Provable Security against Side-Channel Attacks

Matthieu Rivain

matthieu.rivain@cryptoexperts.com

MCrypt Seminar – Aug. 11th 2014

CryptoExperts

WE INNOVATE TO SECURE YOUR BUSINESS

Outline

1. Introduction
2. Modeling side-channel leakage
3. Achieving provable security against SCA

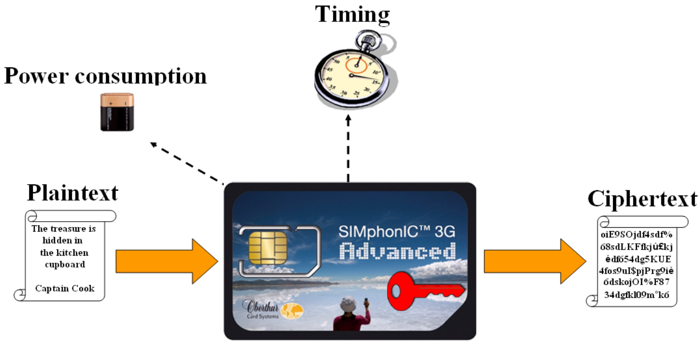
Outline

1. Introduction
2. Modeling side-channel leakage
3. Achieving provable security against SCA

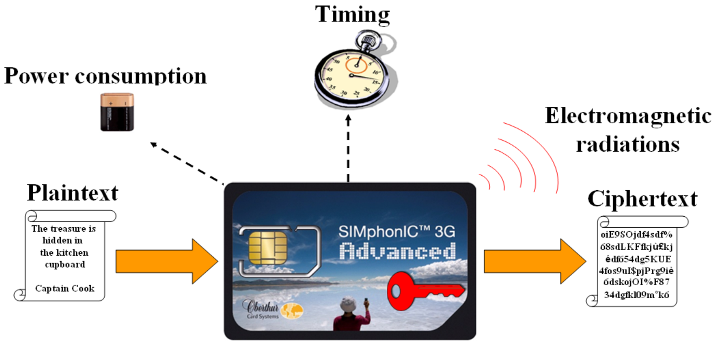
Side-channel attacks

Side-channel attacks

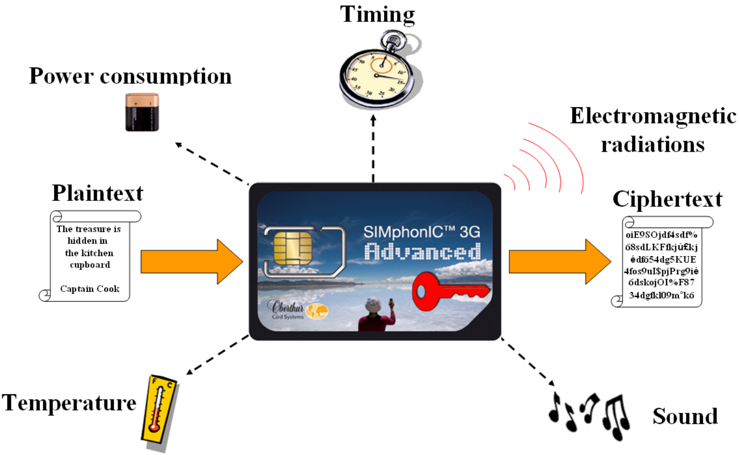
Side-channel attacks



Side-channel attacks



Side-channel attacks



CryptoExperts

Side-channel attacks

Sound and temperature

- Proofs of concept in idealized conditions
- Minor practical threats on embedded systems

Running time

- Trivial solution: constant-time implementations
- Must be carefully addressed
  - timing flaw still discovered in OpenSSL in 2011!
  - timing flaws can be induced by the processor (cache, branch prediction, ...)

Side-channel attacks

Power consumption and EM emanations

- Close by nature (switching activity)
- Can be modeled as weighted sums of the transitions
- EM can be more informative (placing of the probe) but assume a raw access to the circuit
- Both are noisy *i.e.* non-deterministic
- Noise amplification by generating random switching activity

Side-channel attacks

Power consumption and EM emanations

- Close by nature (switching activity)
- Can be modeled as weighted sums of the transitions
- EM can be more informative (placing of the probe) but assume a raw access to the circuit
- Both are noisy *i.e.* non-deterministic
- Noise amplification by generating random switching activity

<ins>This talk</ins>: leakage = power consuption + EM emanations

Provable security

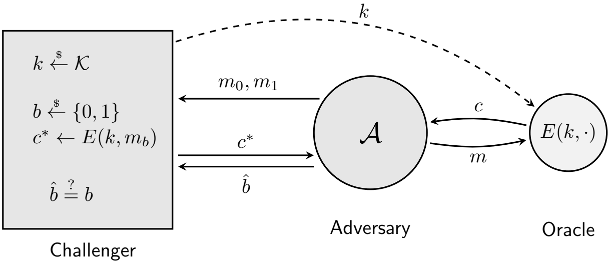
Traditional approach

- define an adversarial model (*e.g.* chosen plaintext attacker)
- define a security goal (*e.g.* distinguish two ciphertexts)

Provable security

Traditional approach

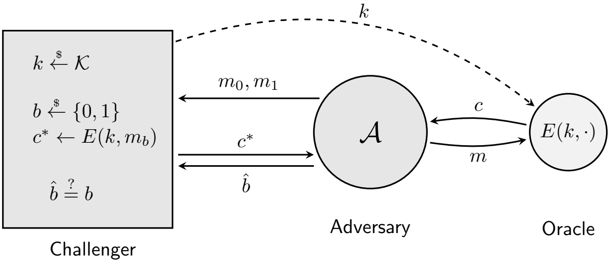
- define an adversarial model (*e.g.* chosen plaintext attacker)
- define a security goal (*e.g.* distinguish two ciphertexts)



Provable security

Traditional approach

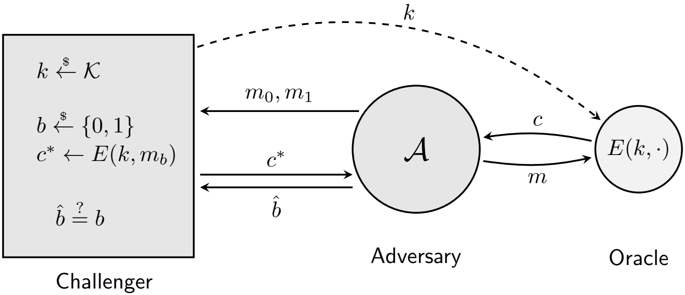
- define an adversarial model (*e.g.* chosen plaintext attacker)
- define a security goal (*e.g.* distinguish two ciphertexts)



Security reduction: If $\mathcal{A}$ exists with non-negligible $|\Pr[\hat{b} = b] - 1/2|$ then I can use $\mathcal{A}$ to efficiently solve a hard problem.

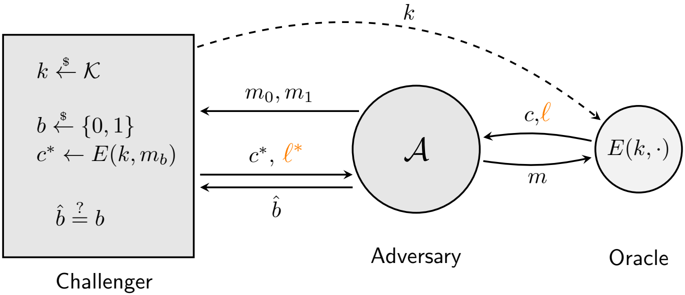
Provable security

… in the presence of leakage



Provable security

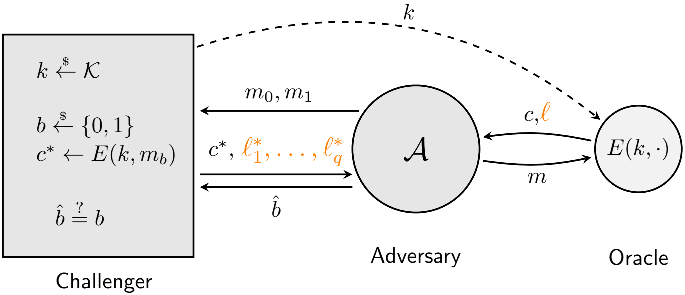
… in the presence of leakage



CryptoExperts

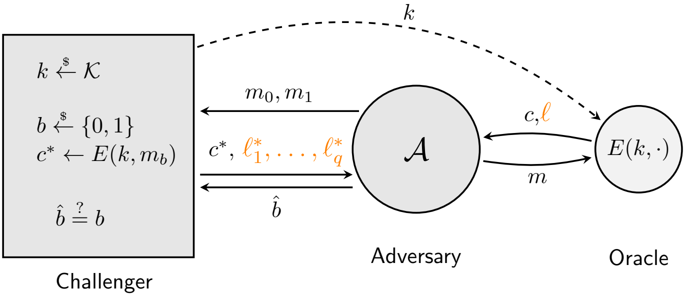
Provable security

… in the presence of leakage



Provable security

... in the presence of leakage



Issue: how to model the leakage?

CryptoExperts

Outline

1. Introduction
2. Modeling side-channel leakage
3. Achieving provable security against SCA

Modeling side-channel leakage

The encryption oracle *cannot* be seen as a mathematical function $E(k, \cdot) : m \mapsto c$ anymore, but as a computation.

- Two classical approaches to model computation:
  - Turing machines (programs)
  - Circuits
- How to model *leaking* computation?

Modeling side-channel leakage

Chronology

- Probing model (circuits, 2003)
- Physically observable cryptography (Turing machines, 2004)
- Leakage resilient cryptography (2008)
- Further leakage models for circuits (2010)
- Noisy leakage model (2013)

Presentation

- Leakage models for circuits
- Leakage models for programs

Modeling side-channel leakage

Chronology

- Probing model (circuits, 2003)
- Physically observable cryptography (Turing machines, 2004)
- Leakage resilient cryptography (2008)
- Further leakage models for circuits (2010)
- Noisy leakage model (2013)

Presentation

- Leakage models for circuits
- Leakage models for programs

CryptoExperts

Modeling side-channel leakage

Chronology

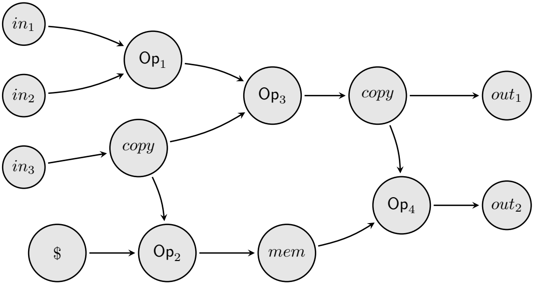
- Probing model (circuits, 2003)
- Physically observable cryptography (Turing machines, 2004)
- Leakage resilient cryptography (2008)
- Further leakage models for circuits (2010)
- Noisy leakage model (2013)

Presentation

- Leakage models for circuits
- Leakage models for programs

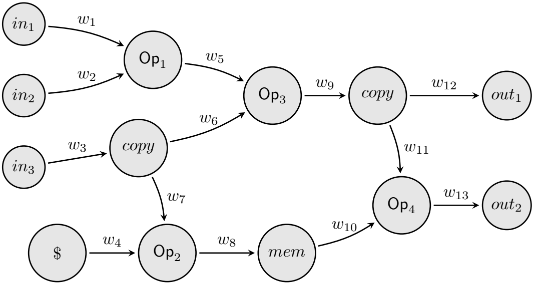
Leakage Models for Circuits

- [Ishai-Sahai-Wagner. CRYPTO 2003]
- Directed graph whose nodes are *gates* and edges are *wires*



Leakage Models for Circuits

- [Ishai-Sahai-Wagner. CRYPTO 2003]
- Directed graph whose nodes are *gates* and edges are *wires*



- At each cycles, the circuit leaks $f(w_1, w_2, \ldots, w_n)$

Leakage Models for Circuits

- Probing security model [Ishai-Sahai-Wagner. CRYPTO 2003]
  - the adversary gets $(w_i)_{i \in \mathcal{I}}$ for some chosen set $|\mathcal{I}| \leq t$
- $\mathcal{AC}_0$ leakage model [Faust et al. EUROCRYPT 2010]
  - the leakage function f belongs to the $\mathcal{AC}_0$ complexity class
  - *i.e.* f is computable by circuits of constant depth d
- Noisy circuit-leakage model [Faust et al. EUROCRYPT 2010]
  - $f : (w_1, w_2, \ldots, w_n) \mapsto (w_1 \oplus \varepsilon_1, w_2 \oplus \varepsilon_2, \ldots, w_n \oplus \varepsilon_n)$

    with $\varepsilon_i = \begin{cases} 1 \text{ with proba } p < 1/2 \\ 0 \text{ with proba } 1 - p \end{cases}$

Leakage Models for Circuits

- Probing security model [Ishai-Sahai-Wagner. CRYPTO 2003]
  - the adversary gets $(w_i)_{i \in \mathcal{I}}$ for some chosen set $|\mathcal{I}| \leq t$
- $\mathcal{AC}_0$ leakage model [Faust et al. EUROCRYPT 2010]
  - the leakage function f belongs to the $\mathcal{AC}_0$ complexity class
  - *i.e.* f is computable by circuits of constant depth d
- Noisy circuit-leakage model [Faust et al. EUROCRYPT 2010]
  - $f : (w_1, w_2, \ldots, w_n) \mapsto (w_1 \oplus \varepsilon_1, w_2 \oplus \varepsilon_2, \ldots, w_n \oplus \varepsilon_n)$

    with $\varepsilon_i = \begin{cases} 1 \text{ with proba } p < 1/2 \\ 0 \text{ with proba } 1-p \end{cases}$
- These models fail in capturing EM and PC leakages!

CryptoExperts

Leakage Models for Circuits

- Probing security model [Ishai-Sahai-Wagner. CRYPTO 2003]
  - the adversary gets $(w_i)_{i \in \mathcal{I}}$ for some chosen set $|\mathcal{I}| \leq t$
- $\mathcal{AC}_0$ leakage model [Faust et al. EUROCRYPT 2010]
  - the leakage function f belongs to the $\mathcal{AC}_0$ complexity class
  - *i.e.* f is computable by circuits of constant depth d
- Noisy circuit-leakage model [Faust et al. EUROCRYPT 2010]
  - $f : (w_1, w_2, \ldots, w_n) \mapsto (w_1 \oplus \varepsilon_1, w_2 \oplus \varepsilon_2, \ldots, w_n \oplus \varepsilon_n)$

    with $\varepsilon_i = \begin{cases} 1 \text{ with proba } p < 1/2 \\ 0 \text{ with proba } 1 - p \end{cases}$
- These models fail in capturing EM and PC leakages!
- Circuits not convenient to model software implementations (or algorithms / protocols)

CryptoExperts

Physically Observable Cryptography

- [Micali-Reyzin. TCC'04]
- Framework for leaking computation
- Strong formalism using Turing machines
- Assumption: *Only Computation Leaks (OCL)*
- Computation divided into subcomputations $y \leftarrow \mathsf{SC}(x)$
- Each SC accesses a part of the state x and leaks $f(x)$
- f adaptively chosen by the adversary
- No actual proposal for f

Leakage Resilient Cryptography

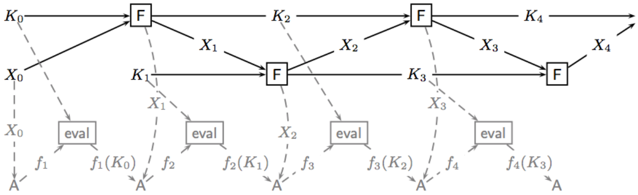
- Model introduced in [Dziembowski-Pietrzak. STOC'08]
- Specialization of the Micali-Reyzin framework
- Leakage functions follow the *bounded retrieval model* [Crescenzo et al. TCC'06]

$$f : \{0,1\}^n \to \{0,1\}^\lambda \qquad \text{for some constant } \lambda < n$$

CryptoExperts

Leakage Resilient Cryptography

- Example: LR stream cipher [Pietrzak. EUROCRYPT'09]



- Many further LR crypto primitives published so far
- Generic LR compilers
  - [Goldwasser-Rothblum. FOCS'12]
  - [Dziembowski-Faust. TCC'12]

Leakage Resilient Cryptography

- Limitation: the leakage of a subcomputation is limited to λ-bit values for $\lambda < n$ (the input size)
- Side-channel leakage far bigger than n bits
  - although it may not remove all the entropy of x

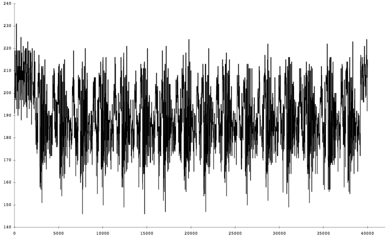


Figure: Power consumption of a DES computation.

Noisy Leakage Model

- [Prouff-Rivain. EUROCRYPT 2013]
- OCL assumption (Micali-Reyzin framework)
- New class of noisy leakage functions
- An observation $f(x)$ introduces a *bounded bias* in $\Pr[x]$
  - very generic

Notion of bias

- Bias of X given $Y = y$:

$$\beta(X|Y = y) = \|\mathsf{Pr}[X] - \mathsf{Pr}[X|Y = y]\|$$

with $\|\cdot\|$ = Euclidean norm.

- Bias of X given Y:

$$\beta(X|Y) = \sum_{y \in \mathcal{Y}} \mathsf{Pr}[Y = y]\, \beta(X|Y = y) \ .$$

- $\beta(X|Y) \in \left[0;\ \sqrt{1 - \frac{1}{|\mathcal{X}|}}\ \right]$ (indep. / deterministic relation)
- Related to MI by:

$$\frac{1}{\ln 2}\beta(X|Y) \leq \mathrm{MI}(X;Y) \leq \frac{|\mathcal{X}|}{\ln 2}\beta(X|Y)$$

CryptoExperts

Noisy Leakage Model

- Every subcomputation leaks a *noisy function* f of its input
  - noise modeled by a fresh random tape argument
- ψ is some *noise parameter*
- $f \in \mathcal{N}(1/\psi) \quad \Rightarrow \quad \beta\big(X|f(X)\big) < \frac{1}{\psi}$
- Capture any form of noisy leakage

CryptoExperts

Noisy Leakage Model

- In practice, the multivariate Gaussian model is widely admitted

$$f(x) \sim \mathcal{N}(\vec{m}_x, \Sigma) \qquad \forall x \in \mathcal{X}$$

- The bias can be efficiently computed:

$$\beta(X|f(X)) \;=\; \sum_{\vec{y}} \mathsf{p}_{\vec{y}} \Big(\sum_{x} \big(\mathsf{p}_{x|\vec{y}} - 1/|\mathcal{X}|\big)^2 \Big)^{1/2}$$

with $\mathsf{p}_{\vec{y}} = \sum_x \frac{\phi_\Sigma(\vec{y}-\vec{m}_x)}{\sum_{\vec{z}} \phi_\Sigma(\vec{z}-\vec{m}_x)}$ and $\mathsf{p}_{x|\vec{y}} = \frac{\phi_\Sigma(\vec{y}-\vec{m}_x)}{\sum_v \phi_\Sigma(\vec{y}-\vec{m}_v)}$

where $\phi_\Sigma : \vec{y} \mapsto \exp\big(-\frac{1}{2}\,\vec{y}\cdot\Sigma\cdot\vec{y}\big)$.

Noisy Leakage Model

Illustration: univariate Hamming weight model with Gaussian noise

$$f(X) = \mathsf{HW}(X) + \mathcal{N}(0, \sigma^2)$$

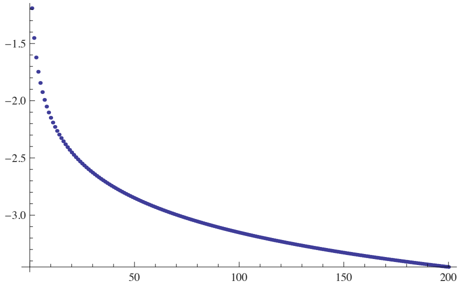


Figure: $\log_{10} \beta(X|f(X))$ w.r.t. σ.

Noisy Leakage Model

Illustration: univariate Hamming weight model with Gaussian noise

$$f(X) = \mathsf{HW}(X) + \mathcal{N}(0, \sigma)$$

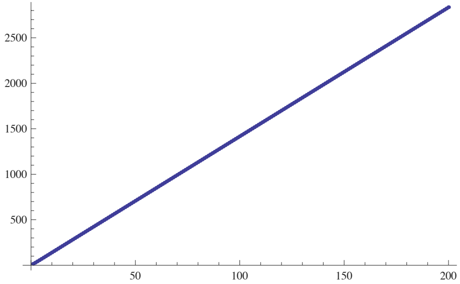
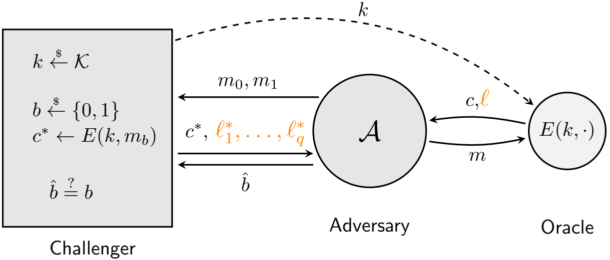


Figure: $\psi = \frac{1}{\beta(X|f(X))}$ w.r.t. σ.

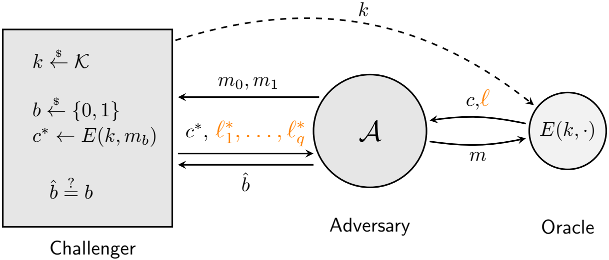
Outline

Achieving provable security against SCA



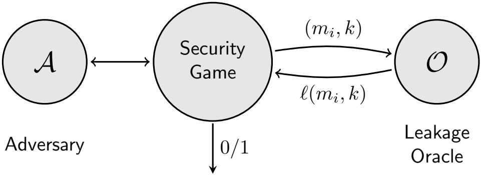
- Describe a (protected) implementation of $E(k, \cdot)$
- Model the leakage
- Provide a security reduction

Achieving provable security against SCA

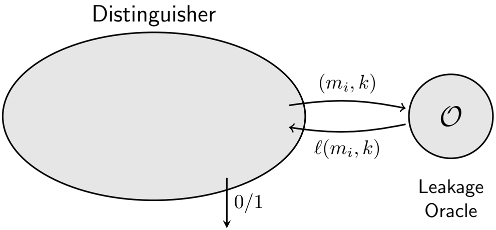


- Describe a (protected) implementation of $E(k, \cdot)$
- Model the leakage
- Provide a security reduction
- What about generic security against SCA?
  - for any cryptosystem, security goal, adversarial model

General setting

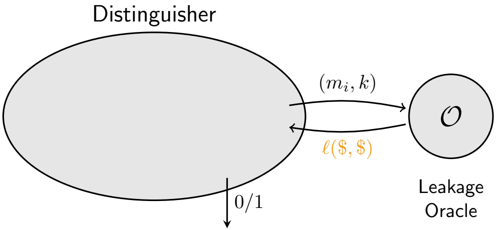


General setting



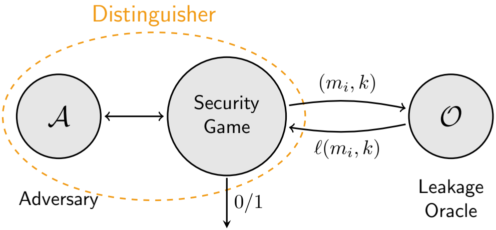
- Security: $\forall$Dist : Adv(Dist$^{\mathcal{O}(\cdot)}$) $\leq 2^{-\kappa}$

General setting



- Security: $\forall$Dist : $\mathsf{Adv}(\mathsf{Dist}^{\mathcal{O}(\cdot)}) \leq 2^{-\kappa}$

General setting

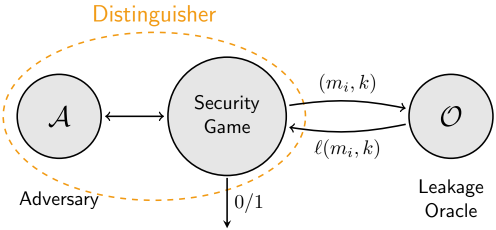


- Security: $\forall \mathsf{Dist}: \ \mathsf{Adv}(\mathsf{Dist}^{\mathcal{O}(\cdot)}) \leq 2^{-\kappa}$

  $\Rightarrow \ \forall \mathcal{A}: \ \mathsf{Adv}(\mathcal{A}\text{-}\mathsf{SG}^{\mathcal{O}(\cdot)}) \approx \mathsf{Adv}(\mathcal{A}\text{-}\mathsf{SG}^{\mathcal{O}^{\$}(\cdot)})$

CryptoExperts

General setting

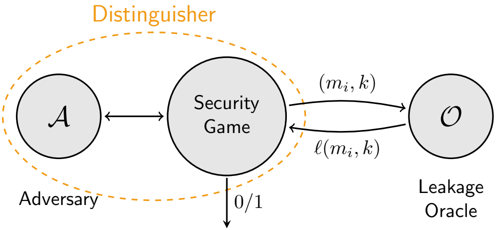


- Security: $\forall \text{Dist}: \ \text{Adv}(\text{Dist}^{\mathcal{O}(\cdot)}) \leq 2^{-\kappa}$

  $\Rightarrow \ \forall \mathcal{A}: \ \text{Adv}(\mathcal{A}\text{-SG}^{\mathcal{O}(\cdot)}) \approx \text{Adv}(\mathcal{A}\text{-SG}^{\mathcal{O}^{\$}(\cdot)})$

- Information theoretic security: $\mathrm{MI}((m,k);\ell(m,k)) \leq 2^{-\kappa}$

General setting



- Security: $\forall \text{Dist}: \ \text{Adv}(\text{Dist}^{\mathcal{O}(\cdot)}) \leq 2^{-\kappa}$

  $\Rightarrow \ \forall \mathcal{A}: \ \text{Adv}(\mathcal{A}\text{-SG}^{\mathcal{O}(\cdot)}) \approx \text{Adv}(\mathcal{A}\text{-SG}^{\mathcal{O}^{\$}(\cdot)})$

- Information theoretic security: $\text{MI}((m,k); \ell(m,k)) \leq 2^{-\kappa}$

$\boxed{\text{IT Security} \Rightarrow \text{Security}}$

Using random sharing

Principle

- Randomly share the internal state of the computation
- A d-sharing of $x \in \mathbb{F}$ is a tuple $(x_1, x_2, \ldots, x_n)$ s.t.

$$x_1 + x_2 + \cdots + x_n = x$$

  with $n - 1$ *degrees of randomness*

- Subcomputations $y \leftarrow \mathsf{SC}(x)$ are replaced by

$$(y_1, y_2, \ldots, y_n) \leftarrow \mathsf{SC}'(x_1, x_2, \ldots, x_n)$$

Using random sharing

Soundness

- [Chari et al. CRYPTO'99]
- Univariate Gaussian leakage model: $\ell_i \sim x_i + \mathcal{N}(\mu, \sigma^2)$
- Distinguishing $\big((\ell_i)_i | x = 0\big)$ from $\big((\ell_i)_i | x = 1\big)$ takes q samples:

$$q \geq cst \cdot \sigma^n$$

- Limitations:
  - univariate leakage model, Gaussian noise assumption
  - static leakage of the shares (*i.e.* without computation)
  - no scheme proposed to securely compute on a shared state

Ishai-Sahai-Wagner Scheme

- [Ishai-Sahai-Wagner. CRYPTO 2003]
- Binary circuit model
- Goal: security against t-probing attacks
- Every wire w is shared in n wires w_1, w_2, ..., w_n
- Issue: how to encode logic gates?
  - NOT gates and AND gates
- NOT gates encoding:

$$\overline{w} = \overline{w_1} \oplus w_2 \cdots \oplus w_n$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ a_2b_1 & a_2b_2 & a_2b_3 \\ a_3b_1 & a_3b_2 & a_3b_3 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ 0 & a_2b_2 & a_2b_3 \\ 0 & 0 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & 0 & 0 \\ a_2b_1 & 0 & 0 \\ a_3b_1 & a_3b_2 & 0 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1b_1 & a_1b_2 & a_1b_3 \\ 0 & a_2b_2 & a_2b_3 \\ 0 & 0 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & a_2b_1 & a_3b_1 \\ 0 & 0 & a_3b_2 \\ 0 & 0 & 0 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example $(n = 3)$:

$$\begin{pmatrix} a_1b_1 & a_1b_2 \oplus a_2b_1 & a_1b_3 \oplus a_3b_1 \\ 0 & a_2b_2 & a_2b_3 \oplus a_3b_2 \\ 0 & 0 & a_3b_3 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1 b_1 & a_1 b_2 \oplus a_2 b_1 & a_1 b_3 \oplus a_3 b_1 \\ 0 & a_2 b_2 & a_2 b_3 \oplus a_3 b_2 \\ 0 & 0 & a_3 b_3 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1b_1 & a_1b_2 \oplus a_2b_1 & a_1b_3 \oplus a_3b_1 \\ 0 & a_2b_2 & a_2b_3 \oplus a_3b_2 \\ 0 & 0 & a_3b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & r_{1,2} & r_{1,3} \\ 0 & 0 & r_{2,3} \\ 0 & 0 & 0 \end{pmatrix}$$

CryptoExperts

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1 b_1 & a_1 b_2 \oplus a_2 b_1 & a_1 b_3 \oplus a_3 b_1 \\ 0 & a_2 b_2 & a_2 b_3 \oplus a_3 b_2 \\ 0 & 0 & a_3 b_3 \end{pmatrix} \oplus \begin{pmatrix} 0 & r_{1,2} & r_{1,3} \\ r_{1,2} & 0 & r_{2,3} \\ r_{1,3} & r_{2,3} & 0 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1b_1 & (a_1b_2 \oplus r_{1,2}) \oplus a_2b_1 & (a_1b_3 \oplus r_{1,3}) \oplus a_3b_1 \\ r_{1,2} & a_2b_2 & (a_2b_3 \oplus r_{2,3}) \oplus a_3b_2 \\ r_{1,3} & r_{2,3} & a_3b_3 \end{pmatrix}$$

CryptoExperts

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{pmatrix} a_1b_1 & (a_1b_2 \oplus r_{1,2}) \oplus a_2b_1 & (a_1b_3 \oplus r_{1,3}) \oplus a_3b_1 \\ r_{1,2} & a_2b_2 & (a_2b_3 \oplus r_{2,3}) \oplus a_3b_2 \\ r_{1,3} & r_{2,3} & a_3b_3 \end{pmatrix}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{array}{c} \begin{pmatrix} a_1 b_1 & (a_1 b_2 \oplus r_{1,2}) \oplus a_2 b_1 & (a_1 b_3 \oplus r_{1,3}) \oplus a_3 b_1 \\ r_{1,2} & a_2 b_2 & (a_2 b_3 \oplus r_{2,3}) \oplus a_3 b_2 \\ r_{1,3} & r_{2,3} & a_3 b_3 \end{pmatrix} \\ \begin{array}{lll} c_1 & & \end{array} \end{array}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{array}{ccc} \left(\begin{array}{ccc} a_1 b_1 & (a_1 b_2 \oplus r_{1,2}) \oplus a_2 b_1 & (a_1 b_3 \oplus r_{1,3}) \oplus a_3 b_1 \\ r_{1,2} & a_2 b_2 & (a_2 b_3 \oplus r_{2,3}) \oplus a_3 b_2 \\ r_{1,3} & r_{2,3} & a_3 b_3 \end{array}\right) \\ \begin{array}{ccc} c_1 & \qquad\qquad c_2 & \qquad\qquad\qquad\qquad \end{array} \end{array}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{array}{ccc} \left(\begin{array}{ccc} a_1 b_1 & (a_1 b_2 \oplus r_{1,2}) \oplus a_2 b_1 & (a_1 b_3 \oplus r_{1,3}) \oplus a_3 b_1 \\ r_{1,2} & a_2 b_2 & (a_2 b_3 \oplus r_{2,3}) \oplus a_3 b_2 \\ r_{1,3} & r_{2,3} & a_3 b_3 \end{array}\right) \\ \begin{array}{ccc} c_1 & c_2 & c_3 \end{array} \end{array}$$

Ishai-Sahai-Wagner Scheme

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. $\bigoplus_i c_i = a \cdot b$

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

- Example ($n = 3$):

$$\begin{array}{ccc} \left(\begin{array}{ccc} a_1 b_1 & (a_1 b_2 \oplus r_{1,2}) \oplus a_2 b_1 & (a_1 b_3 \oplus r_{1,3}) \oplus a_3 b_1 \\ r_{1,2} & a_2 b_2 & (a_2 b_3 \oplus r_{2,3}) \oplus a_3 b_2 \\ r_{1,3} & r_{2,3} & a_3 b_3 \end{array}\right) \\ \begin{array}{ccc} c_1 & c_2 & c_3 \end{array} \end{array}$$

Ishai-Sahai-Wagner Scheme

Sketch of security proof

- t-probing adversary $\Rightarrow n = 2t + 1$ shares
- probed wires: v_1, v_2, . . . , v_t
- construct a set $I = \{$raw and column indices of $v_k\}$
- $(v_1, v_2, \ldots, v_t)$ perfectly simulated from $(a_i)_{i \in I}$ and $(b_i)_{i \in I}$
- $|I| \leq 2t < n \Rightarrow (a_i)_{i \in I}$ and $(b_i)_{i \in I}$ are random $|I|$-tuples

Ishai-Sahai-Wagner Scheme

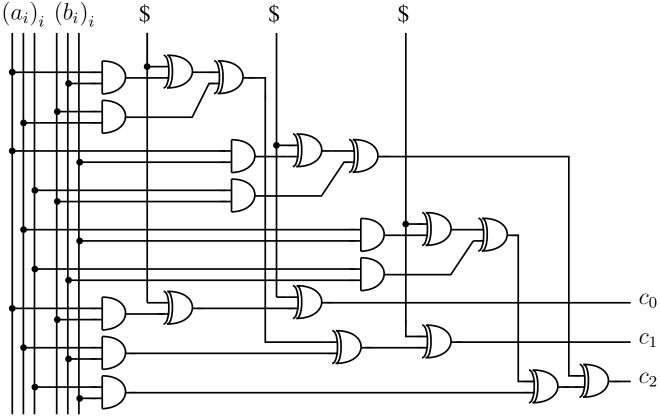


Figure: AND gate for $n = 3$

Ishai-Sahai-Wagner Scheme

- Can be transposed to the dth-order security model
  - the adversary must combined the leakage of at least d subcomputations to recover information
  - in presence of noise d is a relevant security parameter [Chari et al. CRYPTO'99]
- Many dth-order secure schemes based on ISW scheme
- Not fully satisfactory
  - an relevant adversary should use all the leakage

Security in the noisy model

- [Prouff-Rivain. EUROCRYPT 2013]
- Every $y \leftarrow \mathsf{SC}(x)$ leaks $f(x)$ where $\beta\big(X|f(X)\big) < \frac{1}{\psi}$
- Information theoretic security proof:

$$\mathrm{MI}\big((m,k);\ell(m,k)\big) < O(\omega^{-d})$$

- Assumtpion: the noise parameter ψ can be linearly increased
- Need of a *leak-free component* for refreshing

$$\underbrace{\boldsymbol{x} = (x_0, x_1, \ldots, x_d)}_{\bigoplus_i x_i = x} \quad \longmapsto \quad \underbrace{\boldsymbol{x}' = (x'_0, x'_1, \ldots, x'_d)}_{\bigoplus_i x'_i = x}$$

with $(\boldsymbol{x} \mid x)$ and $(\boldsymbol{x}' \mid x)$ mutually independent.

CryptoExperts

Overview of the proof

- Consider a SPN computation

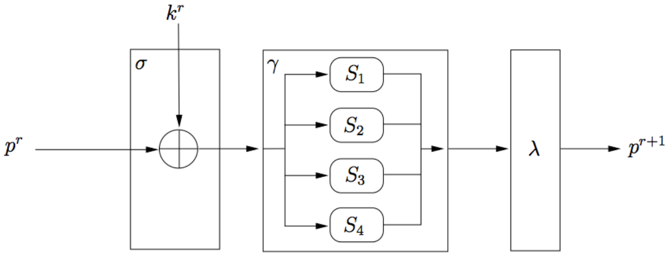


Figure: Example of SPN round.

Overview of the proof

- Classical implementation protected with sharing

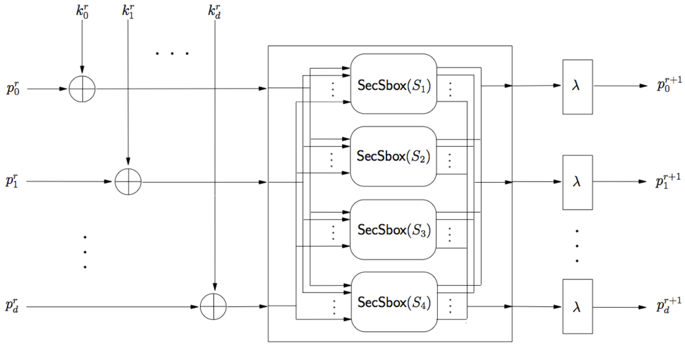


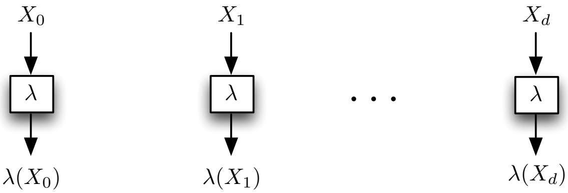
Figure: Example of SPN round protected with sharing.

S-Box computation

- [Carlet et al. FSE'12]
- Polynomial evaluation over GF(2^n)
- Two types of elementary calculations:
  - linear functions (additions, squares, multiplication by coefficients)
  - multiplications over GF(2^n)

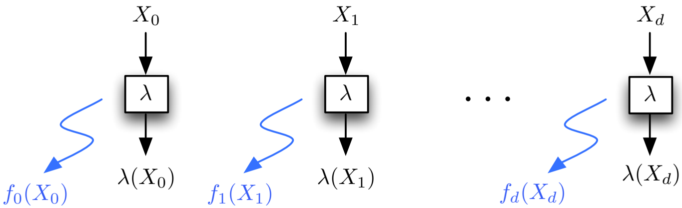
Linear functions

- Given a sharing $X = X_0 \oplus X_1 \oplus \cdots \oplus X_d$



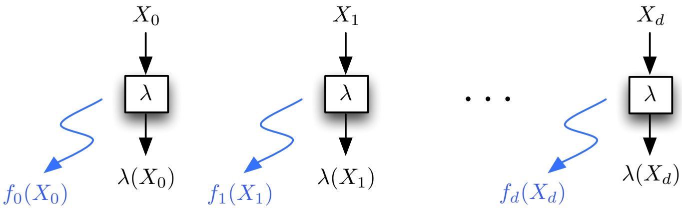
Linear functions

- Given a sharing $X = X_0 \oplus X_1 \oplus \cdots \oplus X_d$



Linear functions

- Given a sharing $X = X_0 \oplus X_1 \oplus \cdots \oplus X_d$



- For $f_i \in \mathcal{N}(1/\psi)$ with $\psi = O(|\mathcal{X}|^{\frac{1}{2}}\,\omega)$, we show

$$\mathrm{MI}\big(X;\ (f_0(X_0), f_1(X_1), \ldots, f_d(X_d))\big) \leq \frac{1}{\omega^{d+1}}$$

CryptoExperts

Multiplications

- Inputs: sharings $\bigoplus_i A_i = g(X)$ and $\bigoplus_i B_i = g(X)$ where $X =$ s-box input
- First step: cross-products

$$\begin{array}{cccc} A_0 \times B_0 & A_0 \times B_1 & \cdots & A_0 \times B_d \\ A_1 \times B_0 & A_1 \times B_1 & \cdots & A_1 \times B_d \\ \vdots & \vdots & \ddots & \vdots \\ A_d \times B_0 & A_d \times B_1 & \cdots & A_d \times B_d \end{array}$$

CryptoExperts

Multiplications

- Inputs: sharings $\bigoplus_i A_i = g(X)$ and $\bigoplus_i B_i = g(X)$ where $X =$ s-box input
- First step: cross-products

$$\begin{matrix} A_0 \times B_0 & A_0 \times B_1 & \cdots & A_0 \times B_d \\ A_1 \times B_0 & A_1 \times B_1 & \cdots & A_1 \times B_d \\ \vdots & \vdots & \ddots & \vdots \\ A_d \times B_0 & A_d \times B_1 & \cdots & A_d \times B_d \end{matrix}$$

$$\begin{matrix} f_{0,0}(A_0, B_0) & f_{0,1}(A_0, B_1) & \cdots & f_{0,d}(A_0, B_d) \\ f_{1,0}(A_1, B_0) & f_{1,1}(A_1, B_1) & \cdots & f_{1,d}(A_1, B_d) \\ \vdots & \vdots & \ddots & \vdots \\ f_{d,0}(A_d, B_0) & f_{d,1}(A_d, B_1) & \cdots & f_{d,d}(A_d, B_d) \end{matrix}$$

Multiplications

- Inputs: sharings $\bigoplus_i A_i = g(X)$ and $\bigoplus_i B_i = g(X)$ where $X =$ s-box input
- First step: cross-products

$$\begin{matrix} A_0 \times B_0 & A_0 \times B_1 & \cdots & A_0 \times B_d \\ A_1 \times B_0 & A_1 \times B_1 & \cdots & A_1 \times B_d \\ \vdots & \vdots & \ddots & \vdots \\ A_d \times B_0 & A_d \times B_1 & \cdots & A_d \times B_d \end{matrix}$$

- For $f_{i,j} \in \mathcal{N}(1/\psi)$ with $\psi = O(|\mathcal{X}|^{\frac{3}{2}} d\, \omega)$ we show

$$\mathrm{MI}\big(X;\ (f_{i,j}(A_i, B_j))_{i,j}\big) \leq \frac{1}{\omega^{d+1}}$$

Multiplications

- Second step: refreshing
- Apply on each column and one row of

$$\begin{array}{cccc}
A_0 \times B_0 & A_0 \times B_1 & \cdots & A_0 \times B_d \\
A_1 \times B_0 & A_1 \times B_1 & \cdots & A_1 \times B_d \\
\vdots & \vdots & \ddots & \vdots \\
A_d \times B_0 & A_d \times B_1 & \cdots & A_d \times B_d
\end{array}$$

- We get a fresh $(d+1)^2$-sharing of $A \times B$

$$\begin{array}{cccc}
V_{0,0} & V_{0,1} & \cdots & V_{0,d} \\
V_{1,0} & V_{1,1} & \cdots & V_{1,d} \\
\vdots & \vdots & \ddots & \vdots \\
V_{d,0} & V_{d,1} & \cdots & V_{d,d}
\end{array}$$

Multiplications

- Third step: summing rows
- Takes d elementary calculations (XORs) per row:

$$
\begin{aligned}
T_{i,1} &\leftarrow V_{i,0} \oplus V_{i,1} \\
T_{i,2} &\leftarrow T_{i,1} \oplus V_{i,2} \\
&\vdots \\
T_{i,d} &\leftarrow T_{i,d-1} \oplus V_{i,d}
\end{aligned}
$$

Multiplications

- Third step: summing rows
- Takes d elementary calculations (XORs) per row:

$$
\begin{aligned}
T_{i,1} &\leftarrow V_{i,0} \oplus V_{i,1} &&\rightsquigarrow f_{i,1}(V_{i,0}, V_{i,1}) \\
T_{i,2} &\leftarrow T_{i,1} \oplus V_{i,2} &&\rightsquigarrow f_{i,2}(T_{i,1}, V_{i,2}) \\
&\vdots && \vdots \\
T_{i,d} &\leftarrow T_{i,d-1} \oplus V_{i,d} &&\rightsquigarrow f_{i,d}(T_{i,d-1}, V_{i,d})
\end{aligned}
$$

Multiplications

- Third step: summing rows
- Takes d elementary calculations (XORs) per row:

$$
\begin{array}{lll}
T_{i,1} \leftarrow V_{i,0} \oplus V_{i,1} & \rightsquigarrow & f_{i,1}(V_{i,0}, V_{i,1}) \\
T_{i,2} \leftarrow T_{i,1} \oplus V_{i,2} & \rightsquigarrow & f_{i,2}(T_{i,1}, V_{i,2}) \\
\quad\vdots & \vdots & \\
T_{i,d} \leftarrow T_{i,d-1} \oplus V_{i,d} & \rightsquigarrow & f_{i,d}(T_{i,d-1}, V_{i,d})
\end{array}
$$

- For $f_{i,j} \in \mathcal{N}(1/\psi)$ with $\psi = O(|\mathcal{X}|^{\frac{3}{2}}\omega)$, we show

$$\mathrm{MI}\big(X;\ (F_0, F_1, \ldots, F_d)\big) \leq \frac{1}{\omega^{d+1}}$$

where $F_i = \big(f_{i,1}(V_{i,0}, V_{i,1}), f_{i,2}(T_{i,1}, V_{i,2}), \ldots, f_{i,d}(T_{i,d-1}, V_{i,d})\big)$

Putting everything together

- Several sequences of subcomputations, each leaking L_t with

$$\mathrm{MI}((m,k);L_t) \leq \frac{1}{\omega^{d+1}}$$

- Use of share-refreshing between each sequence
  - $(L_t)_t$ are mutually independent given (m,k)

- We hence have

$$\mathrm{MI}\big((m,k);(L_1,L_2,\ldots,L_T)\big) \leq \sum_{t=1}^{T} \mathrm{MI}((m,k);L_t) \leq \frac{T}{\omega^{d+1}}$$

Improved security proof

- [Duc-Dziembowski-Faust. EUROCRYPT 2014]
- Security reduction: probing model $\Rightarrow$ noisy model
- ISW scheme secure in the noisy model
- No need for leak-free component !

Improved security proof

- Consider $y_1 \leftarrow \mathsf{SC_1}(x_1),\ y_2 \leftarrow \mathsf{SC_2}(x_2),\ \ldots,\ y_n \leftarrow \mathsf{SC_n}(x_n)$
- t-probing model: $\ell = (x_i)_{i \in I}$ with $|I| = t$
- ε-random probing model: $\ell = (\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n))$
  - where φ_i is a *ε-identity function* *i.e.*

$$\text{with } \varphi_i(x) = \begin{cases} x \text{ with proba } \varepsilon \\ \perp \text{ with proba } 1 - \varepsilon \end{cases}$$

- δ-noisy model: $\ell = (f_1(x_1), f_2(x_2), \ldots, f_n(x_n))$

$$\text{with } \beta(X | f_i(X)) \leq \delta \quad (\text{here } \|\cdot\| = L_1)$$

Improved security proof

- Consider $y_1 \leftarrow \mathsf{SC}_1(x_1),\ y_2 \leftarrow \mathsf{SC}_2(x_2),\ \ldots,\ y_n \leftarrow \mathsf{SC}_n(x_n)$
- t-probing model: $\ell = (x_i)_{i \in I}$ with $|I| = t$
- ε-random probing model: $\ell = (\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n))$
  - where φ_i is a *ε-identity function i.e.*

$$\text{with } \varphi_i(x) = \begin{cases} x \text{ with proba } \varepsilon \\ \perp \text{ with proba } 1 - \varepsilon \end{cases}$$

- δ-noisy model: $\ell = (f_1(x_1), f_2(x_2), \ldots, f_n(x_n))$

$$\text{with } \beta(X | f_i(X)) \leq \delta \quad (\text{here } \|\cdot\| = L_1)$$

t-probing security $\Rightarrow$ ε-random probing security $\Rightarrow$ δ-noisy security

From probing to random probing

- ε-random probing adv. $\mathcal{A}_{rp} \Rightarrow t$-probing adv. $\mathcal{A}_p$
  - with $t = 2n\varepsilon - 1$
- $\mathcal{A}_p$ works as follows
  - sample $(z_1, z_2, \ldots, z_n)$ where $z_i = \begin{cases} 1 \text{ with proba } \varepsilon \\ 0 \text{ with proba } 1 - \varepsilon \end{cases}$
  - set $I = \{i \mid z_i = 1\}$, if $|I| > t$ return $\perp$
  - get $(x_i)_{i \in I}$
  - call $\mathcal{A}_{rp}$ on $(y_1, y_2, \ldots, y_n)$ where $y_i = \begin{cases} x_i \text{ if } i \in I \\ \perp \text{ if } i \notin I \end{cases}$
- If $|I| \leq t$: $(y_1, y_2, \ldots, y_n) \sim (\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n))$
- Chernoff bound: $\Pr[|I| > t] \leq \exp(-t/6)$
- $\forall \mathcal{A}_{rp}\ \exists \mathcal{A}_p$: $\mathsf{Adv}(\mathcal{A}_p) \leq \mathsf{Adv}(\mathcal{A}_{rp}) - \exp(-t/6)$

From probing to random probing

- ε-random probing adv. $\mathcal{A}_{rp} \Rightarrow t$-probing adv. $\mathcal{A}_p$
  - with $t = 2n\varepsilon - 1$
- $\mathcal{A}_p$ works as follows
  - sample $(z_1, z_2, \ldots, z_n)$ where $z_i = \begin{cases} 1 \text{ with proba } \varepsilon \\ 0 \text{ with proba } 1 - \varepsilon \end{cases}$
  - set $I = \{i \mid z_i = 1\}$, if $|I| > t$ return $\perp$
  - get $(x_i)_{i \in I}$
  - call $\mathcal{A}_{rp}$ on $(y_1, y_2, \ldots, y_n)$ where $y_i = \begin{cases} x_i \text{ if } i \in I \\ \perp \text{ if } i \notin I \end{cases}$
- If $|I| \leq t$: $(y_1, y_2, \ldots, y_n) \sim (\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n))$
- Chernoff bound: $\Pr[|I| > t] \leq \exp(-t/6)$
- $\forall \mathcal{A}_{rp}$: $\mathrm{Adv}(\mathcal{A}_{rp}) \leq \max_{\mathcal{A}_p} \mathrm{Adv}(\mathcal{A}_p) + \exp(-t/6)$

CryptoExperts

From random probing to noisy leakage

- Main lemma: every f s.t. $\beta(X|f(X)) \leq \delta$ can be written:

  $$f = f' \circ \varphi$$

  where φ is an ε-identity function with $\varepsilon \leq \delta|\mathcal{X}|$, and

  $$\left.\begin{array}{l} f \text{ efficient to sample} \\ \Pr[f(x) = y] \text{ eff. computable} \end{array}\right\} \Rightarrow f' \text{ efficient to sample}$$

- δ-noisy adversary $\mathcal{A}_n \Rightarrow \varepsilon$-random probing adv. $\mathcal{A}_{rp}$
  - get $(\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n))$
  - call $\mathcal{A}_n$ on $(f_1' \circ \varphi_1(x_1), f_2' \circ \varphi_2(x_1), \ldots, f_n' \circ \varphi_n(x_n))$

- $\forall \mathcal{A}_n \; \exists \mathcal{A}_{rp} : \mathsf{Adv}(\mathcal{A}_{rp}) = \mathsf{Adv}(\mathcal{A}_n)$

From random probing to noisy leakage

- Main lemma: every f s.t. $\beta(X|f(X)) \leq \delta$ can be written:

  $$f = f' \circ \varphi$$

  where φ is an ε-identity function with $\varepsilon \leq \delta|\mathcal{X}|$, and

  $$\left.\begin{array}{l} f \text{ efficient to sample} \\ \Pr[f(x) = y] \text{ eff. computable} \end{array}\right\} \Rightarrow f' \text{ efficient to sample}$$

- δ-noisy adversary $\mathcal{A}_n \Rightarrow \varepsilon$-random probing adv. $\mathcal{A}_{rp}$
  - get $(\varphi_1(x_1), \varphi_2(x_2), \ldots, \varphi_n(x_n))$
  - call $\mathcal{A}_n$ on $(f_1' \circ \varphi_1(x_1), f_2' \circ \varphi_2(x_1), \ldots, f_n' \circ \varphi_n(x_n))$

- $\forall \mathcal{A}_n$: $\mathsf{Adv}(\mathcal{A}_n) \leq \max_{\mathcal{A}_{rp}} \mathsf{Adv}(\mathcal{A}_{rp})$

Combining both reductions

- Security against t-probing $\Rightarrow$ security against δ-noisy

  $$\text{where} \quad \delta = \frac{t+1}{2n|\mathcal{X}|}$$

  - $\exp(-t/6)$ must be negligible $\Rightarrow t \geq 8.65\,\kappa$

- ISW scheme with d-sharing is secure against δ-noisy attackers

  $$\text{where} \quad \delta = \frac{d}{n|\mathcal{X}|} \qquad (\text{and } d \geq 17.5\,\kappa)$$

- For ISW-multiplication $n = O(d^2)$ and $\mathcal{X} = \mathbb{F} \times \mathbb{F}$ giving

  $$\delta = O(1/d|\mathbb{F}|^2) \;\Rightarrow\; \psi = O(d|\mathbb{F}|^2)$$

- Limitation: ψ is still in $O(d)$

CryptoExperts

Conclusion

- New practically relevant model for leaking computation: the noisy model
- Need for practical investigations for the bias estimation
- Only 2 works proposing formal proofs in this model
- Open issues:
  - a scheme secure with constant noise
  - secure implementations with different kind of randomization (*e.g.* exponent/message blinding for RSA/ECC)

CryptoExperts