th nibble of the plaintext

$$\boldsymbol{v} = \mathsf{SB}_{\mathsf{PRESENT}}(k \oplus p),$$

Profiled DPA – step 5

Decide on the target signal

- Before we do further analysis of the profiling traces, we need to choose what information related to the target intermediate value $\boldsymbol{v}$ we are looking for.
- For example, the Hamming weight of $\boldsymbol{v}$; or the 0th bit of $\boldsymbol{v}$

Example

In our running example, we will look at two types of target signals

- The exact value of $\boldsymbol{v}$
- $\mathrm{wt}\left(\boldsymbol{v}\right)$, the Hamming weight of $\boldsymbol{v}$.

Profiled DPA – step 6

Compute SNR and identify the POI

- Compute SNR based on the chosen target signal
- We would like to focus on the time sample where the corresponding SNR is the highest.

Example

- Signal is given by exact value of $\boldsymbol{v}$
  - POI $= 392$.
- Signal is given by the Hamming weight of $\boldsymbol{v}$
  - POI $= 392$.

SNR – exact value

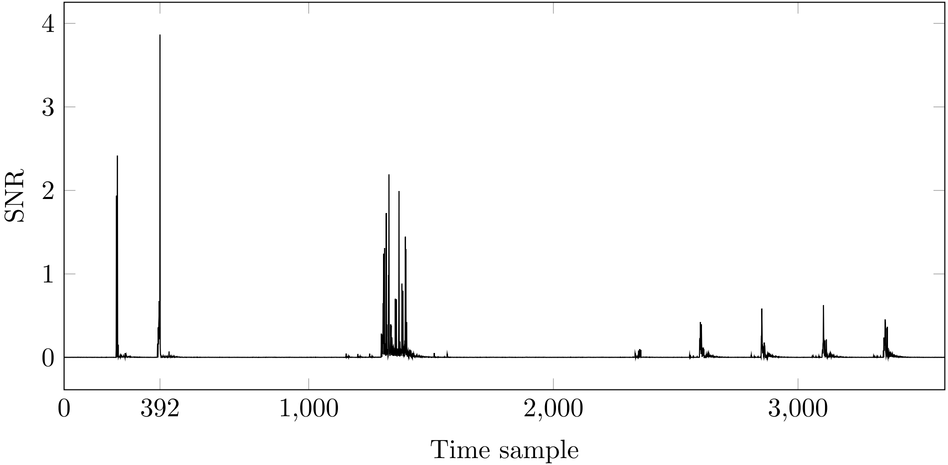


Figure: The signal is given by the exact value of the 0th Sbox output. POI $= 392$

SNR – Hamming weight

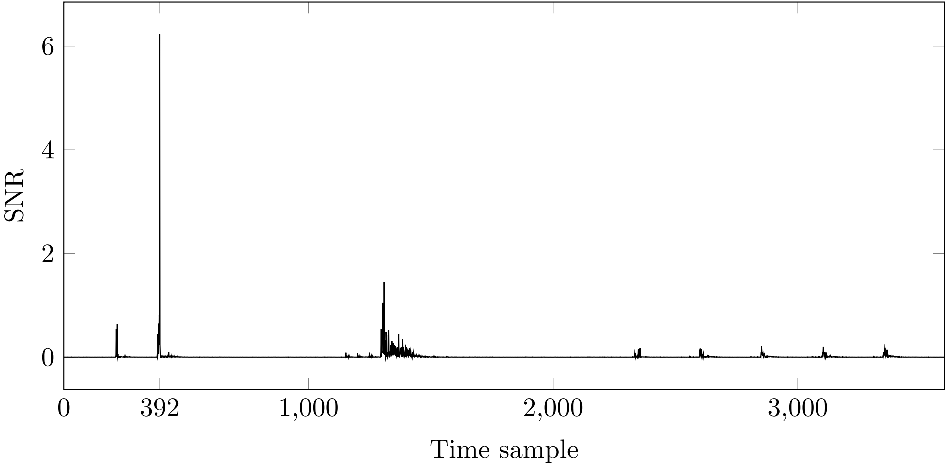


Figure: The signal is given by the Hamming weight of the 0th Sbox output. POI $= 392$

Profiled DPA – step 7

Measurement of attack traces

- After getting our POI, we are ready to carry out the attack.
- The efficiency and success of the attack are highly dependent on the measurement devices the attacker has access to.
- Suppose we have taken measurements of our target device with M_p plaintexts.
- For $j = 1, \dots, M_p$, let $\boldsymbol{\ell}_j$ denote the corresponding power trace. Each trace has q time samples.

Example

- We will use the *Random plaintext dataset* as illustrations.
  - Each trace has 3600 time samples
  - Contains 5000 traces with a fixed round key `0xFEDCBA0123456789` and a random plaintext for each trace.
- $M_p = 5000$, $q = 3600$.

Profiled DPA – step 8

Compute hypothetical target intermediate values

- For each key hypothesis $\hat{k}_i$ of k, and each (part of the) plaintext p_j, we can compute a hypothesis for $\boldsymbol{v}$:

$$\hat{\boldsymbol{v}}_{ij} = \varphi(\hat{k}_i, p_j), \quad i = 1, 2, \ldots, M_k, \quad j = 1, 2, \ldots, M_p.$$

Example

- With each key hypothesis of k, we have a hypothetical value for $\boldsymbol{v}$:

$$\hat{\boldsymbol{v}}_{ij} = \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_i \oplus p_j), \quad i = 1, 2, \ldots, 16, \quad j = 1, 2, \ldots, 5000.$$

- p_j is the 0th nibble of the plaintext corresponding to the attack trace $\boldsymbol{\ell}_j$.
- $\hat{k}_i = i - 1, \quad i = 1, 2, \ldots, 16.$

Profiled DPA – step 8

Compute hypothetical target intermediate values

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Example

For our attacks, with each key hypothesis of the 0th nibble of the first round key, we have a hypothetical value for the 0th Sbox output:

$$\hat{\boldsymbol{v}}_{ij} = \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_i \oplus p_j), \quad i = 1, 2, \ldots, 16, \quad j = 1, 2, \ldots, 5000.$$

We set $\hat{k}_i = i - 1, \quad i = 1, 2, \ldots, 16.$

- $\hat{k}_1 = 0$, $\hat{k}_2 = 1$.
- For *Random plaintext dataset*, we have $p_1 = 9, \quad p_2 = \mathtt{C}$.
- $\hat{\boldsymbol{v}}_{ij} = ?\ i = 1, 2,\ j = 1, 2$

Profiled DPA – step 8

Compute hypothetical target intermediate values

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Example

$$\hat{\boldsymbol{v}}_{ij} = \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_i \oplus p_j), \quad i = 1, 2, \ldots, 16, \quad j = 1, 2, \ldots, 5000.$$

- $\hat{k}_1 = \mathtt{0}$, $\hat{k}_2 = \mathtt{1}$.
- $p_1 = \mathtt{9}, \quad p_2 = \mathtt{C}$.

$$\begin{aligned}
\hat{\boldsymbol{v}}_{11} &= \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_1 \oplus p_1) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{0} \oplus \mathtt{9}) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{9}) = \mathtt{E}, \\
\hat{\boldsymbol{v}}_{12} &= \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_1 \oplus p_2) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{0} \oplus \mathtt{C}) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{C}) = \mathtt{4}, \\
\hat{\boldsymbol{v}}_{21} &= \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_2 \oplus p_1) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{1} \oplus \mathtt{9}) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{8}) = \mathtt{3}, \\
\hat{\boldsymbol{v}}_{22} &= \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_2 \oplus p_2) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{1} \oplus \mathtt{C}) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathtt{D}) = \mathtt{7}.
\end{aligned}$$

Leakage model

- Assume a value $\boldsymbol{v}$ is being processed in the DUT
- Let noise $\sim \mathcal{N}(0, \sigma^2)$ be a normal random variable with mean 0 and variance σ^2.
- *Identity leakage model*
  - The leakage is correlated to $\boldsymbol{v}$

$$\mathcal{L}(\boldsymbol{v}) = \boldsymbol{v} + \text{noise}.$$

- *Hamming weight model*
  - The leakage will then be correlated to $\mathrm{wt}\,(\boldsymbol{v})$, the Hamming weight of $\boldsymbol{v}$[1]

$$\mathcal{L}(\boldsymbol{v}) = \mathrm{wt}\,(\boldsymbol{v}) + \text{noise}.$$

Example

$\boldsymbol{v} = \texttt{A}$

- Identity leakage model: $\mathcal{L}(\boldsymbol{v}) = 10 + \text{noise}$
- Hamming weight leakage model: $\mathcal{L}(\boldsymbol{v}) = 2 + \text{noise}$

[1] The Hamming weight of vector $\boldsymbol{v} \in \mathbb{F}_2^m$ is defined to be the number of 1s in $\boldsymbol{v}$

Profiled DPA – step 9

Identify the leakage model and compute hypothetical signals

- By our choice of the target signal, we have a corresponding leakage model.
- In our analysis, we will consider the identity leakage model and the Hamming weight leakage model corresponding to the signal given by $\boldsymbol{v}$ and $\mathrm{wt}\,(\boldsymbol{v})$ respectively.
- For each hypothetical target intermediate value, we can compute the hypothetical signal depending on our leakage model

$$\mathcal{H}_{ij} := \mathcal{L}(\hat{\boldsymbol{v}}_{ij}) - \text{noise}, \quad i = 1, 2, \ldots, M_k, \quad j = 1, 2, \ldots, M_p.$$

Example

- With each key hypothesis of k, we have a hypothetical value for $\boldsymbol{v}$:

$$\hat{\boldsymbol{v}}_{ij} = \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_i \oplus p_j), \quad i = 1, 2, \ldots, 16, \quad j = 1, 2, \ldots, 5000.$$

- Hamming weight leakage model: $\mathcal{H}_{ij} = \mathrm{wt}\,(\hat{\boldsymbol{v}}_{ij})$
- Identity leakage mode: $\mathcal{H}_{ij} = \hat{\boldsymbol{v}}_{ij}$

Recall what we computed last week

Same computations apply here

Example

- Aim to recover the 0th nibble of the first round key for PRESENT encryption, denoted k
- The target intermediate value is the 0th Sbox output
- Attack traces: *Random plaintext dataset*– 5000 measurements, the corresponding 0th nibble of plaintext is denoted p_j
- With each key hypothesis of k, we have a hypothetical value for $\boldsymbol{v}$:

$$\hat{\boldsymbol{v}}_{ij} = \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_i \oplus p_j), \quad i = 1, 2, \ldots, 16, \quad j = 1, 2, \ldots, 5000.$$

- With a chosen leakage model, we have a hypothetical signal
  - Hamming weight leakage model: $\mathcal{H}_{ij} = \mathrm{wt}\left(\hat{\boldsymbol{v}}_{ij}\right)$
  - Identity leakage mode: $\mathcal{H}_{ij} = \hat{\boldsymbol{v}}_{ij}$

Profiled DPA – step 10

Statistical analysis

- For a fixed key hypothesis $\hat{k}_i$, we view the modeled leakage as a random variable $\mathcal{H}_i$ that varies when the plaintext changes.
- L_{POI}: random variable, leakage at POI
- Then a sample for this pair of random variables $(\mathcal{H}_i, L_{\text{POI}})$ is given by

$$\left\{ (\mathcal{H}_{ij}, l^j_{\text{POI}}) \;\middle|\; j = 1, 2, \ldots, \hat{M}_p \right\},$$

  where l^j_{POI} is the POI-th entry of the attack trace $\boldsymbol{\ell}_j$ and $2 \leq \hat{M}_p \leq M_p$
- With this sample, we can compute the sample correlation coefficient between $\mathcal{H}_i$ and L_{POI} for each key hypothesis $\hat{k}_i$ $(i = 1, 2, \ldots, M_k)$:

$$r^{\hat{M}_p}_{i,\text{POI}} := \frac{\sum_{j=1}^{\hat{M}_p} (\mathcal{H}_{ij} - \overline{\mathcal{H}_i})(l^j_{\text{POI}} - \overline{l_{\text{POI}}})}{\sqrt{\sum_{j=1}^{\hat{M}_p} (\mathcal{H}_{ij} - \overline{\mathcal{H}_i})^2} \sqrt{\sum_{j=1}^{\hat{M}_p} (l^j_{\text{POI}} - \overline{l_{\text{POI}}})^2}}.$$

Profiled DPA – identity leakage model

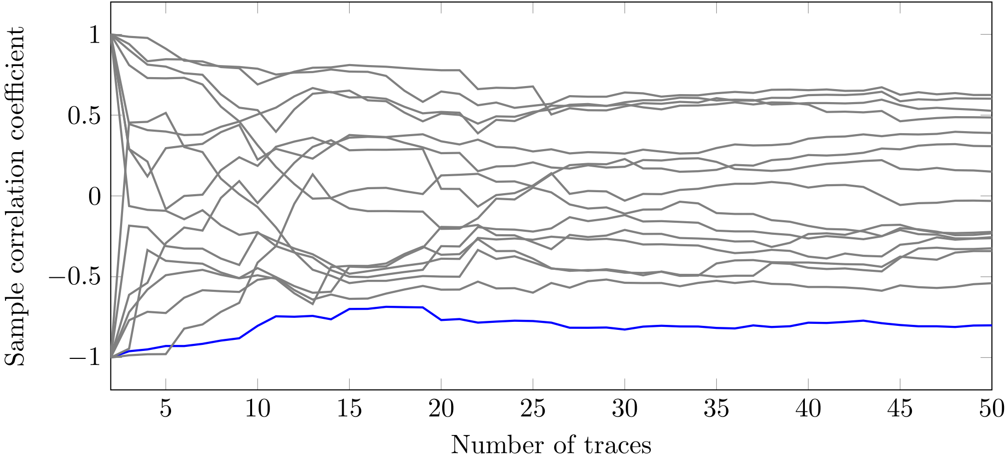


Figure: POI $= 392$. Computed with the *Random plaintext dataset*. The blue line corresponds to the correct key hypothesis $\hat{k}_{10} =$ 9.

Profiled DPA – Hamming weight leakage model

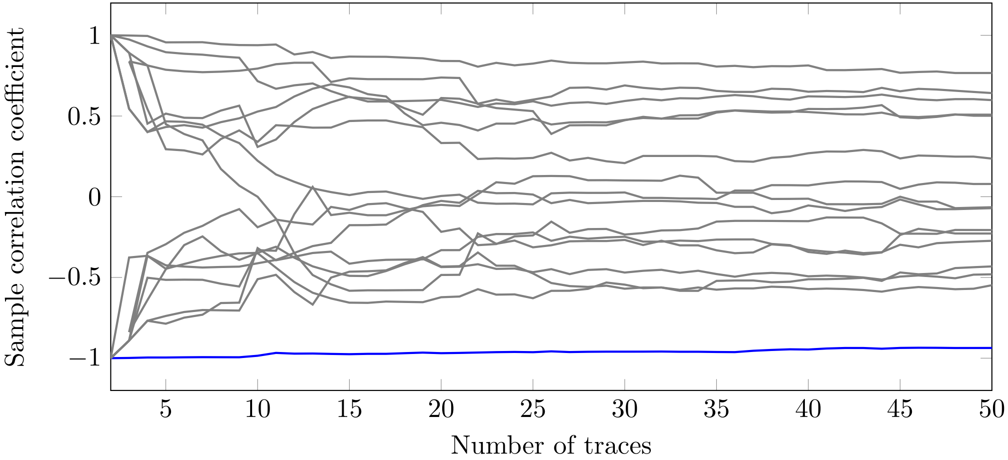


Figure: POI $= 392$. Computed with the *Random plaintext dataset*. The blue line corresponds to the correct key hypothesis $\hat{k}_{10} =$ 9.

Nonprofiled DPA from last week

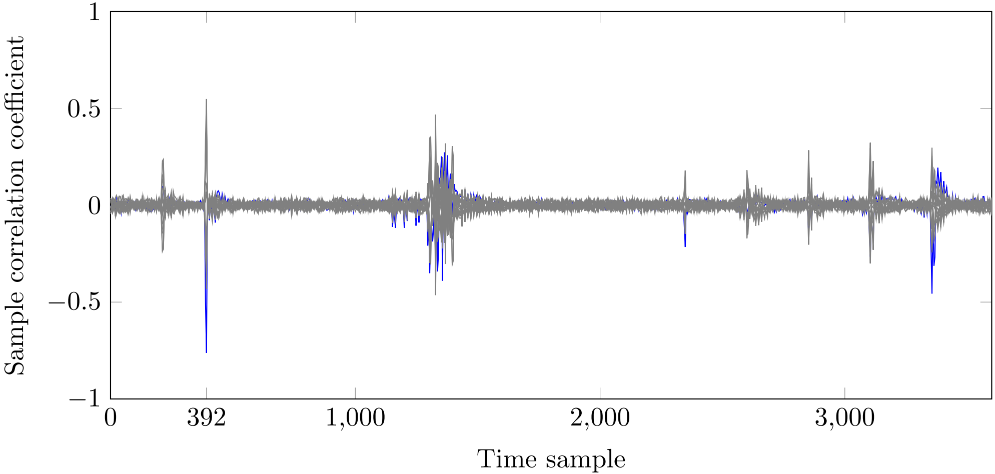


Figure: Sample correlation coefficients for all 3600 time samples and all 16 key hypotheses. Computed with the identity leakage model and *Random plaintext dataset*. The blue line corresponds to the correct key hypothesis $\hat{k}_{10} = 9$.

Observations

- The Hamming weight leakage model is closer to our DUT leakage compared to the identity leakage model
- *A good leakage model is beneficial to our attack.*

Profiled DPA

- Profiled DPA Attack Steps
- Stochastic Leakage Model
- Template-based DPA
- Success Rate and Guessing Entropy
- Further Reading

Stochastic leakage model

- To fully utilize the cloned device in the profiled setting, we can further characterize the leakages instead of just identifying the POI.
- Stochastic leakage model: assumes each bit of the target intermediate value $\boldsymbol{v} = v_{m_v-1}v_{m_v-2}\dots v_1 v_0$ has a different leakage.

$$\mathcal{L}(\boldsymbol{v}) = \sum_{s=0}^{m_v-1} \alpha_s v_s + \text{noise},$$

- noise $\sim \mathcal{N}(0, \sigma^2)$ denotes the noise with variance σ^2
- α_s $(s = 0, 1, \dots, m_v - 1)$ are real numbers – *coefficients* of the stochastic leakage model

Attack with stochastic leakage model

- The attack with stochastic leakage model follows the same steps as described before.
- The only difference is in Step 9, where we need extra effort to find our leakage model by profiling.
- We note that since the stochastic leakage model assumes each value of $\boldsymbol{v}$ has different signals, to identify the POI, we will choose the target signal to be the exact value of $\boldsymbol{v}$ in Step 5.
- Then using the leakages at the POI we will find estimations for α_s values.
- Those estimated values together with

$$\mathcal{L}(\boldsymbol{v}) = \sum_{s=0}^{m_v-1} \alpha_s v_s + \text{noise},$$

provide us with hypothetical signals in Step 9.

Approximating α_s values

- We only focus on the leakage at the POI from each profiling trace.
- Let $\boldsymbol{\ell}_{pf}$ be the vector of leakages at $t =$POI from all M_{pf} profiling traces.
- We will find approximations of α_s with *ordinary least square method* from linear regression.

Example

- Same attack goal and target intermediate value as before
- POI $= 392$ which corresponds to the signal being the exact value of $\boldsymbol{v}$.
- We will use *Random dataset* as our profiling traces, hence $M_{pf} = 10000$.

Approximating α_s values

- For the jth profiling trace, let

$$\boldsymbol{v}_j^{pf} = v_{j(m_v-1)}^{pf} \dots v_{j1}^{pf} v_{j0}^{pf}, \quad j = 1, 2, \dots, M_{pf}$$

  be the corresponding target intermediate value.
- Compute matrix $M_{\boldsymbol{v}}$

$$M_{\boldsymbol{v}} := \begin{pmatrix} v_{10}^{pf} & v_{11}^{pf} & \dots & v_{1(m_v-1)}^{pf} \\ v_{20}^{pf} & v_{21}^{pf} & \dots & v_{2(m_v-1)}^{pf} \\ \vdots & \vdots & \ddots & \vdots \\ v_{M_{pf}0}^{pf} & v_{M_{pf}1}^{pf} & \dots & v_{M_{pf}(m_v-1)}^{pf} \end{pmatrix}$$

- The estimated values $\hat{\alpha}_s$ for α_s are given by

$$\begin{pmatrix} \hat{\alpha}_0 & \hat{\alpha}_1 & \dots & \hat{\alpha}_{m_v-1} \end{pmatrix}^\top = \left(M_{\boldsymbol{v}}^\top M_{\boldsymbol{v}}\right)^{-1} M_{\boldsymbol{v}}^\top \boldsymbol{\ell}_{pf}^\top.$$

Example for matrix M_v

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Table: PRESENT Sbox.

Example

Take POI $= 392$. The first trace in *Random dataset* corresponds to the 0th nibble of the plaintext$=$ 4 and the 0th nibble of the first round key$=$ 7. Thus the intermediate value for the first trace is given by:

$$\boldsymbol{v}_1^{pf} = \mathsf{SB}_{\mathsf{PRESENT}}(4 \oplus 7) = \mathsf{SB}_{\mathsf{PRESENT}}(3) = \mathtt{B} = 1011_2.$$

Example for matrix $M_{\boldsymbol{v}}$

- For the jth profiling trace, let

$$\boldsymbol{v}_j^{pf} = v_{j(m_v-1)}^{pf} \dots v_{j1}^{pf} v_{j0}^{pf}, \quad j = 1, 2, \dots, M_{pf}$$

be the corresponding target intermediate value.

$$M_{\boldsymbol{v}} := \begin{pmatrix} v_{10}^{pf} & v_{11}^{pf} & \cdots & v_{1(m_v-1)}^{pf} \\ v_{20}^{pf} & v_{21}^{pf} & \cdots & v_{2(m_v-1)}^{pf} \\ \vdots & \vdots & \ddots & \vdots \\ v_{M_{pf}0}^{pf} & v_{M_{pf}1}^{pf} & \cdots & v_{M_{pf}(m_v-1)}^{pf} \end{pmatrix}$$

Example

Take POI $= 392$. The first trace in *Random dataset* corresponds to the intermediate value

$$\boldsymbol{v}_1^{pf} = \mathsf{SB}_{\mathsf{PRESENT}}(\mathsf{4} \oplus \mathsf{7}) = \mathsf{SB}_{\mathsf{PRESENT}}(\mathsf{3}) = \mathtt{B} = 1011_2.$$

And the first row of our matrix $M_{\boldsymbol{v}}$ is given by ?

Example for matrix $M_{\boldsymbol{v}}$

- For the jth profiling trace, let

$$\boldsymbol{v}_j^{pf} = v_{j(m_v-1)}^{pf} \dots v_{j1}^{pf} v_{j0}^{pf}, \quad j = 1, 2, \dots, M_{pf}$$

be the corresponding target intermediate value.

$$M_{\boldsymbol{v}} := \begin{pmatrix} v_{10}^{pf} & v_{11}^{pf} & \cdots & v_{1(m_v-1)}^{pf} \\ v_{20}^{pf} & v_{21}^{pf} & \cdots & v_{2(m_v-1)}^{pf} \\ \vdots & \vdots & \ddots & \vdots \\ v_{M_{pf}0}^{pf} & v_{M_{pf}1}^{pf} & \cdots & v_{M_{pf}(m_v-1)}^{pf} \end{pmatrix}$$

Example

Take POI $= 392$. The first trace in *Random dataset* corresponds to the intermediate value $\mathtt{B} = 1011_2$ And the first row of our matrix $M_{\boldsymbol{v}}$ is given by

$$\begin{pmatrix} 1 & 1 & 0 & 1 \end{pmatrix}.$$

Profiling results

- Stochastic leakage model: assumes each bit of the target intermediate value $\boldsymbol{v} = v_{m_v-1}v_{m_v-2}\dots v_1v_0$ has a different leakage.

$$\mathcal{L}(\boldsymbol{v}) = \sum_{s=0}^{m_v-1} \alpha_s v_s + \text{noise},$$

Example

- With POI $= 392$ and the *Random dataset*, we got the following estimated values for α_ss:

$$\hat{\alpha}_0 \approx -0.02019, \quad \hat{\alpha}_1 \approx -0.02027, \quad \hat{\alpha}_2 \approx -0.01920, \quad \hat{\alpha}_3 \approx -0.02039.$$

- The leakage of a $\boldsymbol{v} = v_3v_2v_1v_0$ is given by

$$\mathcal{L}(\boldsymbol{v}) = \hat{\alpha_0}v_0 + \hat{\alpha_1}v_1 + \hat{\alpha_2}v_2 + \hat{\alpha_3}v_3 + \text{noise}.$$

- For example, $\mathcal{L}(\mathtt{E}) =?$

Profiling results

- Stochastic leakage model: assumes each bit of the target intermediate value $\boldsymbol{v} = v_{m_v-1}v_{m_v-2}\dots v_1v_0$ has a different leakage.

$$\mathcal{L}(\boldsymbol{v}) = \sum_{s=0}^{m_v-1} \alpha_s v_s + N,$$

Example

- With POI $= 392$ and the *Random dataset*, we got the following estimated values for α_ss:

$$\hat{\alpha}_0 \approx -0.02019, \quad \hat{\alpha}_1 \approx -0.02027, \quad \hat{\alpha}_2 \approx -0.01920, \quad \hat{\alpha}_3 \approx -0.02039.$$

- The leakage of a $\boldsymbol{v} = v_3v_2v_1v_0$ is given by

$$\mathcal{L}(\boldsymbol{v}) = \hat{\alpha_0}v_0 + \hat{\alpha_1}v_1 + \hat{\alpha_2}v_2 + \hat{\alpha_3}v_3 + \text{noise}.$$

- For example, $\mathcal{L}(\texttt{E}) = \hat{\alpha_1} + \hat{\alpha_2} + \hat{\alpha_3} + \text{noise} = -0.05986 + \text{noise}.$

Attack results

Example

- For our attacks, we aim to recover the 0th nibble of the first round key for PRESENT encryption, denoted k
- The target intermediate value is the 0th Sbox output
- Attack traces: *Random plaintext dataset*– 5000 measurements, the corresponding 0th nibble of plaintext is denoted p_j
- With each key hypothesis of k, we have a hypothetical value for $\boldsymbol{v}$:

$$\hat{\boldsymbol{v}}_{ij} = \mathsf{SB}_{\mathsf{PRESENT}}(\hat{k}_i \oplus p_j), \quad i = 1, 2, \ldots, 16, \quad j = 1, 2, \ldots, 5000.$$

- With the stochastic leakage model, we have a hypothetical signal
  - $\mathcal{H}_{ij} = \mathcal{L}(\boldsymbol{v}) - \text{noise} = \hat{\alpha_0} v_0 + \hat{\alpha_1} v_1 + \hat{\alpha_2} v_2 + \hat{\alpha_3} v_3$
- We have computed that $\hat{\boldsymbol{v}}_{11} = \mathtt{E}, \quad \hat{\boldsymbol{v}}_{12} = 4, \quad \hat{\boldsymbol{v}}_{21} = 3, \quad \hat{\boldsymbol{v}}_{22} = 7.$
- $\mathcal{H}_{11} = -0.05986$, $\mathcal{H}_{12} = -0.03959$, $\mathcal{H}_{21} = -0.02039$, $\mathcal{H}_{22} = -0.05979$

Attack results - stochastic leakage model

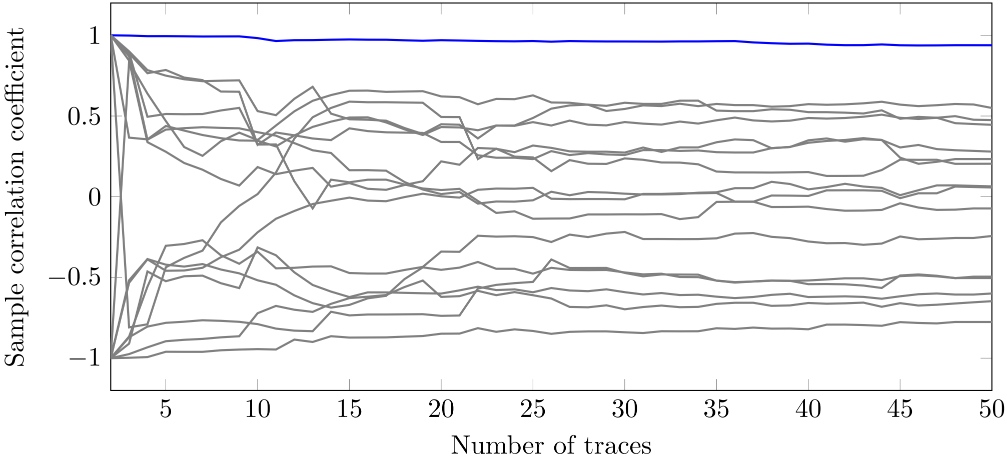


Figure: POI $= 392$. Computed with the stochastic leakage model and *Random plaintext dataset*. The blue line corresponds to the correct key hypothesis $\hat{k}_{10} =$ 9.

Comparison – identity leakage model

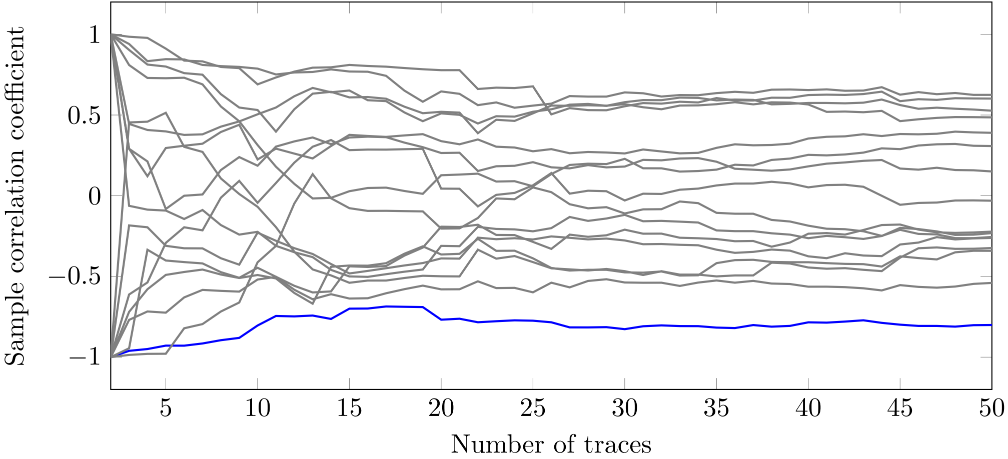


Figure: POI $= 392$. Computed with the *Random plaintext dataset*. The blue line corresponds to the correct key hypothesis $\hat{k}_{10} =$ 9.

Comparison – Hamming weight leakage model

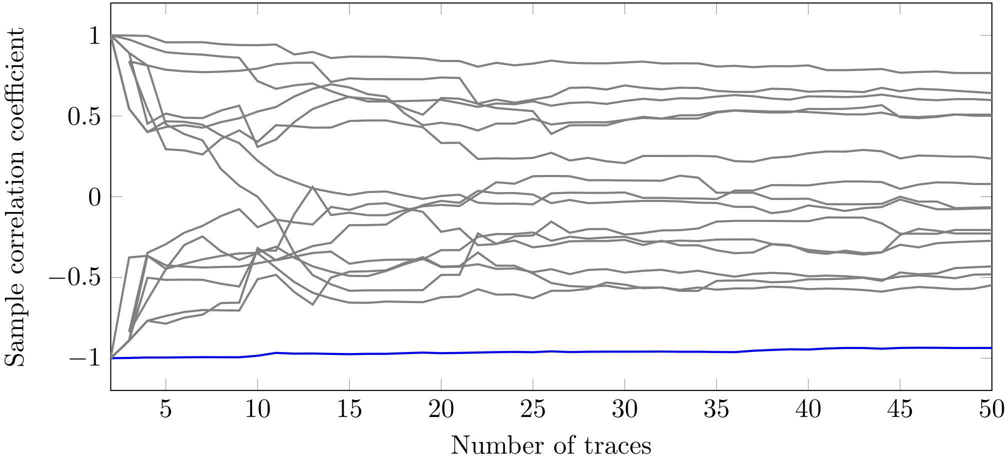


Figure: POI $= 392$. Computed with the *Random plaintext dataset*. The blue line corresponds to the correct key hypothesis $\hat{k}_{10} = 9$.

Profiled DPA

- Profiled DPA Attack Steps
- Stochastic Leakage Model
- Template-based DPA
- Success Rate and Guessing Entropy
- Further Reading

Template

- We have seen how to characterize the leakage assuming each bit of the target intermediate value leaks differently focusing on one POI.
- We can also characterize/profile the leakages of each possible value of the target intermediate value at several POIs.
- The result of this profiling process is a set of *templates*.
- Then during the attack phase, instead of computing correlation coefficients, we use those templates to see which of them fits better to the measured power trace and deduce a probability for each key hypothesis.

Distribution of leakages

- For a fixed time sample t, let L_t, X_t, and N_t denote the random variables corresponding to the leakage, signal, and noise respectively

$$L_t = X_t + N_t.$$

- We consider X_t and N_t to be independent
- We fix the operation and the data, and we get a constant signal, i.e. X_t is a constant
    - The variants in the leakage will be caused by the noise
    - We approximate the distribution induced by L_t with a normal distribution

$$L_t \sim \mathcal{N}(\mu_t, \sigma_t^2).$$

The goal of profiling in template-based DPA is to estimate the mean and variance of the normal random variable L_t. The resulting estimations are our templates.

Recall – profiled DPA steps

Step 1 Identify the target cryptographic implementation

Step 2 Measurement of profiling traces

Step 3 Choose the part of the key to recover

Step 4 Choose the target intermediate value

Step 5 Decide on the target signal

Step 6 Compute SNR and identify the POI

Step 7 Measurement of attack traces

Step 8 Compute hypothetical target intermediate values

Step 9 Identify the leakage model and compute hypothetical signals

Step 10 Statistical analysis

Template-based DPA

Step 1 Identify the target cryptographic implementation

Step 2 Measurement of profiling traces

Step 3 Choose the part of the key to recover

Step 4 Choose the target intermediate value

Step 5 Decide on the target signal

Step 6 Compute SNR and identify the POI

Step 7 Measurement of attack traces

Step 8 Compute hypothetical target intermediate values

Step 9 ~~Identify the leakage model and compute hypothetical signals~~

Step 9 Group the profiling traces

Step 10 Build template

Step 11 Statistical analysis

Template-based DPA – step 9

Group the profiling traces

- We take our set of profiling traces and divide them into M_{signal} sets according to the target signal chosen in profiled DPA step 5
- Let us denote those sets by A_1, A_2, $\ldots$, $A_{M_{signal}}$.

Example

- For our attacks, when the target signal is given by $\boldsymbol{v}$, the exact value of the output of the 0th Sbox in PRESENT, we will divide our profiling traces *Random dataset* into 16 sets, $A_1, A_2, \ldots, A_{16}$, where A_s contains traces corresponding to $\boldsymbol{v} = s - 1$.
- When the target signal is given by $\mathrm{wt}\left(\boldsymbol{v}\right)$, the Hamming weight of $\boldsymbol{v}$, we will divide the profiling traces into 5 sets, $A_1, A_2, \ldots, A_5$, where A_s contains traces corresponding to $\mathrm{wt}\left(\boldsymbol{v}\right) = s - 1$.

Template-based DPA – step 10

Build template

- Let us fix a particular target signal value and only consider inputs to the cryptographic algorithm that result in traces belonging to the corresponding set A_s
- Let L_s denote the random variable representing the leakage for such encryption computations at time sample POI.
- Then L_s can be modeled by a normal random variable.

Normal random variable

- Let $Z \sim \mathcal{N}(0,1)$ be a standard normal random variable
- Take any $\sigma, \mu \in \mathbb{R}$ with $\sigma^2 > 0$
- Define $Y = \sigma Z + \mu$
- It can be shown that Y has PDF

  $$f(y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right).$$

  And

  $$\mathrm{E}\left[Y\right] = \mu, \quad \mathrm{Var}(Y) = \sigma^2$$

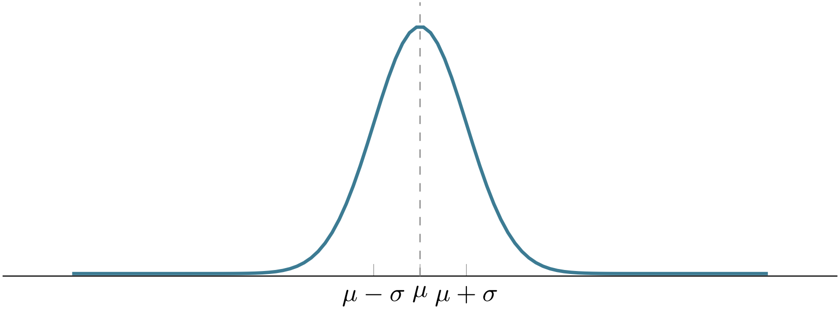
- We say that Y induces a *normal distribution with mean* μ *and variance* σ^2, written $Y \sim \mathcal{N}(\mu, \sigma^2)$. Y is also called *normal/a normal random variable*. We note that the mean and variance fully define a normal distribution.

Normal random variable

$f(y)$ is a bell-shaped curve symmetric about μ and obtains its maximum value of

$$\frac{1}{\sigma\sqrt{2\pi}} \approx \frac{0.399}{\sigma}$$

at $y = \mu$



Template-based DPA – step 10

Build template

- To find the PDF of a normal random variable, we need to identify its mean and variance.
- Using our profiling traces from set A_s, we can compute an approximation for the mean, denoted μ_s, using sample mean of L_s
- Similarly, an approximation for the variance, σ_s^2, is then given by the sample variance of L_s
- The pair (μ_s, σ_s^2) is called a *template*.
- With our profiling traces, we can compute M_{signal} templates.

Example

For our attacks,

- When the target signal is $\boldsymbol{v}$, we will have 16 templates, each corresponding to a possible value of $\boldsymbol{v}$ from 0 to $\mathtt{F}$.
- When the target signal is $\mathrm{wt}\left(\boldsymbol{v}\right)$, we will have 5 templates, each corresponding to a Hamming weight value from 0 to 4.

Template-based DPA – step 11

Statistical analysis

- In this step, we would like to compute a probability for each key hypothesis given the attack traces
- We are only interested in the leakages at the POIs for each attack trace $\boldsymbol{\ell}_j = (l_1^j, l_2^j, \ldots, l_q^j) - l_{\mathsf{POI}}^j$
- For each key hypothesis $\hat{k}_i$ and attack trace $\boldsymbol{\ell}_j$, we compute the hypothetical target intermediate value given the knowledge of the associated plaintext.
- Let $\mu_{s_{ij}}$ and $\sigma^2_{s_{ij}}$ be the template for this hypothetical value, corresponding to $\hat{k}_i$ and $\boldsymbol{\ell}_j$

Template-based DPA – step 11

Statistical analysis

- According to the PDF of a normal random variable

$$f(y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(y-\mu)^2}{2\sigma^2}\right).$$

- We can compute the probability of $\boldsymbol{\ell}_j$ given $\hat{k}_i$

$$P(\boldsymbol{\ell}_j|\hat{k}_i) = P(L_{s_{ij}} = l^j_{\mathsf{POI}}) = \frac{1}{\sqrt{2\sigma^2_{s_{ij}}\pi}} \exp\left(-\frac{(l^j_{\mathsf{POI}} - \mu_{s_{ij}})^2}{2\sigma^2_{s_{ij}}}\right),$$

- The score of $\hat{k}_i$ is given by

$$\mathsf{P}_{\hat{k}_i} = -\sum_{j=1}^{\hat{M}_p} \ln(\sigma^2_{s_{ij}}) + \frac{(l^j_{\mathsf{POI}} - \mu_{s_{ij}})^2}{\sigma^2_{s_{ij}}}.$$

- The higher the score, the more likely the hypothesis is equal to the correct key.

Profiling results

- Mean values

$$-0.042456, \quad -0.046059, \quad -0.046314, \quad -0.047441, \quad -0.045003, \quad -0.048246,$$

$$-0.048646, \quad -0.051325, \quad -0.046065, \quad -0.049202, \quad -0.049648, \quad -0.052415,$$

$$-0.048188, \quad -0.052684, \quad -0.051630, \quad -0.053906$$

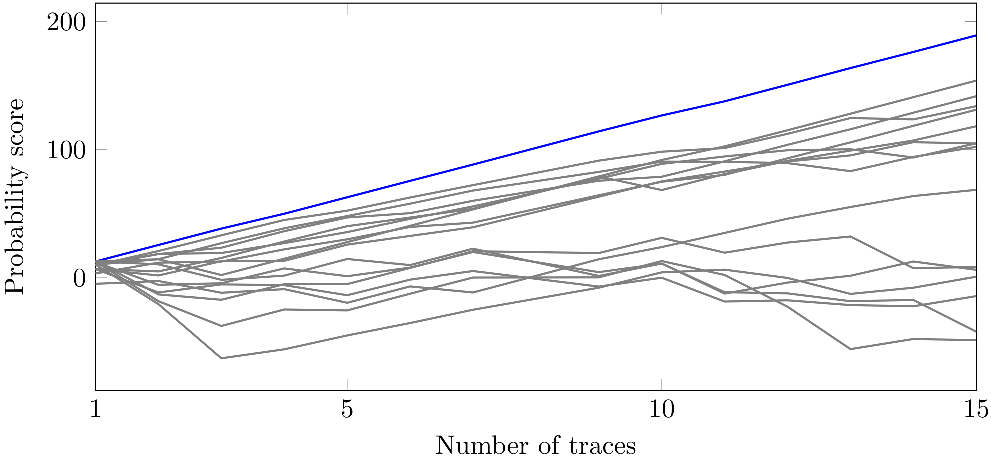
- Variances

$$0.0000230, \quad 0.0000232, \quad 0.0000248, \quad 0.0000229, \quad 0.0000235, \quad 0.0000247,$$

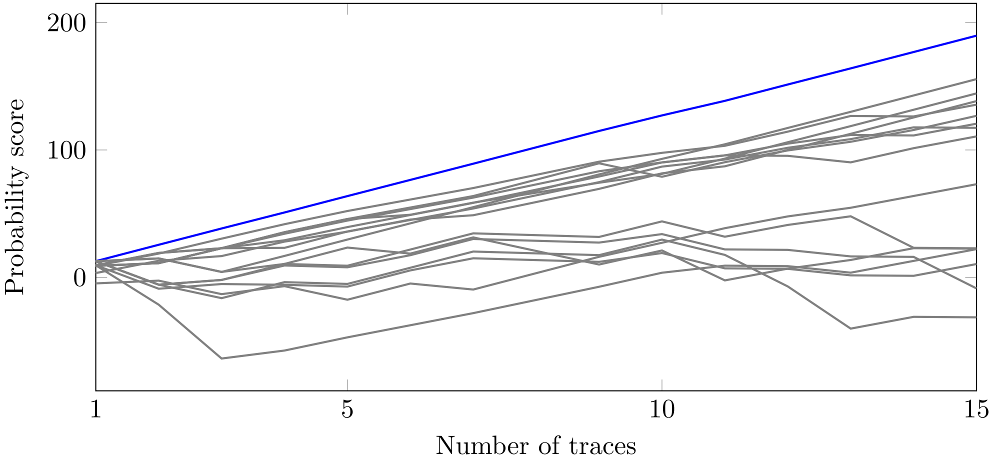
$$0.0000217, \quad 0.0000228, \quad 0.0000234, \quad 0.0000257, \quad 0.0000251, \quad 0.0000203,$$

$$0.0000255, \quad 0.0000243, \quad 0.0000241, \quad 0.0000274,$$

Attack results - target signal given by $\boldsymbol{v}$



Attack results - target signal given by $\mathrm{wt}\,(\boldsymbol{v})$



Note

- We can also use more than one POIs (Gaussian random vector)

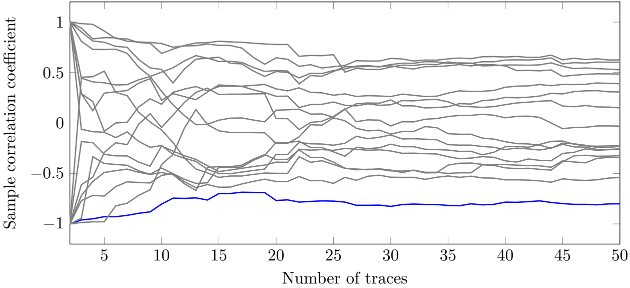
Profiled DPA

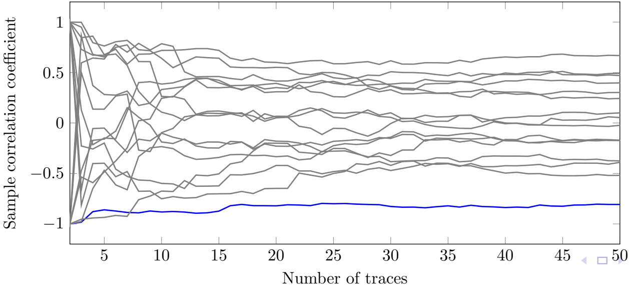
- Profiled DPA Attack Steps
- Stochastic Leakage Model
- Template-based DPA
- Success Rate and Guessing Entropy
- Further Reading

Different ordering of traces

- A different ordering of the traces in *Random plaintext dataset* may affect our attack results.
- For example, if arrange the traces in *Random plaintext dataset* in reverse order, we get different results

Profiled DPA – identity model





Different ordering of traces

- To have a fair comparison between different attack methods (e.g. different choices of leakage models, POIs, etc.), we introduce the notion of *success rate* and *guessing entropy*.

Remark

Our aim is to evaluate the DUT and our implementation against DPA attacks with different settings. Thus, we assume we have the knowledge of the key for the evaluation after the attack.

Key rank

- Fix a number of attack traces $\hat{M}_p$
- For a profiled DPA attack, we assign a *score* to each key hypothesis after the attack: the *absolute value* of the corresponding sample correlation coefficient at POI
- Sort the scores in an array in descending order
  - Rank 1st – highest score
- *Key rank* of a key hypothesis: index of the score for the key hypothesis in this sorted array
- Ultimate goal of an attack: key rank of the correct key hypothesis $= 1$ – We note that if the key rank is low enough, it is possible to use key enumeration algorithms[1] that enable the key recovery

[1] Veyrat-Charvillon, N., Gérard, B., Renauld, M., & Standaert, F. X. (2013). An optimal key enumeration algorithm and its application to side-channel attacks. In Selected Areas in Cryptography: 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers 19 (pp. 390-406). Springer Berlin Heidelberg.

Success rate

- The *success rate* of an attack method with $\hat{M}_p$ traces, denoted $\mathsf{SR}_{\hat{M}_p}$, is defined to be

$$\mathsf{SR}_{\hat{M}_p} = P\left(\text{key rank of the correct key hypothesis} = 1\right).$$

- Empirically, we can estimate the value of $\mathsf{SR}_{\hat{M}_p}$ by computing the frequency of key rank of the correct key hypothesis $= 1$ among a certain number of simulated attacks.

Guessing entropy

- The guessing entropy for an attack method with $\hat{M}_p$ traces is given by the expectation of the key rank of the correct key hypothesis:

$$\mathsf{GE}_{\hat{M}_p} = \mathrm{E}\left[\text{key rank of the correct key hypothesis with } \hat{M}_p \text{ traces}\right].$$

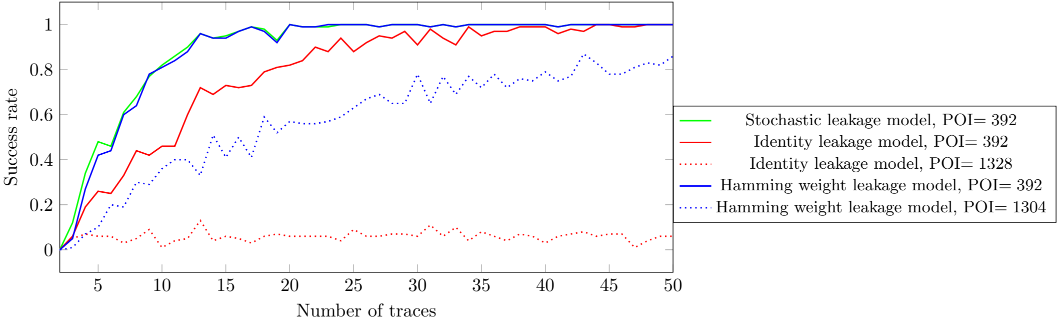
- We can approximate $\mathsf{GE}_{\hat{M}_p}$ with a sample mean of the correct key rank.

Estimating success rate and guessing entropy of an attack

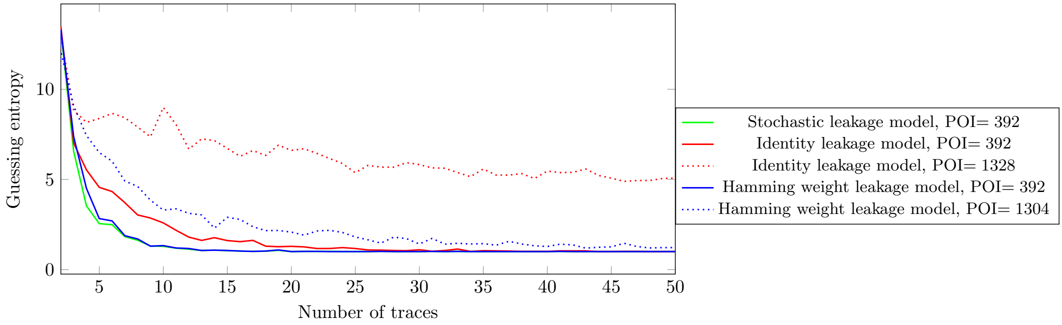
- For a fixed number of traces, we carry out the attacks for a certain number of times, e.g. 100

```
zero array of size $51$    $S_{sr}$, $S_{ge}$// variables to store estimations of success rate
    and guessing entropy, initialized to zero
for $\hat{M}_p = 2$, $\hat{M}_p \leq 50$, $\hat{M}_p++$ do
    array of size $\hat{M}_p \times$no_of_attack    $A \xleftarrow{\text{randomly choose}}$ Random plaintext dataset
    for $i = 0$, $i < 100$, $i++$ do
        $B = A[i \times \hat{M}_p : (i+1) \times \hat{M}_p]$// take $\hat{M}_p$ traces from $A$ without repetition
            for each $i$th attack
        Carry out the attack, compute $rk =$ key rank of the correct key
        $S_{ge}[\hat{M}_p] += rk$
        if $rk == 1$ then
            $S_{sr}[\hat{M}_p] += 1$
    $S_{sr}[\hat{M}_p] = S_{sr}[\hat{M}_p]/$no_of_attack// compute the frequency of successful attacks
    $S_{ge}[\hat{M}_p] = S_{ge}[\hat{M}_p]/$no_of_attack// compute the sample mean
```

Success rate – results



Guessing entropy – results



Observations

- Fewer traces are needed for SR/GE to reach 1 with the Hamming weight model and stochastic model.
- Attack results for the Hamming weight model and stochastic model are similar, with the stochastic model giving slightly better performance.
- The choice of POI is important for the attack. With the wrong POI, the attack will need much more traces.

Results for template-based DPA – success rate

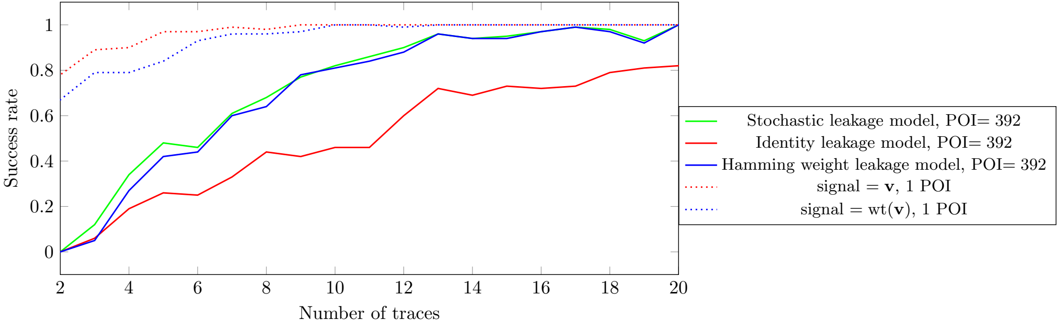


Figure: Estimations of success rate for leakage model based and template-based DPA attacks computed using the *Random plaintext dataset*.

Results for template-based DPA – guessing entropy

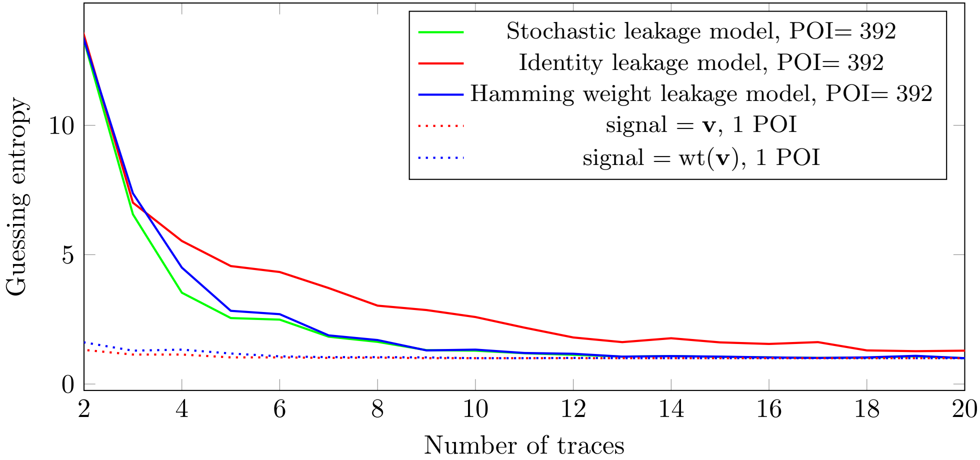


Figure: Estimations of guessing entropy for leakage model based template-based DPA attacks computed using the *Random plaintext dataset*.

Observations

- When the target signal is given by $\boldsymbol{v}$, the attack requires fewer traces as compared to the case when the target signal is given by $\mathrm{wt}\left(\boldsymbol{v}\right)$. This is expected as for the former case we have 16 templates while for the latter we have 5.
    - Of course, the attack results demonstrated that we had enough traces for profiling to get good templates. Without enough profiling traces, different attack results might appear.
- Template-based DPA, in general, performs better than leakage model based DPA. This is not surprising as more information is retrieved from the profiling traces using template-based attacks.

Profiled DPA

- Profiled DPA Attack Steps
- Stochastic Leakage Model
- Template-based DPA
- Success Rate and Guessing Entropy
- Further Reading

Remarks

- Last week we discussed SPA on RSA
- It is more common to see SPA attacks on public key ciphers and DPA on symmetric key block ciphers
- There are also
  - DPA on RSA[1]
  - Profiled SPA[2]

[1] Amiel, F., Feix, B., & Villegas, K. (2007, August). Power analysis for secret recovering and reverse engineering of public key algorithms. In International Workshop on Selected Areas in Cryptography (pp. 110-125). Springer, Berlin, Heidelberg.

[2] Mangard, S., Oswald, E., & Popp, T. (2008). Power analysis attacks: Revealing the secrets of smart cards (Vol. 31). Springer Science & Business Media. Chapter 5

Other Attacks

- Collision attacks: identify collision of intermediate values in power trace to recover secret key
  - Schramm, K., Wollinger, T., & Paar, C. (2003, February). A new class of collision attacks and its application to DES. In International Workshop on Fast Software Encryption (pp. 206-222). Springer, Berlin, Heidelberg.
- Algebraic side-channel attacks (ASCA): express both the target algorithm and its leakages as equations
  - Renauld, M., & Standaert, F. X. (2009, December). Algebraic side-channel attacks. In International Conference on Information Security and Cryptology (pp. 393-410). Springer, Berlin, Heidelberg.
- SCADPA: side-channel assisted differential plaintext attack
  - Breier, J., Jap, D., Hou, X., & Bhasin, S. (2018). On side-channel vulnerabilities of bit permutations: Key recovery and reverse engineering. Cryptology ePrint Archive.

AI-assisted SCA – Motivation

- If we look at DPA, the key recovery is essentially a classification problem.
- In particular, in a profiled setting, the analysis of the leakage traces can be seen as a classification problem where the goal of an attacker is to classify those traces based on the related data (e.g. a specific Sbox output value).
- Various AI-based techniques have been adopted for SCA
- It has also been shown that, with neural networks, protected implementations can be broken.

Various AI Techniques have been Adopted for SCA

- $k-$nearest neighbor algorithm[1]
- Random forest[2]
- Support vector machines[3]
- Multilayer perceptron (MLP)[4]
- Convolutional neural networks (CNN)[5]

[1] Martinasek, Z., Zeman, V., Malina, L., & Martinasek, J. (2016). K-nearest neighbors algorithm in profiling power analysis attacks. Radioengineering, 25(2), 365-382.

[2] L. Lerman, G. Bontempi, O. Markowitch. "A machine learning approach against a masked AES." Journal of Cryptographic Engineering 2015.

[3] A. Heuser, M. Zohner. "Intelligent machine homicide." International Workshop on Constructive Side-Channel Analysis and Secure Design. 2012.

[4] R. Gilmore, N. Hanley, M. O'Neill. "Neural network based attack on a masked implementation of AES." 2015 IEEE International Symposium on Hardware Oriented Security and Trust (HOST).

[5] Zaid, G., Bossuet, L., Habrard, A., Venelli, A. "Methodology for efficient CNN architectures in profiling attacks." IACR Transactions on Cryptographic Hardware and Embedded Systems, 2020.

DPA on AES

- Target intermediate value: 0th Sbox output
- Instead of one nibble of the key, we recover one byte of the key

AI-assisted SCA on AES implementation

- Neural network used for the classification problem in a DPA attack on AES implementations
  - Input of the network: (part of) the traces.
  - Output layer: a softmax activation function and each class corresponds to one possible value of the target Sbox output, hence leading to one key byte hypothesis with the knowledge of the plaintext.
- During the inference, for each input data, the network output indicates the possibilities of the 256 values for the Sbox output, which gives a possibility of each of the corresponding key byte hypotheses.
- Guessing entropy and success rate can be defined in a similar way
- The goal of AI-based SCA is then to achieve low guessing entropy (or high success rate) with as few traces as possible after training.

Hyperparameter Tuning

- Bayesian optimization and random search[1]
- Reinforcement learning[2]
- Genetic algorithm for choosing architectures[3]
- Genetic algorithm for all hyperparameters[4]

[1] Wu, L., Perin, G., & Picek, S. (2020). I choose you: Automated hyperparameter tuning for deep learning-based side-channel analysis. Cryptology ePrint Archive.

[2] Rijsdijk, J., Wu, L., Perin, G., & Picek, S. (2021). Reinforcement learning for hyperparameter tuning in deep learning-based side-channel analysis. IACR Transactions on Cryptographic Hardware and Embedded Systems, 677-707.

[3] Maghrebi, H., Portigliatti, T., & Prouff, E. (2016, December). Breaking cryptographic implementations using deep learning techniques. In International Conference on Security, Privacy, and Applied Cryptography Engineering (pp. 3-26). Springer, Cham.

[4] Acharya, R. Y., Ganji, F., & Forte, D. (2021). InfoNEAT: Information Theory-based NeuroEvolution of Augmenting Topologies for Side-channel Analysis. CHES 2023.

Training Strategy

- Test accuracy in machine learning cannot properly assess SCA performance[1]
- Stopping criteria based on success rate[2]
- Stopping criteria based on mutual information[3]

[1] Picek, S., Heuser, A., Jovic, A., Bhasin, S., Regazzoni, F. “The curse of class imbalance and conflicting metrics with machine learning for side-channel evaluations.” IACR Transactions on Cryptographic Hardware and Embedded Systems, 2019.

[2] Robissout, D., Zaid, G., Colombier, B., Bossuet, L., & Habrard, A. (2020). Online performance evaluation of deep learning networks for side-channel analysis. Cryptology ePrint Archive.

[3] Perin, G., Buhan, I., & Picek, S. (2020). Learning when to stop: a mutual information approach to fight overfitting in profiled side-channel analysis. Cryptology ePrint Archive.

Public Datasets

- ASCAD[1]: masking and artificially introduced random jitter
- AES_HD[2]: unprotected AES hardware implementation on FPGA
- AES_RD[3]: random delay countermeasure

[1] Benadjila, R., Prouff, E., Strullu, R., Cagli, E., & Dumas, C. (2020). Deep learning for side-channel analysis and introduction to ASCAD database. Journal of Cryptographic Engineering, 10(2), 163-188.

[2] `https://github.com/AESHD/AES_HD_Dataset`

[3] `https://github.com/ikizhvatov/randomdelays-traces`

Other Aspects of SCA

- Identify points of interest[1]
- Leakage assessment[2]
  - whether any input-dependent information can be detected in side-channel measurements of the device under test

[1] Lu, X., Zhang, C., Cao, P., Gu, D., & Lu, H. (2021). Pay attention to raw traces: A deep learning architecture for end-to-end profiling attacks. IACR Transactions on Cryptographic Hardware and Embedded Systems, 235-274.

[2] Moos, T., Wegener, F., & Moradi, A. (2021). DL-LA: Deep Learning Leakage Assessment: A modern roadmap for SCA evaluations. IACR Transactions on Cryptographic Hardware and Embedded Systems, 552-598.