Side Channel Attacks against Block Ciphers Implementations and Countermeasures

Emmanuel Prouff
e.prouff@ssi.gouv.fr

ANSSI

Tutorial CHES – August 2013

Part I

Introduction

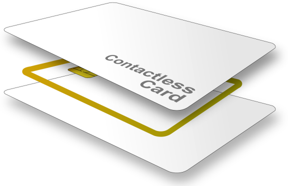
Outline

Plan

Smart Card

- A Smart Card is a circuit embedded on a plastic support. It moreover has communication means, storage capacities and computation capacities.
- The physical characteristics of a smart card are standardized.
- The smart card enables the secure storage of sensitive data: a part of its memory is indeed protected in both writing and reading modes.

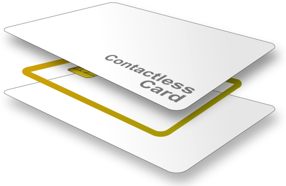
A Smart Card?



- A plastic Support
- Storage and computation means
  - Micro-controller (ST, Atmel, NXP, Samsung, Infineon, etc.)
- Communication means
  - Connectors
  - Antenna

- Main Goal: embed private data and manipulate them in a secure way.

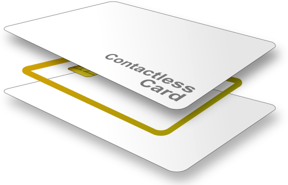
A Smart Card?



- A plastic Support
- Storage and computation means
  - Micro-controller (ST, Atmel, NXP, Samsung, Infineon, etc.)
- Communication means
  - Connectors
  - Antenna

- Main Goal: embed private data and manipulate them in a secure way.

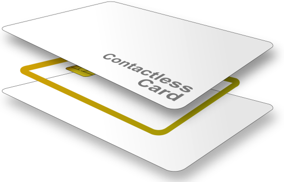
A Smart Card?



- A plastic Support
- Storage and computation means
  - Micro-controller (ST, Atmel, NXP, Samsung, Infineon, etc.)
- Communication means
  - Connectors
  - Antenna

- Main Goal: embed private data and manipulate them in a secure way.

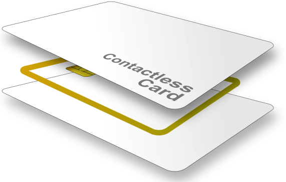
A Smart Card?



- A plastic Support
- Storage and computation means
  - Micro-controller (ST, Atmel, NXP, Samsung, Infineon, etc.)
- Communication means
  - Connectors
  - Antenna

- Main Goal: embed private data and manipulate them in a secure way.

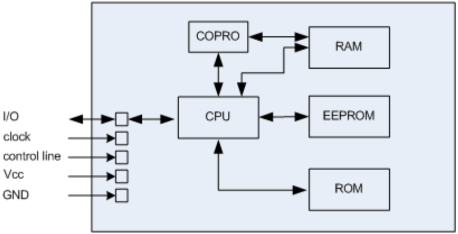
A Smart Card?



- A plastic Support
- Storage and computation means
  - Micro-controller (ST, Atmel, NXP, Samsung, Infineon, etc.)
- Communication means
  - Connectors
  - Antenna

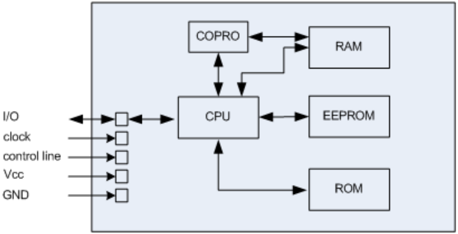
- Main Goal: embed private data and manipulate them in a secure way.

A Smart Card?



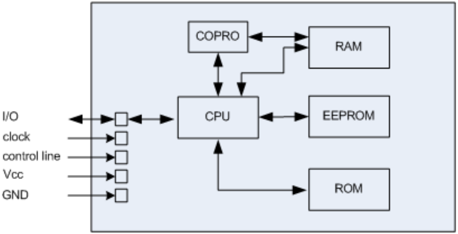
- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic*) *calculations.*

A Smart Card?



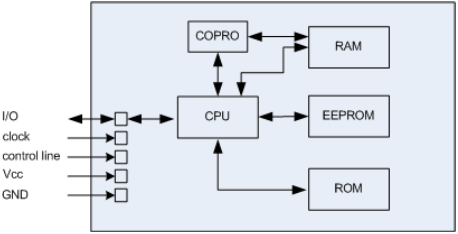
- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic*) calculations.

A Smart Card?



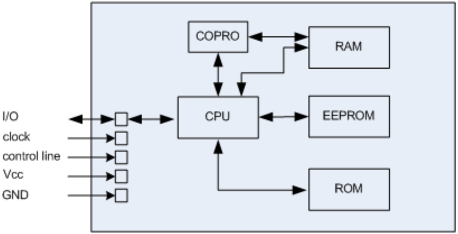
- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic*) *calculations.*

A Smart Card?



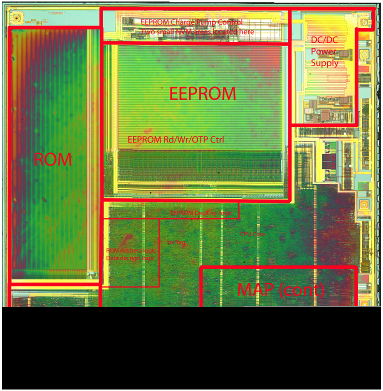
- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic) calculations.*

A Smart Card?



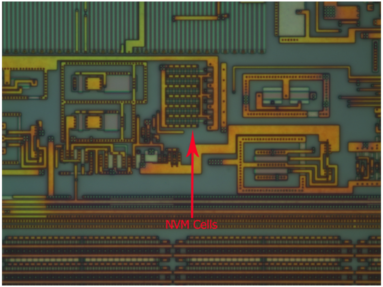
- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic) calculations.*

A Smart Card?



- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic) calculations.*

A Smart Card?



- ROM contains the smart card OS.
- RAM is dedicated to the storage of local and volatile variables during the processings.
- EEPROM contains code and some data.
- Co-processor is dedicated to particular cryptographic (*e.g. arithmetic) calculations.*

Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

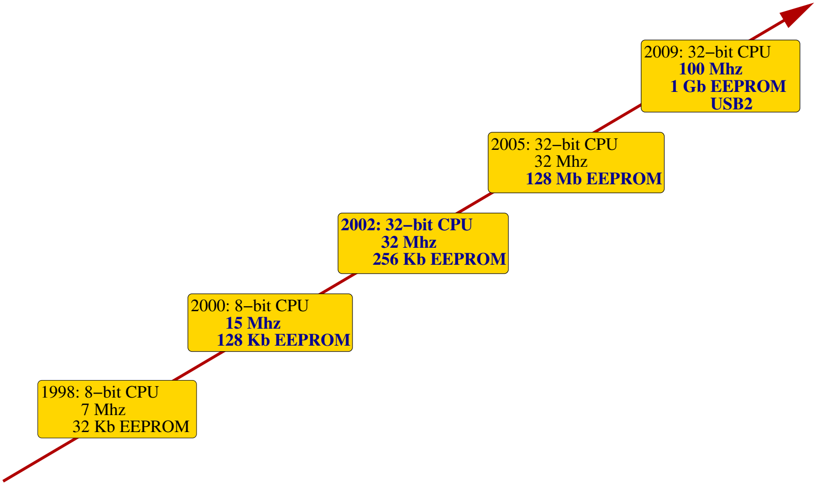
Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

Smart Card Evolution

- The first smart cards (Bull and Motorola) only had 36 bytes of RAM and 1600 bytes of ROM.
- Today, a smart card has;
  - between 16 and 512 Kbytes of ROM,
  - between 1 and 32 Kbytes of RAM,
  - a processor running at 100 MHz.
- ... it can;
  - embed several mega-bytes of Flash memory,
  - communicate in USB2.0,
  - embed a web server.

Smart Card Evolution



Plan

1. Smart Cards
2. Embedded Cryptogaphy

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
    - A standard exists: it will certainly be chosen!
    - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO).*
    - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Which Algorithm for which Security Issue?

- Client (Bank, Operator, Government, etc.) has to deal with security issues: securing transactions, protecting citizen anonymity, limiting access to services, etc.
- In more than 95% of the cases it asks its internal security experts to find a solution.
  - A standard exists: it will certainly be chosen!
  - No satisfying standard exists: a new standardization process is initiated (*e.g. ETSI 3GPP, ISO, ICAO)*.
  - *No satisfying standard exists: a solution is designed by the internal experts (proprietary algorithms).*
- Sometimes (in less than 5% of the cases) the Client asks Industrial experts to find a solution.
- As a consequence, the Smart Card Industry essentially implements standards (and even a very few of them!).

Cryptography in Smart Cards

Smart Cards implement a wide range of cryptographic algorithms:

- Block Ciphers: (Triple-)DES, AES, proprietary algorithms
- Hash functions: SHA family
- Data authentication: CBC-MAC, HMAC
- Symmetric key cryptography: RSA (OAEP, PKCS1-v1.5)
- Signature : RSA (PKCS1-v1.5, PSS), DSA, ECDSA
- Key exchange protocols: Diffie-Hellman, Diffie-Hellman on elliptic curves

Part II

Passive Side Channel Attacks

Outline

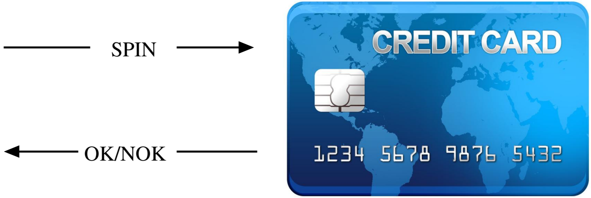
Plan

Exploitation : Simple Attacks (SPA)

SPA refers to attacks where the adversary focus on a single execution of an implementation (with possibility to average the observation for fixed inputs).
In some cases, this gives the adversary information about the manipulated secrets.

- The information leakage must be important.
- The secret must have a simple relationship with the leakage.

Example of SPA; PIN verification



Algo PIN comparison

INPUT(S) : SPIN, PIN
OUTPUT(S) : `ok/nok`

1: **for** $i = 0$ to 3 **do**
2: **if** SPIN[i] $\neq$ PIN[i] **then**
3: **return** `nok`
4: **return** `ok`

The observation of execution timing enables to retrieve PIN with 4×10 tries instead of $10^4 = 10\,000$.

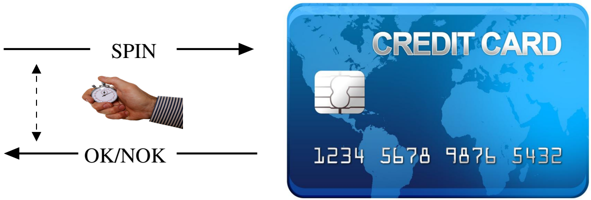
Example of SPA; PIN verification

Algo PIN comparison

```
INPUT(S) : SPIN, PIN
OUTPUT(S) : ok/nok
for i = 0 to 3 do
    if SPIN[i] ≠ PIN[i] then
        return nok
return ok
```

The observation of execution timing enables to retrieve PIN with 4×10 tries instead of $10^4 = 10\,000$.

Example of SPA; PIN verification



Algo PIN comparison

INPUT(S) : SPIN, PIN
OUTPUT(S) : `ok/nok`

```
for i = 0 to 3 do
    if SPIN[i] ≠ PIN[i] then
        return nok

return ok
```

The observation of execution timing enables to retrieve PIN with 4×10 tries instead of $10^4 = 10\,000$.

Plan

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

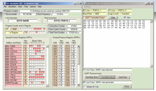
Advanced Side Channel Attacks (DPA like attacks)

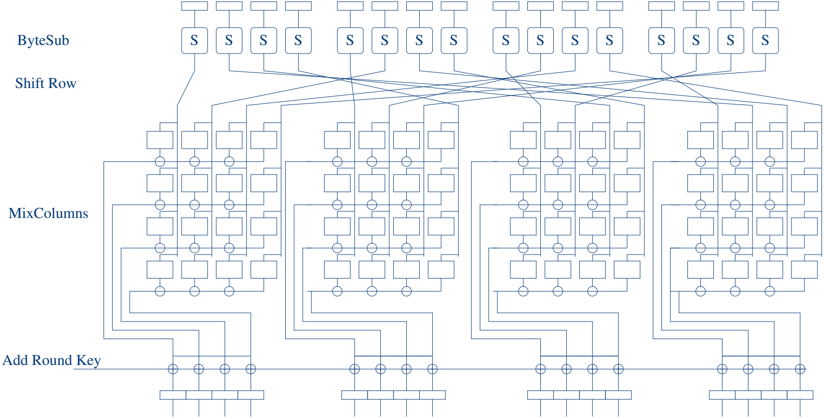
Introduction

- Advanced Side Channel Attacks can extract information from observations in contexts where SPA fails.
- They involve statistical tools (simple – difference of means tests – or sophisticated – mutual information processing –).
- They need several (between 10 and more than 10^6) traces such that:
  - the secret is constant,
  - the inputs are different and [optional] known.
  - [optional] some knowledge about the device architecture, the implementation or the noise characteristics.
- They follow a divide-and-conquer approach: the secret is rebuild piece by piece, where each piece is deduced from the behavior of an intermediate result. The size of the piece usually depends on the architecture size (*e.g.* 8, 16 or 32 bits).

Advanced Side Channel Attacks (DPA like attacks)

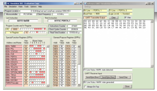
AES Round - 8-bit Software Implementation

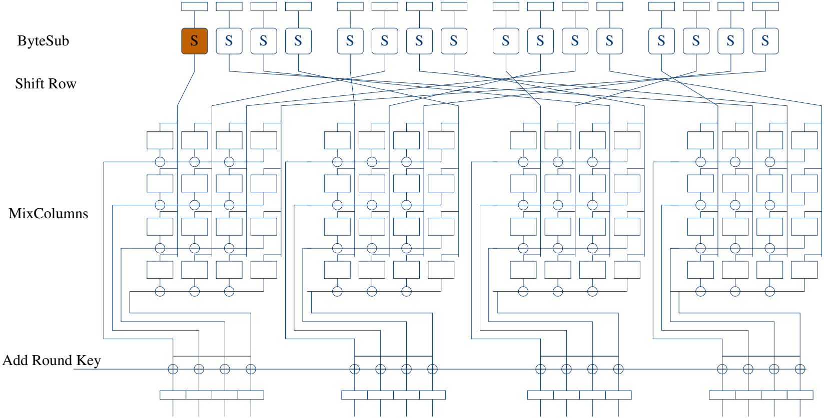




Advanced Side Channel Attacks (DPA like attacks)

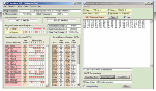
AES Round - 8-bit Software Implementation

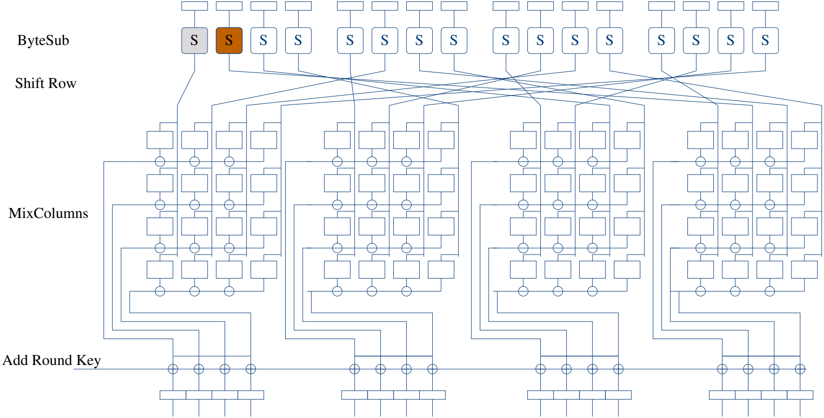




Advanced Side Channel Attacks (DPA like attacks)

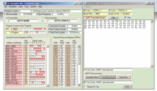
AES Round - 8-bit Software Implementation

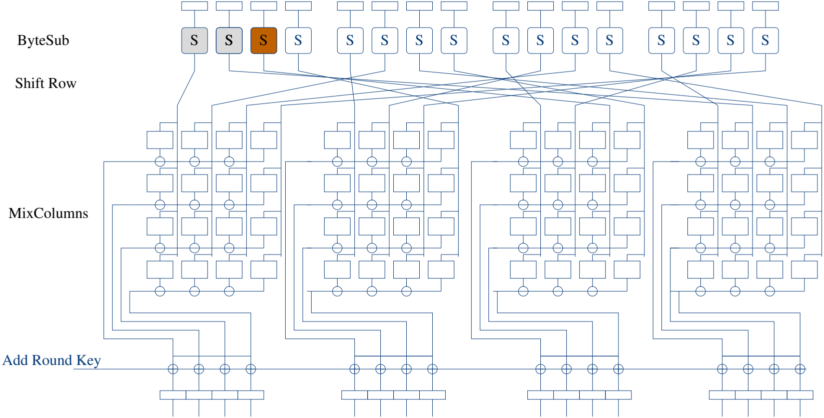




Advanced Side Channel Attacks (DPA like attacks)

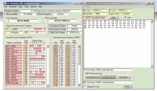
AES Round - 8-bit Software Implementation

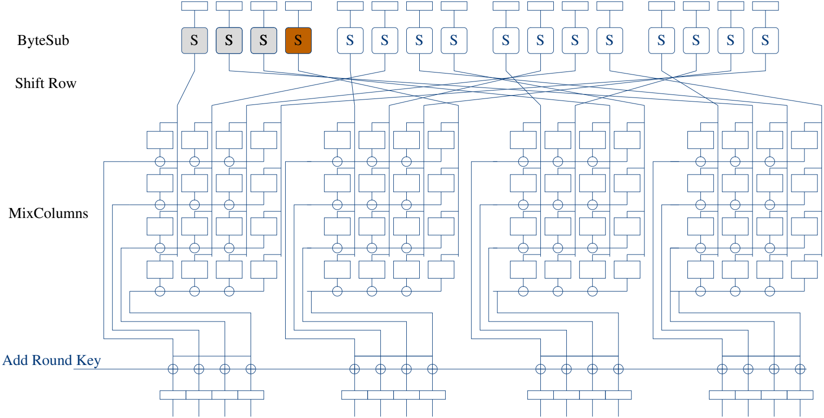




Advanced Side Channel Attacks (DPA like attacks)

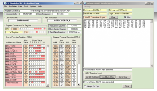
AES Round - 8-bit Software Implementation

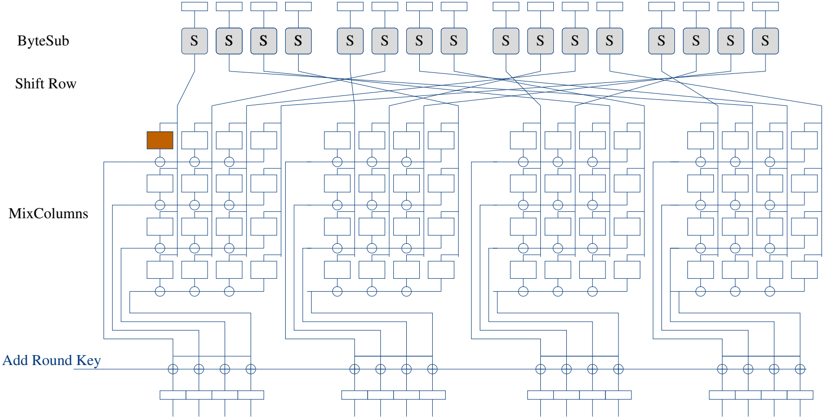




Advanced Side Channel Attacks (DPA like attacks)

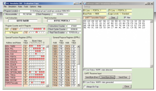
AES Round - 8-bit Software Implementation

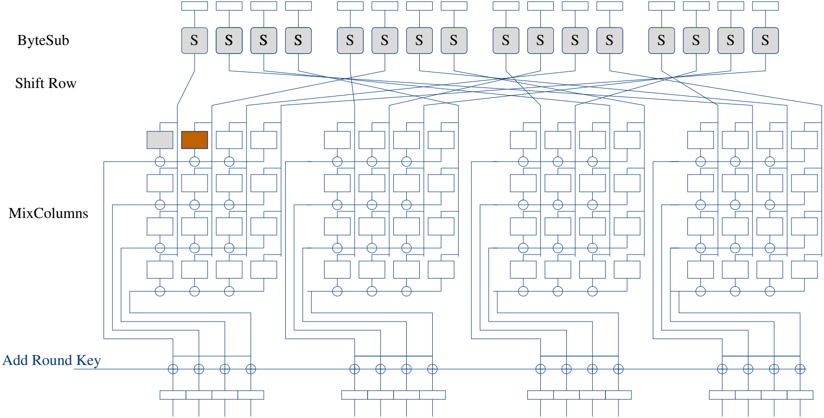




Advanced Side Channel Attacks (DPA like attacks)

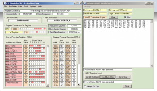
AES Round - 8-bit Software Implementation

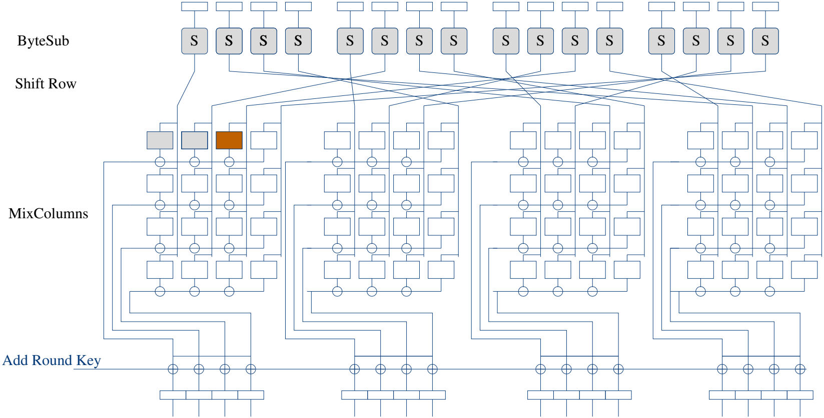




Advanced Side Channel Attacks (DPA like attacks)

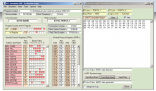
AES Round - 8-bit Software Implementation

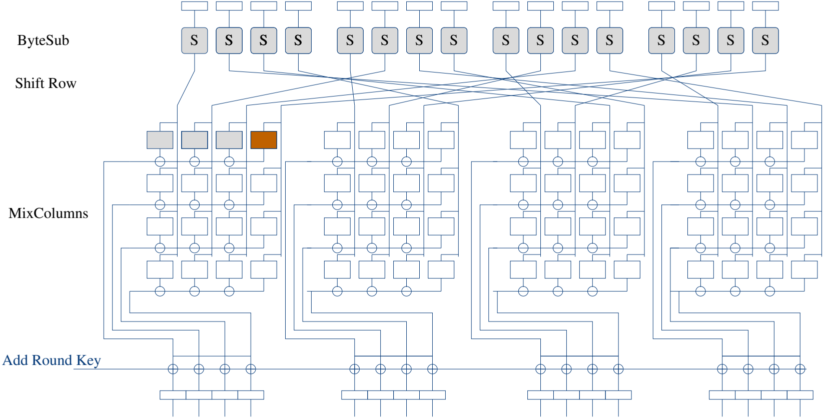




Advanced Side Channel Attacks (DPA like attacks)

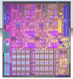
AES Round - 8-bit Software Implementation

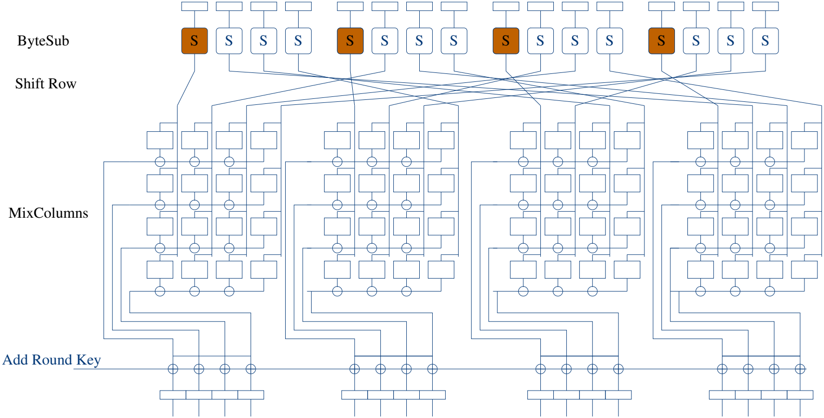




Advanced Side Channel Attacks (DPA like attacks)

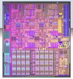
AES Round - Parallel Hardware Implementation

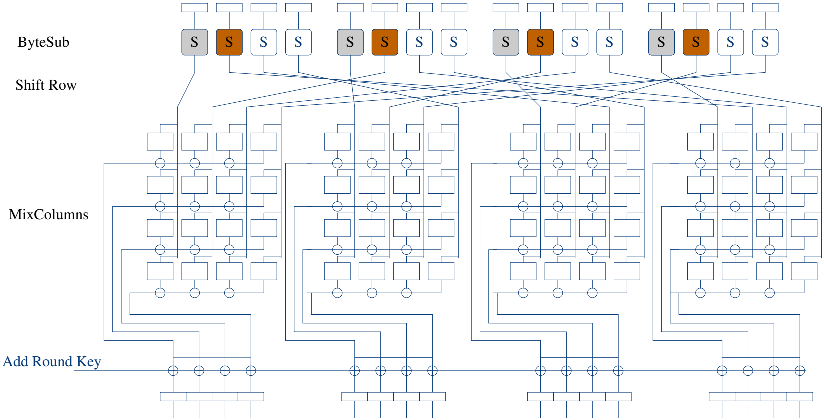




Advanced Side Channel Attacks (DPA like attacks)

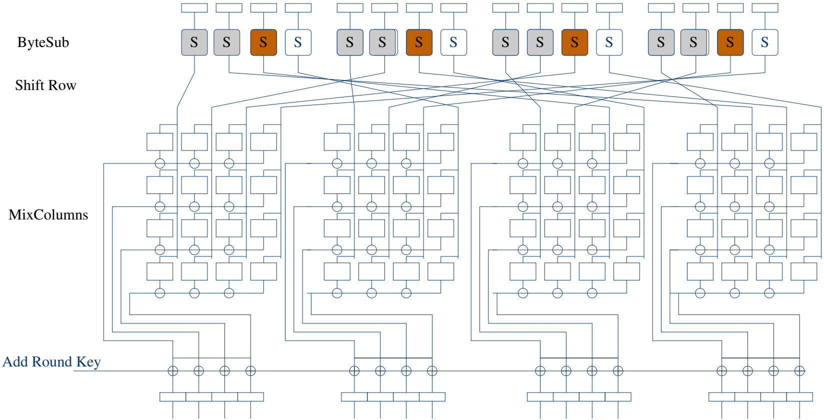
AES Round - Parallel Hardware Implementation





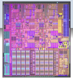
Advanced Side Channel Attacks (DPA like attacks)

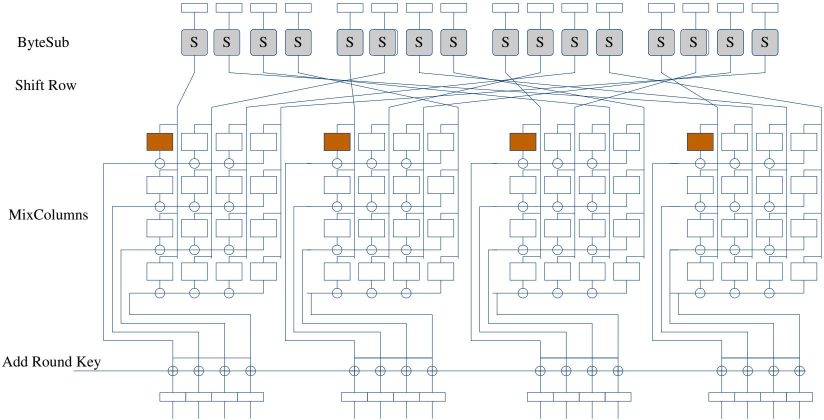
AES Round - Parallel Hardware Implementation



Advanced Side Channel Attacks (DPA like attacks)

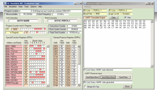
AES Round - Parallel Hardware Implementation

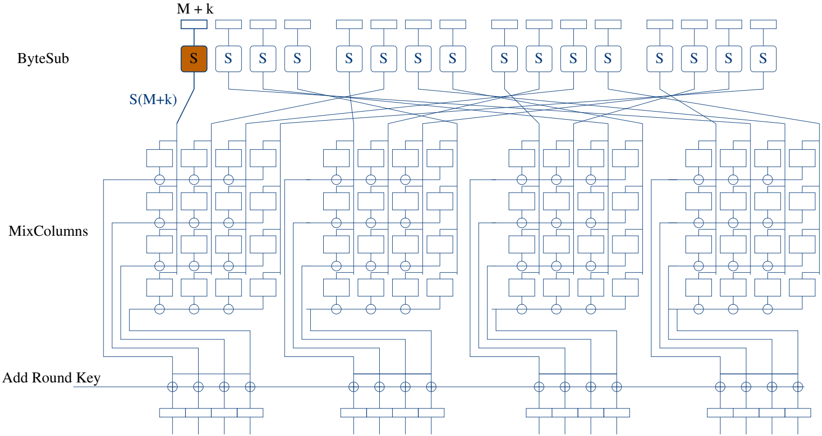




Advanced Side Channel Attacks (DPA like attacks)

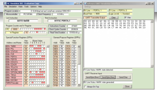
AES Round - Software Implementation – SCA attack

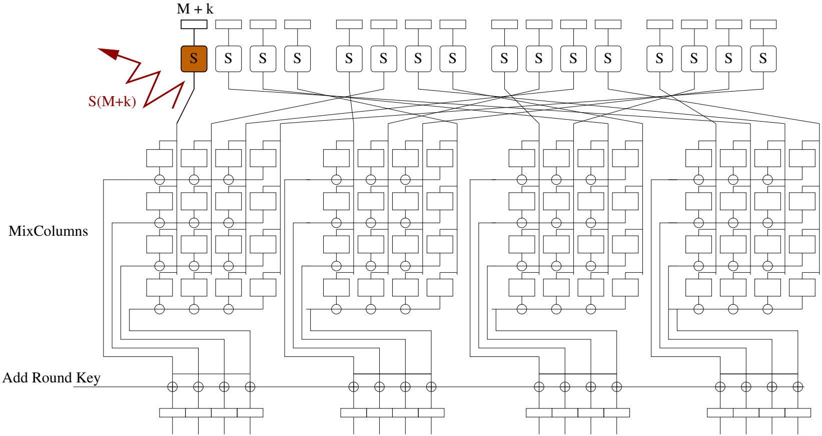




Advanced Side Channel Attacks (DPA like attacks)

AES Round - Software Implementation – SCA attack





Advanced Side Channel Attacks (DPA like attacks)

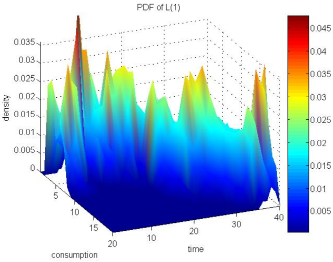
Main Observation

Example: pdf of the leakage for a device processing...

... $\texttt{AES-Sbox}(X + \mathbf{K})$ with $\mathbf{K} = 1$ and $X = \text{cst}$.

For each time (abs.) and each value ℓ in a finite interval (ord.) we plotted in z-axis:

$$Pr[\text{leakage} = \ell] \sim pdf_{\text{leakage}}(\ell)$$



Advanced Side Channel Attacks (DPA like attacks)

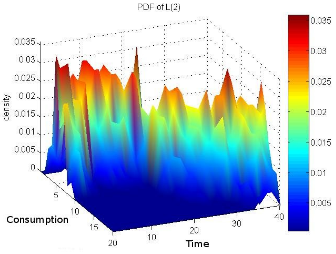
Main Observation

Example: pdf of the leakage for a device processing...

... $\texttt{AES-Sbox}(X + \mathbf{K})$ with $\mathbf{K} = 2$ and $X = \text{cst}$.

For each time (abs.) and each value ℓ in a finite interval (ord.) we plotted in z-axis:

$$Pr[\text{leakage} = \ell] \sim pdf_{\text{leakage}}(\ell)$$



Advanced Side Channel Attacks (DPA like attacks)

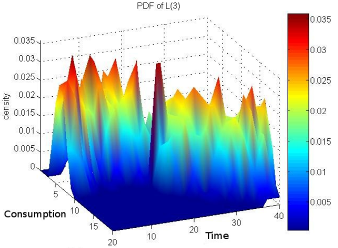
Main Observation

Example: pdf of the leakage for a device processing...

... $\texttt{AES-Sbox}(X + \mathbf{K})$ with $\mathbf{K} = 3$ and $X = \text{cst}$.

For each time (abs.) and each value ℓ in a finite interval (ord.) we plotted in z-axis:

$$Pr[\text{leakage} = \ell] \sim pdf_{\text{leakage}}(\ell)$$



Advanced Side Channel Attacks (DPA like attacks)

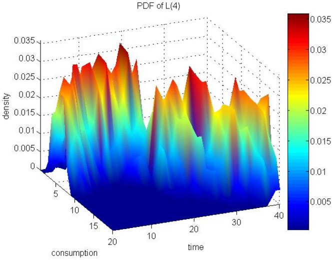
Main Observation

Example: pdf of the leakage for a device processing...

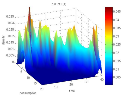
... $\texttt{AES-Sbox}(X + \mathbf{K})$ with $\mathbf{K} = 4$ and $X = \text{cst}$.

For each time (abs.) and each value ℓ in a finite interval (ord.) we plotted in z-axis:

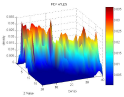
$$Pr[\text{leakage} = \ell] \sim pdf_{\text{leakage}}(\ell)$$



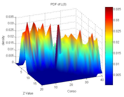
- [Pre-computation] For every possible key $k^\star$ pre-compute the pdf of the leakage $\mathbf{L}$.



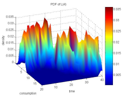
$k^\star = 1$



$k^\star = 2$



$k^\star = 3$

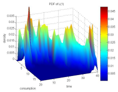


$k^\star = 4$

....

  - [Necessary Condition] Have an open access to a copy of the target device and be able to choose the key value.

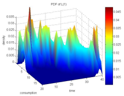
- [Measurement] Measure the consumption for the target device and estimate the pdf of $\mathbf{L}$ for this target.



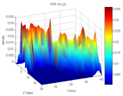
$k^\star = ?$

- [Key-recovery] Compare the pdf estimation with those pre-computed and output the most likely key candidate.

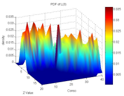
- [Pre-computation] For every possible key $k^\star$ pre-compute the pdf of the leakage $\mathbf{L}$.



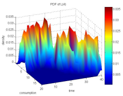
$k^\star = 1$



$k^\star = 2$



$k^\star = 3$

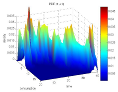


$k^\star = 4$

....

  - [Necessary Condition] Have an open access to a copy of the target device and be able to choose the key value.

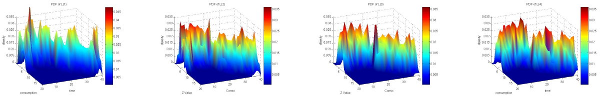
- [Measurement] Measure the consumption for the target device and estimate the pdf of $\mathbf{L}$ for this target.



$k^\star = ?$

- [Key-recovery] Compare the pdf estimation with those pre-computed and output the most likely key candidate.

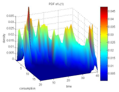
- [Pre-computation] For every possible key $k^\star$ pre-compute the pdf of the leakage $\mathbf{L}$.



$k^\star = 1$ $k^\star = 2$ $k^\star = 3$ $k^\star = 4$ ....

  - [Necessary Condition] Have an open access to a copy of the target device and be able to choose the key value.

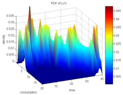
- [Measurement] Measure the consumption for the target device and estimate the pdf of $\mathbf{L}$ for this target.



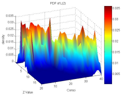
$k^\star = ?$

- [Key-recovery] Compare the pdf estimation with those pre-computed and output the most likely key candidate.

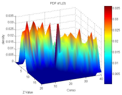
- [Pre-computation] For every possible key $k^\star$ pre-compute the pdf of the leakage $\mathbf{L}$.



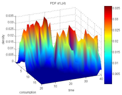
$k^\star = 1$



$k^\star = 2$



$k^\star = 3$

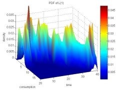


$k^\star = 4$

....

  - [Necessary Condition] Have an open access to a copy of the target device and be able to choose the key value.

- [Measurement] Measure the consumption for the target device and estimate the pdf of $\mathbf{L}$ for this target.

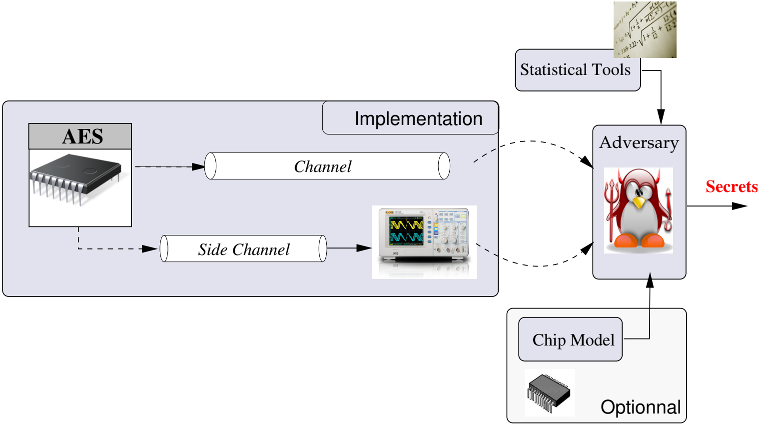


$k^\star = ?$

- [Key-recovery] Compare the pdf estimation with those pre-computed and output the most likely key candidate.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework.



Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X + k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X + k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i+\hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X + k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X + k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X + k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: attack Description Sheet/Form

Attack Description Sheet/Form

Type of Leakage: *e.g. power consumption or electromagnetic emanation*
Model Function: *e.g. one bit of Z or its Hamming weight*
Statistical Distinguisher: *e.g. difference of means, correlation or entropy*
Key Candidate Selection: *e.g. the candidate the maximizes the scores*

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i+\hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) \quad .$$

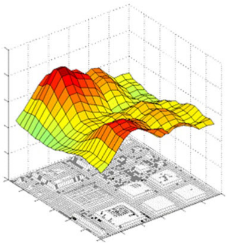
6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: define a model for the consumption.

Goal: define the kind of dependency between the manipulated data and the device behaviour.

- First solution (template/profiled attacks principle):
  - use an exact copy of the attacked device and estimate the pdf of L for every possible pair (X, k).
  - see [Chari et al at CHES 2002].
- Second solution (unprofiled attacks principle):
  - model the function $\mathbf{E}[L|\ X = x, K = k]$.
  - see Messerges PhD Thesis.

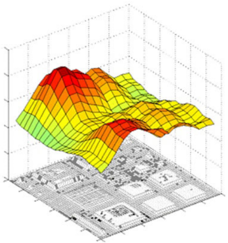


Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: define a model for the consumption.

Goal: define the kind of dependency between the manipulated data and the device behaviour.

- First solution (template/profiled attacks principle):
  - use an exact copy of the attacked device and estimate the pdf of L for every possible pair (X, k).
  - see [Chari et al at CHES 2002].
- Second solution (unprofiled attacks principle):
  - model the function $\mathbf{E}[L|\ X = x, K = k]$.
  - see Messerges PhD Thesis.

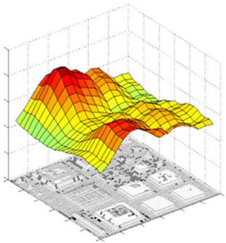


Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: define a model for the consumption.

Goal: define the kind of dependency between the manipulated data and the device behaviour.

- First solution (template/profiled attacks principle):
  - use an exact copy of the attacked device and estimate the pdf of L for every possible pair (X, k).
  - see [Chari et al at CHES 2002].
- Second solution (unprofiled attacks principle):
  - model the function $\mathbf{E}[L| \ X = x, K = k]$.
  - see Messerges PhD Thesis.



Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

Independent Noise Assumption (INA)

The random variable L related to the manipulation of Z equals $Y + B$, where Y is a function of Z and B is independent of Z.

- B is usually called the noise and is viewed as a continuous random variable.
- We usually assume $B \sim \mathcal{N}(0, \sigma^2)$. (Gaussian Noise Assumption).
- Usually, we have $Z = S(X + K)$ where
  - X is known,
  - k is the secret to recover
  - $S(\cdot)$ is a known cryptographic primitive (*e.g. an s-box*).

New problem statement

Modelling = recover the function φ s.t. $Y = \varphi(Z)$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

Independent Noise Assumption (INA)

The random variable L related to the manipulation of Z equals $Y + B$, where Y is a function of Z and B is independent of Z.

- B is usually called the noise and is viewed as a continuous random variable.
- We usually assume $B \sim \mathcal{N}(0, \sigma^2)$. (Gaussian Noise Assumption).
- Usually, we have $Z = S(X + K)$ where
  - X is known,
  - k is the secret to recover
  - $S(\cdot)$ is a known cryptographic primitive (*e.g. an s-box*).

New problem statement

Modelling = recover the function φ s.t. $Y = \varphi(Z)$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

Independent Noise Assumption (INA)

The random variable L related to the manipulation of Z equals $Y + B$, where Y is a function of Z and B is independent of Z.

- B is usually called the noise and is viewed as a continuous random variable.
- We usually assume $B \sim \mathcal{N}(0, \sigma^2)$. (Gaussian Noise Assumption).
- Usually, we have $Z = S(X + K)$ where
  - X is known,
  - k is the secret to recover
  - $S(\cdot)$ is a known cryptographic primitive (*e.g. an s-box*).

New problem statement

Modelling = recover the function φ s.t. $Y = \varphi(Z)$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

Independent Noise Assumption (INA)

The random variable L related to the manipulation of Z equals $Y + B$, where Y is a function of Z and B is independent of Z.

- B is usually called the noise and is viewed as a continuous random variable.
- We usually assume $B \sim \mathcal{N}(0, \sigma^2)$. (Gaussian Noise Assumption).
- Usually, we have $Z = S(X + K)$ where
  - X is known,
  - k is the secret to recover
  - $S(\cdot)$ is a known cryptographic primitive (*e.g. an s-box).*

New problem statement

Modelling = recover the function φ s.t. $Y = \varphi(Z)$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

Independent Noise Assumption (INA)

The random variable L related to the manipulation of Z equals $Y + B$, where Y is a function of Z and B is independent of Z.

- B is usually called the noise and is viewed as a continuous random variable.
- We usually assume $B \sim \mathcal{N}(0, \sigma^2)$. (Gaussian Noise Assumption).
- Usually, we have $Z = S(X + K)$ where
  - X is known,
  - k is the secret to recover
  - $S(\cdot)$ is a known cryptographic primitive (*e.g. an s-box).*

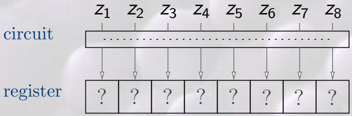
New problem statement

Modelling = recover the function φ s.t. $Y = \varphi(Z)$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$Z = S(X + k)$$

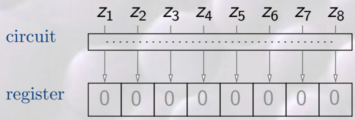


Assumption

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$Z = S(X + k)$$

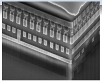


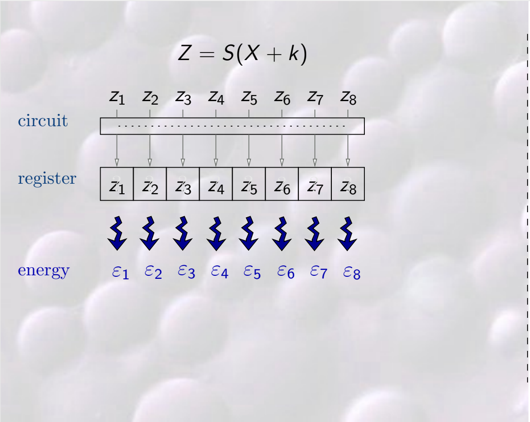
Assumption

0 Precharge

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.





Assumption

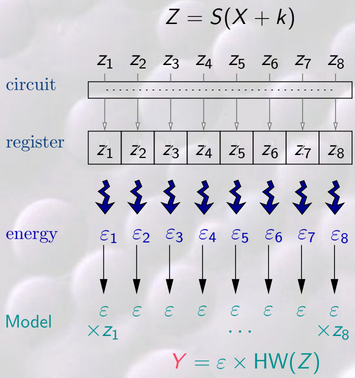
0 Precharge

Linear Regression

the ε_i's are indep.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.



Assumption

0 Precharge

Linear Regression
the ε_i's are indep.

Hamming Weight Model

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

To sum up

- The deterministic part Y in a leakage L may be viewed as a multivariate polynomial in the bit-coordinate z_i of Z with coefficients in $\mathbb{R}$.
  - $\varphi(Z)$ is a polynomial in $\mathbb{R}[z_1, \cdots, z_n]$ and this polynomial is *a priori* unknown to the adversary.
- The modelling problem hence reduces to a problem of polynomial interpolation in noisy context:
  - from noisy observations of $\varphi(Y)$, we want to recover the coefficients ε_0, ε_1, ... such that:

$$\varphi(Z) = \underbrace{\varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots}_{\text{linear part}} + \underbrace{\varepsilon_{0,1} z_0 z_1 + \varepsilon_{0,2} z_0 z_2 + \ldots}_{\text{quadratic part}} + \underbrace{\ldots}_{\text{etc.}}$$

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

To sum up

- The deterministic part Y in a leakage L may be viewed as a multivariate polynomial in the bit-coordinate z_i of Z with coefficients in $\mathbb{R}$.
  - $\varphi(Z)$ is a polynomial in $\mathbb{R}[z_1, \cdots, z_n]$ and this polynomial is *a priori* unknown to the adversary.
- The modelling problem hence reduces to a problem of polynomial interpolation in noisy context:
  - from noisy observations of $\varphi(Y)$, we want to recover the coefficients ε_0, ε_1, ... such that:

$$\varphi(Z) = \underbrace{\varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots}_{\text{linear part}} + \underbrace{\varepsilon_{0,1} z_0 z_1 + \varepsilon_{0,2} z_0 z_2 + \ldots}_{\text{quadratic part}} + \underbrace{\ldots}_{\text{etc.}}$$

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

To sum up

- The deterministic part Y in a leakage L may be viewed as a multivariate polynomial in the bit-coordinate z_i of Z with coefficients in $\mathbb{R}$.
    - $\varphi(Z)$ is a polynomial in $\mathbb{R}[z_1, \cdots, z_n]$ and this polynomial is *a priori* unknown to the adversary.
- The modelling problem hence reduces to a problem of polynomial interpolation in noisy context:
    - from noisy observations of $\varphi(Y)$, we want to recover the coefficients ε_0, ε_1, ... such that:

$$\varphi(Z) = \underbrace{\varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots}_{\text{linear part}} + \underbrace{\varepsilon_{0,1} z_0 z_1 + \varepsilon_{0,2} z_0 z_2 + \ldots}_{\text{quadratic part}} + \underbrace{\ldots}_{\text{etc.}}$$

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

$$\varphi(Z) = \varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots$$

To sum up

- The polynomial interpolation with noise problem is usually solved thanks to linear regression techniques. See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
- Usually, we assume the polynomial $\varphi(Z)$ is of degree 1.
- All the coefficients ε_i for degree-1 monomials are equal (to 1).
- The latter assumption (called Hamming Weight) is today pertinent for almost all smart card technologies.
- For recent ones (*e.g.* 65nm tech.), the non-linear terms must be taken into account. See Veyrat-Charvillon et al's paper at CRYPTO 2011.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

$$\varphi(Z) = \varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots$$

To sum up

- The polynomial interpolation with noise problem is usually solved thanks to linear regression techniques. See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
- Usually, we assume the polynomial $\varphi(Z)$ is of degree 1.
- All the coefficients ε_i for degree-1 monomials are equal (to 1).
- The latter assumption (called Hamming Weight) is today pertinent for almost all smart card technologies.
- For recent ones (*e.g.* 65nm tech.), the non-linear terms must be taken into account. See Veyrat-Charvillon et al's paper at CRYPTO 2011.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

$$\varphi(Z) = \varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots$$

To sum up

- The polynomial interpolation with noise problem is usually solved thanks to linear regression techniques. See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
- Usually, we assume the polynomial $\varphi(Z)$ is of degree 1.
- All the coefficients ε_i for degree-1 monomials are equal (to 1).
- The latter assumption (called Hamming Weight) is today pertinent for almost all smart card technologies.
- For recent ones (*e.g.* 65nm tech.), the non-linear terms must be taken into account. See Veyrat-Charvillon et al's paper at CRYPTO 2011.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

$$\varphi(Z) = \varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots$$

To sum up

- The polynomial interpolation with noise problem is usually solved thanks to linear regression techniques. See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
- Usually, we assume the polynomial $\varphi(Z)$ is of degree 1.
- All the coefficients ε_i for degree-1 monomials are equal (to 1).
- The latter assumption (called Hamming Weight) is today pertinent for almost all smart card technologies.
- For recent ones (*e.g.* 65nm tech.), the non-linear terms must be taken into account. See Veyrat-Charvillon et al's paper at CRYPTO 2011.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

$$\varphi(Z) = \varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots$$

To sum up

- The polynomial interpolation with noise problem is usually solved thanks to linear regression techniques. See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
- Usually, we assume the polynomial $\varphi(Z)$ is of degree 1.
- All the coefficients ε_i for degree-1 monomials are equal (to 1).
- The latter assumption (called Hamming Weight) is today pertinent for almost all smart card technologies.
- For recent ones (*e.g.* 65nm tech.), the non-linear terms must be taken into account. See Veyrat-Charvillon et al's paper at CRYPTO 2011.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: modelling for unprofiled attacks.

$$L \leftarrow Y + B = \varphi(Z) + B$$

$$\varphi(Z) = \varepsilon_0 z_0 + \varepsilon_1 z_1 + \ldots$$

To sum up

- The polynomial interpolation with noise problem is usually solved thanks to linear regression techniques. See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
- Usually, we assume the polynomial $\varphi(Z)$ is of degree 1.
- All the coefficients ε_i for degree-1 monomials are equal (to 1).
- The latter assumption (called Hamming Weight) is today pertinent for almost all smart card technologies.
- For recent ones (*e.g.* 65nm tech.), the non-linear terms must be taken into account. See Veyrat-Charvillon et al's paper at CRYPTO 2011.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) \quad .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function $\mathbf{m}(\cdot)$.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathbf{m}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X + k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function HW.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathrm{HW}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) \quad .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: General Framework (Theoretical)

Context: attack during the manipulation of $S(X+k)$.

1. Measurement :
   - get a leakages sample $(\ell_{k,i})_i$ related to a sample $(x_i)_i$ of plaintexts.
2. Model Selection :
   - Design/Select a function HW.
3. Prediction :
   - For every $\hat{k}$, compute $m_{\hat{k},i} = \mathrm{HW}(S(x_i + \hat{k}))$.
4. Distinguisher Selection :
   - Choose a statistical distinguisher Δ.
5. Key Discrimination :
   - For every $\hat{k}$, compute the distinguishing value $\Delta_{\hat{k}}$:

$$\Delta_{\hat{k}} = \Delta\left((\ell_{k,i})_i, (m_{\hat{k},i})_i\right) .$$

6. Key Candidate Selection :
   - Deduce $\hat{k}$ from all the values $\Delta_{\hat{k}}$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Under INA assumption, the pdf f_L of L is a Gaussian Mixture:

$$f_L(\ell) = \sum_i \Pr[\varphi(Z) = i] \times \mathcal{N}(i, \sigma^2)$$

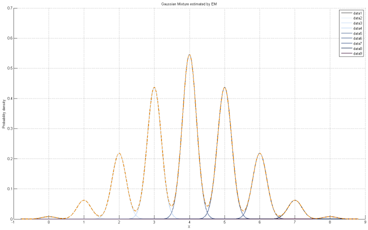


Figure : No noise ($\sigma = 0.2$)

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Under INA assumption, the pdf f_L of L is a Gaussian Mixture:

$$f_L(\ell) = \sum_i \Pr[\varphi(Z) = i] \times \mathcal{N}(i, \sigma^2)$$

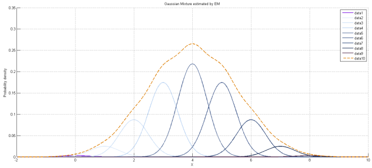


Figure : Small noise ($\sigma = 0.5$)

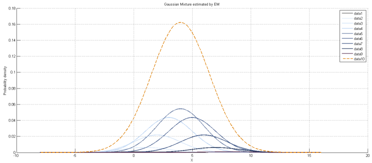


Figure : Medium noise ($\sigma = 2$)

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Question: which property of this mixture depends on the secret k?

Note: difficult question since the adversary does not know φ but a model $\mathbf{m}$ for it!

Many proposals have been done in the literature:

- DPA Kocher et al at CRYPTO 96,
- Multi-bit DPA Messerges in his PhD Thesis,
- CPA Brier et al at CHES 2004,
- Stochastic Attacks Schindler et al at CHES 2006
- or the MIA Gierlichs et al at CHES 2008.
- etc.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Question: which property of this mixture depends on the secret k?

Note: difficult question since the adversary does not know φ but a model $\mathbf{m}$ for it!

Many proposals have been done in the literature:

- DPA Kocher et al at CRYPTO 96,
- Multi-bit DPA Messerges in his PhD Thesis,
- CPA Brier et al at CHES 2004,
- Stochastic Attacks Schindler et al at CHES 2006
- or the MIA Gierlichs et al at CHES 2008.
- etc.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Question: which property of this mixture depends on the secret k?

Note: difficult question since the adversary does not know φ but a model $\mathbf{m}$ for it!

Many proposals have been done in the literature:

- DPA Kocher et al at CRYPTO 96,
- Multi-bit DPA Messerges in his PhD Thesis,
- CPA Brier et al at CHES 2004,
- Stochastic Attacks Schindler et al at CHES 2006
- or the MIA Gierlichs et al at CHES 2008.
- etc.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Question: which property of this mixture depends on the secret k?
Note: difficult question since the adversary does not know φ but a model $\mathbf{m}$ for it!
Many proposals have been done in the literature:

- DPA Kocher et al at CRYPTO 96,
- Multi-bit DPA Messerges in his PhD Thesis,
- CPA Brier et al at CHES 2004,
- Stochastic Attacks Schindler et al at CHES 2006
- or the MIA Gierlichs et al at CHES 2008.
- etc.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

DPA attack Kocher et al at CRYPTO 96.

Attack Description Sheet/Form: DPA

Type of Leakage: no restriction.
Model Function: the function $\mathbf{m} : Z \mapsto z_i$ for some index i.
Statistical Distinguisher: difference of means Test.
Key Candidate Selection: the candidate the maximizes the scores.

Score value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \mathbf{E}(L \mid M_{\hat{k}} = 1) - \mathbf{E}(L \mid M_{\hat{k}} = 0)$$

with $M_{\hat{k}}$ equal to the ith bit of $Z = S(X + \hat{k})$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

DPA attack Kocher et al at CRYPTO 96.

Attack Description Sheet/Form: DPA

Type of Leakage: no restriction.
Model Function: the function $\mathbf{m} : Z \mapsto z_i$ for some index i.
Statistical Distinguisher: difference of means Test.
Key Candidate Selection: the candidate the maximizes the scores.

Score value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \mathbf{E}(L \mid M_{\hat{k}} = 1) - \mathbf{E}(L \mid M_{\hat{k}} = 0)$$

with $M_{\hat{k}}$ equal to the ith bit of $Z = S(X + \hat{k})$.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

DPA attack Kocher et al at CRYPTO 96. Why does it work?

$$\begin{aligned}\Delta_{\hat{k}} &= \mathbf{E}(L \mid M_{\hat{k}} = 1) - \mathbf{E}(L \mid M_{\hat{k}} = 0) \\ &= \mathbf{E}(\varphi(Z) + B \mid M_{\hat{k}} = 1) - \mathbf{E}(\varphi(Z) + B \mid M_{\hat{k}} = 0)\end{aligned}$$

Since the noise B is independent of Z,

$$\begin{aligned}\Delta_{\hat{k}} &= \mathbf{E}(\varphi(Z) \mid M_{\hat{k}} = 1) - \mathbf{E}(\varphi(Z) \mid M_{\hat{k}} = 0) \\ &= \mathbf{E}(\varepsilon_i z_i + (\varphi(Z) - \varepsilon_i z_i) \mid M_{\hat{k}} = 1) - \mathbf{E}(\varepsilon_i z_i + (\varphi(Z) - \varepsilon_i z_i) \mid M_{\hat{k}} = 0)\end{aligned}$$

Let us assume that $(\varphi(Z) - \varepsilon_i z_i)$ is independent of z_i and $M_{\hat{k}}$ (true in practice).

$$\boxed{\Delta_{\hat{k}} = \varepsilon_i \left(\mathbf{E}(z_i \mid M_{\hat{k}} = 1) - \mathbf{E}(z_i \mid M_{\hat{k}} = 0)\right)}$$

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

$$\Delta_{\hat{k}} = \varepsilon_i \left(\mathbf{E}(z_i \mid M_{\hat{k}} = 1) - \mathbf{E}(z_i \mid M_{\hat{k}} = 0)\right)$$

where

- z_i is the ith bit of $S(M + k)$
- $M_{\hat{k}}$ is the ith bit of $S(M + \hat{k})$

If $k = \hat{k}$, then $z_i = M_{\hat{k}}$ and :

$$\Delta_{\hat{k}} = \varepsilon_i \,(1 - 0) = \varepsilon_i$$

If $k = \hat{k}$, then z_i and $M_{\hat{k}}$ are independent (due to properties of S) and

$$\Delta_{\hat{k}} = \varepsilon_i \,(\mathbf{E}(z_i) - \mathbf{E}(z_i)) = 0$$

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

DPA attack Kocher et al at CRYPTO 96.

- Pros: no need for assumption on the device properties, quite efficient in practice.
- Cons: does not use all the information in the trace and attack each bit of the target separately.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Multi-bit DPA attack Messerges in his PhD Thesis.

Attack Description Sheet/Form: Multi-bit DPA

Type of Leakage: no restriction.
Model Function **m**: the Hamming weight function.
Statistical Distinguisher: difference of means for a parameter τ.
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \mathbf{E}(L \mid M_{\hat{k}} \leq \tau) - \mathbf{E}(L \mid M_{\hat{k}} > \tau)$$

with $M_{\hat{k}}$ equal to the $\mathrm{HW}[S(X + \hat{k})]$.

- Pros: exploit more information than the DPA.
- Cons: need assumption (Hamming weight) on the device behaviour.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Multi-bit DPA attack Messerges in his PhD Thesis.

Attack Description Sheet/Form: Multi-bit DPA

Type of Leakage: no restriction.
Model Function **m**: the Hamming weight function.
Statistical Distinguisher: difference of means for a parameter τ.
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \mathbf{E}(L \mid M_{\hat{k}} \leq \tau) - \mathbf{E}(L \mid M_{\hat{k}} > \tau)$$

with $M_{\hat{k}}$ equal to the $\mathrm{HW}[S(X + \hat{k})]$.

- Pros: exploit more information than the DPA.
- Cons: need assumption (Hamming weight) on the device behaviour.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

Multi-bit DPA attack Messerges in his PhD Thesis.

Attack Description Sheet/Form: Multi-bit DPA

Type of Leakage: no restriction.
Model Function **m**: the Hamming weight function.
Statistical Distinguisher: difference of means for a parameter τ.
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \mathbf{E}(L \mid M_{\hat{k}} \leq \tau) - \mathbf{E}(L \mid M_{\hat{k}} > \tau)$$

with $M_{\hat{k}}$ equal to the $\mathrm{HW}[S(X + \hat{k})]$.

- Pros: exploit more information than the DPA.
- Cons: need assumption (Hamming weight) on the device behaviour.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

CPA attack Brier et al at CHES 2004.

Attack Description Sheet/Form: CPA

Type of Leakage: no restriction.
Model Function **m**: possibly any function (in practice HW).
Statistical Distinguisher: linear correlation coefficient.
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \rho(L, M_{\hat{k}})$$

- Pros: exploit more information than the previous ones and is more powerful
- Cons: need assumption (Hamming weight) on the device behaviour.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

CPA attack Brier et al at CHES 2004.

Attack Description Sheet/Form: CPA

Type of Leakage: no restriction.
Model Function **m**: possibly any function (in practice HW).
Statistical Distinguisher: linear correlation coefficient.
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \rho(L, M_{\hat{k}})$$

- Pros: exploit more information than the previous ones and is more powerful
- Cons: need assumption (Hamming weight) on the device behaviour.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

CPA attack Brier et al at CHES 2004.

Attack Description Sheet/Form: CPA

Type of Leakage: no restriction.
Model Function **m**: possibly any function (in practice HW).
Statistical Distinguisher: linear correlation coefficient.
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = \rho(L, M_{\hat{k}})$$

- Pros: exploit more information than the previous ones and is more powerful
- Cons: need assumption (Hamming weight) on the device behaviour.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

MIA attack Gierlichs et al at CHES 2008.

Attack Description Sheet/Form: MIA

Type of Leakage: no restriction.
Model Function **m**: any non-injective function (in practice HW).
Statistical Distinguisher: mutual information (MI).
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = MI(L; M_{\hat{k}}) = \text{entropy}(L) - \text{entropy}(L \mid M_{\hat{k}})$$

- Pros: theoretically able to detect any kind of dependency whatever the quality of the model if the function $x \mapsto \mathbf{m} \circ S(x+k)$ is non-injective!
- Cons: need for efficient estimators of the entropy (currently less efficient than the CPA) Batina *et al, Journal of Cryptology 2011*.

Advanced Side Channel Attacks (DPA like attacks)

Side Channel Analysis: the statistical distinguisher

MIA attack Gierlichs et al at CHES 2008.

Attack Description Sheet/Form: MIA

Type of Leakage: no restriction.
Model Function **m**: any non-injective function (in practice HW).
Statistical Distinguisher: mutual information (MI).
Key Candidate Selection: the candidate the maximizes the scores.

Distinguishing value $\Delta_{\hat{k}}$: a statistical estimator of

$$\Delta_{\hat{k}} = MI(L; M_{\hat{k}}) = \mathrm{entropy}(L) - \mathrm{entropy}(L \mid M_{\hat{k}})$$

- Pros: theoretically able to detect any kind of dependency whatever the quality of the model if the function $x \mapsto \mathbf{m} \circ S(x+k)$ is non-injective!
- Cons: need for efficient estimators of the entropy (currently less efficient than the CPA) Batina *et al, Journal of Cryptology 2011*.

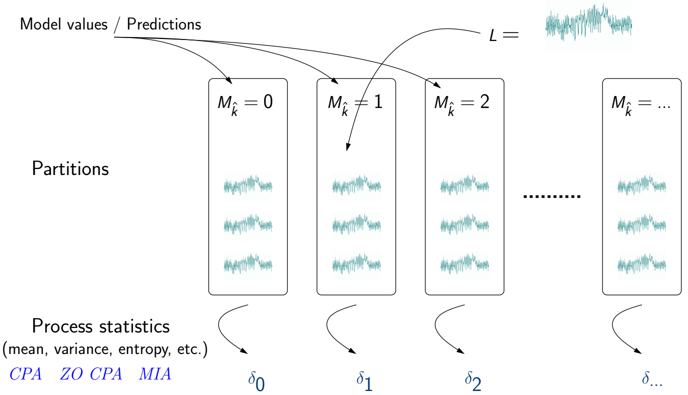
Advanced Side Channel Attacks (DPA like attacks)

Other attacks

- Stochastic attacks: See Schindler et al. at CHES 2005 or Doget et al at JCEN 2011.
  - Good alternative when classical (*e.g. HW) models fail.*
  - *Amounts to process an Euclidean distance between the leakage values and the estimations in the regressed model.*
- Kolmogorov-Smirnov Based attacks: Whitnall et al. at CARDIS 2011.
  - Good alternative to the MIA.
- PPA, EPA, VPA, etc: other attacks exist **but are often very ad hoc ones with no clear advantage to the "classical" ones**.
- Works comparing the attacks:
  - "How to Compare Profiled Side-Channel Attacks?" Standaert *et al, ACNS 2009*.
  - "A fair evaluation framework for comparing side-channel distinguishers" by Withnall *et al, JCEN 2011*.
  - "Univariate Side Channel Attacks and Leakage Modeling" by Doget *et al, JCEN 2011*.

Distinguishers Processing

... a partitioning description.



Combine the statistics

$$\Delta_{\hat{k}} = \sum_i \delta_i \times \mathrm{P}[M_{\hat{k}} = i]$$

Advanced Side Channel Attacks (DPA like attacks)

Attack Efficiency Consideration

Attack Efficiency

The efficiency of an SCA given a success rate β is the smallest value N such that:

$$\Pr(\text{Attack succeeds in recovering } k \text{ with } N \text{ measurements}) \geq \beta \ .$$

Particular case: the attack involves correlation coefficient ($i.e. \Delta = \rho$):

$$\Pr\left(\hat{\rho}_k(N) > \max_{\hat{k} \neq k} \hat{\rho}_{\hat{k}}(N)\right) \geq \beta \ .$$

where $\hat{\rho}_k(N)$ denotes the estimation of ρ_k based on N.

Advanced Side Channel Attacks (DPA like attacks)

Attack Efficiency Consideration

Attack Efficiency

The efficiency of an SCA given a success rate β is the smallest value N such that:

$$\Pr(\text{Attack succeeds in recovering } k \text{ with } N \text{ measurements}) \geq \beta \ .$$

Particular case: the attack involves correlation coefficient ($i.e. \Delta = \rho$):

$$\Pr\left(\hat{\rho}_k(N) > \max_{\hat{k} \neq k} \hat{\rho}_{\hat{k}}(N)\right) \geq \beta \ .$$

where $\hat{\rho}_k(N)$ denotes the estimation of ρ_k based on N.

Advanced Side Channel Attacks (DPA like attacks)

Attack Efficiency Consideration

Attack Efficiency

The efficiency of an SCA given a success rate β is the smallest value N such that:

$$\Pr(\text{Attack succeeds in recovering } k \text{ with } N \text{ measurements}) \geq \beta \ .$$

Particular case: the attack involves correlation coefficient ($i.e. \Delta = \rho$):

$$\Pr\left(\hat{\rho}_k(N) > \max_{\hat{k} \neq k} \hat{\rho}_{\hat{k}}(N)\right) \geq \beta \ .$$

where $\hat{\rho}_k(N)$ denotes the estimation of ρ_k based on N.

Advanced Side Channel Attacks (DPA like attacks)

Attack Efficiency Consideration

- Fisher: when $\hat{\rho}_{\hat{k}}(N)$ is computed between samples that have a joint normal distribution, $Z_{N,\hat{k}} = \frac{1}{2}\ln\left(\frac{1+\hat{\rho}_k(N)}{1-\hat{\rho}_{\hat{k}}(N)}\right)$ has a normal distribution with parameters

$$\mathbf{E}(Z_{N,\hat{k}}) = \frac{1}{2}\ln\left(\frac{1+\rho_k}{1-\rho_{\hat{k}}}\right) \text{ and } \mathrm{Var}(Z_{N,\hat{k}}) = (N-3)^{-2}.$$

- [Mangard at CT-RSA 2004] So, $\Pr(\hat{\rho}_k(N) > \hat{\rho}_{\hat{k}}(N)) = \beta$ implies:

$$N = 3 + 8\left(\frac{\Phi^{-1}(\beta)}{\ln\left(\frac{1+\rho_k}{1-\rho_{\hat{k}}}\right)}\right)^2 ,$$

where Φ denotes the pdf of $\mathcal{N}(0,1)$.

Advanced Side Channel Attacks (DPA like attacks)

Attack Efficiency Consideration

- Fisher: when $\hat{\rho}_{\hat{k}}(N)$ is computed between samples that have a joint normal distribution, $Z_{N,\hat{k}} = \frac{1}{2}\ln\left(\frac{1+\hat{\rho}_k(N)}{1-\hat{\rho}_{\hat{k}}(N)}\right)$ has a normal distribution with parameters

$$\mathbf{E}(Z_{N,\hat{k}}) = \frac{1}{2}\ln\left(\frac{1+\rho_k}{1-\rho_{\hat{k}}}\right) \text{ and } \mathrm{Var}(Z_{N,\hat{k}}) = (N-3)^{-2}.$$

- [Mangard at CT-RSA 2004] So, $\Pr(\hat{\rho}_k(N) > \hat{\rho}_{\hat{k}}(N)) = \beta$ implies:

$$N = 3 + 8\left(\frac{\Phi^{-1}(\beta)}{\ln\left(\frac{1+\rho_k}{1-\rho_{\hat{k}}}\right)}\right)^2,$$

where Φ denotes the pdf of $\mathcal{N}(0,1)$.

Advanced Side Channel Attacks (DPA like attacks)

Attack Efficiency Consideration

- Fisher: when $\hat{\rho}_{\hat{k}}(N)$ is computed between samples that have a joint normal distribution, $Z_{N,\hat{k}} = \frac{1}{2}\ln\left(\frac{1+\hat{\rho}_k(N)}{1-\hat{\rho}_{\hat{k}}(N)}\right)$ has a normal distribution with parameters

$$\mathbf{E}(Z_{N,\hat{k}}) = \frac{1}{2}\ln\left(\frac{1+\rho_k}{1-\rho_{\hat{k}}}\right) \text{ and } \mathrm{Var}(Z_{N,\hat{k}}) = (N-3)^{-2}.$$

- [Mangard at CT-RSA 2004] Assuming $\rho_{\hat{k}}(N) = 0$ we get:

$$\boxed{N \approx 8 \times \Phi^{-1}(\beta)^2 \times \rho_k^{-2}} \ ,$$

since $\ln(1+x) \approx x$ if $|x| < 1$.

Advanced Side Channel Attacks (DPA like attacks)

Link between Efficiency and Signal to Noise Ratio (SNR)

- Let us define the SNR by:

$$\mathrm{SNR} = \frac{\mathbf{Var}[\mathbf{L}] - \mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]} = \frac{\mathbf{Var}[\varphi(Z)]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}$$

  Note: can be computed without knowing φ!

- [Mangard at CT-RSA 2004] If SNR $\ll 1$, we have

$$\rho_{\hat{k}}(N) = \mathrm{SNR} \times \rho^0_{\hat{k}}(N)$$

  where $\rho^0_{\hat{k}}(N)$ denotes the correl. when there is no stoch. noise.

- Consequently,

$$N \sim \frac{1}{SNR}$$

$SNR = 0.01$	$\rightarrow$	around 100 traces	$\rightarrow$	*few seconds*
$SNR = 0.001$	$\rightarrow$	around 1000 traces	$\rightarrow$	*less than $^1/_4$ hour*
$SNR = 0.0001$	$\rightarrow$	around $1O^5$ traces	$\rightarrow$	*several hours*
$SNR = 10^{-6}$	$\rightarrow$	several millions of traces	$\rightarrow$	*several days*

Advanced Side Channel Attacks (DPA like attacks)

Link between Efficiency and Signal to Noise Ratio (SNR)

- Let us define the SNR by:

$$\mathrm{SNR} = \frac{\mathbf{Var}[\mathbf{L}] - \mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]} = \frac{\mathbf{Var}[\varphi(Z)]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}$$

  Note: can be computed without knowing φ!

- [Mangard at CT-RSA 2004] If SNR $\ll 1$, we have

$$\rho_{\hat{k}}(N) = \mathrm{SNR} \times \rho^{0}_{\hat{k}}(N)$$

  where $\rho^{0}_{\hat{k}}(N)$ denotes the correl. when there is no stoch. noise.

- Consequently,

$$N \sim \frac{1}{SNR}$$

$SNR = 0.01$	$\rightarrow$	around 100 traces	$\rightarrow$	*few seconds*
$SNR = 0.001$	$\rightarrow$	around 1000 traces	$\rightarrow$	*less than $^1/_4$ hour*
$SNR = 0.0001$	$\rightarrow$	around 10^5 traces	$\rightarrow$	*several hours*
$SNR = 10^{-6}$	$\rightarrow$	several millions of traces	$\rightarrow$	*several days*

Advanced Side Channel Attacks (DPA like attacks)

Link between Efficiency and Signal to Noise Ratio (SNR)

- Let us define the SNR by:

$$\mathrm{SNR} = \frac{\mathbf{Var}[\mathbf{L}] - \mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]} = \frac{\mathbf{Var}[\varphi(Z)]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}$$

  Note: can be computed without knowing φ!

- [Mangard at CT-RSA 2004] If SNR $\ll 1$, we have

$$\rho_{\hat{k}}(N) = \mathrm{SNR} \times \rho^{0}_{\hat{k}}(N)$$

  where $\rho^{0}_{\hat{k}}(N)$ denotes the correl. when there is no stoch. noise.

- Consequently,

$$N \sim \frac{1}{SNR}$$

$SNR = 0.01$	$\rightarrow$	around 100 traces	$\rightarrow$	*few seconds*
$SNR = 0.001$	$\rightarrow$	around 1000 traces	$\rightarrow$	*less than $^1/_4$ hour*
$SNR = 0.0001$	$\rightarrow$	around 10^5 traces	$\rightarrow$	*several hours*
$SNR = 10^{-6}$	$\rightarrow$	several millions of traces	$\rightarrow$	*several days*

Advanced Side Channel Attacks (DPA like attacks)

Link between Efficiency and Signal to Noise Ratio (SNR)

- Let us define the SNR by:

$$\mathrm{SNR} = \frac{\mathbf{Var}[\mathbf{L}] - \mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]} = \frac{\mathbf{Var}[\varphi(Z)]}{\mathbf{E}[\mathbf{Var}[\mathbf{L} \mid Z]]}$$

Note: can be computed without knowing φ!

- [Mangard at CT-RSA 2004] If SNR $\ll 1$, we have

$$\boxed{\rho_{\hat{k}}(N) = \mathrm{SNR} \times \rho^0_{\hat{k}}(N)}$$

where $\rho^0_{\hat{k}}(N)$ denotes the correl. when there is no stoch. noise.

- Consequently,

$$N \sim \frac{1}{SNR}$$

$SNR = 0.01$	$\rightarrow$	around 100 traces	$\rightarrow$	*few seconds*
$SNR = 0.001$	$\rightarrow$	around 1000 traces	$\rightarrow$	*less than $^1/_4$ hour*
$SNR = 0.0001$	$\rightarrow$	around $1O^5$ traces	$\rightarrow$	*several hours*
$SNR = 10^{-6}$	$\rightarrow$	several millions of traces	$\rightarrow$	*several days*

Advanced Side Channel Attacks (DPA like attacks)

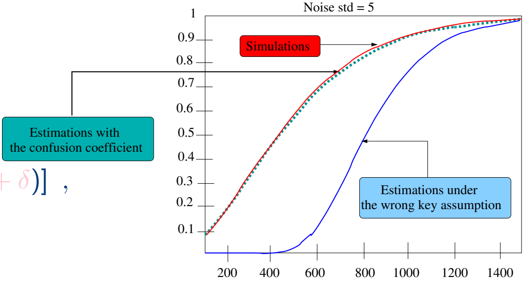
More accurate efficiency evaluations

- Core Idea: relax the assumption $\rho_{\hat{k}}(N) = 0$ for any $\hat{k} \neq 0$.
- Note: this assumption contradicts the ghost Peaks phenomenon ... which is however observed in practice!
- Recent works on this subject: Rivain, SAC 2008, Fei, Luo, Ding, CHES 2012, Thillard, Prouff, Roche, CHES 2013.

Use the notion of Confusion Coefficient, defined for every δ by:

$$\kappa_\delta = \mathbf{E}[\mathrm{m}\circ S(X+k)\times \mathrm{m}\circ S(X+k+\delta)] \ ,$$

where m is the model used in the attack.



Advanced Side Channel Attacks (DPA like attacks)

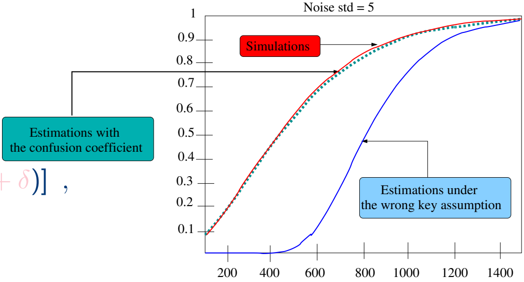
More accurate efficiency evaluations

- Core Idea: relax the assumption $\rho_{\hat{k}}(N) = 0$ for any $\hat{k} \neq 0$.
- Note: this assumption contradicts the ghost Peaks phenomenon ... which is however observed in practice!
- Recent works on this subject: Rivain, SAC 2008, Fei, Luo, Ding, CHES 2012, Thillard, Prouff, Roche, CHES 2013.

Use the notion of Confusion Coefficient, defined for every δ by:

$$\kappa_\delta = \mathbf{E}[\mathrm{m} \circ S(X+k) \times \mathrm{m} \circ S(X+k+\delta)] \ ,$$

where m is the model used in the attack.



Advanced Side Channel Attacks (DPA like attacks)

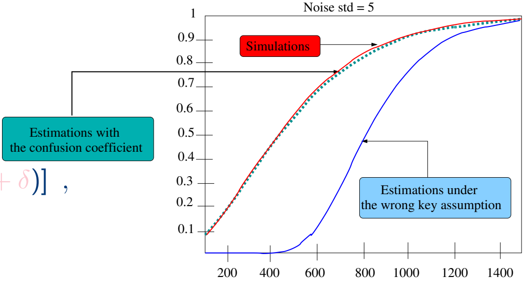
More accurate efficiency evaluations

- Core Idea: relax the assumption $\rho_{\hat{k}}(N) = 0$ for any $\hat{k} \neq 0$.
- Note: this assumption contradicts the ghost Peaks phenomenon ... which is however observed in practice!
- Recent works on this subject: Rivain, SAC 2008, Fei, Luo, Ding, CHES 2012, Thillard, Prouff, Roche, CHES 2013.

Use the notion of Confusion Coefficient, defined for every δ by:

$$\kappa_\delta = \mathbf{E}[\mathrm{m}\circ S(X+k)\times \mathrm{m}\circ S(X+k+\delta)] ,$$

where m is the model used in the attack.



Advanced Side Channel Attacks (DPA like attacks)

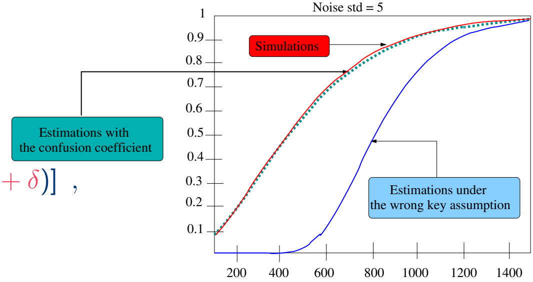
More accurate efficiency evaluations

- Core Idea: relax the assumption $\rho_{\hat{k}}(N) = 0$ for any $\hat{k} \neq 0$.
- Note: this assumption contradicts the ghost Peaks phenomenon ... which is however observed in practice!
- Recent works on this subject: Rivain, SAC 2008, Fei, Luo, Ding, CHES 2012, Thillard, Prouff, Roche, CHES 2013.

Use the notion of Confusion Coefficient, defined for every δ by:

$$\kappa_\delta = \mathbf{E}[\mathrm{m}\circ S(X+k)\times \mathrm{m}\circ S(X+k+\delta)] \ ,$$

where m is the model used in the attack.



Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\Delta(L, g(X, k)) < \varepsilon \ ,$$

where X is some plaintext part and ε is a security parameter.

Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\Delta(L, g(X, k)) < \varepsilon \ ,$$

where X is some plaintext part and ε is a security parameter.

Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\Delta(L, g(X, k)) < \varepsilon \ ,$$

where X is some plaintext part and ε is a security parameter.

Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\Delta(L, g(X, k)) < \varepsilon \ ,$$

where X is some plaintext part and ε is a security parameter.

Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\Delta(L, g(X, k)) < \varepsilon \ ,$$

where X is some plaintext part and ε is a security parameter.

Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\Delta(L, g(X,k)) < \varepsilon \ ,$$

where X is some plaintext part and ε is a security parameter.

Advanced Side Channel Attacks (DPA like attacks)

Efficiency of Other Attacks (MIA, Templates, etc.)

- When provided with the same *a priori* information about the leakage, CPA, MIA, DPA and Gaussian template attacks are asymptotically equivalent Mangard *et al, IET Information Security 2011*.
  - $\Longrightarrow$ Efficiency formula $N \approx 8 \times \Phi^{-1}(\beta)^2 \times \Delta_k^{-2}$ stays true for the corresponding distinguishers.
- Note: for Template attacks, the cost of the on-line phase may be constant but the cost of the off-line templates building will be linear in SNR^{-1}.
- In conclusion, adding security consists in finding efficient way(s) to decrease Δ_k as much as possible.
  - *i.e.*specify the algorithm implementation such that for any instantaneous leakage L, for any key part k and for any function g:

$$\boxed{\Delta(L, g(X, k)) < \varepsilon \ ,}$$

where X is some plaintext part and ε is a security parameter.

Part III

Software Countermeasures for AES and HOSCA

Outline

Plan

SCA Countermeasures

- Masking [IBM Team at CRYPTO 1999].
  - Efficient against SCA in practice.
  - Difficult to implement for non-linear transformations.
- Shuffling [Researchers from Graz University at ACNS 2006].
  - Less efficient against SCA in practice.
  - Easy to implement for every transformation.
- Whitening [Kocher Jaffe June, CRYPTO 1999].
  - Less efficient than masking when used alone and costly in Hardware.
  - Easy to implement for every transformation.

SCA Countermeasures

- Masking [IBM Team at CRYPTO 1999].
  - Efficient against SCA in practice.
  - Difficult to implement for non-linear transformations.
- Shuffling [Researchers from Graz University at ACNS 2006].
  - Less efficient against SCA in practice.
  - Easy to implement for every transformation.
- Whitening [Kocher Jaffe June, CRYPTO 1999].
  - Less efficient than masking when used alone and costly in Hardware.
  - Easy to implement for every transformation.

SCA Countermeasures

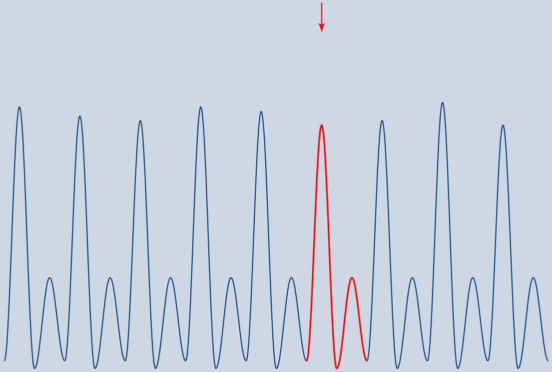
- Masking [IBM Team at CRYPTO 1999].
  - Efficient against SCA in practice.
  - Difficult to implement for non-linear transformations.
- Shuffling [Researchers from Graz University at ACNS 2006].
  - Less efficient against SCA in practice.
  - Easy to implement for every transformation.
- Whitening [Kocher Jaffe June, CRYPTO 1999].
  - Less efficient than masking when used alone and costly in Hardware.
  - Easy to implement for every transformation.

SCA Countermeasures

- Masking [IBM Team at CRYPTO 1999].
  - Efficient against SCA in practice.
  - Difficult to implement for non-linear transformations.
- Shuffling [Researchers from Graz University at ACNS 2006].
  - Less efficient against SCA in practice.
  - Easy to implement for every transformation.
- Whitening [Kocher Jaffe June, CRYPTO 1999].
  - Less efficient than masking when used alone and costly in Hardware.
  - Easy to implement for every transformation.

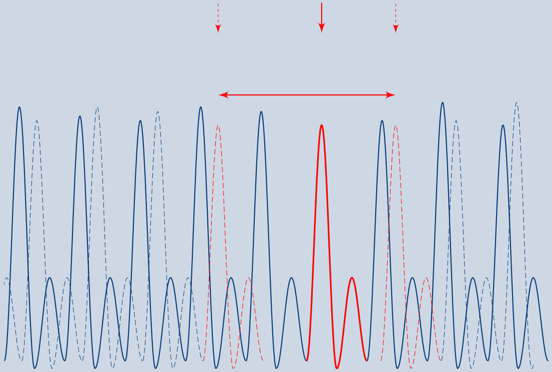
SCA Countermeasures

Core Ideas



SCA Countermeasures

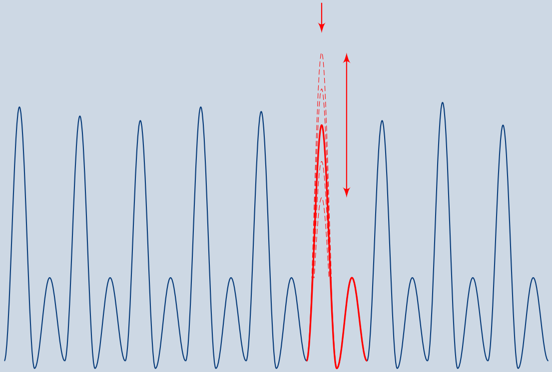
Core Ideas



Desynchronisation = different points, same amplitutde

SCA Countermeasures

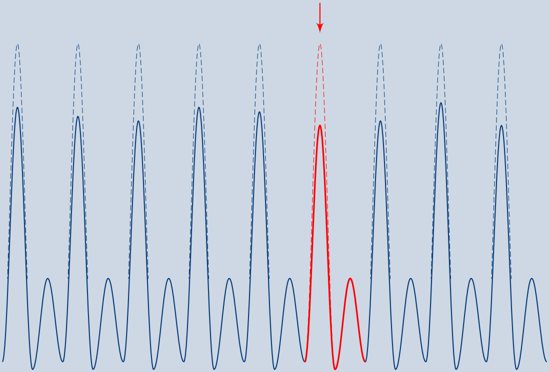
Core Ideas



Masking = same point, random amplitude

SCA Countermeasures

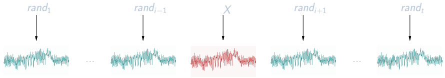
Core Ideas



Balanced Logic = same point, constant amplitude

SCA Countermeasures
Shuffling Method

- **Core Idea:** spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:

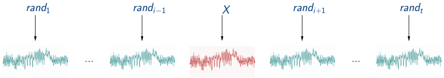


- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures

Shuffling Method

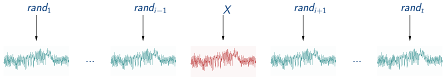
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

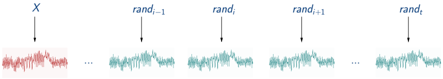
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

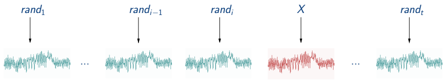
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

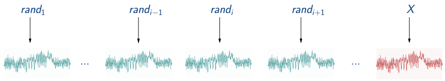
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

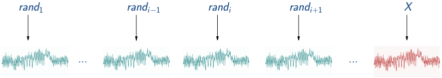
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

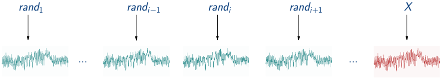
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

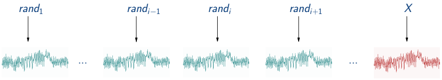
- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Shuffling Method

- Core Idea: spread the sensitive signal related to Z over t different signals $S_1, \ldots, S_t$ leaking at different times.
- Select an index at random:



- Impact: decreases the attack efficiency by a factor of t (*i.e.* $\Delta_k^2 \longrightarrow \Delta_k^2/t$)
- Asset: can be used to protect any operation Op on Z.
- Issue: t must be very large to have satisfying security.

SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares $M_0, \ldots, M_d$ s.t

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g.* *an s-box*).

SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares M_0,..., M_d s.t

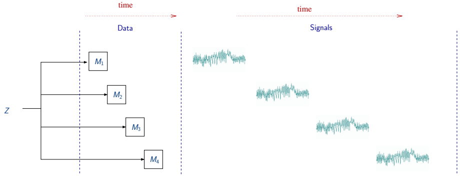
$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g. an s-box*).

SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares $M_0, \ldots, M_d$ s.t

$$M_0 \oplus \cdots \oplus M_d = Z \quad .$$



- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g.* an s-box).

SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares M_0,..., M_d s.t

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g.* *an s-box*).

SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares $M_0, \ldots, M_d$ s.t

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g. an s-box*).

SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares M_0,..., M_d s.t

$$M_0 \oplus \cdots \oplus M_d = Z \quad .$$

- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g.* *an s-box*).

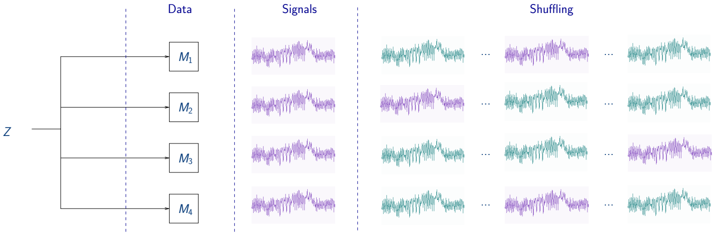
SCA Countermeasures
Masking Method

- Core idea: randomly split Z into $d+1$ shares $M_0, \ldots, M_d$ s.t

$$M_0 \oplus \cdots \oplus M_d = Z \quad .$$

- Impact: attack efficiency decreases exponentially w.r.t. d, the base of the exponential being the noise std [CJRR99].
- Impact: for $d = 2$ the distinguisher value Δ_k of first-order SCA is reduced to 0!
- Asset: easy to apply when Op is linear.
- Issue: even for small d, it is costly when Op is non linear (*e.g. an s-box*).

SCA Countermeasures
Masking-and-Shuffling Method



$Z = M_1 \oplus M_2 \oplus M_3 \oplus M_4$

Plan

SCA Countermeasures
Masking Scheme for Block Ciphers

- SPN networks (*e.g. DES, AES*)
- *The different transformations must satisfy:*

Completeness

The masked variable M_0 and the masks M_i must verify:

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

Security

All the shares M_i must be manipulated at different times.

SCA Countermeasures

Masking Scheme for Block Ciphers

- SPN networks (*e.g. DES, AES*)
- *The different transformations must satisfy:*

Completeness

The masked variable M_0 and the masks M_i must verify:

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

Security

All the shares M_i must be manipulated at different times.

SCA Countermeasures
Masking Scheme for Block Ciphers

- SPN networks (*e.g. DES, AES*)
- *The different transformations must satisfy:*

Completeness

The masked variable M_0 and the masks M_i must verify:

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

Security

All the shares M_i must be manipulated at different times.

SCA Countermeasures

Masking Scheme for Block Ciphers

- Propagation through **linear transformation**

$$
\begin{tikzcd}
M_0 \arrow[d] & M_1 \arrow[d] & \cdots & M_d \arrow[d] \\
\boxed{\mathrm{L}} \arrow[d] & \boxed{\mathrm{L}} \arrow[d] & & \boxed{\mathrm{L}} \arrow[d] \\
L(M_0) & L(M_1) & \cdots & L(M_d)
\end{tikzcd}
$$

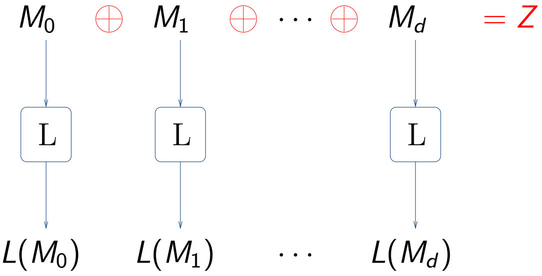
Issue

How to compute $(M_0', \ldots, M_d')$ from $(M_0, \ldots, M_d)$ in a secure way?

SCA Countermeasures

Masking Scheme for Block Ciphers

- Propagation through **linear transformation**



Issue

How to compute $(M'_0, \ldots, M'_d)$ from $(M_0, \ldots, M_d)$ in a secure way?

SCA Countermeasures
Masking Scheme for Block Ciphers

- Propagation through **linear transformation**

$$M_0 \oplus M_1 \oplus \cdots \oplus M_d = Z$$

$$L(M_0) \oplus L(M_1) \oplus \cdots \oplus L(M_d) = L(Z)$$

Issue

How to compute $(M'_0, \ldots, M'_d)$ from $(M_0, \ldots, M_d)$ in a secure way?

SCA Countermeasures

Masking Scheme for Block Ciphers

- Propagation through **s-box**

$$M_0 \oplus M_1 \oplus \cdots \oplus M_d = Z$$

$$S(M_0) \oplus S(M_1) \oplus \cdots \oplus S(M_d) \neq S(Z)$$

Issue

How to compute $(M'_0, \ldots, M'_d)$ from $(M_0, \ldots, M_d)$ in a secure way?

SCA Countermeasures
Masking Scheme for Block Ciphers

- Propagation through **s-box**

$$
\begin{array}{ccccccccc}
M_0 & \oplus & M_1 & \oplus & \cdots & \oplus & M_d & & = Z \\
\downarrow & & \downarrow & & & & \downarrow & & \\
?? & & ?? & & & & ?? & & \\
\downarrow & & \downarrow & & & & \downarrow & & \\
M'_0 & \oplus & M'_1 & \oplus & \cdots & \oplus & M'_d & & = S(Z)
\end{array}
$$

Issue

How to compute $(M'_0, \ldots, M'_d)$ from $(M_0, \ldots, M_d)$ in a secure way?

SCA Countermeasures
Masking Scheme for first order

- Method by table recomputation for $d = 1$

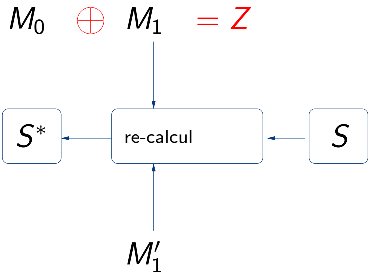


Table Recomputation

For every x: $S^*(x) \leftarrow S(x \oplus M_1) \oplus M'_1$

- $M'_0 \leftarrow S^*(M_0)$

SCA Countermeasures

Masking Scheme for first order

- Method by table recomputation for $d = 1$

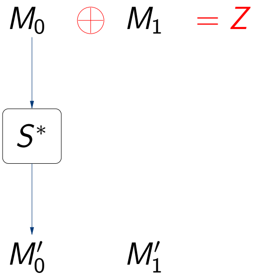


Table Recomputation

For every x: $S^*(x) \leftarrow S(x \oplus M_1) \oplus M'_1$

- $M'_0 \leftarrow S^*(M_0)$

SCA Countermeasures
Masking Scheme for first order

- Method by table recomputation for $d = 1$

$$
\begin{array}{ccccc}
M_0 & \oplus & M_1 & = & Z \\
\downarrow & & & & \\
\boxed{S^*} & & & & \\
\downarrow & & & & \\
M'_0 & \oplus & M'_1 & = & S(Z)
\end{array}
$$

Table Recomputation

For every x: $\quad S^*(x) \leftarrow S(x \oplus M_1) \oplus M'_1$

- $M'_0 \leftarrow S^*(M_0) = S(M_0 \oplus M_1) \oplus M'_1 = S(Z) \oplus M'_1$

SCA Countermeasures

Masking Scheme for first order

Algo Firs-Order Masking Pre-processing

INPUT(S) : A table representation of the function S, an input mask M_1 and an output mask M_1'

OUTPUT(S) : The table representation of the function $X \mapsto S(X \oplus M_1) \oplus M_1'$

1: **for** $x = 0$ to $2^n - 1$ **do**

2: $\quad T[x \oplus M_1] \leftarrow T[x] \oplus M_1'$

3: **return** T

Algo Firs-Order Masking of an s-box processing

INPUT(S) : A masked input $Z + M_1$ (*e.g.* $Z = S(M \oplus k)$)

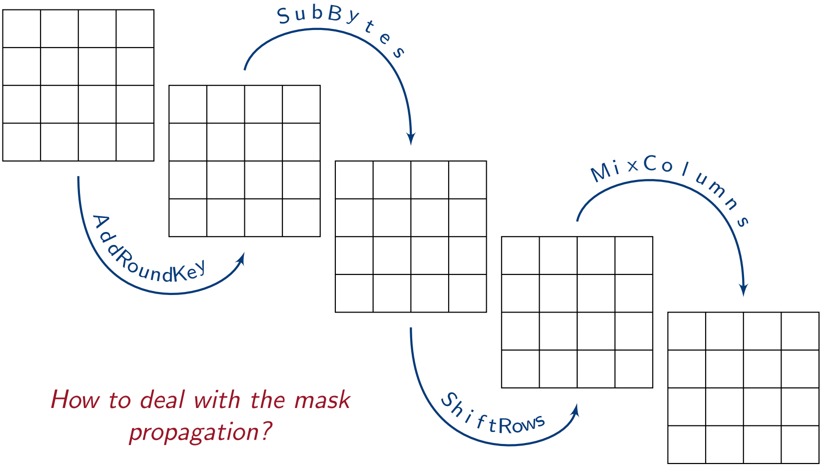
OUTPUT(S) : *The value* $Y = S(Z) \oplus N_1$ *where* M_1' *is a known random value*

1: $Y \leftarrow T^\star(Z)$

2: **return** Y

SCA Countermeasures

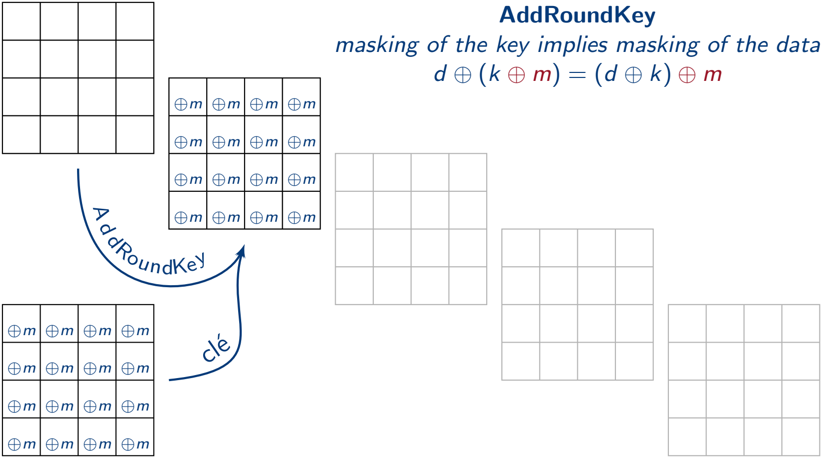
Illustration with a software AES Herbst et al., ACNS 2006



How to deal with the mask propagation?

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006



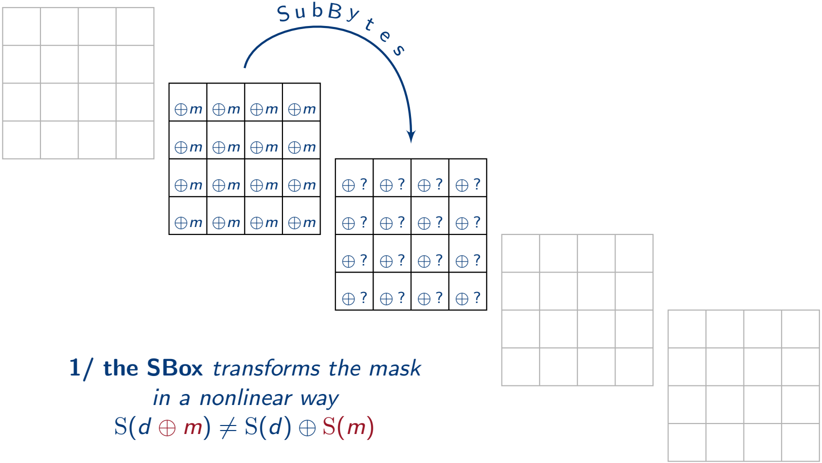
AddRoundKey

masking of the key implies masking of the data

$$d \oplus (k \oplus m) = (d \oplus k) \oplus m$$

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006



1/ the SBox *transforms the mask in a nonlinear way*

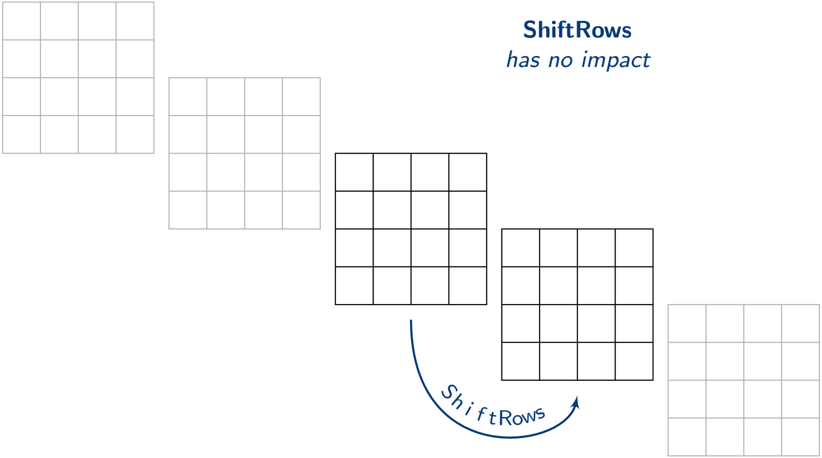
$$\mathrm{S}(d \oplus m) \neq \mathrm{S}(d) \oplus \mathrm{S}(m)$$

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006

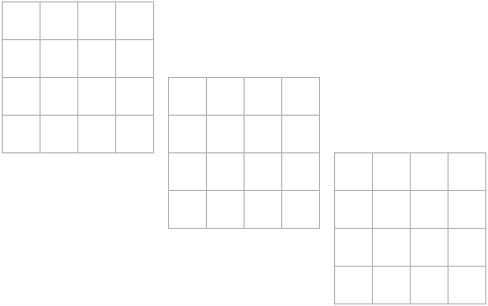
ShiftRows

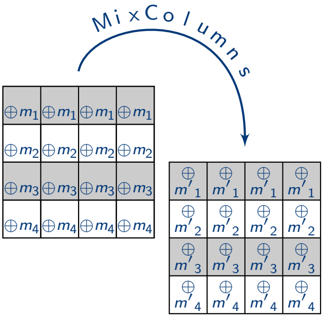
has no impact



SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006





MixColumns

performs a linear combination of the col. elts

$\rightarrow$ *requires at least 2 masks*

$\rightarrow$ *better (perfs) to use 4 masks, one per line*

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006

To sum-up

1. Generate 6 masks
   - m et m' will be resp. the input and output mask of the **sbox**
   - m_1, m_2, m_3 et m_4 will be used for **MixColumns**
2. Pre-compte a new **sbox** $\mathrm{S}_{m,m'}$ s.t.

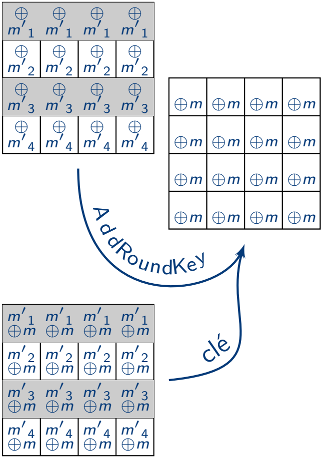
$$\mathrm{S}_{m,m'}(d \oplus m) = \mathrm{S}(d) \oplus m' \quad \textbf{(costly in RAM \ldots)}$$

3. At each MixColumns processing, apply MixColumns to the masked data and to (m_1, m_2, m_3, m_4) s.t.

$$(m'_1, m'_2, m'_3, m'_4) = \mathrm{MixColumns}(m_1, m_2, m_3, m_4)$$

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006

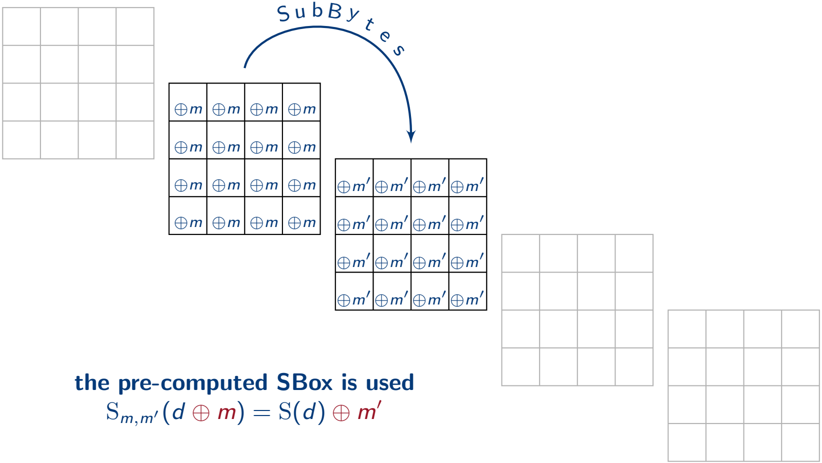


AddRoundKey

- each line of the sate is masked with (m'_1, m'_2, m'_3, m'_4)
- each line of the key state is masked with $(m'_1 \oplus m, m'_2 \oplus m, m'_3 \oplus m, m'_4 \oplus m)$

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006



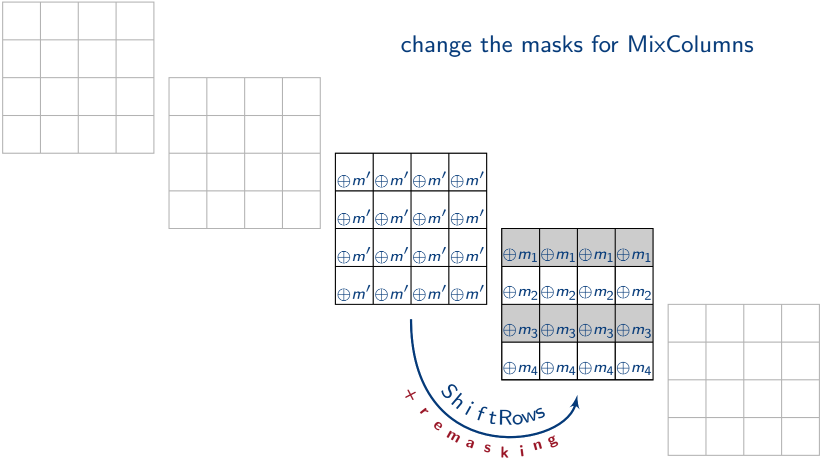
the pre-computed SBox is used

$$\mathrm{S}_{m,m'}(d \oplus m) = \mathrm{S}(d) \oplus m'$$

SCA Countermeasures

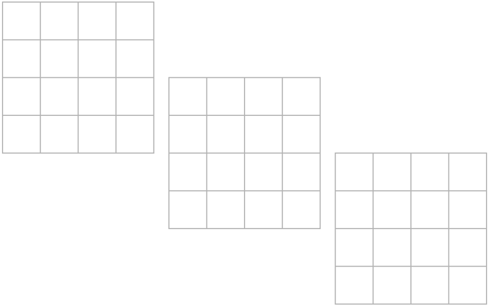
Illustration with a software AES Herbst et al., ACNS 2006

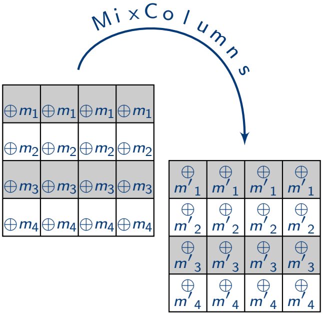
change the masks for MixColumns



SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006



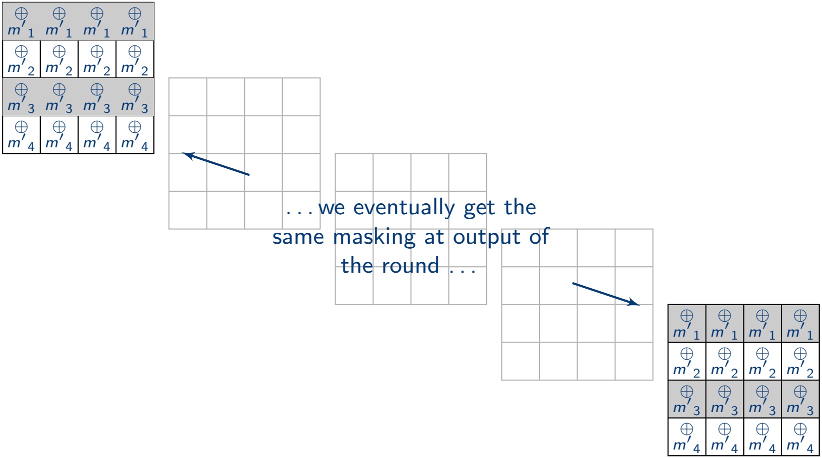


Such that

$$\begin{pmatrix} m'_1 \\ m'_2 \\ m'_3 \\ m'_4 \end{pmatrix} = \textbf{MixColumns} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \end{pmatrix}$$

SCA Countermeasures

Illustration with a software AES Herbst et al., ACNS 2006



. . . we eventually get the same masking at output of the round . . .

SCA Countermeasures

Masking Schemes for first order: other proposals...

- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

- **Multiplicative Masking.** Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- **Affine Masking.** von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- **Modular Additive Masking.** Coron, CHES 1999: $M_0 + Z \mod n$.
- **Homographic Masking.** Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- **Leakage squeeezing** Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).

Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

SCA Countermeasures

Masking Schemes for first order: other proposals...

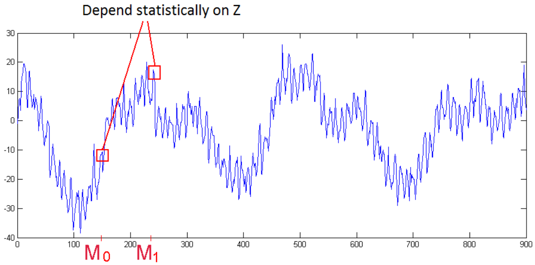
- Multiplicative Masking. Gollic et al at CHES 2002 or Genelle et al at ACNS 2010: $M_0 \times Z$ with $M_0 \neq 0$.
- Affine Masking. von Willich at IMAI 2001 or Fumarolli et al at SCA 2010: $M_0 \times Z + M_1$ with $M_0 \neq 0$.
- Modular Additive Masking. Coron, CHES 1999: $M_0 + Z \mod n$.
- Homographic Masking. Courtois and Goubin, ICISC 2005
  - $\frac{M_0 \times Z + M_1}{M_2 \times Z + M_3}$ or ∞ if $Z = -\frac{M_3}{M_2}$ or $\frac{M_0}{M_2}$ if $Z = \infty$.
  - $M_0 \times M_3 \neq M_1 \times M_2$ and Z belongs to $\mathbb{K} \cup \{\infty\}$ where $\mathbb{K}$ is a field.
- Leakage squeeezing Bhasin *et al, Eprint 2013*
  - $Z \oplus M_0$ where M_0 belongs to a Code with high dual distance.

Note: all those masking does not lead to perfect security against first-order SCA (*i.e.* $\Delta_k \neq 0$).
Practical security is however sometimes achieved since the information leakage is significantly reduced (*i.e.* $\Delta_k < \varepsilon$).

Plan

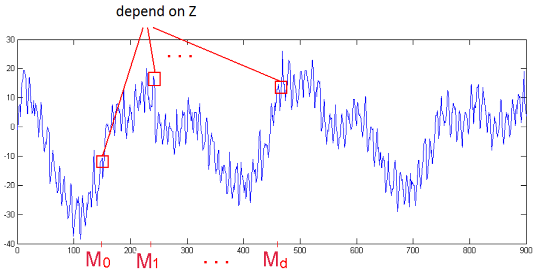
Higher Order Side Channel Attacks
Core Principle

- First Order Masking: $M_0 = Z \oplus M_1$
- $\Longrightarrow$ Second Order SCA:



Higher Order Side Channel Attacks
Core Principle

- Masking of order d: $M_0 = Z \oplus M_1 \oplus \cdots \oplus M_d$
- Attack of order $d+1$:



Higher Order Side Channel Attacks
Introduction

Advanced SCA have been defined to target each CM

- d^{th}-order Masking: HO-SCA
  - [Messerges in his PhD Thesis]
  - Improved latter in Prouff et al at IEEE TC 2009 or in Gierlichs et al at Journal of Cryptology 2011
- t^{th}-order Shuffling: Integrated Attacks
  - [Clavier et al at CHES 2000]
- (d^{th}-order Masking)-and-(t^{th}-order shuffling): Integrated HO-SCA
  - [Tillich et al at ACNS 2007]
  - Improved in [Rivain et al at CHES 2009]

Higher Order Side Channel Attacks
Introduction

Advanced SCA have been defined to target each CM

- d^{th}-order Masking: HO-SCA
  - [Messerges in his PhD Thesis]
  - Improved latter in Prouff et al at IEEE TC 2009 or in Gierlichs et al at Journal of Cryptology 2011
- t^{th}-order Shuffling: Integrated Attacks
  - [Clavier et al at CHES 2000]
- (d^{th}-order Masking)-and-(t^{th}-order shuffling): Integrated HO-SCA
  - [Tillich et al at ACNS 2007]
  - Improved in [Rivain et al at CHES 2009]

Higher Order Side Channel Attacks
Introduction

Advanced SCA have been defined to target each CM

- d^{th}-order Masking: HO-SCA
  - [Messerges in his PhD Thesis]
  - Improved latter in Prouff et al at IEEE TC 2009 or in Gierlichs et al at Journal of Cryptology 2011
- t^{th}-order Shuffling: Integrated Attacks
  - [Clavier et al at CHES 2000]
- (d^{th}-order Masking)-and-(t^{th}-order shuffling): Integrated HO-SCA
  - [Tillich et al at ACNS 2007]
  - Improved in [Rivain et al at CHES 2009]

Higher Order Side Channel Attacks

Introduction

Advanced SCA have been defined to target each CM

- d^{th}-order Masking: HO-SCA
  - [Messerges in his PhD Thesis]
  - Improved latter in Prouff et al at IEEE TC 2009 or in Gierlichs et al at Journal of Cryptology 2011
- t^{th}-order Shuffling: Integrated Attacks
  - [Clavier et al at CHES 2000]
- (d^{th}-order Masking)-and-(t^{th}-order shuffling): Integrated HO-SCA
  - [Tillich et al at ACNS 2007]
  - Improved in [Rivain et al at CHES 2009]

Higher Order Side Channel Attacks
Introduction

Advanced SCA have been defined to target each CM

- d^{th}-order Masking: HO-SCA
  - [Messerges in his PhD Thesis]
  - Improved latter in Prouff et al at IEEE TC 2009 or in Gierlichs et al at Journal of Cryptology 2011
- t^{th}-order Shuffling: Integrated Attacks
  - [Clavier et al at CHES 2000]
- (d^{th}-order Masking)-and-(t^{th}-order shuffling): Integrated HO-SCA
  - [Tillich et al at ACNS 2007]
  - Improved in [Rivain et al at CHES 2009]

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\text{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\text{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $M_{\hat{k}}$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(M_{\hat{k}}, f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $M_{\hat{k}}$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(M_{\hat{k}}, f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Example: if $Z = S(M+k)$ and $M_{\hat{k}} = \mathrm{HW}[S(M+\hat{k})]$, we have ...

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.

In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.

In other cases, the single difference is the function f.

Higher Order Side Channel Attacks

Introduction - General Principle

All the previous SCA follow the same outlines.

1. Input: set of observations for the signals $(L_i)_i$ related to a sensitive datum Z
2. Choose a statistical distintguisher Δ and a pre-processing function f
3. From the observations, estimate $f(L_i)$
4. For every hypothesis $\mathrm{HW}[S(M+\hat{k})]$ on Z, estimate

$$\Delta_{\hat{k}} = |\Delta(\mathrm{HW}[S(M+\hat{k})], f((L_i)_i))| \ .$$

5. Select the hypothesis that maximizes the estimation of $\Delta_{\hat{k}}$.

Note: if the mutual information is used instead of the correlation coefficient, there is not need for a pre-processing function f.
In other cases, the single difference is the function f.

HO-SCA against Higher Order Masking

Illustration with Δ being Pearson' Correlation Coefficient

Context: sensitive variable Z split into $d+1$ shares M_0, ..., M_d
Notation: L_i is the signal related to M_i.

Function f is a normalized product:

$$f(L_0, \cdots, L_d) = \prod_{i=0}^{d} (L_i - \mathbf{E}(L_i)) \ .$$

In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{cst_1}{\left(\sqrt{1 + cst_2 \cdot \sigma^2}\right)^{d+1}} \ .$$

It is denoted by $\rho(d, \sigma)$.

HO-SCA against Higher Order Masking

Illustration with Δ being Pearson' Correlation Coefficient

Context: sensitive variable Z split into $d+1$ shares M_0, ..., M_d

Notation: L_i is the signal related to M_i.

Function f is a normalized product:

$$f(L_0, \cdots, L_d) = \prod_{i=0}^{d} (L_i - \mathbf{E}(L_i)) \ .$$

In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{cst_1}{\left(\sqrt{1 + cst_2 \cdot \sigma^2}\right)^{d+1}} \ .$$

It is denoted by $\rho(d, \sigma)$.

HO-SCA against Higher Order Masking

Illustration with Δ being Pearson' Correlation Coefficient

Context: sensitive variable Z split into $d+1$ shares M_0, ..., M_d
Notation: L_i is the signal related to M_i.

Function f is a normalized product:

$$f(L_0,\cdots,L_d) = \prod_{i=0}^{d}(L_i - \mathbf{E}(L_i)) \ .$$

In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{cst_1}{\left(\sqrt{1 + cst_2 \cdot \sigma^2}\right)^{d+1}} \ .$$

It is denoted by $\rho(d, \sigma)$.

HO-SCA against Higher Order Masking

Illustration with Δ being Pearson' Correlation Coefficient

Context: sensitive variable Z split into $d+1$ shares M_0, ..., M_d

Notation: L_i is the signal related to M_i.

Function f is a normalized product:

$$f(L_0, \cdots, L_d) = \prod_{i=0}^{d} (L_i - \mathbf{E}(L_i)) \ .$$

In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{cst_1}{\left(\sqrt{1 + cst_2 \cdot \sigma^2}\right)^{d+1}} \ .$$

It is denoted by $\rho(d, \sigma)$.

HO-SCA against Higher Order Masking

Illustration with Δ being Pearson' Correlation Coefficient

Context: sensitive variable Z split into $d+1$ shares M_0, ..., M_d
Notation: L_i is the signal related to M_i.

Function f is a normalized product:

$$f(L_0, \cdots, L_d) = \prod_{i=0}^{d} (L_i - \mathbf{E}(L_i)) \ .$$

In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{cst_1}{\left(\sqrt{1 + cst_2 \cdot \sigma^2}\right)^{d+1}} \ .$$

It is denoted by $\rho(d, \sigma)$.

Integrated SCA Against Shuffling

Illustration with Δ being Pearson' Correlation Coefficient

Context: the signal S containing information about Z is randomly spread over t different signals L_1, ..., L_t.

Function f is an Integrated signal:

$$f(L_1, \cdots, L_t) = L_1 + L_2 + \ldots + L_t$$

Note: the sum always contains the term S.
In the Hamming Weight Model, the efficiency satisfies:

$$\boxed{\rho_k = \frac{1}{\sqrt{t}\sqrt{1 + cst_2 \cdot \sigma^2}} .}$$

Integrated SCA Against Shuffling

Illustration with Δ being Pearson' Correlation Coefficient

Context: the signal S containing information about Z is randomly spread over t different signals L_1, ..., L_t.

Function f is an Integrated signal:

$$f(L_1, \cdots, L_t) = L_1 + L_2 + \ldots + L_t$$

Note: the sum always contains the term S.
In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{1}{\sqrt{t}\sqrt{1 + cst_2 \cdot \sigma^2}} \ .$$

Integrated SCA Against Shuffling

Illustration with Δ being Pearson' Correlation Coefficient

Context: the signal S containing information about Z is randomly spread over t different signals L_1, ..., L_t.

Function f is an Integrated signal:

$$f(L_1, \cdots, L_t) = L_1 + L_2 + \ldots + L_t$$

Note: the sum always contains the term S.

In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{1}{\sqrt{t}\sqrt{1 + cst_2 \cdot \sigma^2}} \ .$$

Integrated SCA Against Shuffling

Illustration with Δ being Pearson' Correlation Coefficient

Context: the signal S containing information about Z is randomly spread over t different signals L_1, ..., L_t.

Function f is an Integrated signal:

$$f(L_1, \cdots, L_t) = L_1 + L_2 + \ldots + L_t$$

Note: the sum always contains the term S.
In the Hamming Weight Model, the efficiency satisfies:

$$\rho_k = \frac{1}{\sqrt{t}\sqrt{1 + cst_2 \cdot \sigma^2}} \ .$$

Part IV

Deafeating HOSCA and Proven Security

Outline

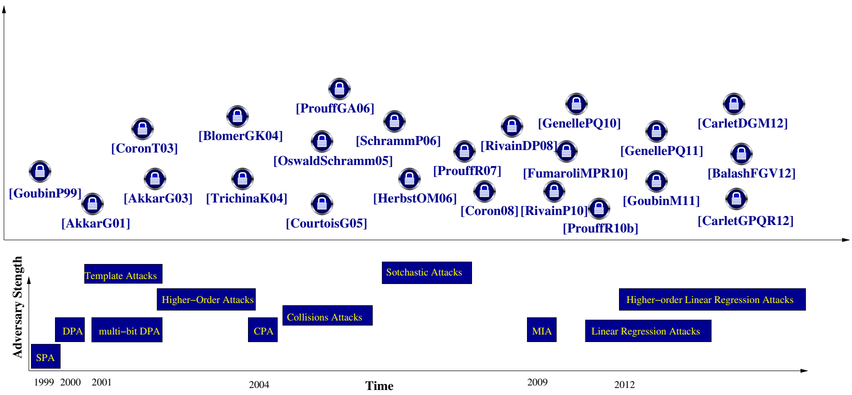
Plan

Need for security proofs?

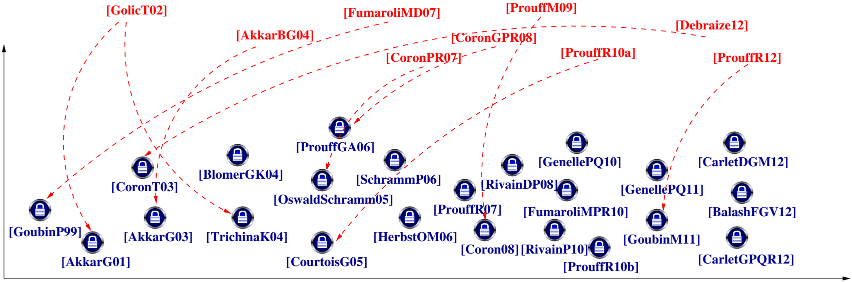
A brief history of countermeasures and attacks

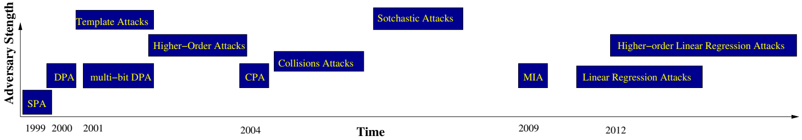
Need for security proofs?

A brief history of countermeasures and attacks



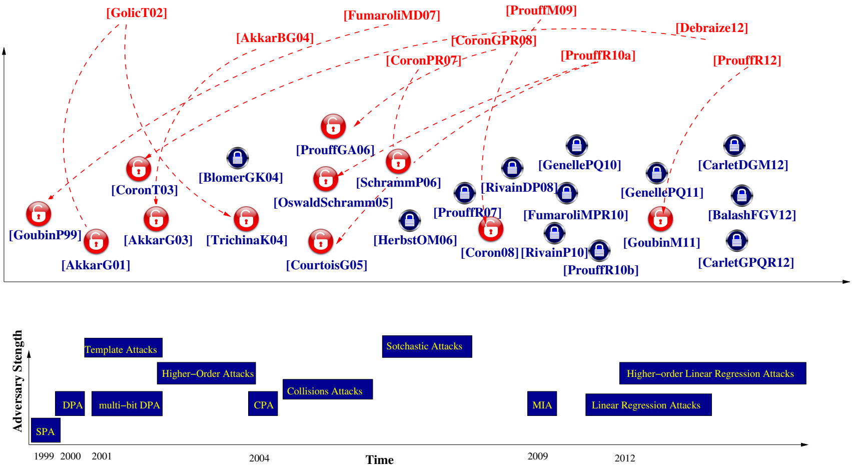
Need for security proofs?
A brief history of countermeasures and attacks





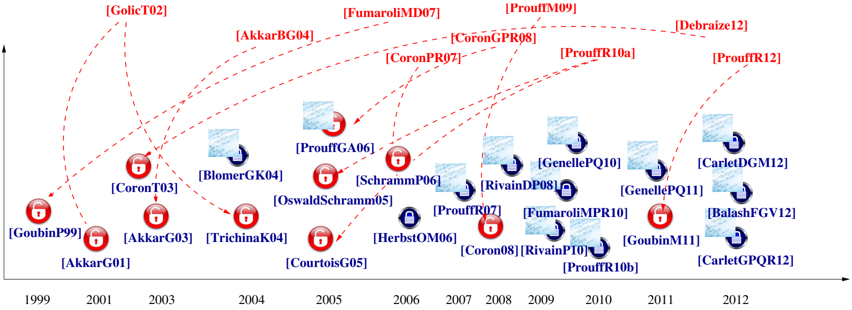
Need for security proofs?

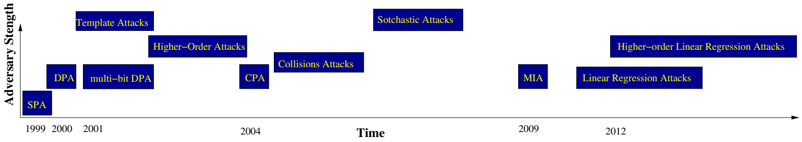
A brief history of countermeasures and attacks



Need for security proofs?

A brief history of countermeasures and attacks





Which security guaranty?

Provable security for embedded systems. Two main approaches...

First approach consists in designing cryptosystems that can be proved secure for some leakage models.

- Recent interest from the crypto theory community (start with DziembowskiPietrzak2007).
- Proofs are given for some leakage models:
    - Bounded Retrieval Model (BRM): the overall sensitive leakage is bounded.
    - (coutinuous) Leakage-resilient cryptography (LRC): the leakage is limited for each invocation only.
- BRM primitives are insecure against DPA and its practical relevence is still under discussion.
- LRC primitives aims at DPA-security
    - Based on re-keying techniques
    - The kind of adversary catched by those models is too strong, which strongly impacts the efficiency.

Which security guaranty?

Provable security for embedded systems. Two main approaches...

First approach consists in designing cryptosystems that can be proved secure for some leakage models.

- Recent interest from the crypto theory community (start with DziembowskiPietrzak2007).
- Proofs are given for some leakage models:
  - Bounded Retrieval Model (BRM): the overall sensitive leakage is bounded.
  - (coutinuous) Leakage-resilient cryptography (LRC): the leakage is limited for each invocation only.
- BRM primitives are insecure against DPA and its practical relevence is still under discussion.
- LRC primitives aims at DPA-security
  - Based on re-keying techniques
  - The kind of adversary catched by those models is too strong, which strongly impacts the efficiency.

Which security guaranty?

Provable security for embedded systems. Two main approaches...

First approach consists in designing cryptosystems that can be proved secure for some leakage models.

- Recent interest from the crypto theory community (start with DziembowskiPietrzak2007).
- Proofs are given for some leakage models:
  - Bounded Retrieval Model (BRM): the overall sensitive leakage is bounded.
  - (coutinuous) Leakage-resilient cryptography (LRC): the leakage is limited for each invocation only.
- BRM primitives are insecure against DPA and its practical relevence is still under discussion.
- LRC primitives aims at DPA-security
  - Based on re-keying techniques
  - The kind of adversary catched by those models is too strong, which strongly impacts the efficiency.

Which security guaranty?

Provable security for embedded systems. Two main approaches...

First approach consists in designing cryptosystems that can be proved secure for some leakage models.

- Recent interest from the crypto theory community (start with DziembowskiPietrzak2007).
- Proofs are given for some leakage models:
  - Bounded Retrieval Model (BRM): the overall sensitive leakage is bounded.
  - (coutinuous) Leakage-resilient cryptography (LRC): the leakage is limited for each invocation only.
- BRM primitives are insecure against DPA and its practical relevence is still under discussion.
- LRC primitives aims at DPA-security
  - Based on re-keying techniques
  - The kind of adversary catched by those models is too strong, which strongly impacts the efficiency.

Which security guaranty?

Provable security for embedded systems. Two main approaches...

First approach consists in designing cryptosystems that can be proved secure for some leakage models.

- Recent interest from the crypto theory community (start with DziembowskiPietrzak2007).
- Proofs are given for some leakage models:
  - Bounded Retrieval Model (BRM): the overall sensitive leakage is bounded.
  - (coutinuous) Leakage-resilient cryptography (LRC): the leakage is limited for each invocation only.
- BRM primitives are insecure against DPA and its practical relevence is still under discussion.
- LRC primitives aims at DPA-security
  - Based on re-keying techniques
  - The kind of adversary catched by those models is too strong, which strongly impacts the efficiency.

Which security guaranty?

Provable security for embedded systems. Two main approaches...

First approach consists in designing cryptosystems that can be proved secure for some leakage models.

- Recent interest from the crypto theory community (start with DziembowskiPietrzak2007).
- Proofs are given for some leakage models:
  - Bounded Retrieval Model (BRM): the overall sensitive leakage is bounded.
  - (coutinuous) Leakage-resilient cryptography (LRC): the leakage is limited for each invocation only.
- BRM primitives are insecure against DPA and its practical relevence is still under discussion.
- LRC primitives aims at DPA-security
  - Based on re-keying techniques
  - The kind of adversary catched by those models is too strong, which strongly impacts the efficiency.

Which security guaranty?

Provable security for embedded systems (second approach)

Second approach consists in securing the implementation using secret sharing techniques.

- First Ideas in GoubinPatarin99 and ChariJutlaRaoRohatgi99.
- Soundness based on the following remark:

- Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
- Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
- Number q of leakage samples to test $((L_i)_i | x = 0) \stackrel{?}{=} ((L_i)_i | x = 1)$:

$$q \geq O(1)\sigma^d$$

- Until now, two options exist to prove the security:
  - the probing Adversary model
  - the Information Bounded model.

Which security guaranty?

Provable security for embedded systems (second approach)

Second approach consists in securing the implementation using secret sharing techniques.

- First Ideas in GoubinPatarin99 and CharіJutlaRaoRohatgi99.
- Soundness based on the following remark:

- Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
- Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
- Number q of leakage samples to test $\left((L_i)_i | x = 0\right) \stackrel{?}{=} \left((L_i)_i | x = 1\right)$:

$$q \geq O(1)\sigma^d$$

- Until now, two options exist to prove the security:
  - the probing Adversary model
  - the Information Bounded model.

Which security guaranty?

Provable security for embedded systems (second approach)

Second approach consists in securing the implementation using secret sharing techniques.

- First Ideas in GoubinPatarin99 and CharIJutlaRaoRohatgi99.
- Soundness based on the following remark:

- Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
- Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
- Number q of leakage samples to test $\left((L_i)_i | x = 0\right) \stackrel{?}{=} \left((L_i)_i | x = 1\right)$:

$$q \geq O(1)\sigma^d$$

- Until now, two options exist to prove the security:
  - the probing Adversary model
  - the Information Bounded model.

Which security guaranty?

Provable security for embedded systems (second approach)

Second approach consists in securing the implementation using secret sharing techniques.

- First Ideas in GoubinPatarin99 and CharìJutlaRaoRohatgi99.
- Soundness based on the following remark:

- Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
- Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
- Number q of leakage samples to test $\left((L_i)_i | x = 0\right) \stackrel{?}{=} \left((L_i)_i | x = 1\right)$:

$$q \geq O(1)\sigma^d$$

- Until now, two options exist to prove the security:
  - the probing Adversary model
  - the Information Bounded model.

Which security guaranty?

Provable security for embedded systems (second approach)

Second approach consists in securing the implementation using secret sharing techniques.

- First Ideas in GoubinPatarin99 and ChariJutlaRaoRohatgi99.
- Soundness based on the following remark:

- Bit x masked $\mapsto x_0, x_1, \ldots, x_d$
- Leakage : $L_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
- Number q of leakage samples to test $\left((L_i)_i | x = 0\right) \stackrel{?}{=} \left((L_i)_i | x = 1\right)$:

$$q \geq O(1)\sigma^d$$

- Until now, two options exist to prove the security:
  - the probing Adversary model
  - the Information Bounded model.

Probing Adversary Model

IshaiSahaiWagner, CRYPTO 2003

- A d^{th}-order probing adversary is allowed to observe **at most** d intermediate results during the overall algorithm processing.
  - Hardware interpretation: d is the maximum of wires observed in the circuit.
  - Software interpretation: d is the maximum of different timings during the processing.
- d^{th}-order probing adversary = d^{th}-order SCA as introduced in Messerges99.
- Countermeasures proved to be secure against a d^{th}-order probing adv.:
  - $d = 1$: KocherJaffeJune99, BlömerGuajardoKrummel04, ProuffRivain07.
  - $d = 2$: RivainDottaxProuff08.
  - $d \geq 1$: IshaiSahaiWagner03, ProuffRoche11, GenelleProuffQuisquater11, CarletGoubinProuffQuisquaterRivain12.

Probing Adversary Model

IshaiSahaiWagner, CRYPTO 2003

- A d^{th}-order probing adversary is allowed to observe **at most** d intermediate results during the overall algorithm processing.
  - Hardware interpretation: d is the maximum of wires observed in the circuit.
  - Software interpretation: d is the maximum of different timings during the processing.
- d^{th}-order probing adversary $=$ d^{th}-order SCA as introduced in Messerges99.
- Countermeasures proved to be secure against a d^{th}-order probing adv.:
  - $d = 1$: KocherJaffeJune99, BlömerGuajardoKrummel04, ProuffRivain07.
  - $d = 2$: RivainDottaxProuff08.
  - $d \geq 1$: IshaiSahaiWagner03, ProuffRoche11, GenelleProuffQuisquater11, CarletGoubinProuffQuisquaterRivain12.

Probing Adversary Model

IshaiSahaiWagner, CRYPTO 2003

- A d^{th}-order probing adversary is allowed to observe **at most** d intermediate results during the overall algorithm processing.
  - Hardware interpretation: d is the maximum of wires observed in the circuit.
  - Software interpretation: d is the maximum of different timings during the processing.
- d^{th}-order probing adversary = d^{th}-order SCA as introduced in Messerges99.
- Countermeasures proved to be secure against a d^{th}-order probing adv.:
  - $d = 1$: KocherJaffeJune99, BlömerGuajardoKrummel04, ProuffRivain07.
  - $d = 2$: RivainDottaxProuff08.
  - $d \geq 1$: IshaiSahaiWagner03, ProuffRoche11, GenelleProuffQuisquater11, CarletGoubinProuffQuisquaterRivain12.

Probing Adversary Model

IshaiSahaiWagner, CRYPTO 2003

- A d^{th}-order probing adversary is allowed to observe **at most** d intermediate results during the overall algorithm processing.
  - Hardware interpretation: d is the maximum of wires observed in the circuit.
  - Software interpretation: d is the maximum of different timings during the processing.
- d^{th}-order probing adversary $=$ d^{th}-order SCA as introduced in Messerges99.
- Countermeasures proved to be secure against a d^{th}-order probing adv.:
  - $d = 1$: KocherJaffeJune99, BlömerGuajardoKrummel04, ProuffRivain07.
  - $d = 2$: RivainDottaxProuff08.
  - $d \geq 1$: IshaiSahaiWagner03, ProuffRoche11, GenelleProuffQuisquater11, CarletGoubinProuffQuisquaterRivain12.

Probing Adversary Model
Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Probing Adversary Model
Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Probing Adversary Model

Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Probing Adversary Model
Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Probing Adversary Model

Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication*), *it is difficult to apply otherwise!*

Probing Adversary Model
Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Probing Adversary Model
Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Probing Adversary Model
Proofs Outlines

To prove the security of an implementation...

- for $d = 1, 2$: list all the intermediate variables and check that none of them is sensitive.
- for $d \geq 3$: the method above starts is too costly!
- Issue: how to prove that a scheme can be made d^{th}-order secure for any given d?
- Ishai-Sahai-Wagner's approach:
  - Two players: the **Adversary** who can observe any d-tuple of intermediate results and an **Oracle** with no access to the implementation
  - The game: for any d-tuple, prove that the oracle can simulate the adversary's view of the implementation execution.
- Method works well for simple schemes (*e.g. the implementation of a multiplication), it is difficult to apply otherwise!*

Information Bounded Model

Chari *et al, CRYPTO 1999* and Prouff and Rivain, Eurocrypt 2013

- Implementation Model. Micali-Reyzin, TCC 2004

  Implementation = seq. of elem. computations producing a list of interm. results $(Z_i)_i$.

- Leakage Model. The leakage on each Z_i is modelled by a probabilistic function f_i s.t.

  $$\mathrm{MI}(Z_i; f_i(Z_i) \leq O(1/\psi)\ ,$$

  where ψ is a security parameter which only depends on the stochastic noise.
- Security Proof goal: find a deterministic function P s.t. the following bound is as tight as possible:

  $$\mathrm{MI}((X,k); (f_i(Z_i)))_i \leq P(1/\psi)$$

  where X is the plaintext and k is the key.
- Example of scheme benefiting from such a security bound:
  - $\longrightarrow$ Ishai-Sahai-Wagner d^{th}-order Boolean Masking of AES and its extensions Ishai,Sahai and Wagner, CRYPTO 2004 and Rivain-Prouff, CHES 2010.

Information Bounded Model

Chari *et al*, *CRYPTO 1999* and Prouff and Rivain, Eurocrypt 2013

- **Implementation Model.** Micali-Reyzin, TCC 2004

  Implementation = seq. of elem. computations producing a list of interm. results $(Z_i)_i$.

- **Leakage Model.** The leakage on each Z_i is modelled by a probabilistic function f_i s.t.

  $$\mathrm{MI}(Z_i; f_i(Z_i) \leq O(1/\psi)\ ,$$

  where ψ is a security parameter which only depends on the stochastic noise.

- **Security Proof goal:** find a deterministic function P s.t. the following bound is as tight as possible:

  $$\mathrm{MI}((X,k);(f_i(Z_i)))_i \leq P(1/\psi)$$

  where X is the plaintext and k is the key.

- Example of scheme benefiting from such a security bound:
  - $\longrightarrow$ Ishai-Sahai-Wagner d^{th}-order Boolean Masking of AES and its extensions Ishai,Sahai and Wagner, CRYPTO 2004 and Rivain-Prouff, CHES 2010.

Information Bounded Model

Chari *et al, CRYPTO 1999* and Prouff and Rivain, Eurocrypt 2013

- Implementation Model. Micali-Reyzin, TCC 2004

  Implementation = seq. of elem. computations producing a list of interm. results $(Z_i)_i$.

- Leakage Model. The leakage on each Z_i is modelled by a probabilistic function f_i s.t.

$$\mathrm{MI}(Z_i; f_i(Z_i) \leq O(1/\psi) \ ,$$

  where ψ is a security parameter which only depends on the stochastic noise.

- Security Proof goal: find a deterministic function P s.t. the following bound is as tight as possible:

$$\mathrm{MI}((X,k);(f_i(Z_i)))_i \leq P(1/\psi)$$

  where X is the plaintext and k is the key.

- Example of scheme benefiting from such a security bound:
  - $\longrightarrow$ Ishai-Sahai-Wagner d^{th}-order Boolean Masking of AES and its extensions Ishai,Sahai and Wagner, CRYPTO 2004 and Rivain-Prouff, CHES 2010.

Information Bounded Model

Chari *et al, CRYPTO 1999* and Prouff and Rivain, Eurocrypt 2013

- Implementation Model. Micali-Reyzin, TCC 2004

  Implementation = seq. of elem. computations producing a list of interm. results $(Z_i)_i$.

- Leakage Model. The leakage on each Z_i is modelled by a probabilistic function f_i s.t.

$$\mathrm{MI}(Z_i; f_i(Z_i) \leq O(1/\psi) \ ,$$

  where ψ is a security parameter which only depends on the stochastic noise.

- Security Proof goal: find a deterministic function P s.t. the following bound is as tight as possible:

$$\mathrm{MI}((X,k); (f_i(Z_i)))_i \leq P(1/\psi)$$

  where X is the plaintext and k is the key.

- Example of scheme benefiting from such a security bound:
  - $\longrightarrow$ Ishai-Sahai-Wagner d^{th}-order Boolean Masking of AES and its extensions Ishai,Sahai and Wagner, CRYPTO 2004 and Rivain-Prouff, CHES 2010.

Information Bounded Model

Chari *et al, CRYPTO 1999* and Prouff and Rivain, Eurocrypt 2013

- Implementation Model. Micali-Reyzin, TCC 2004

  Implementation = seq. of elem. computations producing a list of interm. results $(Z_i)_i$.

- Leakage Model. The leakage on each Z_i is modelled by a probabilistic function f_i s.t.

  $$\mathrm{MI}(Z_i; f_i(Z_i) \leq O(1/\psi) \ ,$$

  where ψ is a security parameter which only depends on the stochastic noise.

- Security Proof goal: find a deterministic function P s.t. the following bound is as tight as possible:

  $$\mathrm{MI}((X,k); (f_i(Z_i)))_i \leq P(1/\psi)$$

  where X is the plaintext and k is the key.

- Example of scheme benefiting from such a security bound:
  - $\longrightarrow$ Ishai-Sahai-Wagner d^{th}-order Boolean Masking of AES and its extensions Ishai,Sahai and Wagner, CRYPTO 2004 and Rivain-Prouff, CHES 2010.

Plan

Higher-Order Masking Schemes

Achieving security in the probing adversary model

Definition

A *dth-order masking scheme* for an encryption algorithm $c \leftarrow \mathcal{E}(m, k)$ is an algorithm

$$(c_0, c_1, \ldots, c_d) \leftarrow \mathcal{E}'((m_0, m_1, \ldots, m_d), (k_0, k_1, \ldots, k_d))$$

- Completeness: there exists R s.t.:

$$R(c_0, \cdots, c_d) = \mathcal{E}(m, k)$$

- Security: $\forall\{iv_1, iv_2, \ldots, iv_d\} \subseteq \{\text{intermediate var. of } \mathcal{E}'\}$:

$$\Pr(k \mid iv_1, iv_2, \ldots, iv_d) = \Pr(k)$$

For SPN (*eg.* DES, AES) the main issue is masking the S-box.

Higher-Order Masking Schemes

Achieving security in the probing adversary model

Definition

A *dth-order masking scheme* for an encryption algorithm $c \leftarrow \mathcal{E}(m, k)$ is an algorithm

$$(c_0, c_1, \ldots, c_d) \leftarrow \mathcal{E}'((m_0, m_1, \ldots, m_d), (k_0, k_1, \ldots, k_d))$$

- Completeness: there exists R s.t.:

$$R(c_0, \cdots, c_d) = \mathcal{E}(m, k)$$

- Security: $\forall \{iv_1, iv_2, \ldots, iv_d\} \subseteq \{\text{intermediate var. of } \mathcal{E}'\}$:

$$\Pr(k \mid iv_1, iv_2, \ldots, iv_d) = \Pr(k)$$

For SPN (*eg.* DES, AES) the main issue is masking the S-box.

Higher-Order Masking Schemes

Achieving security in the probing adversary model

Definition

A *dth-order masking scheme* for an encryption algorithm $c \leftarrow \mathcal{E}(m, k)$ is an algorithm

$$(c_0, c_1, \ldots, c_d) \leftarrow \mathcal{E}'((m_0, m_1, \ldots, m_d), (k_0, k_1, \ldots, k_d))$$

- Completeness: there exists R s.t.:

$$R(c_0, \cdots, c_d) = \mathcal{E}(m, k)$$

- Security: $\forall \{iv_1, iv_2, \ldots, iv_d\} \subseteq \{\text{intermediate var. of } \mathcal{E}'\}$:

$$\Pr(k \mid iv_1, iv_2, \ldots, iv_d) = \Pr(k)$$

For SPN (*eg.* DES, AES) the main issue is masking the S-box.

Higher-Order Masking Schemes

Achieving security in the probing adversary model

Definition

A *dth-order masking scheme* for an encryption algorithm $c \leftarrow \mathcal{E}(m, k)$ is an algorithm

$$(c_0, c_1, \ldots, c_d) \leftarrow \mathcal{E}'((m_0, m_1, \ldots, m_d), (k_0, k_1, \ldots, k_d))$$

- Completeness: there exists R s.t.:

$$R(c_0, \cdots, c_d) = \mathcal{E}(m, k)$$

- Security: $\forall \{iv_1, iv_2, \ldots, iv_d\} \subseteq \{\text{intermediate var. of } \mathcal{E}'\}$:

$$\Pr(k \mid iv_1, iv_2, \ldots, iv_d) = \Pr(k)$$

For SPN (*eg.* DES, AES) the main issue is masking the S-box.

Masking a S-box

Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- `NOT` gate masking (*a.k.a. encoding*):

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \mathit{NOT}\,(x_0) & \mathit{NOT}\,(x_1) & \cdots & \mathit{NOT}\,(x_d) \end{array}$$

- `AND` gate masking: issue since the operations cannot be done on each shares separately.

Masking a S-box
Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- NOT gate masking (*a.k.a. encoding*):

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \mathit{NOT}\,(x_0) & \mathit{NOT}\,(x_1) & \cdots & \mathit{NOT}\,(x_d) \end{array}$$

- AND gate masking: issue since the operations cannot be done on each shares separately.

Masking a S-box

Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- NOT gate masking (*a.k.a. encoding*):

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ NOT\ (x_0) & NOT\ (x_1) & \cdots & NOT\ (x_d) \end{array}$$

- AND gate masking: issue since the operations cannot be done on each shares separately.

Masking a S-box

Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- NOT gate masking (*a.k.a. encoding*):

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ NOT\ (x_0) & NOT\ (x_1) & \cdots & NOT\ (x_d) \end{array}$$

- AND gate masking: issue since the operations cannot be done on each shares separately.

Masking a S-box
Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- NOT gate masking (*a.k.a. encoding*):

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \mathit{NOT}\,(x_0) & \mathit{NOT}\,(x_1) & \cdots & \mathit{NOT}\,(x_d) \end{array}$$

- AND gate masking: issue since the operations cannot be done on each shares separately.

Masking a S-box

Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- `NOT` gate masking (*a.k.a. encoding*):

$$\begin{array}{ccccccccc} & x_0 & & x_1 & & \cdots & & x_d \\ & \downarrow & & \downarrow & & \downarrow & & \downarrow \\ \mathtt{NOT}(x) = & \mathtt{NOT}(x_0) & \oplus & \mathtt{NOT}(x_1) & \oplus & \cdots & \oplus & \mathtt{NOT}(x_d) \end{array}$$

- `AND` gate masking: issue since the operations cannot be done on each shares separately.

Masking a S-box

Original work of Ishai, Sahai and Wagner

Main idea: split the S-box computation into elementary fields operations and protect each of them individually.

- Original idea limited to GF(2), then extended to any field in RivainProuff2010 and FaustRabinReyzinTromerVaikuntanathan2011.
- Data are split by bitwise addition: $x \longrightarrow x_0, \cdots, x_d$ s.t. $x_i \leftarrow \$$, $i > 0$, and $x_0 = \bigoplus_i x_i$.
- `NOT` gate masking (*a.k.a. encoding*):

$$
\begin{array}{cccccccc}
 & x_0 & & x_1 & & \cdots & & x_d \\
 & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
\mathtt{NOT}\,(x) = & \mathtt{NOT}\,(x_0) & \oplus & \mathtt{NOT}\,(x_1) & \oplus & \cdots & \oplus & \mathtt{NOT}\,(x_d)
\end{array}
$$

- `AND` gate masking: issue since the operations cannot be done on each shares separately.

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 & a_0 b_2 \\ a_1 b_0 & a_1 b_1 & a_1 b_2 \\ a_2 b_0 & a_2 b_1 & a_2 b_2 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 & a_0 b_2 \\ 0 & a_1 b_1 & a_1 b_2 \\ 0 & 0 & a_2 b_2 \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 & 0 \\ a_1 b_0 & 0 & 0 \\ a_2 b_0 & a_2 b_1 & 0 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 & a_0 b_2 \\ 0 & a_1 b_1 & a_1 b_2 \\ 0 & 0 & a_2 b_2 \end{pmatrix} \oplus \begin{pmatrix} 0 & a_1 b_0 & a_2 b_0 \\ 0 & 0 & a_2 b_1 \\ 0 & 0 & 0 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 \oplus a_1 b_0 & a_0 b_2 \oplus a_2 b_0 \\ 0 & a_1 b_1 & a_1 b_2 \oplus a_2 b_1 \\ 0 & 0 & a_2 b_2 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 \oplus a_1 b_0 & a_0 b_2 \oplus a_2 b_0 \\ 0 & a_1 b_1 & a_1 b_2 \oplus a_2 b_1 \\ 0 & 0 & a_2 b_2 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 \oplus a_1 b_0 & a_0 b_2 \oplus a_2 b_0 \\ 0 & a_1 b_1 & a_1 b_2 \oplus a_2 b_1 \\ 0 & 0 & a_2 b_2 \end{pmatrix} \oplus \begin{pmatrix} 0 & r_{1,2} & r_{1,3} \\ 0 & 0 & r_{2,3} \\ 0 & 0 & 0 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & a_0 b_1 \oplus a_1 b_0 & a_0 b_2 \oplus a_2 b_0 \\ 0 & a_1 b_1 & a_1 b_2 \oplus a_2 b_1 \\ 0 & 0 & a_2 b_2 \end{pmatrix} \oplus \begin{pmatrix} 0 & r_{1,2} & r_{1,3} \\ r_{1,2} & 0 & r_{2,3} \\ r_{1,3} & r_{2,3} & 0 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 & (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ r_{1,2} & a_1 b_1 & (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ r_{1,3} & r_{2,3} & a_2 b_2 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{pmatrix} a_0 b_0 & (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 & (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ r_{1,2} & a_1 b_1 & (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ r_{1,3} & r_{2,3} & a_2 b_2 \end{pmatrix}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{array}{c} \begin{pmatrix} a_0 b_0 & (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 & (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ r_{1,2} & a_1 b_1 & (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ r_{1,3} & r_{2,3} & a_2 b_2 \end{pmatrix} \\ c_1 \end{array}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an `AND` gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{array}{ccc} \left(\begin{array}{c} a_0 b_0 \\ r_{1,2} \\ r_{1,3} \end{array}\right. & \begin{array}{c} (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 \\ a_1 b_1 \\ r_{2,3} \end{array} & \left.\begin{array}{c} (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ a_2 b_2 \end{array}\right) \\ c_1 & c_2 & \end{array}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{array}{ccc} \left(\begin{array}{c} a_0 b_0 \\ r_{1,2} \\ r_{1,3} \end{array}\right. & \begin{array}{c} (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 \\ a_1 b_1 \\ r_{2,3} \end{array} & \left.\begin{array}{c} (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ a_2 b_2 \end{array}\right) \\ c_1 & c_2 & c_3 \end{array}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{array}{ccc} \multicolumn{3}{c}{\left(\begin{array}{ccc} a_0 b_0 & (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 & (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ r_{1,2} & a_1 b_1 & (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ r_{1,3} & r_{2,3} & a_2 b_2 \end{array}\right)} \\ c_1 & c_2 & c_3 \end{array}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \Big(\bigoplus_i a_i\Big)\Big(\bigoplus_i b_i\Big) = \bigoplus_{i,j} a_i b_j$$

- Illustration of ISW scheme for $d = 2$:

$$\begin{array}{ccc} \left(\begin{array}{ccc} a_0 b_0 & (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 & (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ r_{1,2} & a_1 b_1 & (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ r_{1,3} & r_{2,3} & a_2 b_2 \end{array}\right) \\ \begin{array}{ccc} c_1 & c_2 & c_3 \end{array} \end{array}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Masking an AND gate

- AND gates encoding:
  - Input: $(a_i)_i$, $(b_i)_i$ s.t. $\bigoplus_i a_i = a$, $\bigoplus_i b_i = b$
  - Output: $(c_i)_i$ s.t. $\bigoplus_i c_i = ab$

$$\bigoplus_i c_i = \left(\bigoplus_i a_i\right)\left(\bigoplus_i b_i\right) = \bigoplus_{i,j} a_i b_j$$

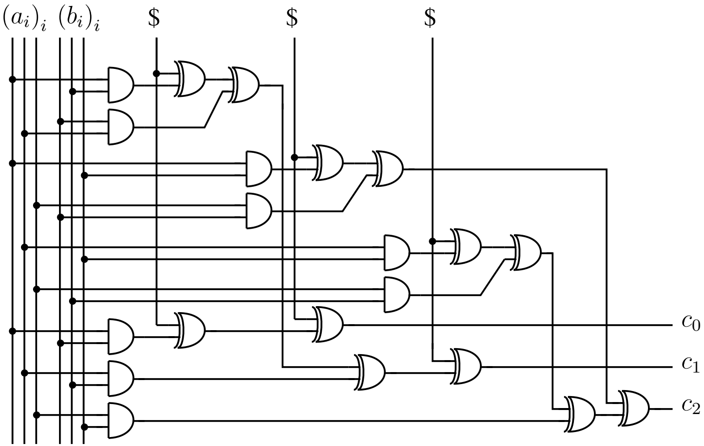
- Illustration of ISW scheme for $d = 2$:

$$\begin{array}{ccc} \left(\begin{array}{c} a_0 b_0 \\ r_{1,2} \\ r_{1,3} \end{array}\right. & \begin{array}{c} (a_0 b_1 \oplus r_{1,2}) \oplus a_1 b_0 \\ a_1 b_1 \\ r_{2,3} \end{array} & \left.\begin{array}{c} (a_0 b_2 \oplus r_{1,3}) \oplus a_2 b_0 \\ (a_1 b_2 \oplus r_{2,3}) \oplus a_2 b_1 \\ a_2 b_2 \end{array}\right) \\ c_1 & c_2 & c_3 \end{array}$$

- Ishai *et al.* prove $(d/2)$th-order security
  - Extended to get dth-order security in RivainProuff10

Ishai-Sahai-Wagner (ISW) Scheme

Example: AND gate for $d = 2$



Ishai-Sahai-Wagner (ISW) Scheme

Practical Issues

- Important area overhead for the masked circuit
  - A wire is encoded by $d+1$ wires
  - One AND gate encoded by
    - $(d+1)^2$ ANDs $+$ $2d(d+1)$ XORs $+$ $d(d+1)/2$ \$
  - Example: AES S-box circuit

	ISW		
No masking	$d=1$	$d=2$	$d=3$
200 gates	500 gates	1.1 Kgates	2 Kgates

- Not suitable for software implementations
- Idea: apply ISW in larger fields

ProuffRivain10,FaustRabinReyzinTromerVaikuntanathan2011.

Ishai-Sahai-Wagner (ISW) Scheme

Practical Issues

- Important area overhead for the masked circuit
  - A wire is encoded by $d+1$ wires
  - One AND gate encoded by
    - $(d+1)^2$ ANDs $+$ $2d(d+1)$ XORs $+$ $d(d+1)/2$ \$
  - Example: AES S-box circuit

	ISW		
No masking	$d=1$	$d=2$	$d=3$
200 gates	500 gates	1.1 Kgates	2 Kgates

- Not suitable for software implementations
- Idea: apply ISW in larger fields ProuffRivain10,FaustRabinReyzinTromerVaikuntanathan2011.

Ishai-Sahai-Wagner (ISW) Scheme

Practical Issues

- Important area overhead for the masked circuit
  - A wire is encoded by $d+1$ wires
  - One AND gate encoded by
    - $(d+1)^2$ ANDs $+$ $2d(d+1)$ XORs $+$ $d(d+1)/2$ \$
  - Example: AES S-box circuit

	ISW		
No masking	$d=1$	$d=2$	$d=3$
200 gates	500 gates	1.1 Kgates	2 Kgates

- Not suitable for software implementations
- Idea: apply ISW in larger fields

ProuffRivain10,FaustRabinReyzinTromerVaikuntanathan2011.

Ishai-Sahai-Wagner (ISW) Scheme

Practical Issues

- Important area overhead for the masked circuit
  - A wire is encoded by $d+1$ wires
  - One AND gate encoded by
    - $(d+1)^2$ ANDs $+$ $2d(d+1)$ XORs $+$ $d(d+1)/2$ \$
  - Example: AES S-box circuit

	ISW		
No masking	$d=1$	$d=2$	$d=3$
200 gates	500 gates	1.1 Kgates	2 Kgates

- Not suitable for software implementations
- Idea: apply ISW in larger fields

ProuffRivain10,FaustRabinReyzinTromerVaikuntanathan2011.

Application to AES in Software

RivainProuff10

- AES S-box: $\mathrm{S} = \mathrm{Af} \circ \mathrm{Exp}$:
  - Af: affine transformation over $\mathrm{GF}(2)^8$
  - $\mathrm{Exp} : x \mapsto x^{254}$ over $\mathrm{GF}(2^8)$

- Masking Af is efficient:

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \mathrm{Af}(x_0) & \mathrm{Af}(x_1) & \cdots & \mathrm{Af}(x_d) \end{array}$$

- Masking Exp:
  - masked square: easy since $x^2 = x_0^2 \oplus x_1^2 \oplus \cdots \oplus x_d^2$
  - masked multiplications : apply ISW on $\mathrm{GF}(2^8)$
  - To minimize the number of multiplications: find a small **addition chain** for 254
    - done in RivainProuff10: only 4 multiplications (and 7 squares).

Application to AES in Software

RivainProuff10

- AES S-box: $\mathrm{S} = \mathrm{Af} \circ \mathrm{Exp}$:
  - Af: affine transformation over $\mathrm{GF}(2)^8$
  - $\mathrm{Exp} : x \mapsto x^{254}$ over $\mathrm{GF}(2^8)$
- Masking Af is efficient:

$$\begin{array}{cccc} x_0 & x_1 & \cdots & x_d \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \mathrm{Af}(x_0) & \mathrm{Af}(x_1) & \cdots & \mathrm{Af}(x_d) \end{array}$$

- Masking Exp:
  - masked square: easy since $x^2 = x_0^2 \oplus x_1^2 \oplus \cdots \oplus x_d^2$
  - masked multiplications : apply ISW on $\mathrm{GF}(2^8)$
  - To minimize the number of multiplications: find a small **addition chain** for 254
    - done in RivainProuff10: only 4 multiplications (and 7 squares).

Application to AES in Software

RivainProuff10

- AES S-box: $\mathrm{S} = \mathrm{Af} \circ \mathrm{Exp}$:
  - Af: affine transformation over $\mathrm{GF}(2)^8$
  - $\mathrm{Exp} : x \mapsto x^{254}$ over $\mathrm{GF}(2^8)$
- Masking Af is efficient:

$$
\begin{array}{cccccccccc}
 & x_0 & & x_1 & & \cdots & & x_d & \\
 & \downarrow & & \downarrow & & \downarrow & & \downarrow & \\
\mathrm{Af}(x) = & \mathrm{Af}(x_0) & \oplus & \mathrm{Af}(x_1) & \oplus & \cdots & \oplus & \mathrm{Af}(x_d) & (\oplus \text{0x63 if } d \text{ odd})
\end{array}
$$

- Masking Exp:
  - masked square: easy since $x^2 = x_0^2 \oplus x_1^2 \oplus \cdots \oplus x_d^2$
  - masked multiplications : apply ISW on $\mathrm{GF}(2^8)$
  - To minimize the number of multiplications: find a small **addition chain** for 254
    - done in RivainProuff10: only 4 multiplications (and 7 squares).

Application to AES in Software

RivainProuff10

- AES S-box: $\mathrm{S} = \mathrm{Af} \circ \mathrm{Exp}$:
  - Af: affine transformation over $\mathrm{GF}(2)^8$
  - $\mathrm{Exp} : x \mapsto x^{254}$ over $\mathrm{GF}(2^8)$
- Masking Af is efficient:

$$\begin{array}{cccccccccc} & x_0 & & x_1 & & \cdots & & x_d & \\ & \downarrow & & \downarrow & & \downarrow & & \downarrow & \\ \mathrm{Af}(x) = & \mathrm{Af}(x_0) & \oplus & \mathrm{Af}(x_1) & \oplus & \cdots & \oplus & \mathrm{Af}(x_d) & (\oplus 0\mathrm{x}63 \text{ if } d \text{ odd}) \end{array}$$

- Masking Exp:
  - masked square: easy since $x^2 = x_0^2 \oplus x_1^2 \oplus \cdots \oplus x_d^2$
  - masked multiplications : apply ISW on $\mathrm{GF}(2^8)$
  - To minimize the number of multiplications: find a small **addition chain** for 254
    - done in RivainProuff10: only 4 multiplications (and 7 squares).

Application to AES in Software

RivainProuff10

- AES S-box: $\mathrm{S} = \mathrm{Af} \circ \mathrm{Exp}$:
  - Af: affine transformation over $\mathrm{GF}(2)^8$
  - $\mathrm{Exp} : x \mapsto x^{254}$ over $\mathrm{GF}(2^8)$
- Masking Af is efficient:

$$\begin{array}{cccccccccc} & x_0 & & x_1 & & \cdots & & x_d & \\ & \downarrow & & \downarrow & & \downarrow & & \downarrow & \\ \mathrm{Af}(x) = & \mathrm{Af}(x_0) & \oplus & \mathrm{Af}(x_1) & \oplus & \cdots & \oplus & \mathrm{Af}(x_d) & (\oplus \text{0x63 if } d \text{ odd}) \end{array}$$

- Masking Exp:
  - masked square: easy since $x^2 = x_0^2 \oplus x_1^2 \oplus \cdots \oplus x_d^2$
  - masked multiplications : apply ISW on $\mathrm{GF}(2^8)$
  - To minimize the number of multiplications: find a small **addition chain** for 254
    - done in RivainProuff10: only 4 multiplications (and 7 squares).

Extension to Any S-Box

CarletGoubinProuffQuisquaterRivain12

- Write the s-box $\mathrm{S} : \{0,1\}^n \to \{0,1\}^m$ as a polynomial function over $\mathrm{GF}(2^n)$:

$$\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$$

- Securely evaluate this polynomial on the shared input $(x_i)_i$

Extension to Any S-Box

CarletGoubinProuffQuisquaterRivain12

- Write the s-box $\mathrm{S} : \{0,1\}^n \to \{0,1\}^m$ as a polynomial function over $\mathrm{GF}(2^n)$:

$$\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$$

- Securely evaluate this polynomial on the shared input $(x_i)_i$

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
  1. additions
  2. scalar multiplications (*i.e.* by constants)
  3. squares
  4. regular multiplications $\Rightarrow$ *nonlinear multiplications*
- Masking is efficient for the 3 first kinds
  - $(x + y) = (x_0 + y_0) + (x_1 + y_1) + \cdots + (x_d + y_d)$
  - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
  - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$
- nonlinear multiplication masked with ISW scheme

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
  1. additions
  2. scalar multiplications (*i.e.* by constants)
  3. squares
  4. regular multiplications $\Rightarrow$ *nonlinear multiplications*
- Masking is efficient for the 3 first kinds
  - $(x + y) = (x_0 + y_0) + (x_1 + y_1) + \cdots + (x_d + y_d)$
  - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
  - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$
- nonlinear multiplication masked with ISW scheme

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
  1. additions
  2. scalar multiplications (*i.e.* by constants)
  3. squares
  4. regular multiplications $\Rightarrow$ *nonlinear multiplications*
- Masking is efficient for the 3 first kinds
  - $(x + y) = (x_0 + y_0) + (x_1 + y_1) + \cdots + (x_d + y_d)$
  - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
  - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$
- nonlinear multiplication masked with ISW scheme

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
  1. additions
  2. scalar multiplications (*i.e.* by constants)
  3. squares
  4. regular multiplications $\Rightarrow$ *nonlinear multiplications*
- Masking is efficient for the 3 first kinds
  - $(x+y) = (x_0+y_0) + (x_1+y_1) + \cdots + (x_d+y_d)$
  - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
  - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$
- nonlinear multiplication masked with ISW scheme

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
  1. additions
  2. scalar multiplications (*i.e.* by constants)
  3. squares
  4. regular multiplications $\Rightarrow$ *nonlinear multiplications*

- Masking is efficient for the 3 first kinds
  - $(x + y) = (x_0 + y_0) + (x_1 + y_1) + \cdots + (x_d + y_d)$
  - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
  - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$

- nonlinear multiplication masked with ISW scheme

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
  1. additions
  2. scalar multiplications (*i.e.* by constants)
  3. squares
  4. regular multiplications $\Rightarrow$ *nonlinear multiplications*

- Masking is efficient for the 3 first kinds
  - $(x + y) = (x_0 + y_0) + (x_1 + y_1) + \cdots + (x_d + y_d)$
  - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
  - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$

- nonlinear multiplication masked with ISW scheme

General Method

- Four kinds of operations over $\mathrm{GF}(2^n)$:
    1. additions
    2. scalar multiplications (*i.e.* by constants)
    3. squares
    4. regular multiplications $\Rightarrow$ *nonlinear multiplications*
- Masking is efficient for the 3 first kinds
    - $(x+y) = (x_0+y_0) + (x_1+y_1) + \cdots + (x_d+y_d)$
    - $x^2 = x_0^2 + x_1^2 + \cdots + x_d^2$
    - $a \cdot x = a \cdot x_0 + a \cdot x_1 + \cdots + a \cdot x_d$
- nonlinear multiplication masked with ISW scheme

Masking Complexity

- Masking an operation $\in$ {addition, square, scalar mult.}
  $\Rightarrow$ $d+1$ operations
- Masking a nonlinear multiplication
  $\Rightarrow$ $(d+1)^2$ mult. $+$ $2d(d+1)$ add. $+$ $nd(d+1)/2$ random bits

Definition

The *masking complexity* of a (n,m) s-box is the minimal number of nonlinear multiplications required to evaluate its polynomial representation over $\mathrm{GF}(2^n)$.

Masking Complexity

- Masking an operation $\in$ {addition, square, scalar mult.}
  $\Rightarrow$ $d+1$ operations
- Masking a nonlinear multiplication
  $\Rightarrow$ $(d+1)^2$ mult. $+$ $2d(d+1)$ add. $+$ $nd(d+1)/2$ random bits

Definition

The *masking complexity* of a (n,m) s-box is the minimal number of nonlinear multiplications required to evaluate its polynomial representation over $\mathrm{GF}(2^n)$.

Straightforward schemes

- Goal: evaluate $\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$
- first solution :
  - compute $\mathrm{S}(x) = a_0 + x(a_1 + x(a_2 + x(\cdots)))$
  - $\Rightarrow 2^n - 2$ nonlinear multiplications
- second solution :
  - first compute x^2, x^3, x^4, ... then evaluate $\mathrm{S}(x)$
  - $x^j \leftarrow (x^{j/2})^2$ when j even, $x^j \leftarrow x \cdot x^{j-1}$ when j odd
  - $\Rightarrow 2^{n-1} - 1$ nonlinear multiplications
- But we can do much better Carlet, Goubin, Prouff, Quisquater, Prouff, FSE 2013 and Roy and Vivek, CHES 2013!
  - Optimal methods for power functions.
  - Efficient heuristic for the general case: around $2^{n-r-1} + 2^r - 2$ multiplications where 2^r is the degree of the polynomial.

Straightforward schemes

- Goal: evaluate $\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$
- first solution :
  - compute $\mathrm{S}(x) = a_0 + x(a_1 + x(a_2 + x(\cdots)))$
  - $\Rightarrow 2^n - 2$ nonlinear multiplications
- second solution :
  - first compute x^2, x^3, x^4, ... then evaluate $\mathrm{S}(x)$
  - $x^j \leftarrow (x^{j/2})^2$ when j even, $x^j \leftarrow x \cdot x^{j-1}$ when j odd
  - $\Rightarrow 2^{n-1} - 1$ nonlinear multiplications
- But we can do much better Carlet, Goubin, Prouff, Quisquater, Prouff, FSE 2013 and Roy and Vivek, CHES 2013!
  - Optimal methods for power functions.
  - Efficient heuristic for the general case: around $2^{n-r-1} + 2^r - 2$ multiplications where 2^r is the degree of the polynomial.

Straightforward schemes

- Goal: evaluate $\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$
- first solution :
  - compute $\mathrm{S}(x) = a_0 + x(a_1 + x(a_2 + x(\cdots)))$
  - $\Rightarrow 2^n - 2$ nonlinear multiplications
- second solution :
  - first compute x^2, x^3, x^4, ... then evaluate $\mathrm{S}(x)$
  - $x^j \leftarrow (x^{j/2})^2$ when j even, $x^j \leftarrow x \cdot x^{j-1}$ when j odd
  - $\Rightarrow 2^{n-1} - 1$ nonlinear multiplications
- But we can do much better Carlet, Goubin, Prouff, Quisquater, Prouff, FSE 2013 and Roy and Vivek, CHES 2013!
  - Optimal methods for power functions.
  - Efficient heuristic for the general case: around $2^{n-r-1} + 2^r - 2$ multiplications where 2^r is the degree of the polynomial.

Straightforward schemes

- Goal: evaluate $\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$
- first solution :
  - compute $\mathrm{S}(x) = a_0 + x(a_1 + x(a_2 + x(\cdots)))$
  - $\Rightarrow 2^n - 2$ nonlinear multiplications
- second solution :
  - first compute x^2, x^3, x^4, ... then evaluate $\mathrm{S}(x)$
  - $x^j \leftarrow (x^{j/2})^2$ when j even, $x^j \leftarrow x \cdot x^{j-1}$ when j odd
  - $\Rightarrow 2^{n-1} - 1$ nonlinear multiplications
- But we can do much better Carlet, Goubin, Prouff, Quisquater, Prouff, FSE 2013 and Roy and Vivek, CHES 2013!
  - Optimal methods for power functions.
  - Efficient heuristic for the general case: around $2^{n-r-1} + 2^r - 2$ multiplications where 2^r is the degree of the polynomial.

Straightforward schemes

- Goal: evaluate $\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{2^n-1} x^{2^n-1}$
- first solution :
  - compute $\mathrm{S}(x) = a_0 + x(a_1 + x(a_2 + x(\cdots)))$
  - $\Rightarrow 2^n - 2$ nonlinear multiplications
- second solution :
  - first compute x^2, x^3, x^4, ... then evaluate $\mathrm{S}(x)$
  - $x^j \leftarrow (x^{j/2})^2$ when j even, $x^j \leftarrow x \cdot x^{j-1}$ when j odd
  - $\Rightarrow 2^{n-1} - 1$ nonlinear multiplications
- But we can do much better Carlet, Goubin, Prouff, Quisquater, Prouff, FSE 2013 and Roy and Vivek, CHES 2013!
  - Optimal methods for power functions.
  - Efficient heuristic for the general case: around $2^{n-r-1} + 2^r - 2$ multiplications where 2^r is the degree of the polynomial.

Optimal Masking of Power Functions

Problem

For a given $\alpha \in [1; 2^n - 1]$ compute x^α with the least number of nonlinear multiplications.

$$\Leftrightarrow$$

Problem

Find the shortest 2-addition chain for α (modulo $2^n - 1$).

Optimal Masking of Power Functions

Problem

For a given $\alpha \in [1; 2^n - 1]$ compute x^α with the least number of nonlinear multiplications.

$$\Longleftrightarrow$$

Problem

Find the shortest 2-addition chain for α (modulo $2^n - 1$).

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \to x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \to x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \to x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \to x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \to x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \to x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \to x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \to x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \to x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \to x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \to x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \to x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \to x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \to x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \to x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

Optimal Masking of Power Functions

- *Cyclotomic class of* α : $C_\alpha = \{\alpha \cdot 2^j \bmod (2^n - 1);\ j \leq n\}$
- If $\beta \in C_\alpha$ ($\Leftrightarrow C_\beta = C_\alpha$)
  - x^α can be computed from x^β with 0 nonlinear multiplication
  - x^α and x^β have the same masking complexity
- Exhaustive search for best 2-addition chains
  - $x \rightarrow x^2, x^4, x^8, \ldots$ (0 nonlinear multiplications)
  - with 1 nonlinear multiplication
    - $x^3 = x \cdot x^2 \rightarrow x^6, x^{12}, x^{24}, \ldots$
    - $x^5 = x \cdot x^4 \rightarrow x^{10}, x^{20}, x^{40}, \ldots$
    - *etc.*
  - with 2 nonlinear multiplications
    - $x^7 = x^3 \cdot x^4 \rightarrow x^{14}, x^{28}, \ldots$
    - $x^{11} = x^3 \cdot x^8 \rightarrow x^{22}, x^{44}, \ldots$
    - *etc.*

k	Cyclotomic classes in $\mathcal{M}_k^n$
	$n = 4$
0	$C_0 = \{0\},\ C_1 = \{1, 2, 4, 8\}$
1	$C_3 = \{3, 6, 12, 9\},\ C_5 = \{5, 10\}$
2	$C_7 = \{7, 14, 13, 11\}$
	$n = 6$
0	$C_0 = \{0\},\ C_1 = \{1, 2, 4, 8, 16, 32\}$
1	$C_3 = \{3, 6, 12, 24, 48, 33\},\ C_5 = \{5, 10, 20, 40, 17, 34\},\ C_9 = \{9, 18, 36\}$
2	$C_7 = \{7, 14, 28, 56, 49, 35\},\ C_{11} = \{11, 22, 44, 25, 50, 37\},\ C_{13} = \{13, 26, 52, 41, 19, 38\},$ $C_{15} = \{15, 30, 29, 27, 23\},\ C_{21} = \{21, 42\},\ C_{27} = \{27, 54, 45\}$
3	$C_{23} = \{23, 46, 29, 58, 53, 43\},\ C_{31} = \{31, 62, 61, 59, 55, 47\}$
	$n = 8$
0	$C_0 = \{0\},\ C_1 = \{1, 2, 4, 8, 16, 32, 64, 128\}$
1	$C_3 = \{3, 6, 12, 24, 48, 96, 192, 129\},\ C_5 = \{5, 10, 20, 40, 80, 160, 65, 130\},$ $C_9 = \{9, 18, 36, 72, 144, 33, 66, 132\},\ C_{17} = \{17, 34, 68, 136\}$
2	$C_7 = \{7, 14, 28, 56, 112, 224, 193, 131\},\ C_{11} = \{11, 22, 44, 88, 176, 97, 194, 133\},$ $C_{13} = \{13, 26, 52, 104, 208, 161, 67, 134\},\ C_{15} = \{15, 30, 60, 120, 240, 225, 195, 135\},$ $C_{19} = \{19, 38, 76, 152, 49, 98, 196, 137\},\ C_{21} = \{21, 42, 84, 168, 81, 162, 69, 138\},$ $C_{25} = \{25, 50, 100, 200, 145, 35, 70, 140\},\ C_{27} = \{27, 54, 108, 216, 177, 99, 198, 141\},$ $C_{37} = \{37, 74, 148, 41, 82, 164, 73, 146\},\ C_{45} = \{45, 90, 180, 105, 210, 165, 75, 150\},$ $C_{51} = \{51, 102, 204, 153\},\ C_{85} = \{85, 170\}$
3	$C_{23} = \{23, 46, 92, 184, 113, 226, 197, 139\},\ C_{29} = \{29, 58, 116, 232, 209, 163, 71, 142\},$ $C_{31} = \{31, 62, 124, 248, 241, 227, 199, 143\},\ C_{39} = \{39, 78, 156, 57, 114, 228, 201, 147\},$ $C_{43} = \{43, 86, 172, 89, 178, 101, 202, 149\},\ C_{47} = \{47, 94, 188, 121, 242, 229, 203, 151\},$ $C_{53} = \{53, 106, 212, 169, 83, 166, 77, 154\},\ C_{55} = \{55, 110, 220, 185, 115, 230, 205, 155\},$ $C_{59} = \{59, 118, 236, 217, 179, 103, 206, 157\},\ C_{61} = \{61, 122, 244, 233, 211, 167, 79, 158\},$ $C_{63} = \{63, 126, 252, 249, 243, 231, 207, 159\},\ C_{87} = \{87, 174, 93, 186, 117, 234, 213, 171\},$ $C_{91} = \{91, 182, 109, 218, 181, 107, 214, 173\},\ C_{95} = \{95, 190, 125, 250, 245, 235, 215, 175\},$ $C_{111} = \{111, 222, 189, 123, 246, 237, 219, 183\},\ C_{119} = \{119, 238, 221, 187\}$
4	$C_{127} = \{127, 254, 253, 251, 247, 239, 223, 191\}$

Cyclotomic Method

$$
\begin{aligned}
\mathrm{S}(x) &= a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 \\
&\quad + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11} + a_{12}x^{12} + \ldots \\
&= a_0 + a_1x + a_2x^2 + a_4x^4 + a_8x^8 + \ldots \\
&\quad + a_3x^3 + a_6x^6 + a_{12}x^{12} + a_{24}x^{24} + \ldots \\
&\quad + a_5x^5 + a_{10}x^{10} + a_{20}x^{20} + a_{40}x^{40} + \ldots \\
&\quad + \ldots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \ldots
\end{aligned}
$$

where

- $L_1(X) = a_1X + a_2X^2 + a_4X^4 + a_8X^8 + \ldots$
- $L_3(X) = a_3X + a_6X^2 + a_{12}X^4 + a_{24}X^8 + \ldots$
- $L_5(X) = a_5X + a_{10}X^2 + a_{20}X^4 + a_{40}X^8 + \ldots$
- ...

Cyclotomic Method

$$
\begin{aligned}
S(x) &= a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 \\
&\quad + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11} + a_{12}x^{12} + \dots \\
&= a_0 + a_1x + a_2x^2 + a_4x^4 + a_8x^8 + \dots \\
&\quad + a_3x^3 + a_6x^6 + a_{12}x^{12} + a_{24}x^{24} + \dots \\
&\quad + a_5x^5 + a_{10}x^{10} + a_{20}x^{20} + a_{40}x^{40} + \dots \\
&\quad + \dots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \dots
\end{aligned}
$$

where

- $L_1(X) = a_1X + a_2X^2 + a_4X^4 + a_8X^8 + \dots$
- $L_3(X) = a_3X + a_6X^2 + a_{12}X^4 + a_{24}X^8 + \dots$
- $L_5(X) = a_5X + a_{10}X^2 + a_{20}X^4 + a_{40}X^8 + \dots$
- ...

Cyclotomic Method

$$
\begin{aligned}
\mathrm{S}(x) &= a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \\
&\quad + a_8 x^8 + a_9 x^9 + a_{10} x^{10} + a_{11} x^{11} + a_{12} x^{12} + \ldots \\
&= a_0 + a_1 x + a_2 x^2 + a_4 x^4 + a_8 x^8 + \ldots \\
&\quad + a_3 x^3 + a_6 x^6 + a_{12} x^{12} + a_{24} x^{24} + \ldots \\
&\quad + a_5 x^5 + a_{10} x^{10} + a_{20} x^{20} + a_{40} x^{40} + \ldots \\
&\quad + \ldots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \ldots
\end{aligned}
$$

where

- $L_1(X) = a_1 X + a_2 X^2 + a_4 X^4 + a_8 X^8 + \ldots$
- $L_3(X) = a_3 X + a_6 X^2 + a_{12} X^4 + a_{24} X^8 + \ldots$
- $L_5(X) = a_5 X + a_{10} X^2 + a_{20} X^4 + a_{40} X^8 + \ldots$
- ...

Cyclotomic Method

$$
\begin{aligned}
S(x) &= a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 \\
&\quad + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11} + a_{12}x^{12} + \dots \\
&= a_0 + a_1x + a_2x^2 + a_4x^4 + a_8x^8 + \dots \\
&\quad + a_3x^3 + a_6x^6 + a_{12}x^{12} + a_{24}x^{24} + \dots \\
&\quad + a_5x^5 + a_{10}x^{10} + a_{20}x^{20} + a_{40}x^{40} + \dots \\
&\quad + \dots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \dots
\end{aligned}
$$

where

- $L_1(X) = a_1X + a_2X^2 + a_4X^4 + a_8X^8 + \dots$
- $L_3(X) = a_3X + a_6X^2 + a_{12}X^4 + a_{24}X^8 + \dots$
- $L_5(X) = a_5X + a_{10}X^2 + a_{20}X^4 + a_{40}X^8 + \dots$
- ...

Cyclotomic Method

$$
\begin{aligned}
S(x) &= a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \\
&\quad + a_8 x^8 + a_9 x^9 + a_{10} x^{10} + a_{11} x^{11} + a_{12} x^{12} + \dots \\
&= a_0 + a_1 x + a_2 x^2 + a_4 x^4 + a_8 x^8 + \dots \\
&\quad + a_3 x^3 + a_6 x^6 + a_{12} x^{12} + a_{24} x^{24} + \dots \\
&\quad + a_5 x^5 + a_{10} x^{10} + a_{20} x^{20} + a_{40} x^{40} + \dots \\
&\quad + \dots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \dots
\end{aligned}
$$

where

- $L_1(X) = a_1 X + a_2 X^2 + a_4 X^4 + a_8 X^8 + \dots$
- $L_3(X) = a_3 X + a_6 X^2 + a_{12} X^4 + a_{24} X^8 + \dots$
- $L_5(X) = a_5 X + a_{10} X^2 + a_{20} X^4 + a_{40} X^8 + \dots$
- ...

Cyclotomic Method

$$
\begin{aligned}
S(x) &= a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 + a_6x^6 + a_7x^7 \\
&\quad + a_8x^8 + a_9x^9 + a_{10}x^{10} + a_{11}x^{11} + a_{12}x^{12} + \ldots \\
&= a_0 + a_1x + a_2x^2 + a_4x^4 + a_8x^8 + \ldots \\
&\quad + a_3x^3 + a_6(x^3)^2 + a_{12}(x^3)^4 + a_{24}(x^3)^8 + \ldots \\
&\quad + a_5x^5 + a_{10}(x^5)^2 + a_{20}(x^5)^4 + a_{40}(x^5)^8 + \ldots \\
&\quad + \ldots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \ldots
\end{aligned}
$$

where

- $L_1(X) = a_1X + a_2X^2 + a_4X^4 + a_8X^8 + \ldots$
- $L_3(X) = a_3X + a_6X^2 + a_{12}X^4 + a_{24}X^8 + \ldots$
- $L_5(X) = a_5X + a_{10}X^2 + a_{20}X^4 + a_{40}X^8 + \ldots$
- ...

Cyclotomic Method

$$
\begin{aligned}
S(x) &= a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 \\
&\quad + a_8 x^8 + a_9 x^9 + a_{10} x^{10} + a_{11} x^{11} + a_{12} x^{12} + \dots \\
&= a_0 + a_1 x + a_2 x^2 + a_4 x^4 + a_8 x^8 + \dots \\
&\quad + a_3 x^3 + a_6 (x^3)^2 + a_{12} (x^3)^4 + a_{24} (x^3)^8 + \dots \\
&\quad + a_5 x^5 + a_{10} (x^5)^2 + a_{20} (x^5)^4 + a_{40} (x^5)^8 + \dots \\
&\quad + \dots \\
&= a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + \dots
\end{aligned}
$$

where

- $L_1(X) = a_1 X + a_2 X^2 + a_4 X^4 + a_8 X^8 + \dots$
- $L_3(X) = a_3 X + a_6 X^2 + a_{12} X^4 + a_{24} X^8 + \dots$
- $L_5(X) = a_5 X + a_{10} X^2 + a_{20} X^4 + a_{40} X^8 + \dots$
- ...

Cyclotomic Method

1. Compute one power per cyclotomic class x, x^3, x^5, x^7, ...
2. Evaluate the corresponding linearized polynomials $L_1(x)$, $L_3(x^3)$, $L_5(x^5)$, $L_7(x^7)$, ...
3. Compute the sum $\mathrm{S}(x) = a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + L_7(x^7) + \ldots$

Number of nonlinear multiplication

$$=$$

$$\#\{\textbf{cyclotomic classes}\}\backslash(C_0 \cup C_1)$$

n	3	4	5	6	7	8	9	10
# nlm	1	3	5	11	17	33	53	105

Cyclotomic Method

1. Compute one power per cyclotomic class x, x^3, x^5, x^7, ...
2. Evaluate the corresponding linearized polynomials $L_1(x)$, $L_3(x^3)$, $L_5(x^5)$, $L_7(x^7)$, ...
3. Compute the sum $S(x) = a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + L_7(x^7) + \ldots$

Number of nonlinear multiplication

$=$

$\#\{$**cyclotomic classes**$\}\backslash(C_0 \cup C_1)$

n	3	4	5	6	7	8	9	10
# nlm	1	3	5	11	17	33	53	105

Cyclotomic Method

1. Compute one power per cyclotomic class x, x^3, x^5, x^7, ...
2. Evaluate the corresponding linearized polynomials $L_1(x)$, $L_3(x^3)$, $L_5(x^5)$, $L_7(x^7)$, ...
3. Compute the sum
$$\mathrm{S}(x) = a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + L_7(x^7) + \dots$$

Number of nonlinear multiplication

$$=$$

$$\#\{\textbf{cyclotomic classes}\}\backslash(C_0 \cup C_1)$$

n	3	4	5	6	7	8	9	10
# nlm	1	3	5	11	17	33	53	105

Cyclotomic Method

1. Compute one power per cyclotomic class x, x^3, x^5, x^7, ...
2. Evaluate the corresponding linearized polynomials $L_1(x)$, $L_3(x^3)$, $L_5(x^5)$, $L_7(x^7)$, ...
3. Compute the sum
$$\mathrm{S}(x) = a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + L_7(x^7) + \ldots$$

Number of nonlinear multiplication

$=$

$\#\{$**cyclotomic classes**$\}\backslash(C_0 \cup C_1)$

n	3	4	5	6	7	8	9	10
# nlm	1	3	5	11	17	33	53	105

Cyclotomic Method

1. Compute one power per cyclotomic class x, x^3, x^5, x^7, ...
2. Evaluate the corresponding linearized polynomials $L_1(x)$, $L_3(x^3)$, $L_5(x^5)$, $L_7(x^7)$, ...
3. Compute the sum
$$\mathrm{S}(x) = a_0 + L_1(x) + L_3(x^3) + L_5(x^5) + L_7(x^7) + \ldots$$

Number of nonlinear multiplication

$=$

$\#\{$**cyclotomic classes**$\}\backslash(C_0 \cup C_1)$

n	3	4	5	6	7	8	9	10
# nlm	1	3	5	11	17	33	53	105

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 + a_8 x^8 + a_9 x^9 + a_{10} x^{10} + a_{11} x^{11} + a_{12} x^{12} + \ldots$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7 + a_8 x^8 + a_9 x^9 + a_{10} x^{10} + a_{11} x^{11} + a_{12} x^{12} + \ldots$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$
\begin{aligned}
S(x) \quad = \quad & a_0 + a_2x^2 + a_4x^4 + a_6x^6 + a_8x^8 + \ldots \\
& a_1x + a_3x^3 + a_5x^5 + a_7x^7 + a_9x^9 + \ldots
\end{aligned}
$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \quad \text{nonlinear mult.}$$

Parity-Split Method

$$
\begin{aligned}
\mathrm{S}(x) &= a_0 + a_2x^2 + a_4x^4 + a_6x^6 + a_8x^8 + \ldots \\
&\quad (a_1 + a_3x^2 + a_5x^4 + a_7x^6 + a_9x^8 + \ldots) \cdot x
\end{aligned}
$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \ \text{ nonlinear mult.}$$

Parity-Split Method

$$\begin{aligned} S(x) &= a_0 + a_2x^2 + a_4x^4 + a_6x^6 + a_8x^8 + \ldots \\ & (a_1 + a_3x^2 + a_5x^4 + a_7x^6 + a_9x^8 + \ldots) \cdot x \end{aligned}$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \ \text{ nonlinear mult.}$$

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_2X + a_4X^2 + a_6X^3 + a_8X^4 + \dots$$
$$(a_1 + a_3X + a_5X^2 + a_7X^3 + a_9X^4 + \dots) \cdot x$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \ \text{ nonlinear mult.}$$

Parity-Split Method

$$\begin{aligned} S(x) &= a_0 + a_2X + a_4X^2 + a_6X^3 + a_8X^4 + \ldots \\ &\quad (a_1 + a_3X + a_5X^2 + a_7X^3 + a_9X^4 + \ldots) \cdot x \end{aligned}$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_4X^2 + a_8X^4 + \ldots + a_2X + a_6X^3 + \ldots$$
$$(a_1 + a_5X^2 + a_9X^4 + \ldots + a_3x^2 + a_7X^3 + \ldots) \cdot x$$

where $X = x^2$

- Nonlinear mult. : 1
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\begin{aligned}\mathrm{S}(x) \;=\;& a_0 + a_4X^2 + a_8X^4 + \ldots + (a_2 + a_6X^2 + \ldots)\cdot X + \\ & (a_1 + a_5X^2 + a_9X^4 + \ldots + (a_3 + a_7X^2 + \ldots)\cdot X)\cdot x\end{aligned}$$

where $X = x^2$

- Nonlinear mult. : 1+2
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\begin{aligned} \mathrm{S}(x) \quad = \quad & a_0 + a_4 x^4 + a_8 x^8 + \ldots + (a_2 + a_6 x^4 + \ldots) \cdot x^2 + \\ & (a_1 + a_5 x^4 + a_9 x^8 + \ldots + (a_3 + a_7 x^4 + \ldots) \cdot x^2) \cdot x \end{aligned}$$

where $X = x^4$

- Nonlinear mult. : 1+2
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \quad \text{nonlinear mult.}$$

Parity-Split Method

$$\begin{aligned} \mathrm{S}(x) &= a_0 + a_4 X + a_8 X^2 + \ldots + (a_2 + a_6 X + \ldots) \cdot x^2 + \\ & (a_1 + a_5 X + a_9 X^2 + \ldots + (a_3 + a_7 X + \ldots) \cdot x^2) \cdot x \end{aligned}$$

where $X = x^4$

- Nonlinear mult. : 1+2
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\begin{aligned} S(x) &= a_0 + a_4 X + a_8 X^2 + \ldots + (a_2 + a_6 X + \ldots)\cdot x^2 + \\ &\quad (a_1 + a_5 X + a_9 X^2 + \ldots + (a_3 + a_7 X + \ldots)\cdot x^2)\cdot x \end{aligned}$$

where $X = x^4$

- Nonlinear mult. : $1+2+\cdots+2^{r-1} = 2^r - 1$
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \ \text{ nonlinear mult.}$$

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_4X + a_8X^2 + \ldots + (a_2 + a_6X + \ldots) \cdot x^2 + (a_1 + a_5X + a_9X^2 + \ldots + (a_3 + a_7X + \ldots) \cdot x^2) \cdot x$$

where $X = x^4$

- Nonlinear mult. : $1+2+\cdots+2^{r-1} = 2^r - 1$
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_4 X + a_8 X^2 + \ldots + (a_2 + a_6 X + \ldots) \cdot x^2 + (a_1 + a_5 X + a_9 X^2 + \ldots + (a_3 + a_7 X + \ldots) \cdot x^2) \cdot x$$

where $X = x^4$

- Nonlinear mult. : $1+2+\cdots+2^{r-1} = 2^r - 1$
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$\mathrm{S}(x) = a_0 + a_4X + a_8X^2 + \ldots + (a_2 + a_6X + \ldots) \cdot x^2 + (a_1 + a_5X + a_9X^2 + \ldots + (a_3 + a_7X + \ldots) \cdot x^2) \cdot x$$

where $X = x^4$

- Nonlinear mult. : $1+2+\cdots+2^{r-1} = 2^r - 1$
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \text{ nonlinear mult.}$$

Parity-Split Method

$$
\begin{aligned}
\mathrm{S}(x) \; = \; & a_0 + a_4X + a_8X^2 + \ldots + (a_2 + a_6X + \ldots) \cdot x^2 + \\
& (a_1 + a_5X + a_9X^2 + \ldots + (a_3 + a_7X + \ldots) \cdot x^2) \cdot x
\end{aligned}
$$

where $X = x^4$

- Nonlinear mult. : $1+2+\cdots+2^{r-1} = 2^r - 1$
- and the evaluation of 2^{r+1} polynomials in $X = x^{2^r}$
  - we derive X^j for $j < 2^{n-r}$
  - $2^{n-r-1} - 1$ nonlinear mult.

$$\Rightarrow 2^{n-r-1} + 2^r - 2 \;\; \text{nonlinear mult.}$$

Comparison

Number of nonlinear multiplications w.r.t. the evaluation method

Method \ n	3	4	5	6	7	8	9	10
Cyclotomic	1	3	5	11	17	33	53	105
Parity-Split	2	4	6	10	14	22	30	46

- For PRESENT ($n = 4$), we shall prefer the cyclotomic method
- For DES ($n = 6$), we shall prefer the parity-split method

Comparison

Number of nonlinear multiplications w.r.t. the evaluation method

Method \ n	3	4	5	6	7	8	9	10
Cyclotomic	1	3	5	11	17	33	53	105
Parity-Split	2	4	6	10	14	22	30	46

- For PRESENT ($n = 4$), we shall prefer the cyclotomic method
- For DES ($n = 6$), we shall prefer the parity-split method

Alternative Approach

... for algorithms combining only affine and power functions

- Idea: Mix additive with multiplicative masking defined on the same field.
- Recall (Additive masking):
  $x \in \mathrm{GF}(2^n) \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{d+1}$ s.t.

$$\sum_i x_i = x \ .$$

- Recall (Multiplicative masking):
  $x \in \mathrm{GF}(2^n)^* \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{*\,d+1}$ s.t.

$$\prod_i x_i = x \ .$$

- So, use additive masking for affine transformations and multiplicative masking for power functions.

Alternative Approach

... for algorithms combining only affine and power functions

- Idea: Mix additive with multiplicative masking defined on the same field.
- Recall (Additive masking):
  $x \in \mathrm{GF}(2^n) \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{d+1}$ s.t.
  $$\sum_i x_i = x \ .$$
- Recall (Multiplicative masking):
  $x \in \mathrm{GF}(2^n)^* \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{*d+1}$ s.t.
  $$\prod_i x_i = x \ .$$
- So, use additive masking for affine transformations and multiplicative masking for power functions.

Alternative Approach

... for algorithms combining only affine and power functions

- Idea: Mix additive with multiplicative masking defined on the same field.
- Recall (Additive masking):
  $x \in \mathrm{GF}(2^n) \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{d+1}$ s.t.

$$\sum_i x_i = x \ .$$

- Recall (Multiplicative masking):
  $x \in \mathrm{GF}(2^n)^* \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{*\,d+1}$ s.t.

$$\prod_i x_i = x \ .$$

- So, use additive masking for affine transformations and multiplicative masking for power functions.

Alternative Approach

... for algorithms combining only affine and power functions

- Idea: Mix additive with multiplicative masking defined on the same field.
- Recall (Additive masking):
  $x \in \mathrm{GF}(2^n) \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{d+1}$ s.t.

$$\sum_i x_i = x \ .$$

- Recall (Multiplicative masking):
  $x \in \mathrm{GF}(2^n)^* \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{*\,d+1}$ s.t.

$$\prod_i x_i = x \ .$$

- So, use additive masking for affine transformations and multiplicative masking for power functions.

Alternative Approach

... for algorithms combining only affine and power functions

- Idea: Mix additive with multiplicative masking defined on the same field.
- Recall (Additive masking):
  $x \in \mathrm{GF}(2^n) \mapsto (x_0, \cdots x_d) \in \mathrm{GF}(2^n)^{d+1}$ s.t.
  $$\sum_i x_i = x \ .$$
- Recall (Multiplicative masking):
  $x \in \mathrm{GF}(2^n)^* \mapsto (x_0, \cdots x_d) \in {\mathrm{GF}(2^n)^*}^{d+1}$ s.t.
  $$\prod_i x_i = x \ .$$
- So, use additive masking for affine transformations and multiplicative masking for power functions.

Alternative Approach

... for algorithms combining only affine and power functions

- Issue 1: convert additive masking into multiplicative masking without leaking information in the d^{th}-order probing model?
  - Solution: conversions algorithms proposed in GenelleProuffQuisquater11 (complexity: d^2 additions and $d(3+d)/2$ multiplications).
- Issue 2: multiplicative is only sound in the multiplicative group! How to deal with the 0 value problem?
  - Solution: map the sharing of 0 into the sharing of 1 and keep trace of this modification for further correction.
  - Amounts to secure the processing of the function

$$x \mapsto x \oplus \delta_0(x) \text{ with } \delta_0(x) = x_0 \texttt{ AND } x_1 \texttt{ AND } \dots \texttt{ AND } x_n \ .$$

  - Soundness: for any power e, we have

$$(x \oplus \delta_0(x))^e = x^e \oplus \delta_0(x)$$

Alternative Approach

... for algorithms combining only affine and power functions

- Issue 1: convert additive masking into multiplicative masking without leaking information in the d^{th}-order probing model?
  - Solution: conversions algorithms proposed in GenelleProuffQuisquater11 (complexity: d^2 additions and $d(3+d)/2$ multiplications).
- Issue 2: multiplicative is only sound in the multiplicative group! How to deal with the 0 value problem?
  - Solution: map the sharing of 0 into the sharing of 1 and keep trace of this modification for further correction.
  - Amounts to secure the processing of the function

$$x \mapsto x \oplus \delta_0(x) \text{ with } \delta_0(x) = x_0 \texttt{ AND } x_1 \texttt{ AND } \ldots \texttt{ AND } x_n \ .$$

  - Soundness: for any power e, we have

$$(x \oplus \delta_0(x))^e = x^e \oplus \delta_0(x)$$

Alternative Approach

... for algorithms combining only affine and power functions

- Issue 1: convert additive masking into multiplicative masking without leaking information in the d^{th}-order probing model?
  - Solution: conversions algorithms proposed in GenelleProuffQuisquater11 (complexity: d^2 additions and $d(3+d)/2$ multiplications).
- Issue 2: multiplicative is only sound in the multiplicative group! How to deal with the 0 value problem?
  - Solution: map the sharing of 0 into the sharing of 1 and keep trace of this modification for further correction.
  - Amounts to secure the processing of the function

$$x \mapsto x \oplus \delta_0(x) \text{ with } \delta_0(x) = x_0 \texttt{ AND } x_1 \texttt{ AND } \ldots \texttt{ AND } x_n \ .$$

  - Soundness: for any power e, we have

$$(x \oplus \delta_0(x))^e = x^e \oplus \delta_0(x)$$

Alternative Approach

... for algorithms combining only affine and power functions

- Issue 1: convert additive masking into multiplicative masking without leaking information in the d^{th}-order probing model?
  - Solution: conversions algorithms proposed in GenelleProuffQuisquater11 (complexity: d^2 additions and $d(3+d)/2$ multiplications).
- Issue 2: multiplicative is only sound in the multiplicative group! How to deal with the 0 value problem?
  - Solution: map the sharing of 0 into the sharing of 1 and keep trace of this modification for further correction.
  - Amounts to secure the processing of the function

$$x \mapsto x \oplus \delta_0(x) \text{ with } \delta_0(x) = x_0 \text{ AND } x_1 \text{ AND } \ldots \text{ AND } x_n .$$

  - Soundness: for any power e, we have

$$(x \oplus \delta_0(x))^e = x^e \oplus \delta_0(x)$$

Alternative Approach

... for algorithms combining only affine and power functions

- Issue 1: convert additive masking into multiplicative masking without leaking information in the d^{th}-order probing model?
  - Solution: conversions algorithms proposed in GenelleProuffQuisquater11 (complexity: d^2 additions and $d(3+d)/2$ multiplications).
- Issue 2: multiplicative is only sound in the multiplicative group! How to deal with the 0 value problem?
  - Solution: map the sharing of 0 into the sharing of 1 and keep trace of this modification for further correction.
  - Amounts to secure the processing of the function

$$x \mapsto x \oplus \delta_0(x) \text{ with } \delta_0(x) = x_0 \texttt{ AND } x_1 \texttt{ AND } \dots \texttt{ AND } x_n \ .$$

  - Soundness: for any power e, we have

$$(x \oplus \delta_0(x))^e = x^e \oplus \delta_0(x)$$

Alternative Approach

... for algorithms combining only affine and power functions

- Issue 1: convert additive masking into multiplicative masking without leaking information in the d^{th}-order probing model?
  - Solution: conversions algorithms proposed in GenelleProuffQuisquater11 (complexity: d^2 additions and $d(3+d)/2$ multiplications).
- Issue 2: multiplicative is only sound in the multiplicative group! How to deal with the 0 value problem?
  - Solution: map the sharing of 0 into the sharing of 1 and keep trace of this modification for further correction.
  - Amounts to secure the processing of the function

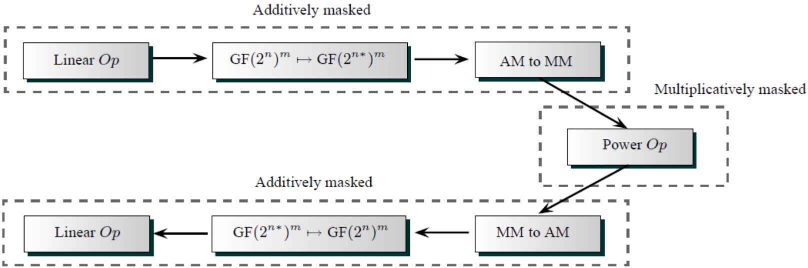
$$x \mapsto x \oplus \delta_0(x) \text{ with } \delta_0(x) = x_0 \texttt{ AND } x_1 \texttt{ AND } \ldots \texttt{ AND } x_n \ .$$

  - Soundness: for any power e, we have

$$(x \oplus \delta_0(x))^e = x^e \oplus \delta_0(x)$$

Alternative Approach

… for algorithms combining only affine and power functions



Performances

Table : Comparison of secure AES implementations

	Method	cycles (10^3)	RAM (bytes)
	Unprotected Implementation		
1.	No Masking	2	32
	First Order Masking		
2.	Re-computation	10	256
3.	Tower Field in $\mathrm{GF}(2^2)$	77	42
4.	Multiplicative Masking	22	256
5.	Secure exponentiation for $d = 1$	73	24
6.	Additive and Multiplicative Masking for $d = 1$	25	50
	Second Order Masking		
7.	Double Re-computations	594	$512 + 28$
8.	Single Re-computation	672	$256 + 22$
9.	Secure exponentiation for $d = 2$	189	48
10.	Additive and Multiplicative Masking f for $d = 2$	69	86
	Third Order Masking		
11.	Secure exponentiation for $d = 3$	326	72
12.	Additive and Multiplicative Masking f for $d = 3$	180	128

Masking Schemes for Block Ciphers

Still other alternatives...

- Apply Tower Field Approach: $\mathrm{GF}(2^8) \sim \mathrm{GF}(2^4)[X]/p(X) \sim \mathrm{GF}(2^2)[X]/p'(X) \sim \mathrm{GF}(2)[X]/p''(X)$.
  - see *e.g.* *OswaldMangardPramstallerRijmen05*.
  - *sometimes lead to efficiency improvement as some operations can be tabulated.*
- Split exponentiation in more complex sequences than simply squarings and multiplications.
  - *e.g. also consider bilinear operations as* $x \mapsto x \times L(x)$ *where* L *is linear* *ProuffRivainRoche13*.
- Develop masking schemes for security in presence of glitches.
  - current propositions based on MPC techniques NikovaRijmenSchläffer11,ProuffRoche11.
- Find alternatives to reduce the consumption of random values.

Conclusion

- Security of current implementations is usually only evaluated w.r.t. first-order (a.k.a. univariate) SCA.
  - Against those attacks efficient and effective solutions exist: shuffling + first-order masking + noise addition.
- Higher-order SCA start to be also considered by security evaluators.
  - Effective countermeasures exist but their efficiency is low (see the masking schemes presented here).
  - Best alternative for software AES: shuffling + additive/multiplicative masking + noise.
  - For other block ciphers: only ISW extensions may be applied but they are costly. **This topic needs more studies.**
- Security of today implementations must be formally proved
  - Give upper bound on the information leakage, obtained with sound models. **This topic needs more studies.**
  - Prove the security against some classes of adversaries.
  - Find protocols which ensure that no SCA can be performed.
  - Develop automatic provers.

Conclusion

- Security of current implementations is usually only evaluated w.r.t. first-order (a.k.a. univariate) SCA.
    - Against those attacks efficient and effective solutions exist: shuffling + first-order masking + noise addition.
- Higher-order SCA start to be also considered by security evaluators.
    - Effective countermeasures exist but their efficiency is low (see the masking schemes presented here).
    - Best alternative for software AES: shuffling + additive/multiplicative masking + noise.
    - For other block ciphers: only ISW extensions may be applied but they are costly. **This topic needs more studies.**
- Security of today implementations must be formally proved
    - Give upper bound on the information leakage, obtained with sound models. **This topic needs more studies.**
    - Prove the security against some classes of adversaries.
    - Find protocols which ensure that no SCA can be performed.
    - Develop automatic provers.

Conclusion

- Security of current implementations is usually only evaluated w.r.t. first-order (a.k.a. univariate) SCA.
  - Against those attacks efficient and effective solutions exist: shuffling + first-order masking + noise addition.
- Higher-order SCA start to be also considered by security evaluators.
  - Effective countermeasures exist but their efficiency is low (see the masking schemes presented here).
  - Best alternative for software AES: shuffling + additive/multiplicative masking + noise.
  - For other block ciphers: only ISW extensions may be applied but they are costly. **This topic needs more studies.**
- Security of today implementations must be formally proved
  - Give upper bound on the information leakage, obtained with sound models. **This topic needs more studies.**
  - Prove the security against some classes of adversaries.
  - Find protocols which ensure that no SCA can be performed.
  - Develop automatic provers.

Conclusion

- Security of current implementations is usually only evaluated w.r.t. first-order (a.k.a. univariate) SCA.
  - Against those attacks efficient and effective solutions exist: shuffling + first-order masking + noise addition.
- Higher-order SCA start to be also considered by security evaluators.
  - Effective countermeasures exist but their efficiency is low (see the masking schemes presented here).
  - Best alternative for software AES: shuffling + additive/multiplicative masking + noise.
  - For other block ciphers: only ISW extensions may be applied but they are costly. **This topic needs more studies.**
- Security of today implementations must be formally proved
  - Give upper bound on the information leakage, obtained with sound models. **This topic needs more studies.**
  - Prove the security against some classes of adversaries.
  - Find protocols which ensure that no SCA can be performed.
  - Develop automatic provers.

Thank you for your attention!

Acknowledgement: some slides of the countermeasures' section have been directly inspired by presentations done by Karim Khalfallah, Matthieu Rivain and Thomas Roche. I thank them very much for their help.