

SCARE of Secret Ciphers with SPN Structures

Matthieu Rivain

Joint work with Thomas Roche (ANSSI)

ASIACRYPT 2013 – December 3rd



Outline

- 1 ■ Introduction
- 2 ■ Substitution-Permutation Networks
- 3 ■ Basic SCARE of Classical SPN Structures
- 4 ■ SCARE in the Presence of Noisy Leakage
- 5 ■ Attack Experiments

Outline

- 1 ■ Introduction
- 2 ■ Substitution-Permutation Networks
- 3 ■ Basic SCARE of Classical SPN Structures
- 4 ■ SCARE in the Presence of Noisy Leakage
- 5 ■ Attack Experiments

Introduction

SCARE: Side-Channel Analysis for Reverse Engineering

- private code recovery
- secret crypto design recovery

Introduction

SCARE: Side-Channel Analysis for Reverse Engineering

- private code recovery
- secret crypto design recovery \Leftarrow This paper

Introduction

SCARE: Side-Channel Analysis for Reverse Engineering

- private code recovery
- secret crypto design recovery \Leftarrow This paper
- usual in mobile SIM / pay-TV cards

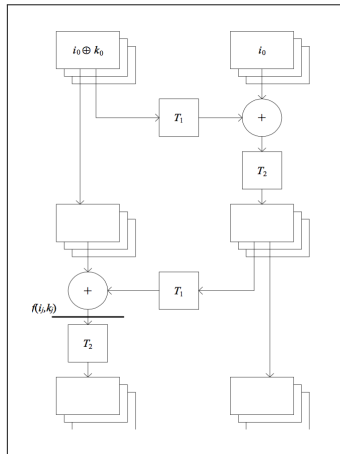
Previous works

[Novak. ACNS 2003]

- secret instance of the GSM A3/A8 algorithm
- side-channel assumption: detection of colliding s-boxes
- recovery of one secret s-box

[Clavier. ePrint 2004/ICISS 2007]

- recovery of the two s-boxes and the secret key



Limitations

- **Target:** specific cipher structure
- **Assumption:** idealized leakage model
 - ⇒ perfect collision detection

Our work

- Consider a generic class of ciphers:
Substitution-Permutation Networks (SPN)
- Relax the idealized leakage assumption
 - ▶ consider noisy leakages
 - ▶ experiments in a practical leakage model

Further works

[Daudigny et al. ACNS 2005] (DES)

[Réal et al. CARDIS 2008] (hardware Feistel)

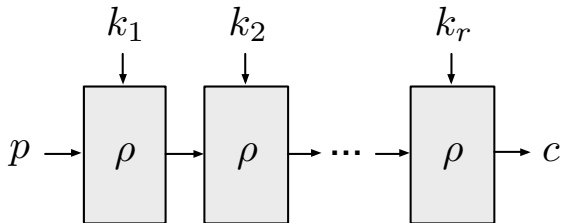
[Guilley et al. LATINCRYPT 2010] (stream ciphers)

[Clavier et al. INDOCRYPT 2013] (modified AES)

Outline

- 1 ■ Introduction
- 2 ■ Substitution-Permutation Networks
- 3 ■ Basic SCARE of Classical SPN Structures
- 4 ■ SCARE in the Presence of Noisy Leakage
- 5 ■ Attack Experiments

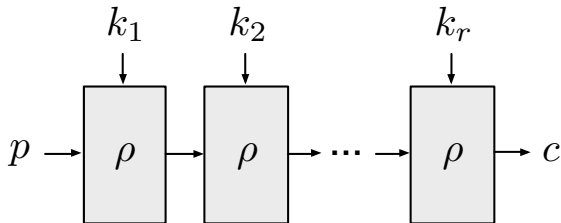
Substitution-Permutation Networks



We consider two types of round functions:

- Classical SPN structures
- Feistel structures

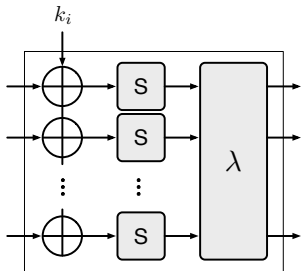
Substitution-Permutation Networks



We consider two types of round functions:

- Classical SPN structures \Leftarrow This talk
- Feistel structures

Classical SPN Structure



- State: $n \times m$ bits
- n s-box computations
- m -bits s-box inputs

$$\lambda : \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \mapsto \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \quad \text{with } a_{i,j} \in \mathbb{F}_{2^m}$$

Outline

- 1 ■ Introduction
- 2 ■ Substitution-Permutation Networks
- 3 ■ Basic SCARE of Classical SPN Structures**
- 4 ■ SCARE in the Presence of Noisy Leakage
- 5 ■ Attack Experiments

Attacker Model

Basic assumption:

Colliding s-box computations can be detected from the side-channel leakage.

Specifically, we assume that the attacker is able to

- (i) identify the s-box computations in the side-channel leakage trace and extract the leakage corresponding to each s-box computation,
- (ii) decide whether two s-box computations $y_1 \leftarrow S(x_1)$ and $y_2 \leftarrow S(x_2)$ are such that $x_1 = x_2$ or not from their respective leakages.

Equivalent Representations

One cipher has several representations

1. Change the s-box: $S'(x) = S(x \oplus \delta)$
and the round keys: $k'_i = (k_{i,1} \oplus \delta, k_{i,2} \oplus \delta, \dots, k_{i,n} \oplus \delta)$

Equivalent Representations

One cipher has several representations

1. Change the s-box: $S'(x) = S(x \oplus \delta)$
and the round keys: $k'_i = (k_{i,1} \oplus \delta, k_{i,2} \oplus \delta, \dots, k_{i,n} \oplus \delta)$
2. Change the s-box: $S'(x) = \alpha \cdot S(x)$
and the matrix coefficients: $a'_{i,j} = \frac{a_{i,j}}{\alpha}$

Equivalent Representations

One cipher has several representations

1. Change the s-box: $S'(x) = S(x \oplus \delta)$
and the round keys: $k'_i = (k_{i,1} \oplus \delta, k_{i,2} \oplus \delta, \dots, k_{i,n} \oplus \delta)$
2. Change the s-box: $S'(x) = \alpha \cdot S(x)$
and the matrix coefficients: $a'_{i,j} = \frac{a_{i,j}}{\alpha}$

The attack can recover the cipher up to equivalent representations

Equivalent Representations

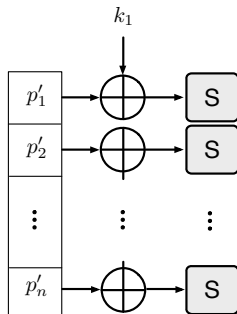
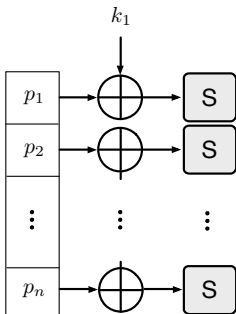
One cipher has several representations

1. Change the s-box: $S'(x) = S(x \oplus \delta)$
and the round keys: $k'_i = (k_{i,1} \oplus \delta, k_{i,2} \oplus \delta, \dots, k_{i,n} \oplus \delta)$
2. Change the s-box: $S'(x) = \alpha \cdot S(x)$
and the matrix coefficients: $a'_{i,j} = \frac{a_{i,j}}{\alpha}$

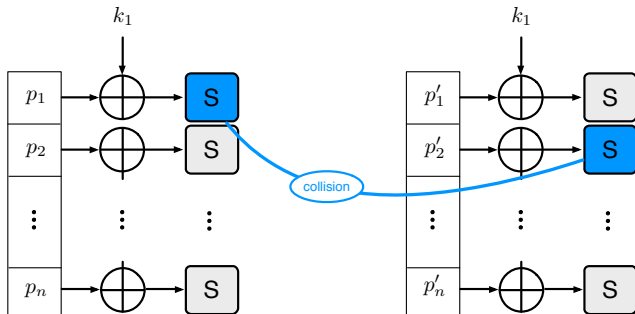
The attack can recover the cipher up to equivalent representations

We fix a representation by setting $k_{1,1} = 0$ and $a_{1,1} = 1$

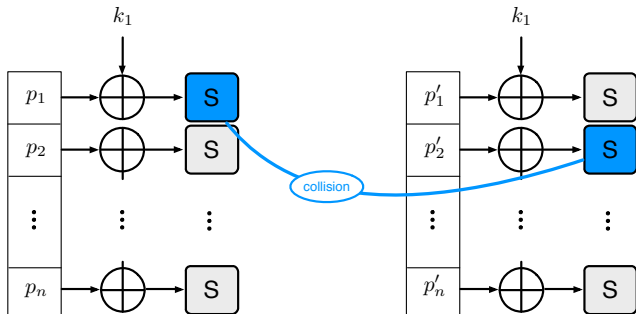
Stage 1: Recovering k_1



Stage 1: Recovering k_1

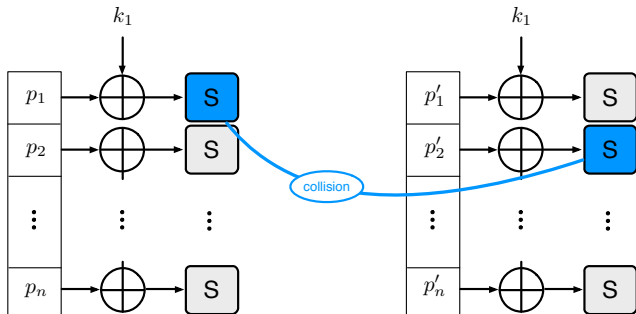


Stage 1: Recovering k_1



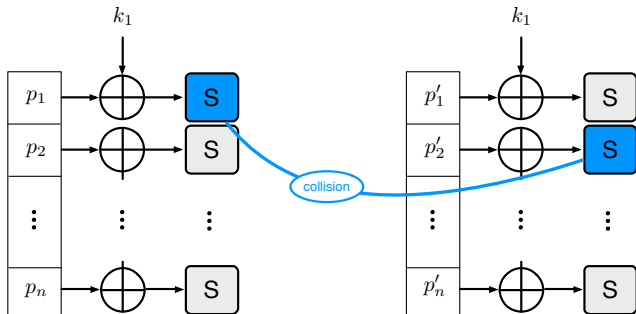
$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2}$$

Stage 1: Recovering k_1



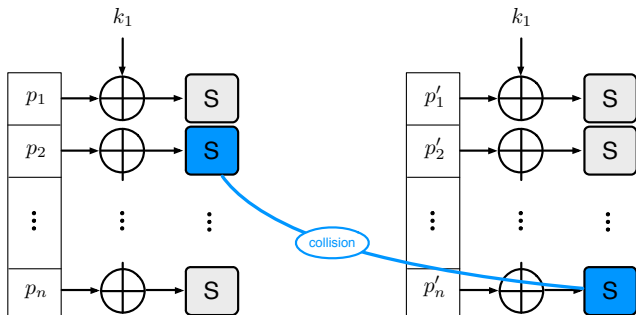
$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2}$$

Stage 1: Recovering k_1



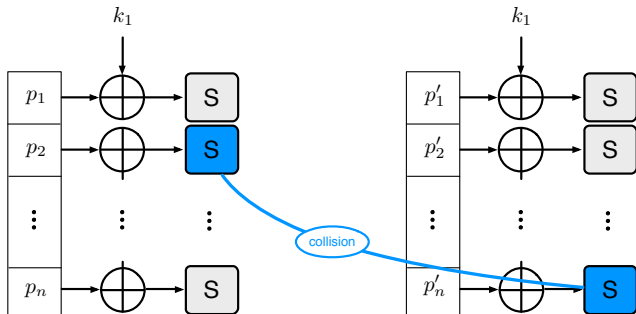
$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2} \Rightarrow k_{1,2} = p_1 \oplus p'_2 \oplus k_{1,1}$$

Stage 1: Recovering k_1



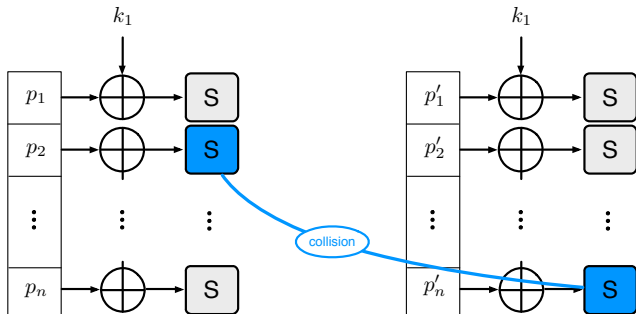
$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2} \Rightarrow k_{1,2} = p_1 \oplus p'_2 \oplus k_{1,1}$$

Stage 1: Recovering k_1



$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2} \Rightarrow k_{1,2} = p_1 \oplus p'_2 \oplus k_{1,1}$$
$$p_2 \oplus k_{1,2} = p'_n \oplus k_{1,n}$$

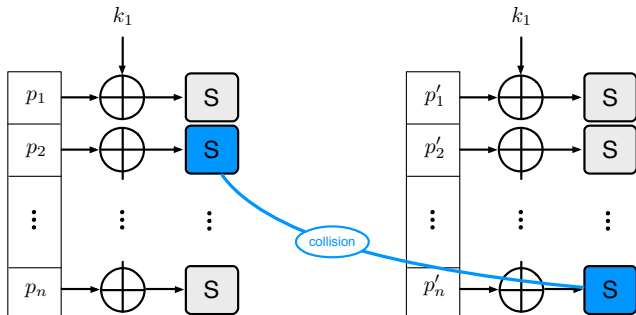
Stage 1: Recovering k_1



$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2} \Rightarrow k_{1,2} = p_1 \oplus p'_2 \oplus k_{1,1}$$

$$p_2 \oplus k_{1,2} = p'_n \oplus k_{1,n}$$

Stage 1: Recovering k_1

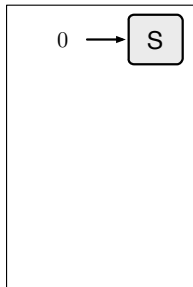


$$p_1 \oplus k_{1,1} = p'_2 \oplus k_{1,2} \Rightarrow k_{1,2} = p_1 \oplus p'_2 \oplus k_{1,1}$$

$$p_2 \oplus k_{1,2} = p'_n \oplus k_{1,n} \Rightarrow k_{1,n} = p_1 \oplus p'_n \oplus k_{1,2}$$

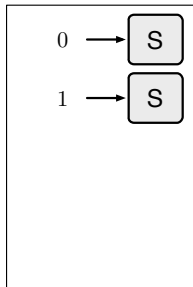
and so on ...

Stage 2: Recovering λ , S and k_2



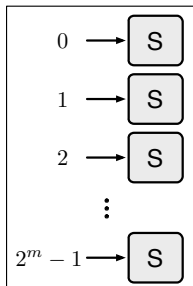
leakage basis

Stage 2: Recovering λ , S and k_2



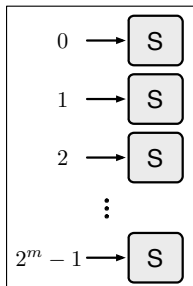
leakage basis

Stage 2: Recovering λ , S and k_2

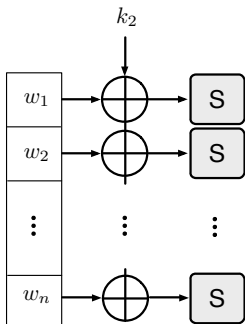


leakage basis

Stage 2: Recovering λ , S and k_2

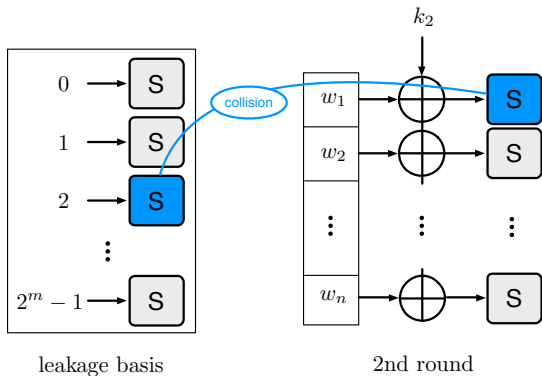


leakage basis

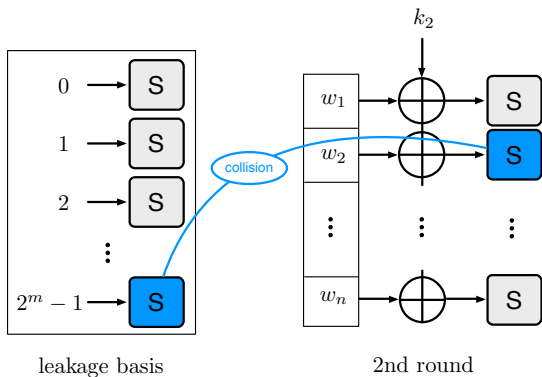


2nd round

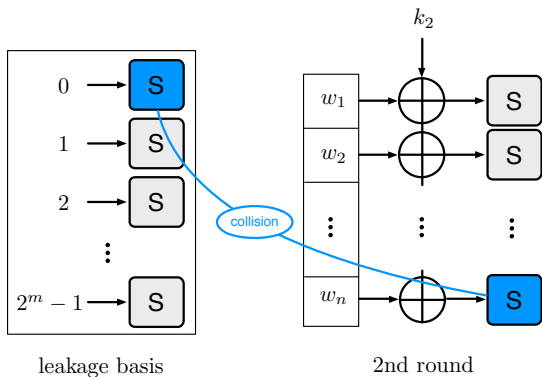
Stage 2: Recovering λ , S and k_2



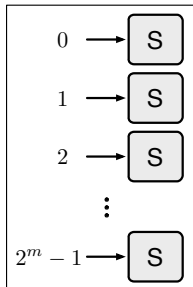
Stage 2: Recovering λ , S and k_2



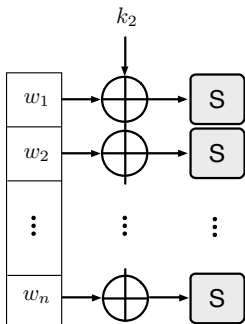
Stage 2: Recovering λ , S and k_2



Stage 2: Recovering λ , S and k_2

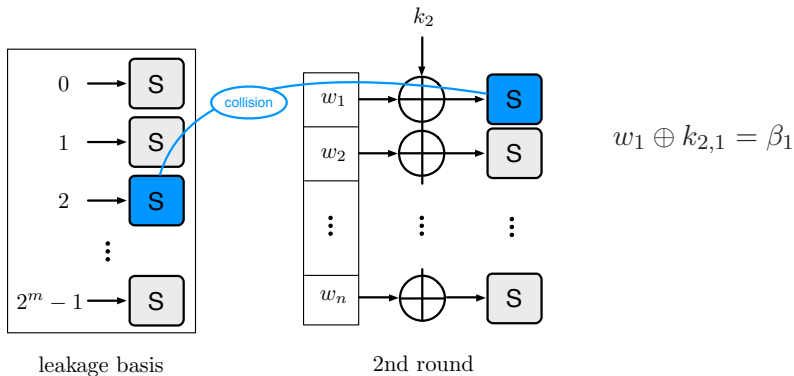


leakage basis

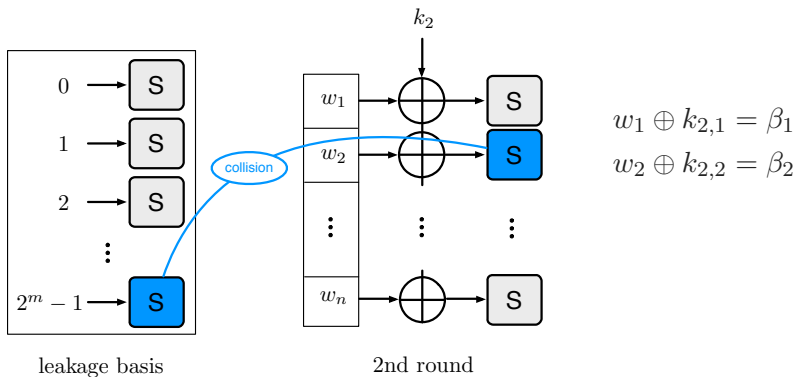


2nd round

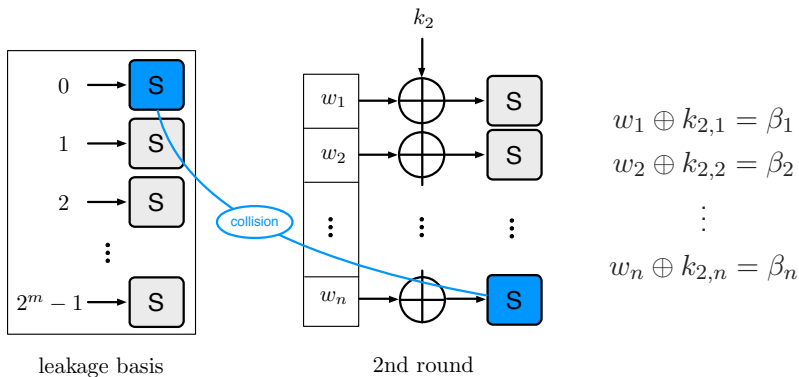
Stage 2: Recovering λ , S and k_2



Stage 2: Recovering λ , S and k_2



Stage 2: Recovering λ , S and k_2



Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} S(p_1 \oplus k_{1,1}) \\ S(p_2 \oplus k_{1,2}) \\ \vdots \\ S(p_n \oplus k_{1,n}) \end{pmatrix}$$

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} S(p_1 \oplus k_{1,1}) \\ S(p_2 \oplus k_{1,2}) \\ \vdots \\ S(p_n \oplus k_{1,n}) \end{pmatrix}$$

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} S(j_1) \\ S(j_2) \\ \vdots \\ S(j_n) \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$ and $x_j = S(j)$.

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$ and $x_j = S(j)$.

We get equations of the form:

$$k_{2,i} \oplus \beta_i = a_{i,1} \cdot x_{j_1} \oplus a_{i,2} \cdot x_{j_2} \oplus \cdots \oplus a_{i,n} \cdot x_{j_n}$$

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$ and $x_j = S(j)$.

We get **quadratic** equations of the form:

$$k_{2,i} \oplus \beta_i = a_{i,1} \cdot x_{j_1} \oplus a_{i,2} \cdot x_{j_2} \oplus \cdots \oplus a_{i,n} \cdot x_{j_n}$$

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$ and $x_j = S(j)$.

We get **quadratic** equations of the form:

$$k_{2,i} \oplus \beta_i = a_{i,1} \cdot x_{j_1} \oplus a_{i,2} \cdot x_{j_2} \oplus \cdots \oplus a_{i,n} \cdot x_{j_n}$$

Using linearization, we get a system with $2^m \cdot n^2 + n$ unknowns

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$ and $x_j = S(j)$.

We get **quadratic** equations of the form:

$$k_{2,i} \oplus \beta_i = a_{i,1} \cdot x_{j_1} \oplus a_{i,2} \cdot x_{j_2} \oplus \cdots \oplus a_{i,n} \cdot x_{j_n}$$

Using linearization, we get a system with $2^m \cdot n^2 + n$ unknowns
 \Rightarrow solvable with $2^m \cdot n + 1$ encryptions

Stage 2: Recovering λ , S and k_2

We have

$$\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix} \oplus \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \cdot \begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}$$

where $j_t = p_t \oplus k_{1,t}$ and $x_j = S(j)$.

We get **quadratic** equations of the form:

$$k_{2,i} \oplus \beta_i = a_{i,1} \cdot x_{j_1} \oplus a_{i,2} \cdot x_{j_2} \oplus \cdots \oplus a_{i,n} \cdot x_{j_n}$$

Using linearization, we get a system with $2^m \cdot n^2 + n$ unknowns

⇒ solvable with $2^m \cdot n + 1$ encryptions

⇒ solvable with 4097 encryptions for $m = 8$, $n = 16$

A better way

$$\underbrace{\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}}_A \cdot \underbrace{\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}}_{\vec{x}} = \underbrace{\begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix}}_{\vec{k}_2} \oplus \underbrace{\begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}}_{\vec{\beta}}$$

$$A \cdot \vec{x} = \vec{k}_2 \oplus \vec{\beta}$$

A better way

$$\underbrace{\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}}_A \cdot \underbrace{\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}}_{\vec{x}} = \underbrace{\begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix}}_{\vec{k}_2} \oplus \underbrace{\begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}}_{\vec{\beta}}$$

$$A \cdot \vec{x} = \vec{k}_2 \oplus \vec{\beta}$$

$$\vec{x} = A^{-1} \cdot \vec{k}_2 \oplus A^{-1} \cdot \vec{\beta}$$

A better way

$$\underbrace{\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix}}_A \cdot \underbrace{\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix}}_{\vec{x}} = \underbrace{\begin{pmatrix} k_{2,1} \\ k_{2,2} \\ \vdots \\ k_{2,n} \end{pmatrix}}_{\vec{k}_2} \oplus \underbrace{\begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}}_{\vec{\beta}}$$

$$A \cdot \vec{x} = \vec{k}_2 \oplus \vec{\beta}$$

$$\vec{x} = \underbrace{A^{-1} \cdot \vec{k}_2}_{\vec{k}'_2} \oplus A^{-1} \cdot \vec{\beta}$$

A better way

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix} = \begin{pmatrix} k'_{2,1} \\ k'_{2,2} \\ \vdots \\ k'_{2,n} \end{pmatrix} \oplus \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} \\ a'_{2,1} & a'_{2,2} & \cdots & a'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{n,1} & a'_{n,2} & \cdots & a'_{n,n} \end{pmatrix} \cdot \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

A better way

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix} = \begin{pmatrix} k'_{2,1} \\ k'_{2,2} \\ \vdots \\ k'_{2,n} \end{pmatrix} \oplus \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} \\ a'_{2,1} & a'_{2,2} & \cdots & a'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{n,1} & a'_{n,2} & \cdots & a'_{n,n} \end{pmatrix} \cdot \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

We get equations of the form:

$$x_{j_i} = k'_{2,i} \oplus a'_{i,1} \cdot \beta_1 \oplus a'_{i,2} \cdot \beta_2 \oplus \cdots \oplus a'_{i,n} \cdot \beta_n$$

A better way

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix} = \begin{pmatrix} k'_{2,1} \\ k'_{2,2} \\ \vdots \\ k'_{2,n} \end{pmatrix} \oplus \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} \\ a'_{2,1} & a'_{2,2} & \cdots & a'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{n,1} & a'_{n,2} & \cdots & a'_{n,n} \end{pmatrix} \cdot \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

We get **linear** equations of the form:

$$x_{j_i} = k'_{2,i} \oplus a'_{i,1} \cdot \beta_1 \oplus a'_{i,2} \cdot \beta_2 \oplus \cdots \oplus a'_{i,n} \cdot \beta_n$$

A better way

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix} = \begin{pmatrix} k'_{2,1} \\ k'_{2,2} \\ \vdots \\ k'_{2,n} \end{pmatrix} \oplus \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} \\ a'_{2,1} & a'_{2,2} & \cdots & a'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{n,1} & a'_{n,2} & \cdots & a'_{n,n} \end{pmatrix} \cdot \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

We get **linear** equations of the form:

$$x_{j_i} = k'_{2,i} \oplus a'_{i,1} \cdot \beta_1 \oplus a'_{i,2} \cdot \beta_2 \oplus \cdots \oplus a'_{i,n} \cdot \beta_n$$

We get a linear system with $2^m + n^2 + n$ unknowns

A better way

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix} = \begin{pmatrix} k'_{2,1} \\ k'_{2,2} \\ \vdots \\ k'_{2,n} \end{pmatrix} \oplus \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} \\ a'_{2,1} & a'_{2,2} & \cdots & a'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{n,1} & a'_{n,2} & \cdots & a'_{n,n} \end{pmatrix} \cdot \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

We get **linear** equations of the form:

$$x_{j_i} = k'_{2,i} \oplus a'_{i,1} \cdot \beta_1 \oplus a'_{i,2} \cdot \beta_2 \oplus \cdots \oplus a'_{i,n} \cdot \beta_n$$

We get a linear system with $2^m + n^2 + n$ unknowns
 \Rightarrow solvable with $2^m/n + n + 1$ encryptions

A better way

$$\begin{pmatrix} x_{j_1} \\ x_{j_2} \\ \vdots \\ x_{j_n} \end{pmatrix} = \begin{pmatrix} k'_{2,1} \\ k'_{2,2} \\ \vdots \\ k'_{2,n} \end{pmatrix} \oplus \begin{pmatrix} a'_{1,1} & a'_{1,2} & \cdots & a'_{1,n} \\ a'_{2,1} & a'_{2,2} & \cdots & a'_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a'_{n,1} & a'_{n,2} & \cdots & a'_{n,n} \end{pmatrix} \cdot \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

We get **linear** equations of the form:

$$x_{j_i} = k'_{2,i} \oplus a'_{i,1} \cdot \beta_1 \oplus a'_{i,2} \cdot \beta_2 \oplus \cdots \oplus a'_{i,n} \cdot \beta_n$$

We get a linear system with $2^m + n^2 + n$ unknowns

⇒ solvable with $2^m/n + n + 1$ encryptions

⇒ solvable with 33 encryptions for $m = 8$, $n = 16$

And finally

Stage 3: recovering k_3, k_4, \dots, k_r

⇒ similar as stage 1

Outline

- 1 ■ Introduction
- 2 ■ Substitution-Permutation Networks
- 3 ■ Basic SCARE of Classical SPN Structures
- 4 ■ SCARE in the Presence of Noisy Leakage
- 5 ■ Attack Experiments

SCARE in the Presence of Noisy Leakage

Gaussian noise assumption:



SCARE in the Presence of Noisy Leakage

Gaussian noise assumption:



Stage 1 (Recovering k_1): usual scenario of *linear collision attacks*
[Gérard-Standaert. CHES 2012]

SCARE in the Presence of Noisy Leakage

Gaussian noise assumption:



Stage 1 (Recovering k_1): usual scenario of *linear collision attacks*
[Gérard-Standaert. CHES 2012]

Stage 2 (Recovering λ , S and k_2) composed of 4 steps:

- building leakage templates
- collecting equations
- solving a subsystem (Stage 2.1)
- recovering remaining unknowns (Stage 2.2)

Building leakage templates

Construct a template basis:

$$\mathcal{B} = \{(\hat{m}_\beta, \hat{\Sigma}_\beta)_\beta \mid \beta \in \mathbb{F}_{2^m}\},$$

with

- \hat{m}_β : sample mean
- $\hat{\Sigma}_\beta$: sample covariance matrix

Collecting equations

We collect several groups of equations $\vec{x} = \vec{k}'_2 \oplus A^{-1} \cdot \vec{\beta}$

Noisy leakage \Rightarrow we cannot determine $\vec{\beta}$ with a 100% confidence

- ▷ we use averaging (each encryption N times)
- ▷ maximum likelihood approach based on \mathcal{B}

Problem: we cannot tolerate one single wrong β_i

Success probability:

- for one s-box: p
- for one encryption: p^n
- for the attack: $(p^n)^t$
 - ▶ where t is the number of required encryptions

Solving a subsystem

Increasing the success probability:

- reduce the number t
- subsystem only involving x_0, x_1, \dots, x_{s-1}
- chosen plaintext attack

Obtained system:

- $n^2 + n + s - 2$ unknowns
- taking $s \leq n + 2$
 - ▶ we get at most $n^2 + 2n$ unknowns
 - ▶ we need $t = n + 2$
- e.g. $t = 18$ instead of $t = 33$ for $n = 16$ and $m = 8$

Recovering remaining unknowns

Maximum likelihood approach for

- remaining s-box output $x_s, x_{s+1}, \dots, x_{2^m-1}$ (Stage 2.2)
- remaining round keys k_3, k_4, \dots, k_r (Stage 3)

Outline

- 1 ■ Introduction
- 2 ■ Substitution-Permutation Networks
- 3 ■ Basic SCARE of Classical SPN Structures
- 4 ■ SCARE in the Presence of Noisy Leakage
- 5 ■ Attack Experiments

Attack Experiments

Attack simulations using a **practical leakage model**

- s-box computation on an AVR chip (ATMega 32A, 8-bit)
- profiled electromagnetic leakage
- Gaussian noise assumption
- 3 leakage points depending on the s-box input
- 3 leakage points depending on the s-box output

Attack Experiments

Two different settings:

- **(128,8)-setting:**

- ▶ 128-bit message block
- ▶ 8-bit s-box ($m = 8 \Rightarrow n = 16$)
- ▶ e.g. AES block cipher

- **(64,4)-setting:**

- ▶ 64-bit message block
- ▶ 4-bit s-box ($m = 4 \Rightarrow n = 16$)
- ▶ e.g. LED and PRESENT lightweight block ciphers

Attack results

Stage 1: 100% success rate with

- a few hundred traces for the (64,4)-setting
- a few thousand traces for the (128,8)-setting

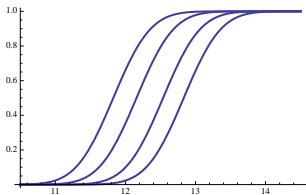
Attack results

Stage 1: 100% success rate with

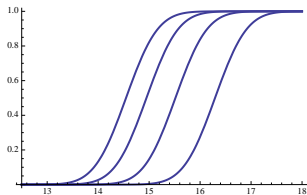
- a few hundred traces for the (64,4)-setting
- a few thousand traces for the (128,8)-setting

Stage 2.1: bottleneck of the attack

SR w.r.t. #encryptions (for 1, 2, 2^8 , 2^{32} system solving trials)



(64,4)-setting



(128,8)-setting

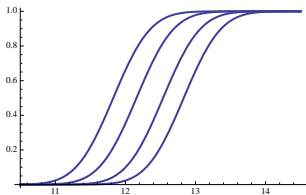
Attack results

Stage 1: 100% success rate with

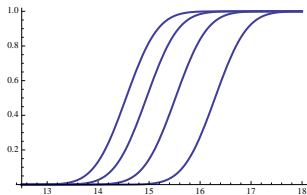
- a few hundred traces for the (64,4)-setting
- a few thousand traces for the (128,8)-setting

Stage 2.1: bottleneck of the attack

SR w.r.t. #encryptions (for 1, 2, 2^8 , 2^{32} system solving trials)



(64,4)-setting



(128,8)-setting

Stages 2.2, 3: a few dozens/hundreds of traces.

The end

Questions?

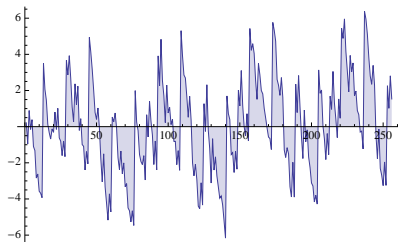
The end

Questions?

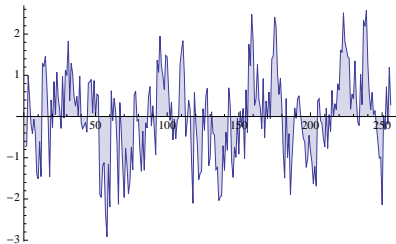
The end

Questions?

Profiled leakage parameters

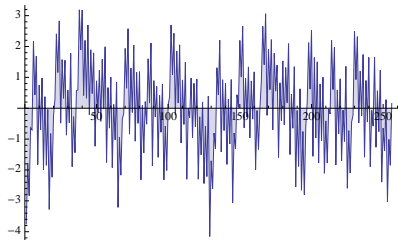


1st point mean w.r.t input

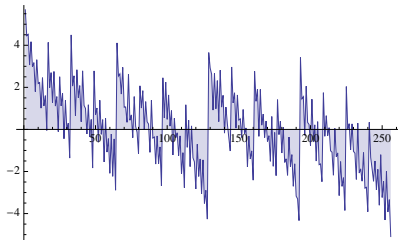


2nd point mean w.r.t input

Profiled leakage parameters

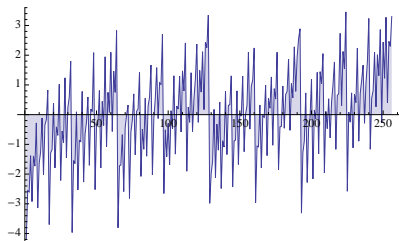


3rd point mean w.r.t input

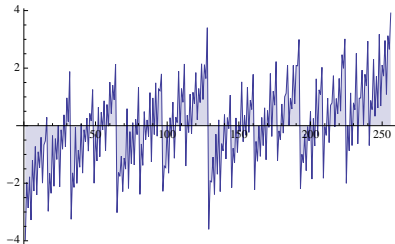


4th point mean w.r.t output

Profiled leakage parameters



5th point mean w.r.t output



6th point mean w.r.t output

Profiled leakage parameters

$$\Sigma = \begin{pmatrix} \mathbf{36.7} & \mathbf{-13.7} & -1.8 & 2.9 & -2.2 & -0.7 \\ \mathbf{-13.7} & \mathbf{30.7} & 0.6 & 0.7 & -0.5 & -0.1 \\ -1.8 & 0.6 & \mathbf{27.5} & -0.9 & 0.7 & 0.4 \\ 2.9 & 0.7 & -0.9 & \mathbf{38.7} & \mathbf{-27.0} & -5.4 \\ -2.2 & -0.5 & 0.7 & \mathbf{-27.0} & \mathbf{37.2} & 3.9 \\ -0.7 & -0.1 & 0.4 & -5.4 & 3.9 & \mathbf{26.2} \end{pmatrix}$$