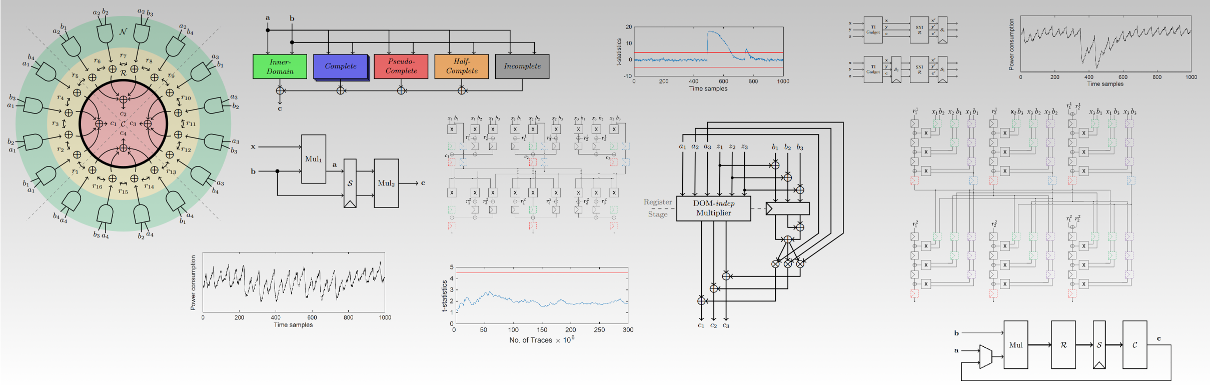



# Glitch-Resistant Masking Revisited

## or Why Proofs in the Robust Probing Model are Needed

**Thorben Moos[1], Amir Moradi[1], Tobias Schneider[2] and François-Xavier Standaert[2]**

[1]Horst Görtz Institute for IT Security, Ruhr-Universität Bochum, Germany

[2]ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium

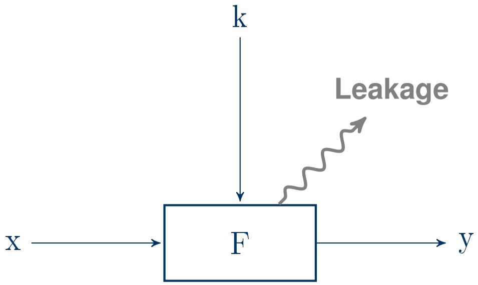
August 27th, 2019

Section 1

# Introduction

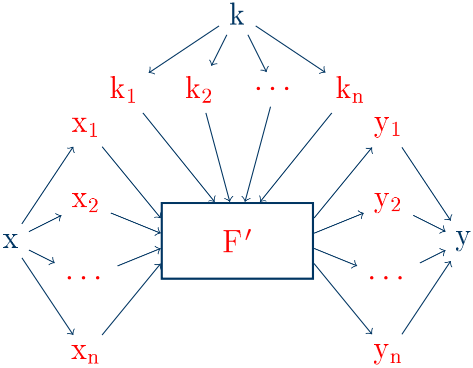
# Physical Attacks

Introduction




- **Physical** characteristics used to extract secrets:
  - Timing
  - Power
  - EM
- Countermeasures to increase attack complexity:
  - **Masking**
  - Hiding
  - Re-keying

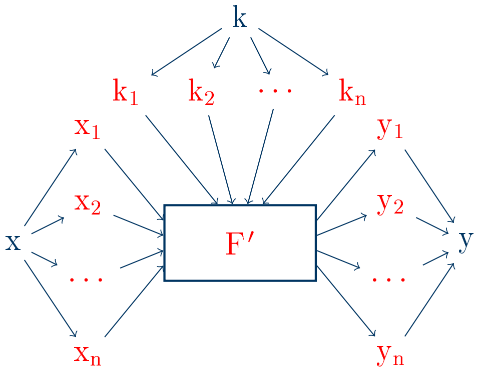
# Concept of Masking
## Introduction




- Encode sensitive variables into shares
- Compute **securely** on shares
- Decode at end to recover result

# Concept of Masking

## Introduction




- Encode sensitive variables into shares
- Compute **securely** on shares
- Decode at end to recover result

Masking if implemented **correctly** increases the attack complexity **exponentially** in the number of shares.
*(assuming sufficient noise)*

# Security Notions

## Introduction

- Masked algorithms can be proven secure
- **Common Solution:** Probing model[1]

**Definition ($t$-Probing Security)**

A circuit C is $t$-probing secure if and only if every $t$-tuple of its intermediate variables is independent of any sensitive variable.

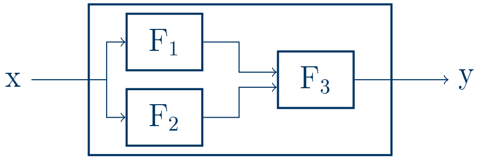
[1] Y. Ishai, A. Sahai and D. Wagner, *Private Circuits: Securing Hardware against Probing Attacks*, CRYPTO 2003

# Security Notions
## Introduction

- Masked algorithms can be proven secure
- **Common Solution:** Probing model[1]

**Definition ($t$-Probing Security)**

A circuit C is $t$-probing secure if and only if every $t$-tuple of its intermediate variables is independent of any sensitive variable.




**Example:**

- 3rd-order masking
- Any possible combination of **three** probes should not reveal secret

[1] Y. Ishai, A. Sahai and D. Wagner, *Private Circuits: Securing Hardware against Probing Attacks*, CRYPTO 2003

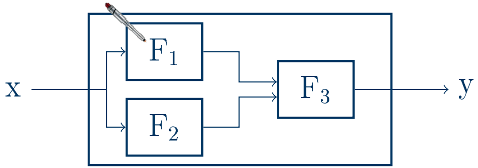
# Security Notions

## Introduction

- Masked algorithms can be proven secure
- **Common Solution:** Probing model[1]

**Definition ($t$-Probing Security)**

A circuit C is $t$-probing secure if and only if every $t$-tuple of its intermediate variables is independent of any sensitive variable.




**Example:**

- 3rd-order masking
- Any possible combination of **three** probes should not reveal secret

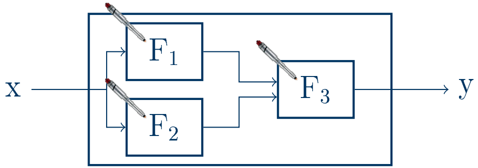
[1] Y. Ishai, A. Sahai and D. Wagner, *Private Circuits: Securing Hardware against Probing Attacks*, CRYPTO 2003

# Security Notions
## Introduction

- Masked algorithms can be proven secure
- **Common Solution:** Probing model[1]

**Definition ($t$-Probing Security)**

A circuit C is $t$-probing secure if and only if every $t$-tuple of its intermediate variables is independent of any sensitive variable.




**Example:**

- 3rd-order masking
- Any possible combination of **three** probes should not reveal secret

[1] Y. Ishai, A. Sahai and D. Wagner, *Private Circuits: Securing Hardware against Probing Attacks*, CRYPTO 2003

# Security Notions

## Introduction

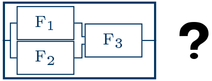
- Scales badly with number of probes and complexity of algorithm
- Prove smaller sub-gadgets and compose **securely**

[2] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Gregoire, P.-Y. Strub and R. Zucchini, *Strong Non-Interference and Type-Directed Higher-Order Masking*, CCS 2016

# Security Notions

## Introduction

- Scales badly with number of probes and complexity of algorithm
- Prove smaller sub-gadgets and compose **securely**

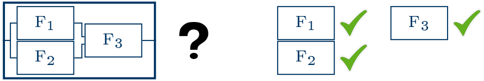



[2] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Gregoire, P.-Y. Strub and R. Zucchini, *Strong Non-Interference and Type-Directed Higher-Order Masking*, CCS 2016

# Security Notions

Introduction

- Scales badly with number of probes and complexity of algorithm
- Prove smaller sub-gadgets and compose **securely**




[2] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Gregoire, P.-Y. Strub and R. Zucchini, *Strong Non-Interference and Type-Directed Higher-Order Masking*, CCS 2016

# Security Notions
Introduction

- Scales badly with number of probes and complexity of algorithm
- Prove smaller sub-gadgets and compose **securely**


[2] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Gregoire, P.-Y. Strub and R. Zucchini, *Strong Non-Interference and Type-Directed Higher-Order Masking*, CCS 2016

# Security Notions
Introduction

- Scales badly with number of probes and complexity of algorithm
- Prove smaller sub-gadgets and compose **securely**


- **Common Solution:** (Strong) Non-Interference[2]

**Definition (t−(Strong) Non-Interference)**

A circuit gadget G is $\mathrm{t}-$*(Strong) Non-Interfering* (t-(S)NI) if and only if for any set of $\mathrm{t}_1$ probes on its intermediate values and every set of $\mathrm{t}_2$ probes on its output shares with $\mathrm{t}_1 + \mathrm{t}_2 \leqslant \mathrm{t}$, the totality of the probes can be simulated with $\mathrm{t}_1 + \mathrm{t}_2$ (only $\mathrm{t}_1$) shares of each input.

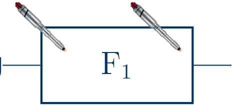
[2] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Gregoire, P.-Y. Strub and R. Zucchini, *Strong Non-Interference and Type-Directed Higher-Order Masking*, CCS 2016

# Potential Flaws

Introduction

**Local Flaw:** Probing security of masked **module** is reduced.

**Example:** 2nd-order masking

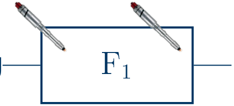


# Potential Flaws

Introduction

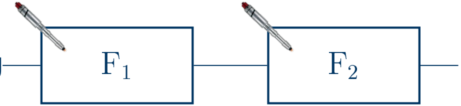
**Local Flaw:** Probing security of masked **module** is reduced.

**Example:** 2nd-order masking




**Compositional Flaw:** Probing security of **composition** of modules is reduced.

**Example:** 2nd-order masking



# Robust Probing

## Introduction

- Physical defaults (glitches, transitions, coupling) **reduce masking order** in practice
- Numerous **higher-order** hardware-oriented masking schemes:
  - **CMS:** Consolidated Masking Schemes
  - **DOM:** Domain-Oriented Masking
  - **UMA:** Unified Masking Approach
  - **GLM:** Generic Low-Latency Masking

# Robust Probing
## Introduction

- Physical defaults (glitches, transitions, coupling) **reduce masking order** in practice
- Numerous **higher-order** hardware-oriented masking schemes:
  - **CMS:** Consolidated Masking Schemes
  - **DOM:** Domain-Oriented Masking
  - **UMA:** Unified Masking Approach
  - **GLM:** Generic Low-Latency Masking
- Due to lack of model: Mostly focused on glitch-resistant (local) probing security
- Dedicated extension of probing model to hardware masking:

**Composable Masking Schemes in the Presence of Physical Defaults and the Robust Probing Model**

Sebastian Faust[1], Vincent Grosso[1,2], Santos Merino Del Pozo[3], Clara Paglialonga[1], François-Xavier Standaert[3]

# Overview

Introduction

In this paper:

- Analysis of higher-order HW masking schemes
  - CMS - **local**
  - DOM - **local**
  - UMA - **compositional**
  - GLM - **local** + **compositional**

  } Strong case for unified HW security notion (e.g., robust probing model)
- Experiments and evaluation of practical impact of flaws
- **Conclusion:** Always verify local and compositional security in adequate model

# Overview
Introduction

In this paper:
- Analysis of higher-order HW masking schemes
  - CMS - **local**
  - DOM - **local**
  - UMA - **compositional**
  - GLM - **local** + **compositional**

  } Strong case for unified HW security notion (e.g., robust probing model)
- Experiments and evaluation of practical impact of flaws
- **Conclusion:** Always verify local and compositional security in adequate model

**Disclaimer**

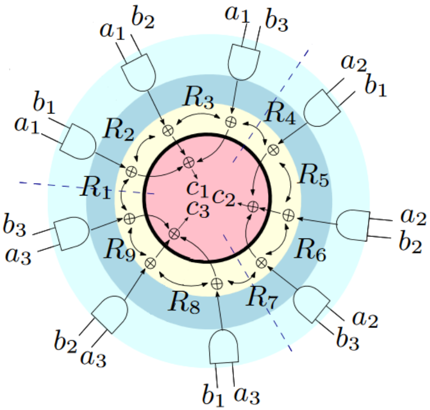
Most of the flaws are in instantiations/compositions which are not explicitly given in the sources, and their specific instantiations at lower orders should not be affected by our flaws. The discussed flaws can still result in insecure designs when used by others.

Section 2

# Local Flaws

# Consolidated Masking Scheme

## Local Flaws

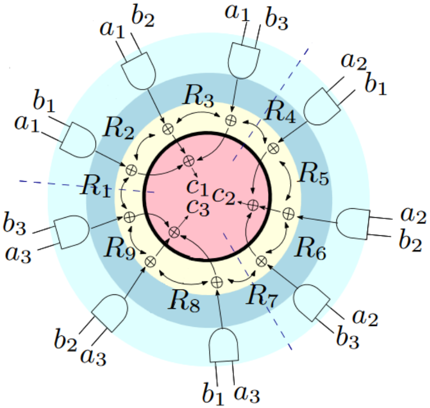



2nd-order masking

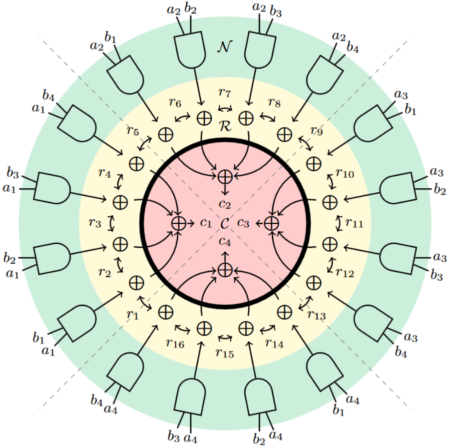
- First proposed at CRYPTO 2015 as d+1 masking scheme
- Then used at CHES 2016 to mask AES with d+1 shares for d=1 and d=2
- *"Our construction is generic and can be extended to higher orders"*
- *"The ring structure of the refreshing in the general, higher-order case..."*

# Consolidated Masking Scheme

## Local Flaws




2nd-order masking

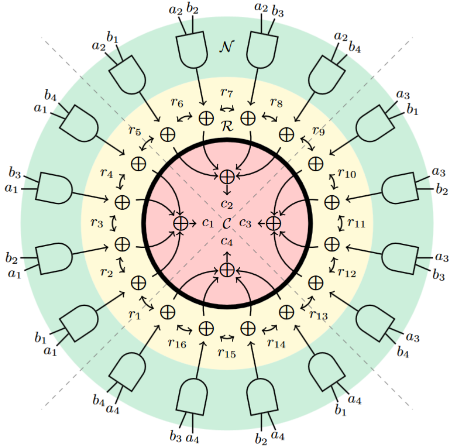



3rd-order masking

# Consolidated Masking Scheme

## Local Flaws

- **Local Flaw:** Attack with 3 *standard* probes
- Authors already proposed fix
- **Compositional** security is still open issue

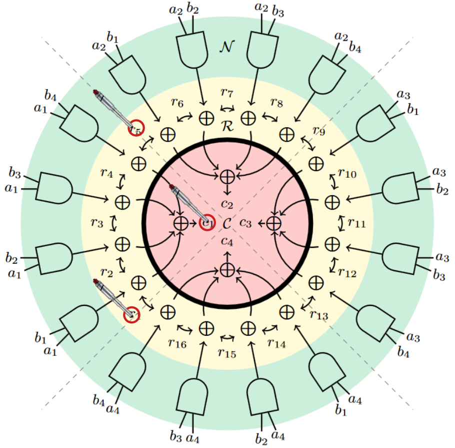



3rd-order masking

# Consolidated Masking Scheme

## Local Flaws

- **Local Flaw:** Attack with 3 *standard* probes
- Authors already proposed fix
- **Compositional** security is still open issue




3rd-order masking

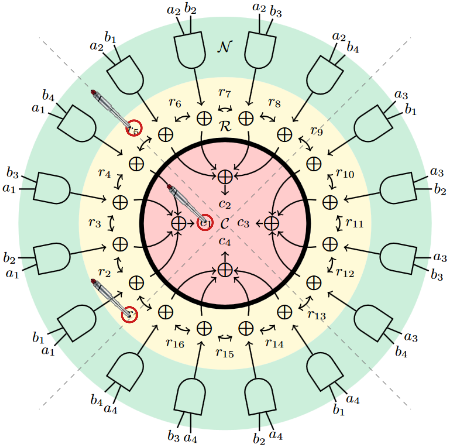
# Consolidated Masking Scheme

## Local Flaws

- **Local Flaw:** Attack with 3 *standard* probes
- Authors already proposed fix
- **Compositional** security is still open issue

**In Paper:** Domain-Oriented Masking

$(\lceil d/2 \rceil + 1)$th-order flaw with *extended* probes for DOM-*dep* multiplication



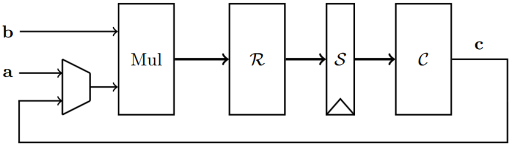
3rd-order masking

Section 3

# Compositional Flaws

# Generic Low-Latency Masking

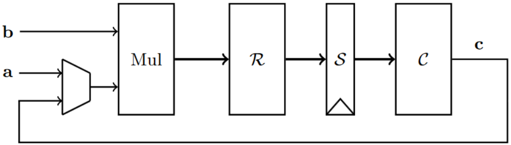
## Compositional Flaws




- Introduced at CHES 2018
- Proposes to use **CMS** refresh $\mathcal{R}$
- Suffers from same flaws
  - **Local Flaw**
  - **Compositional Flaw**
- Fix requires secure refresh algorithm with low-latency

# Generic Low-Latency Masking

## Compositional Flaws



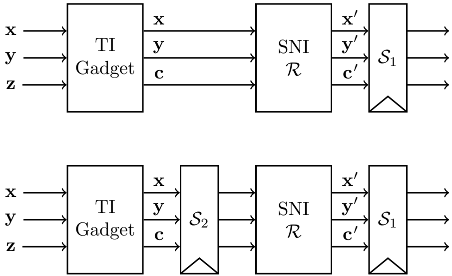

**In Paper:** Unified Masking Approach

A systematic composability flaw

- Introduced at CHES 2018
- Proposes to use **CMS** refresh $\mathcal{R}$
- Suffers from same flaws
  - **Local Flaw**
  - **Compositional Flaw**
- Fix requires secure refresh algorithm with low-latency

# On the Need of the Robust Probing Model

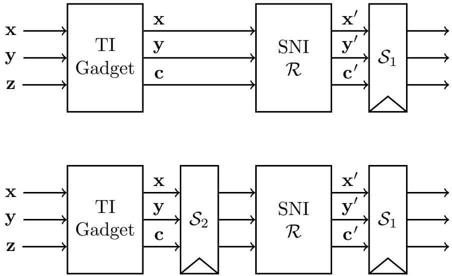
## Compositional Flaws




- Security depends on combinatorial combinations, refreshs, register stages
- Not sufficient to solve **glitch-resistance** and **composability** separately
- **Example:** Non-completeness and SNI

# On the Need of the Robust Probing Model

## Compositional Flaws




- Security depends on combinatorial combinations, refreshs, register stages
- Not sufficient to solve **glitch-resistance** and **composability** separately
- **Example:** Non-completeness and SNI
- **Solution:** Unified model
- **Note:** TI can be composable, but hard to formally prove for higher orders

Section 4

# Practical Impact

# Experiments
## Practical Impact

- SAKURA-G (Spartan-6 FPGA), Clock: 6 MHz, Sampling: 500 MS/s
- Leakage detection with fixed-vs-random $t$-test

**Results:**

- All flaws are practically **detecable** / **Not** necessarily **reduce** practical security
- Bias caused by the flaws have low amplitude
- All order reductions multivariate

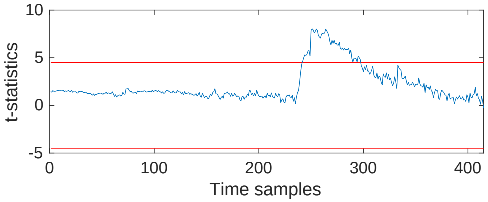
# Experiments

Practical Impact

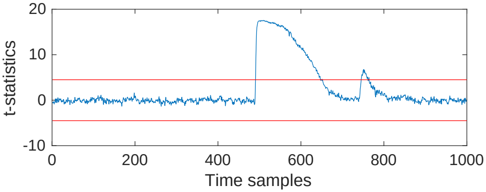
- SAKURA-G (Spartan-6 FPGA), Clock: 6 MHz, Sampling: 500 MS/s
- Leakage detection with fixed-vs-random $t$-test

**Results:**

- All flaws are practically **detecable** / **Not** necessarily **reduce** practical security
- Bias caused by the flaws have low amplitude
- All order reductions multivariate



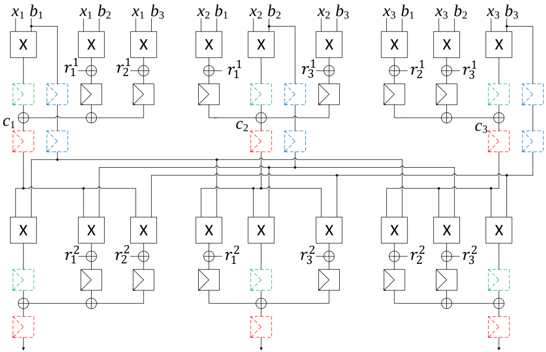

(c) 3rd-order multivariate (CMS)




(d) 4th-order univariate (CMS)

# Composability in Hardware - A Matter of Registers

## Practical Impact




- Register placement is essential
- Used by TI glitch propagation
- For DOM initially claimed that the DOM-*indep* multiplier does not require output registers
- Without output registers (red) the construction is not composable
- Pipeline registers can be important

Section 5

# Conclusion

# Summary
## Conclusion

- Extensive security proofs not yet established in HW masking
- Lack of appropriate model for higher orders and composability

# Summary
## Conclusion

- Extensive security proofs not yet established in HW masking
- Lack of appropriate model for higher orders and composability

**Our results show:**

- No HW masking provides **local and compositional** higher-order security
- Practical impact could be **limited**, flaws are still an **undesirable** source of risk
- **Currently:** Only adapted DOM-*indep* multiplication was robustly proven secure

# Summary
## Conclusion

- Extensive security proofs not yet established in HW masking
- Lack of appropriate model for higher orders and composability

**Our results show:**

- No HW masking provides **local and compositional** higher-order security
- Practical impact could be **limited**, flaws are still an **undesirable** source of risk
- **Currently:** Only adapted DOM-*indep* multiplication was robustly proven secure

**In the future:**

- Fix flaws and prove existing schemes
- Design new (improved) schemes

Thank you for your attention.

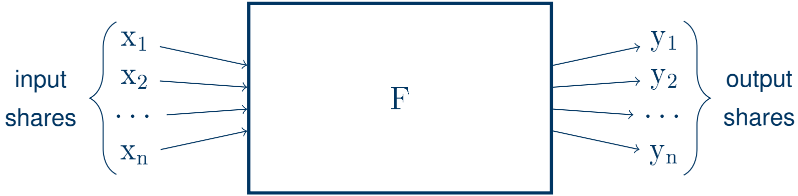
Any questions?

Section 6

# Backup

# Security Notions
Backup

## Example:

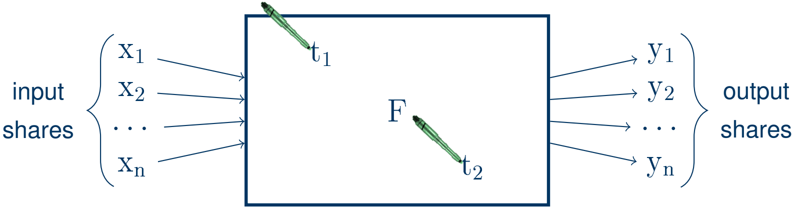



[3] G. Cassiers, F.-X. Standaert, *Trivially and Efficiently Composing Masked Gadgets with Probe Isolating Non-Interference*, eprint 2018/438

# Security Notions

Backup

## Example:

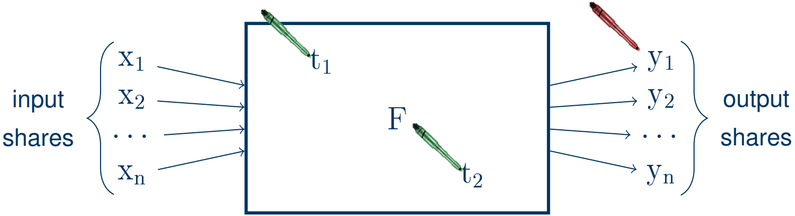



[3] G. Cassiers, F.-X. Standaert, *Trivially and Efficiently Composing Masked Gadgets with Probe Isolating Non-Interference*, eprint 2018/438

# Security Notions

Backup

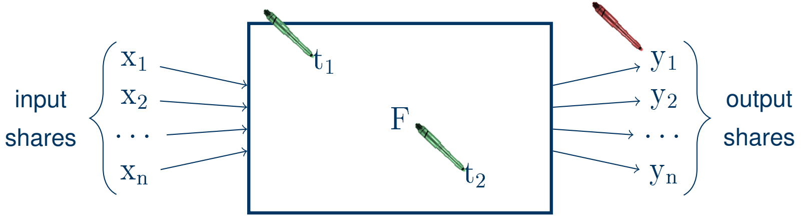
## Example:




[3] G. Cassiers, F.-X. Standaert, *Trivially and Efficiently Composing Masked Gadgets with Probe Isolating Non-Interference*, eprint 2018/438

# Security Notions
Backup

**Example:**




Simulate with

- **NI:** 2 + 1 = 3
- **SNI:** 2 = 2

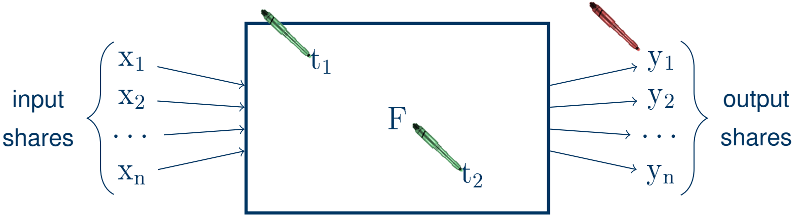
input shares.

[3] G. Cassiers, F.-X. Standaert, *Trivially and Efficiently Composing Masked Gadgets with Probe Isolating Non-Interference*, eprint 2018/438

# Security Notions

Backup

**Example:**




Simulate with

- **NI:** 2 + 1 = 3
- **SNI:** 2 = 2

input shares.

- Enables reasoning about **secure** composition of modules
- Has been used to prove various SW-oriented masked algorithms/gadgets
- Alternative notions allow trade-offs, e.g., PINI[3]

[3] G. Cassiers, F.-X. Standaert, *Trivially and Efficiently Composing Masked Gadgets with Probe Isolating Non-Interference*, eprint 2018/438