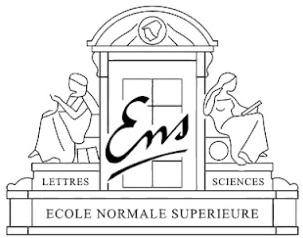


THALES

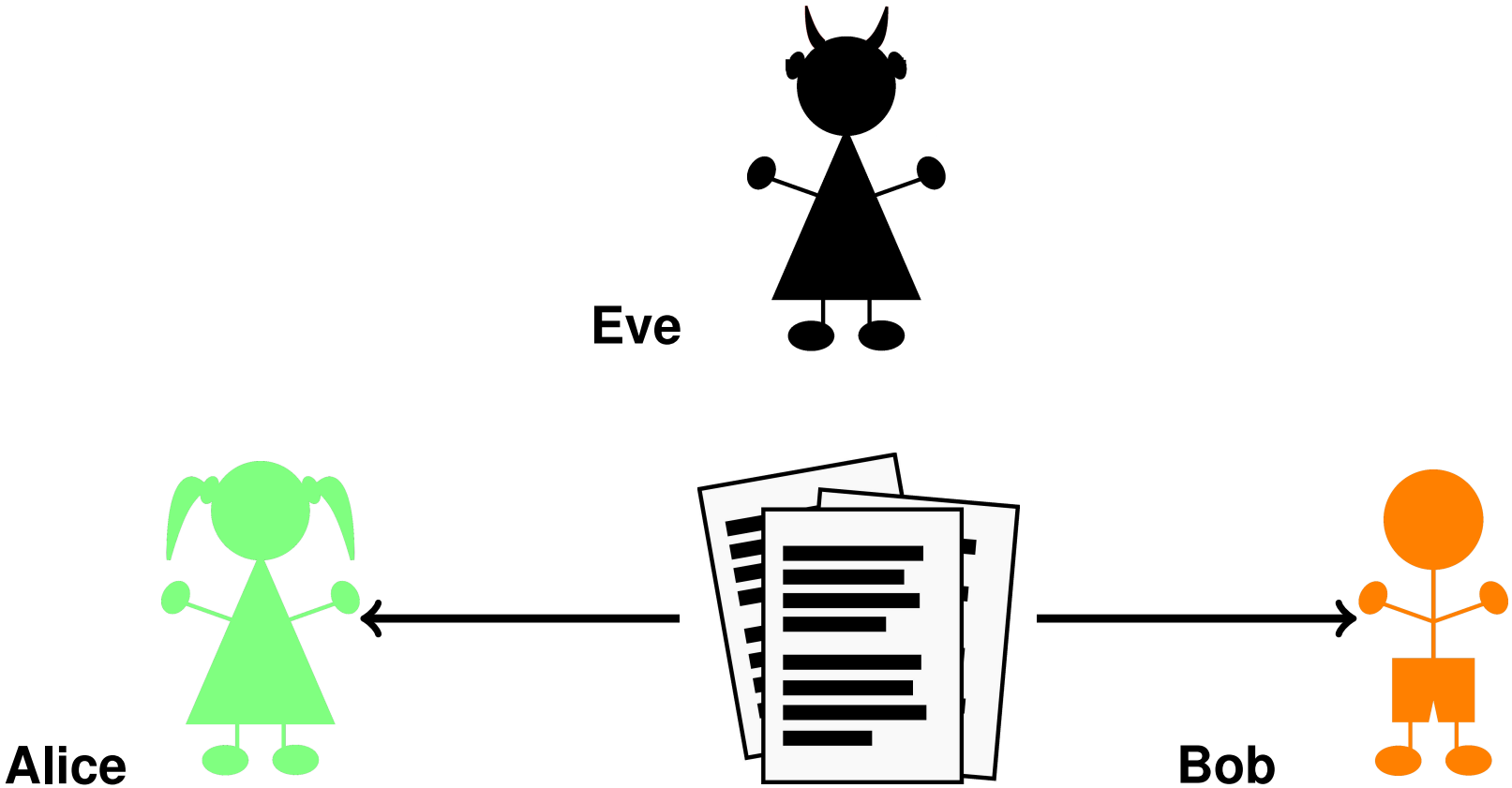
Security of Cryptosystems Against Power-Analysis Attacks

PhD Defense

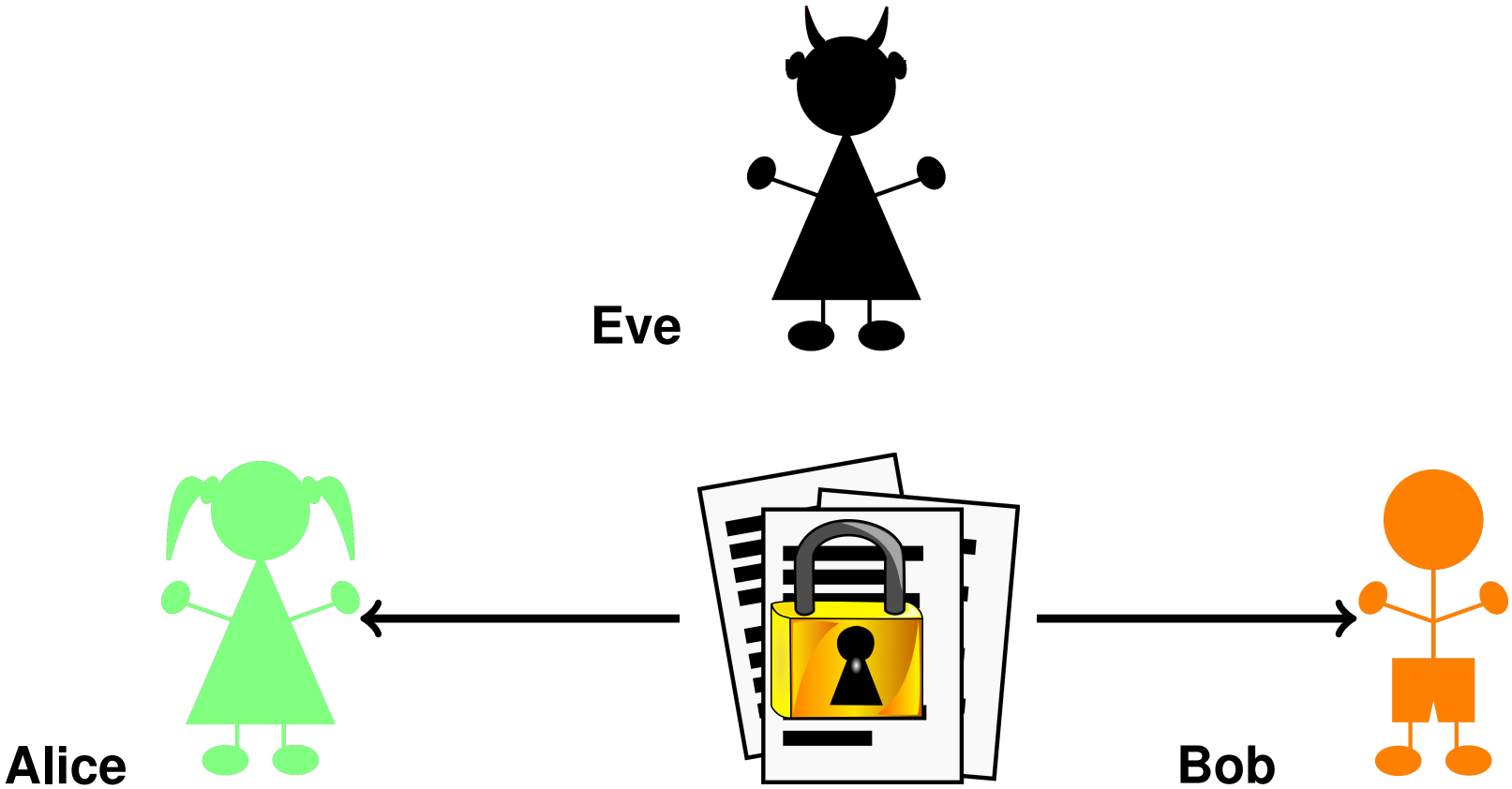
October 22, 2015

Presented by Sonia Belaïd

Cryptology



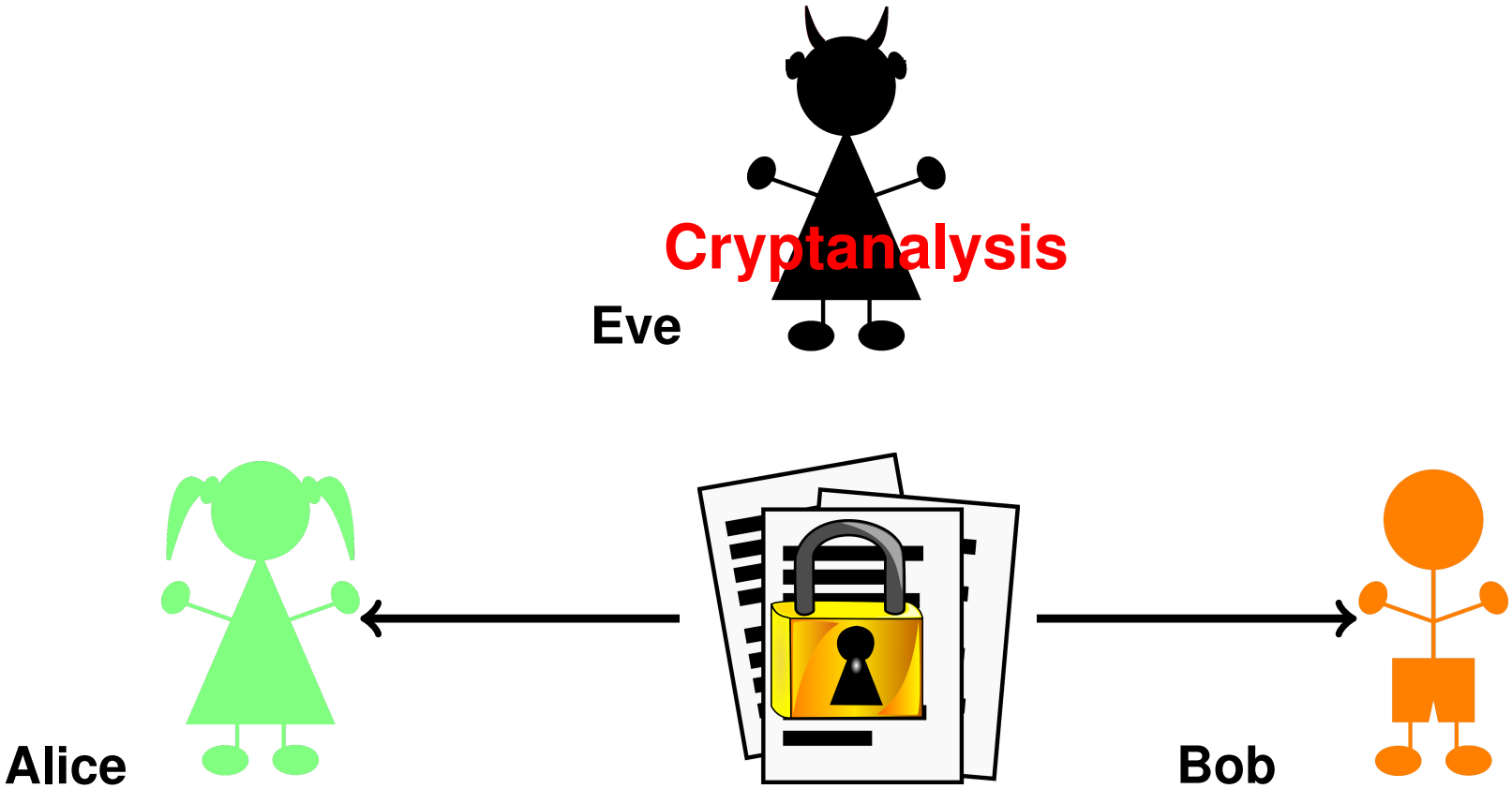
Cryptology



Cryptography

- ➜ Confidentiality
- ➜ Authentication
- ➜ Integrity

Cryptology



Cryptography

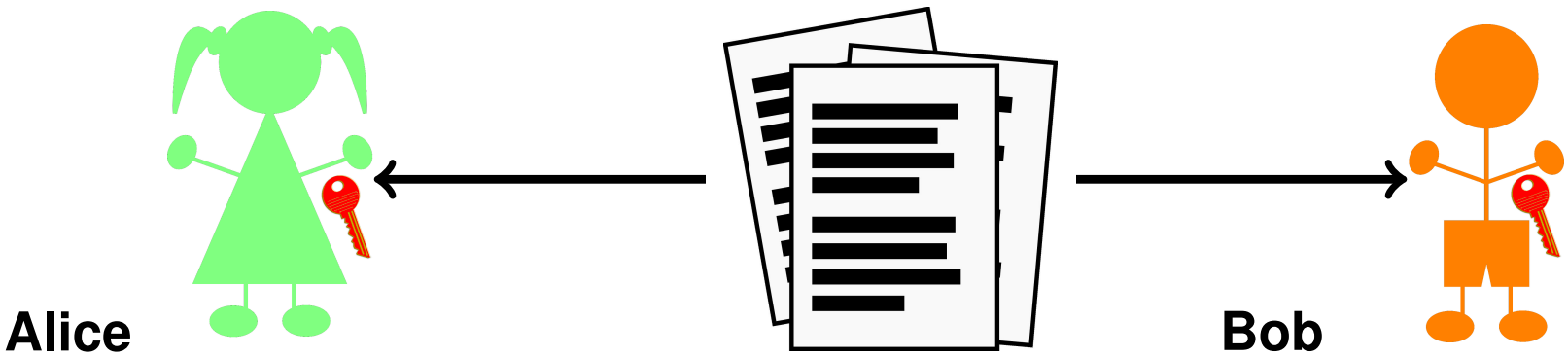
- ➜ Confidentiality
- ➜ Authentication
- ➜ Integrity

Cryptography is divided into

- Asymmetric cryptography
- Symmetric cryptography

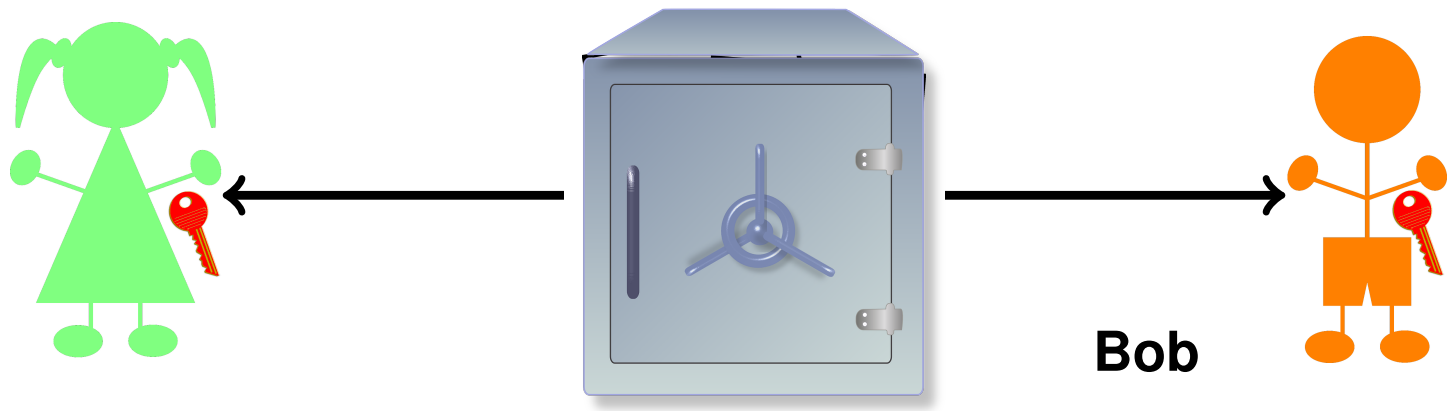
Cryptography is divided into

- Asymmetric cryptography
- Symmetric cryptography



Cryptography is divided into

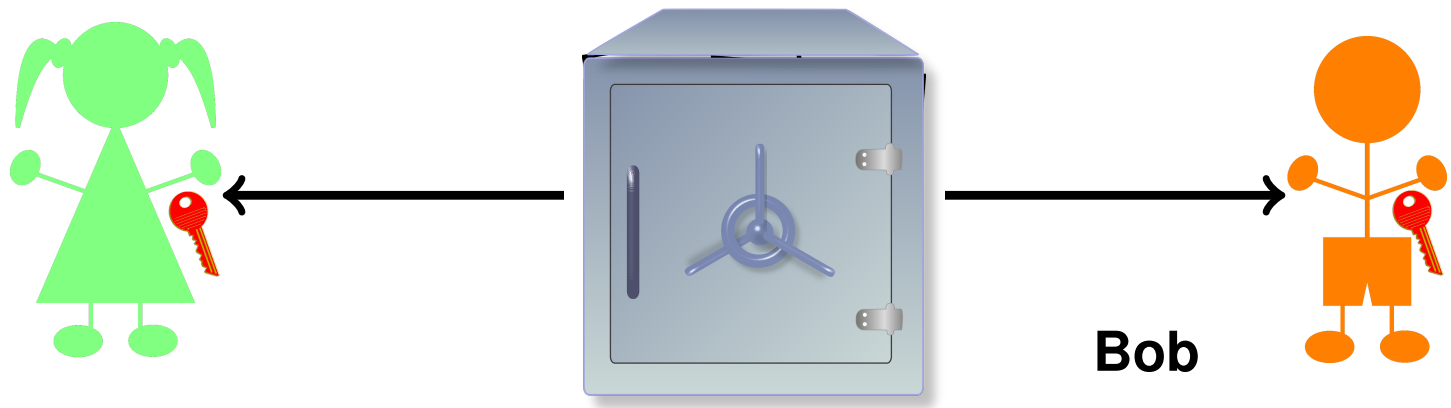
- Asymmetric cryptography
- Symmetric cryptography



Cryptography is divided into

- Asymmetric cryptography
- Symmetric cryptography

c = ‘fbjdsiqfesarizom’



Alice

$m \leftarrow$ ‘Hello Bob’
$c \leftarrow \mathrm{ENC}_k(m)$

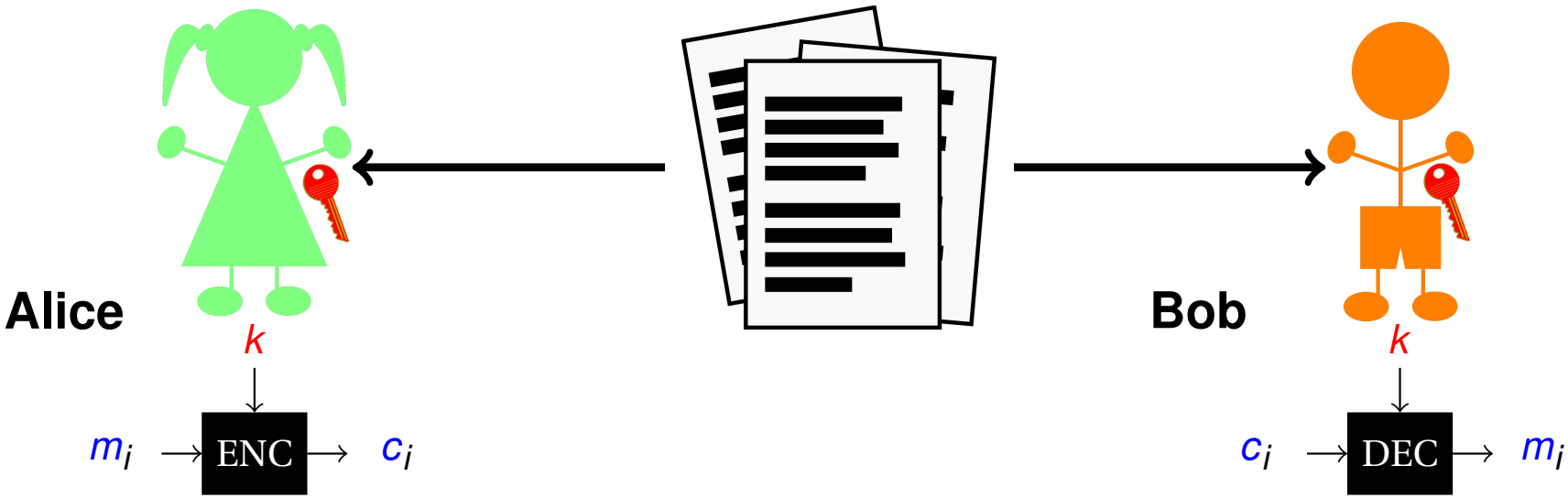
Bob

$m \leftarrow \mathrm{DEC}_k(c)$
$m =$ ‘Hello Bob’

Example: confidentiality with encryption

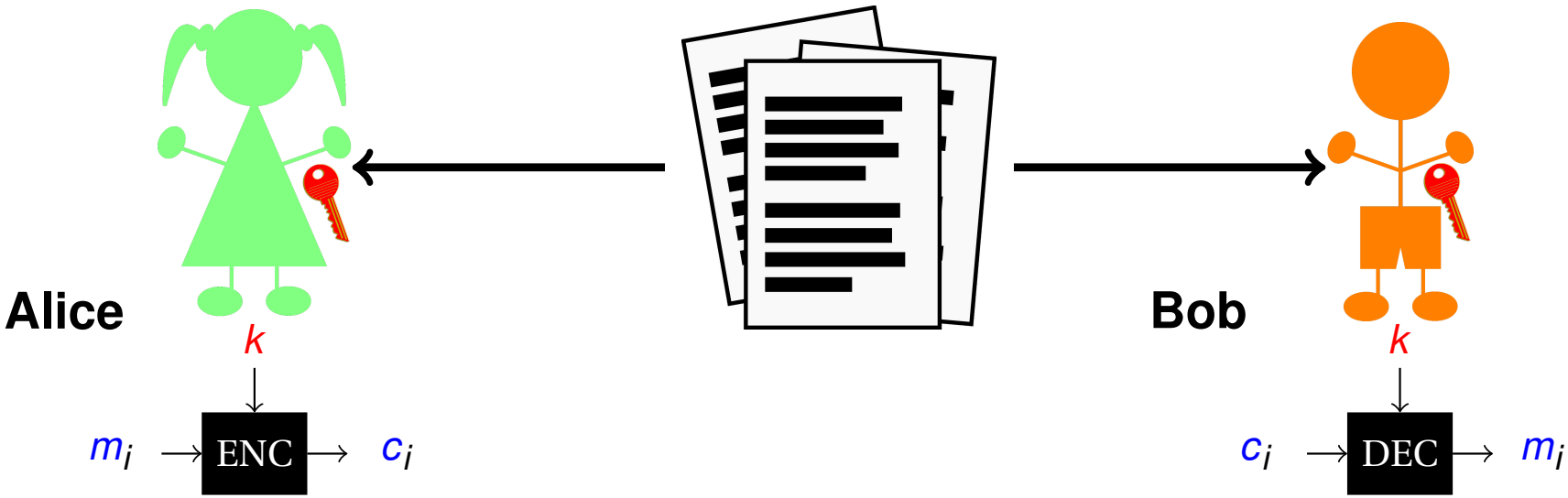
Cryptanalysis

- ➜ Black-box cryptanalysis
- ➜ Side-channel analysis



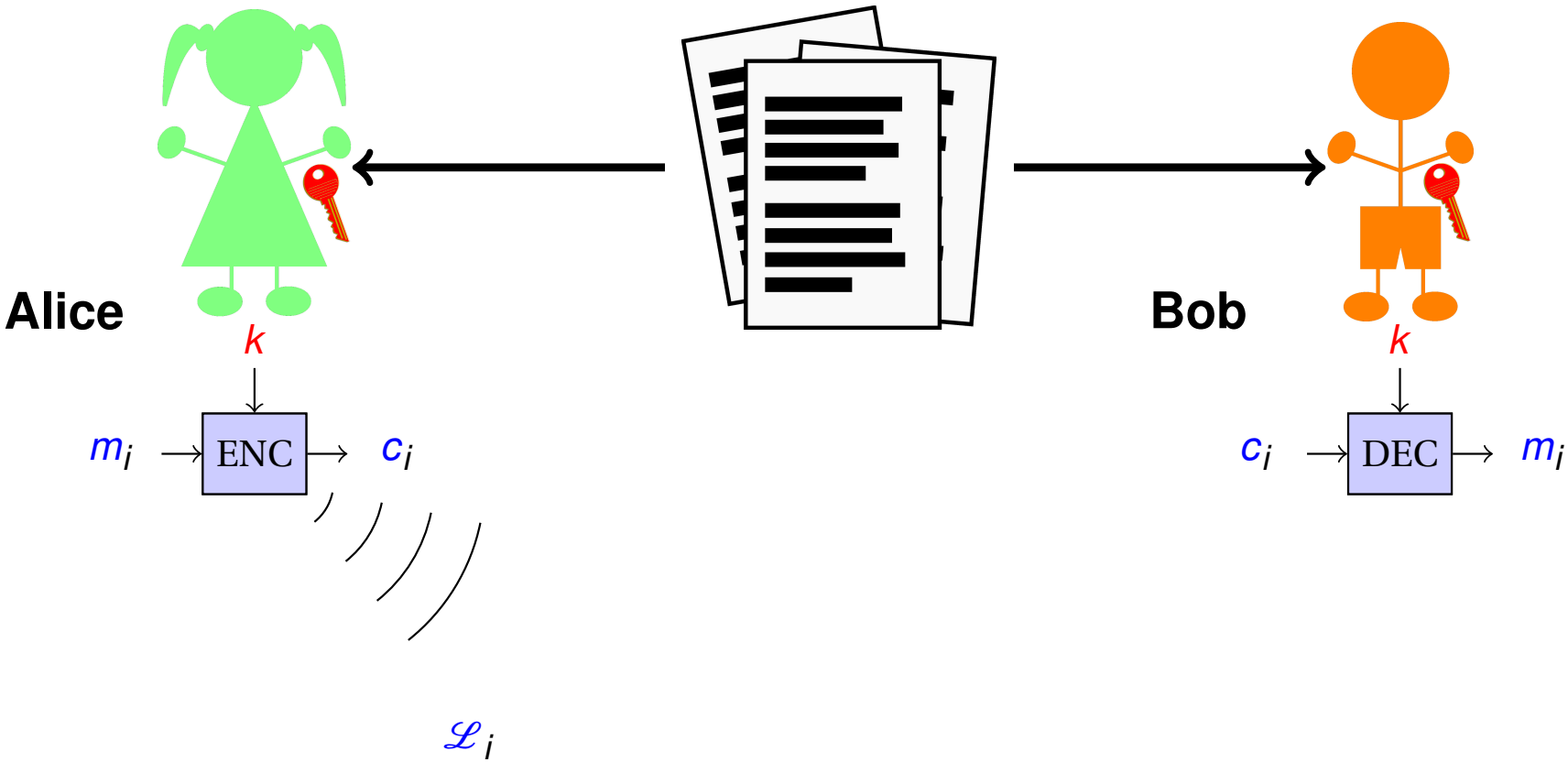
Cryptanalysis

- ➜ Black-box cryptanalysis: $\mathscr{A} \leftarrow (m_i, c_i)$
- ➜ Side-Channel Analysis



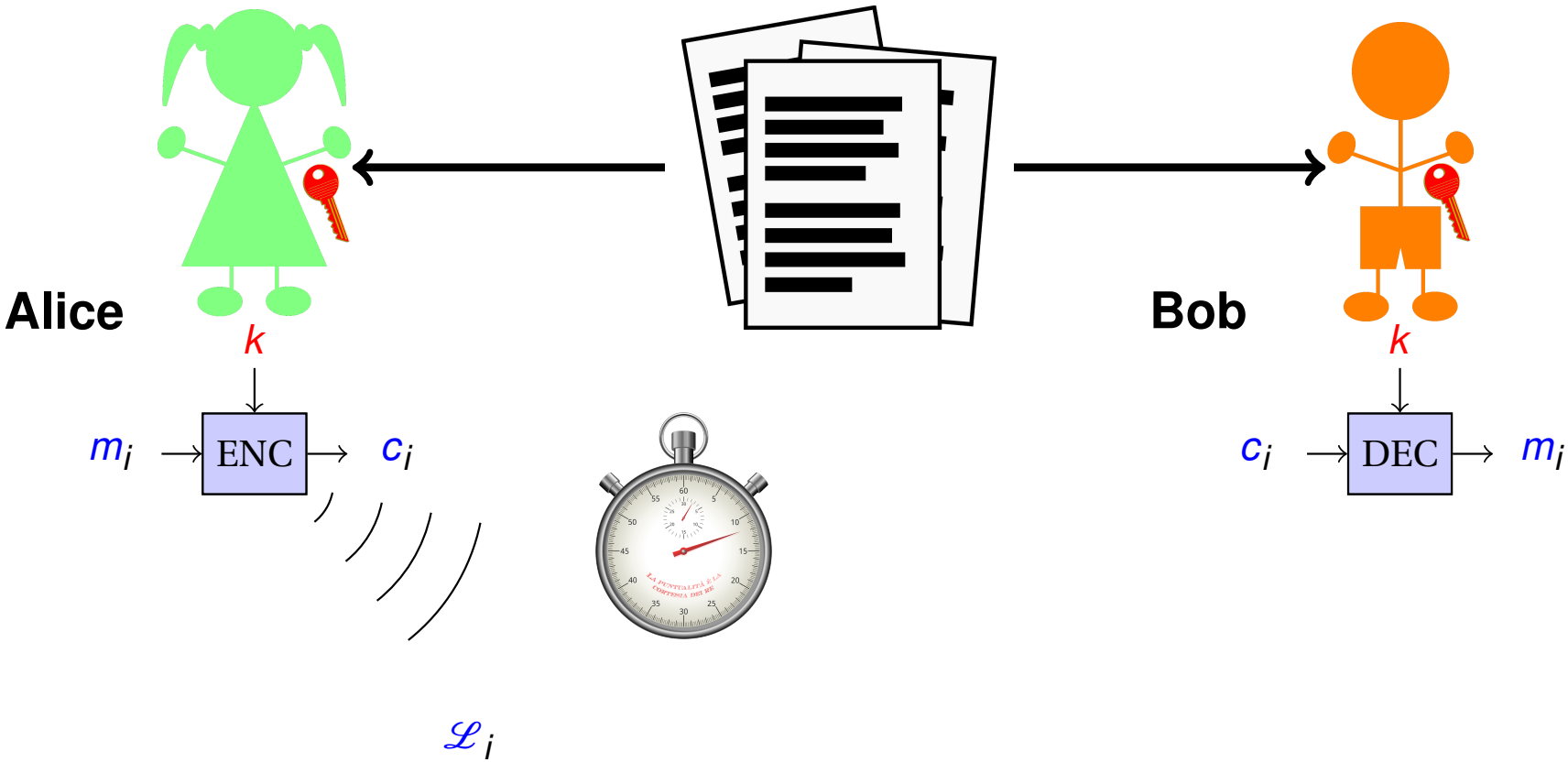
Cryptanalysis

- Black-box cryptanalysis
- Side-Channel Analysis: $\mathscr{A} \leftarrow (m_i, c_i, \mathscr{L}_i)$



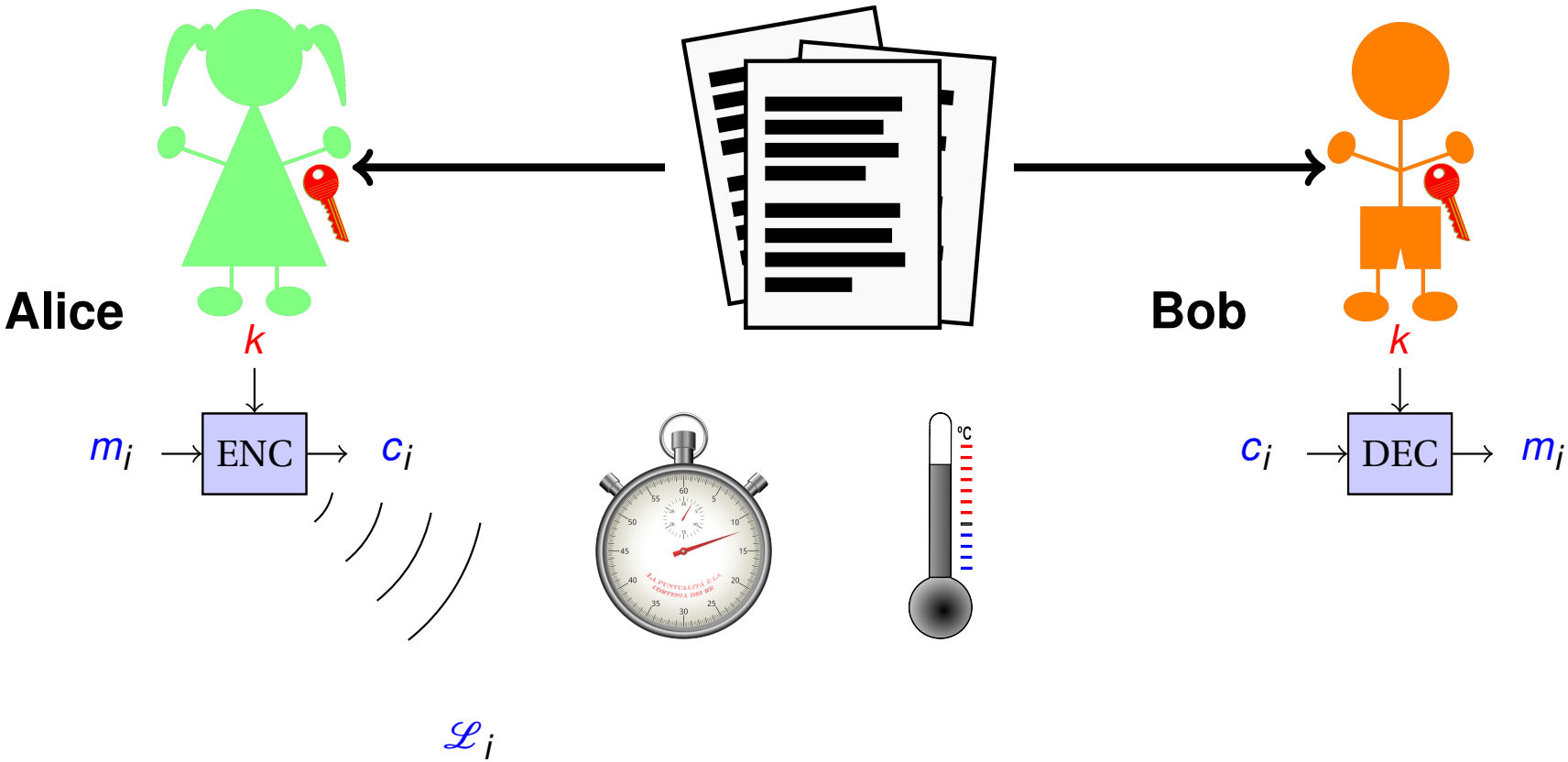
Cryptanalysis

- Black-box cryptanalysis
- Side-Channel Analysis: $\mathscr{A} \leftarrow (m_i, c_i, \mathscr{L}_i)$



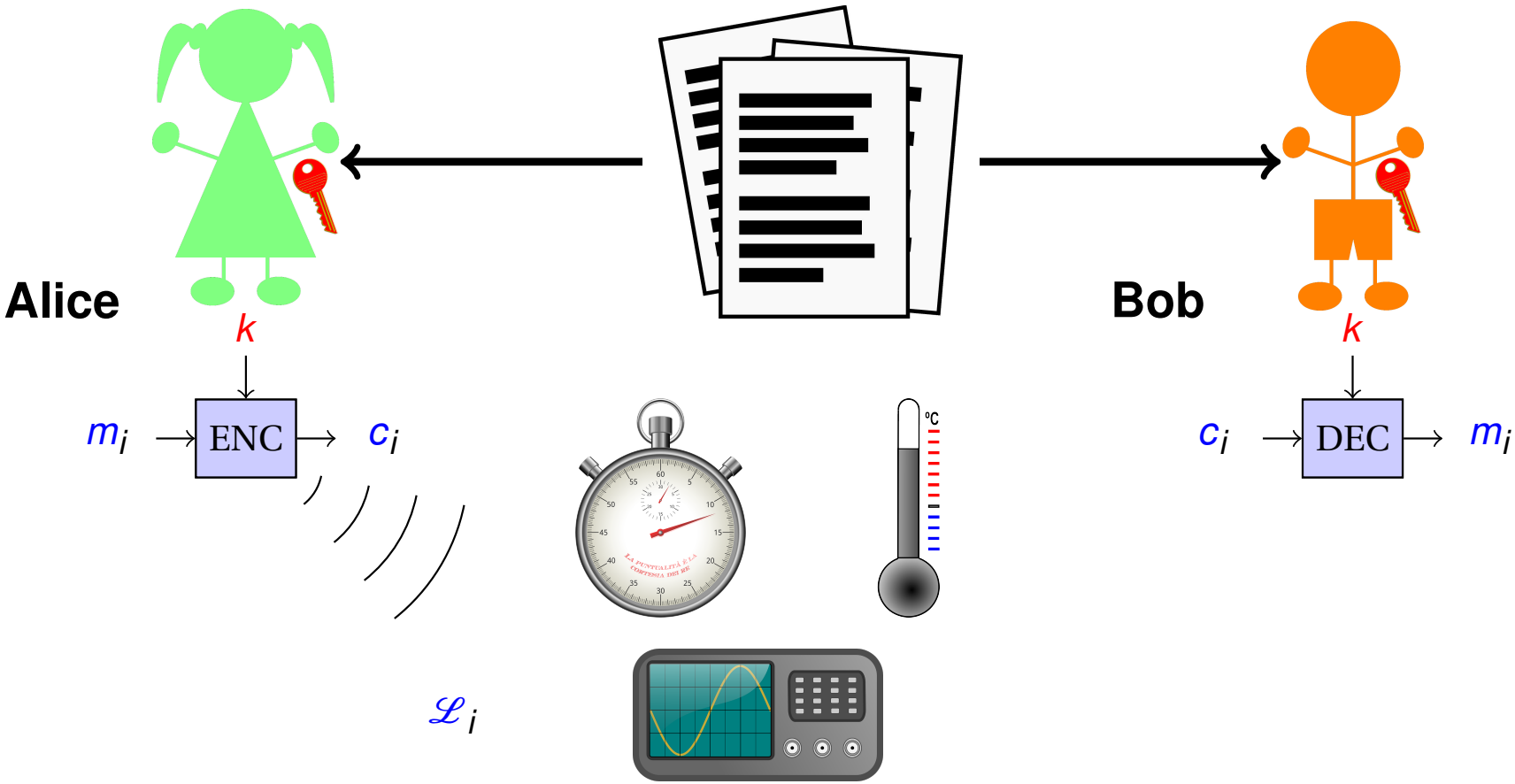
Cryptanalysis

- Black-box cryptanalysis
- Side-Channel Analysis: $\mathscr{A} \leftarrow (m_i, c_i, \mathscr{L}_i)$



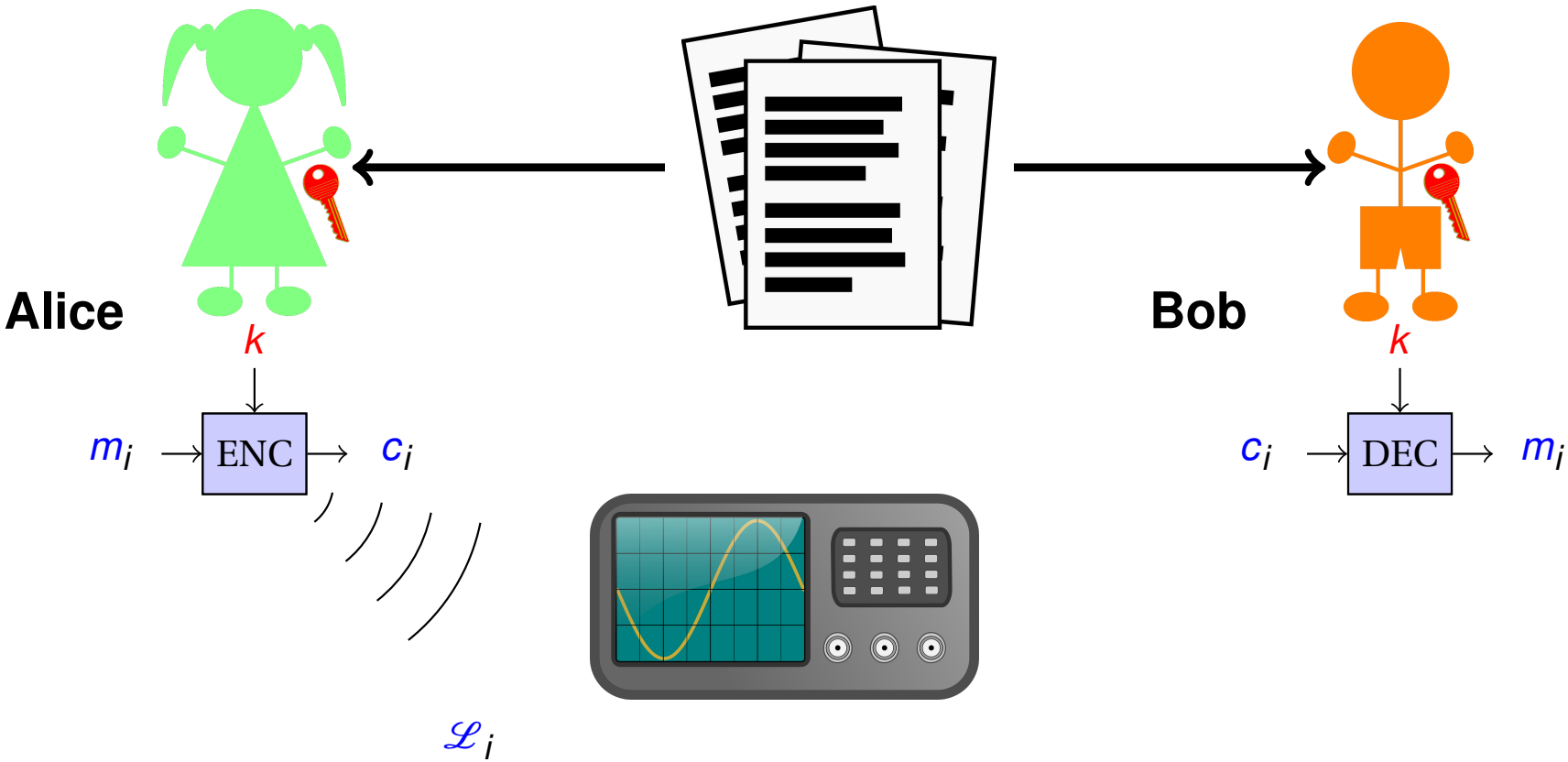
Cryptanalysis

- Black-box cryptanalysis
- Side-Channel Analysis: $\mathscr{A} \leftarrow (m_i, c_i, \mathscr{L}_i)$



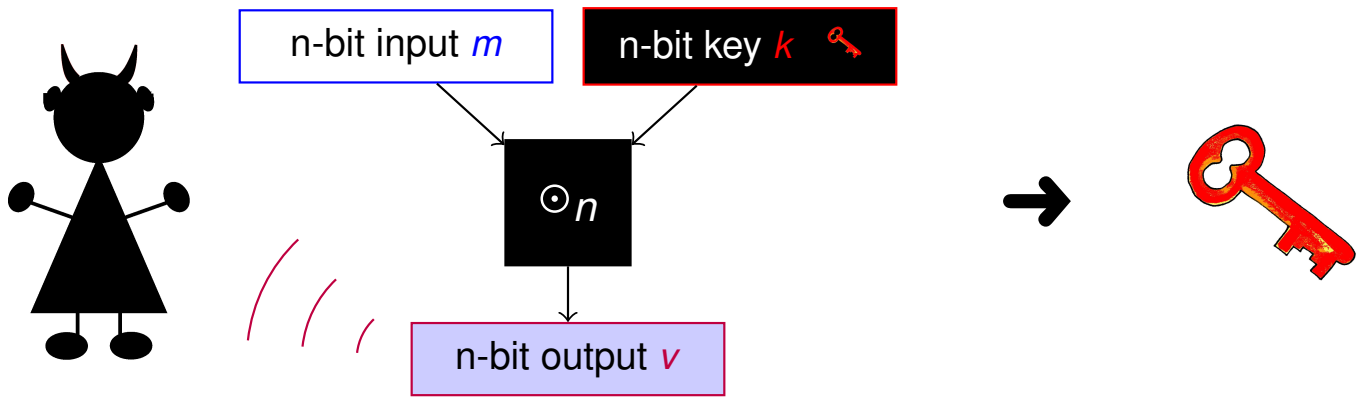
Cryptanalysis

- Black-box cryptanalysis
- Side-Channel Analysis: $\mathscr{A} \leftarrow (m_i, c_i, \mathscr{L}_i)$

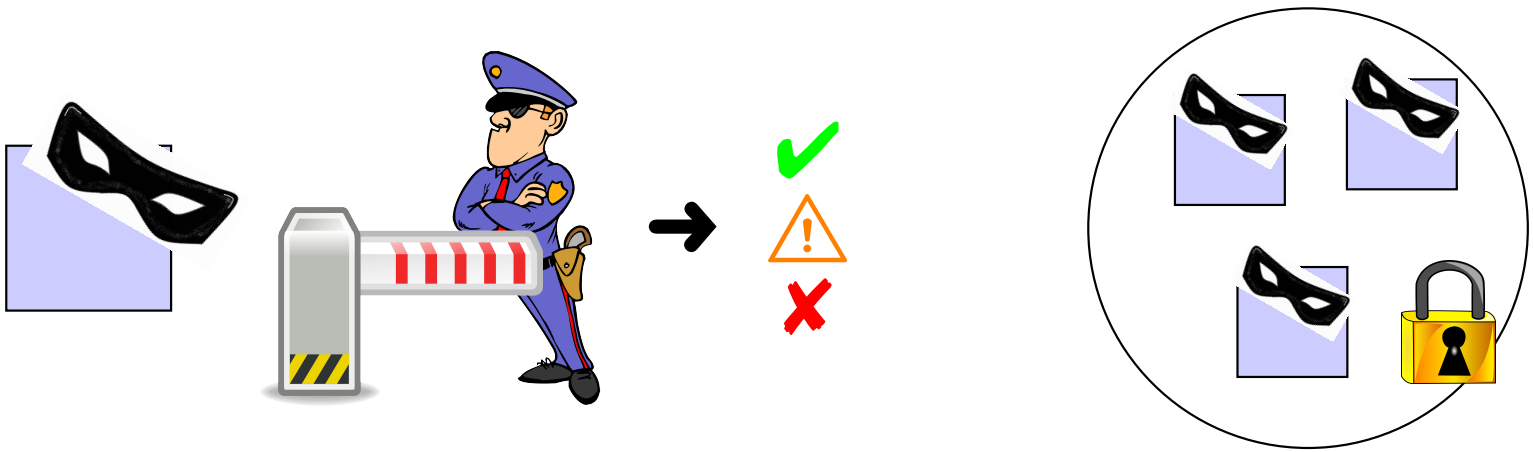


Chosen Contributions

Cryptanalysis: Power-Analysis Attacks

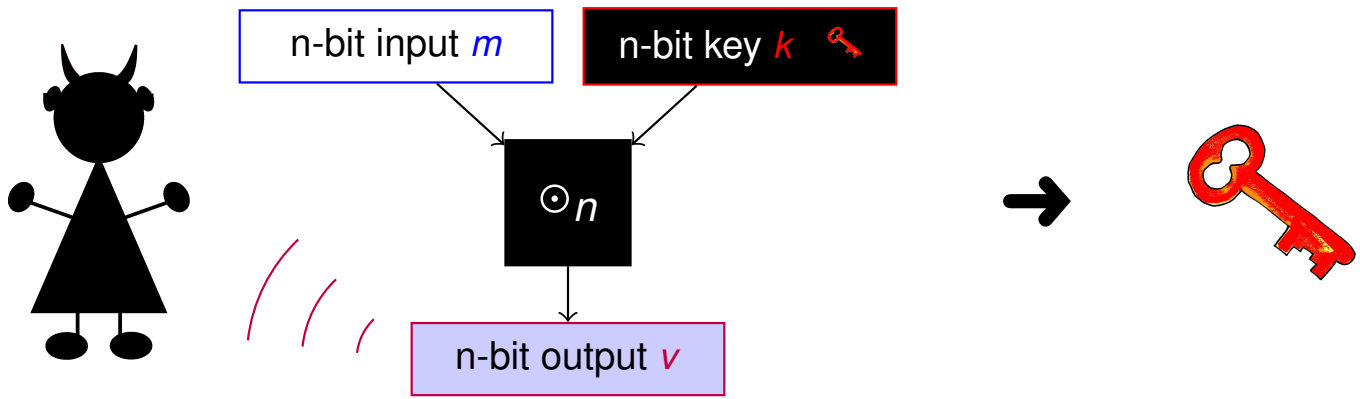


Cryptography: countermeasures against Power-Analysis Attacks



Chosen Contributions

Cryptanalysis: Power-Analysis Attacks

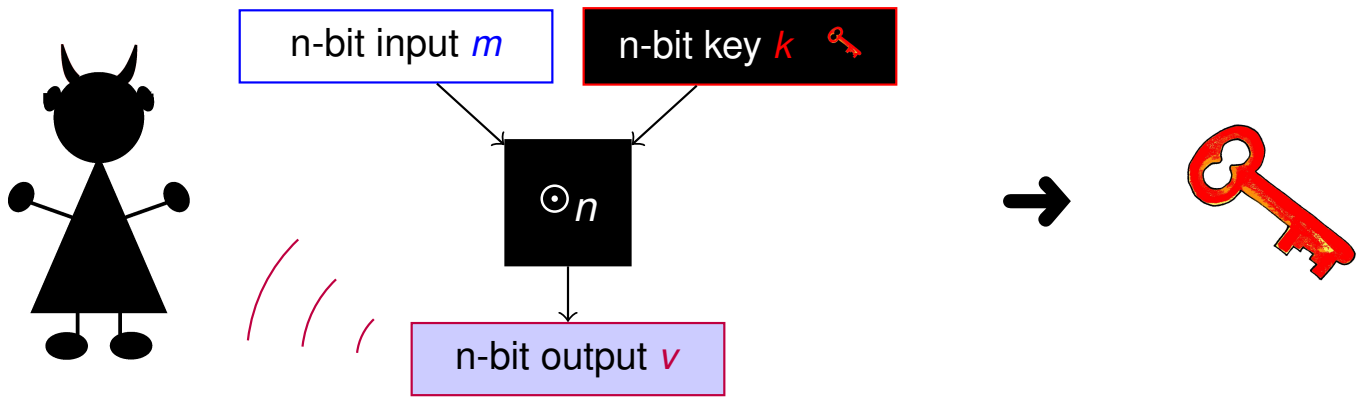


Hidden Multiplier Problem:

1. Successful attack with low noise → S. Belaïd, P-A. Fouque, and B. Gérard — Asiacrypt 2014
2. Improved attack for higher noise → S. Belaïd, J-S. Coron, B. Gérard, P-A. Fouque, J-G. Kammerer, and E. Prouff — CHES 2015

Chosen Contributions

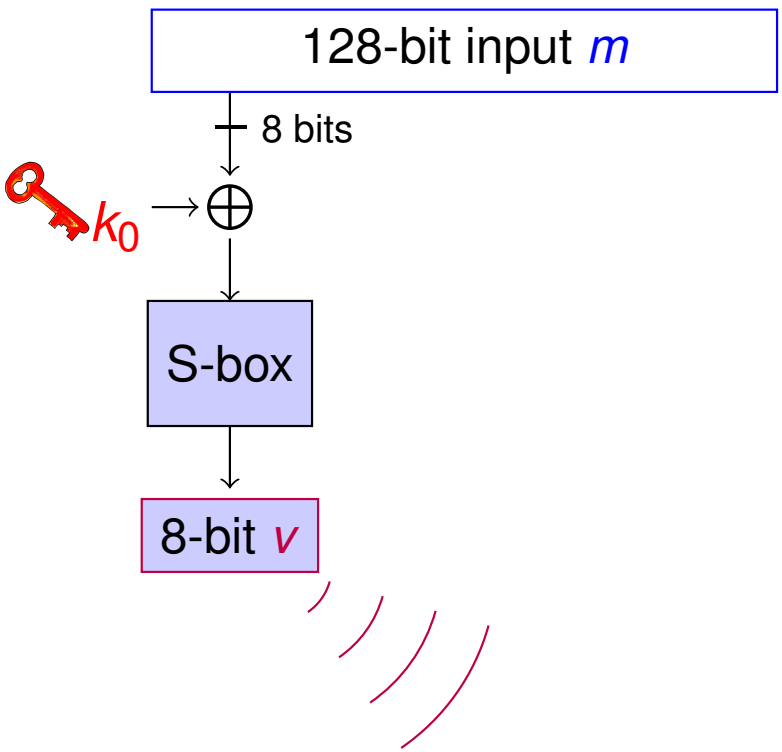
Cryptanalysis: Power-Analysis Attacks



Hidden Multiplier Problem:

1. Successful attack with low noise → S. Belaïd, P-A. Fouque, and B. Gérard — Asiacrypt 2014
2. Improved attack for higher noise → S. Belaïd, J-S. Coron, B. Gérard, P-A. Fouque, J-G. Kammerer, and E. Prouff — CHES 2015

Classical Power-Analysis Attack against AES-128



Attack on 8 bits

- prediction of the outputs for the 256 possible 8-bit secret
- correlation between predictions and leakage
- selection of the best correlation to find the correct 8-bit secret

Attack on 128 bits

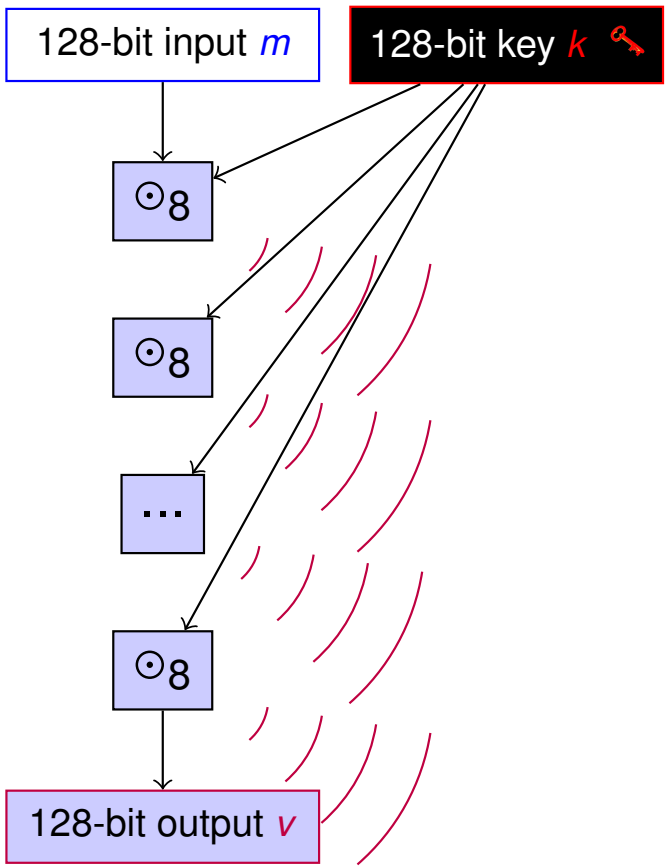
- repetition of the attack on 8 bits on each S-box

Power-Analysis Attack against AES-GCM authentication, multiplication-based fresh re-keying, ...

➜ k is only manipulated in multiplications

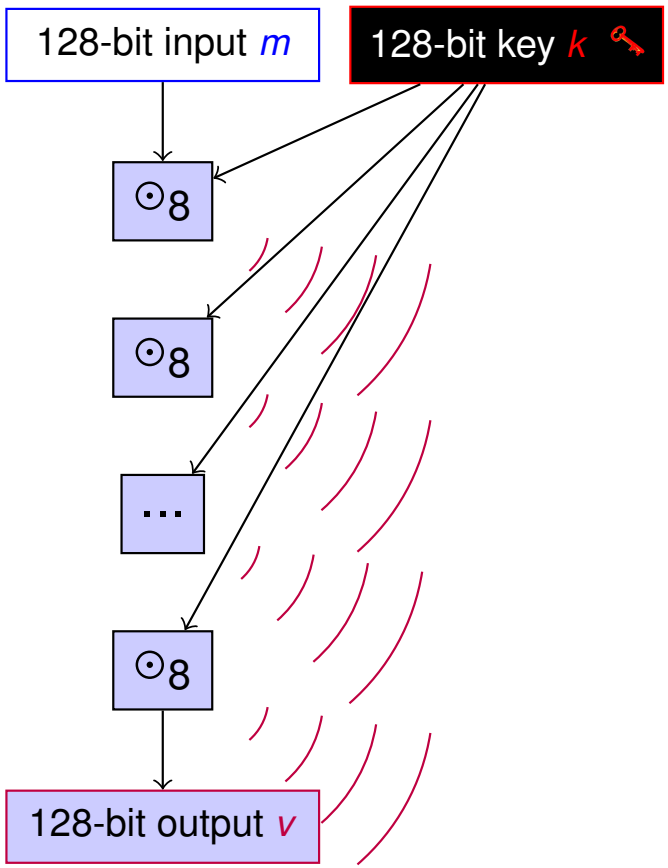
Power-Analysis Attack against AES-GCM authentication, multiplication-based fresh re-keying, ...

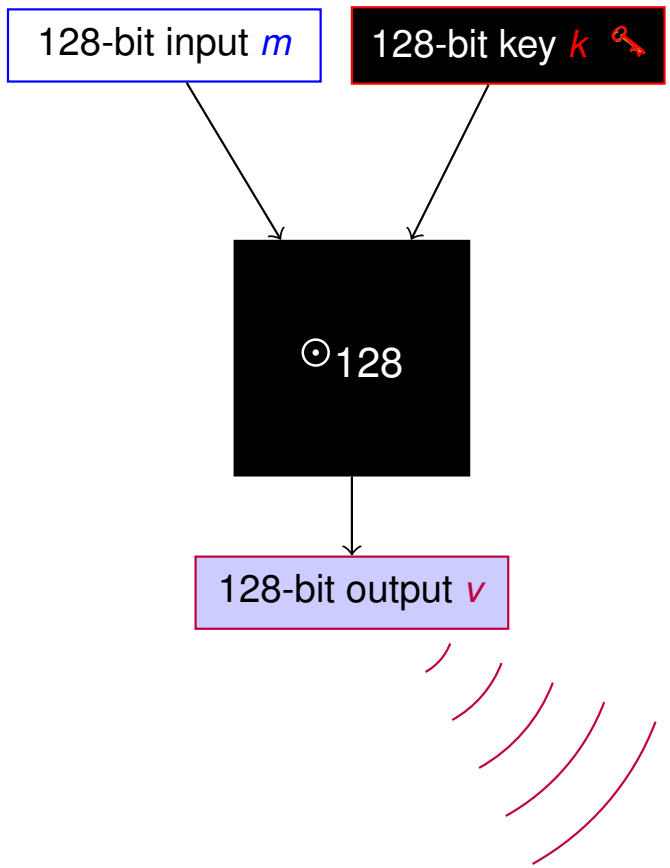
➜ k is only manipulated in multiplications



Power-Analysis Attack against AES-GCM authentication, multiplication-based fresh re-keying, ...

➜ k is only manipulated in multiplications



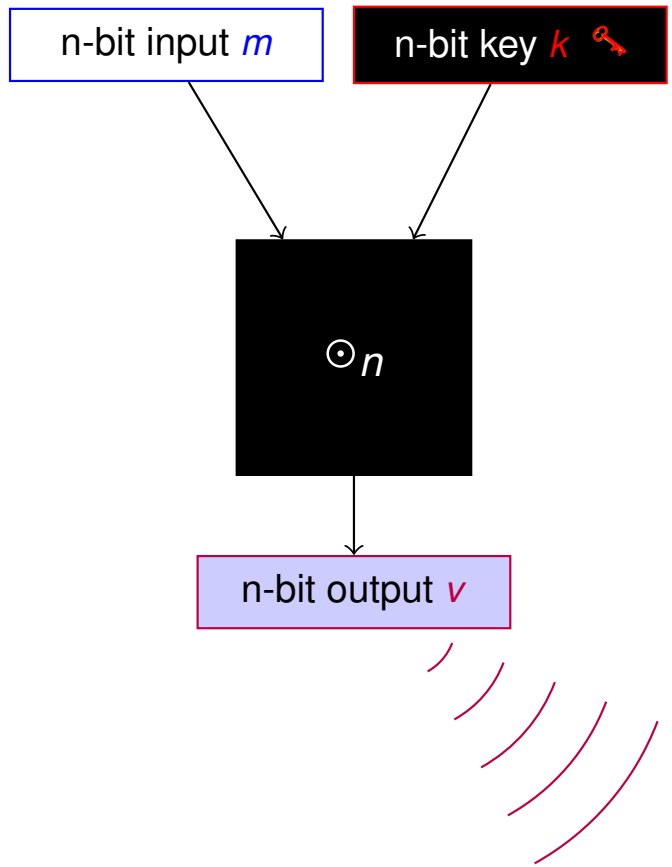


Hidden Multiplier Problem

Let $k \leftarrow \mathrm{GF}(2^n)$. Let $\ell \in \mathbb{N}$.
Given a sequence $\{m^i, \mathscr{L}^i\}_{1 \le i \le \ell}$
where

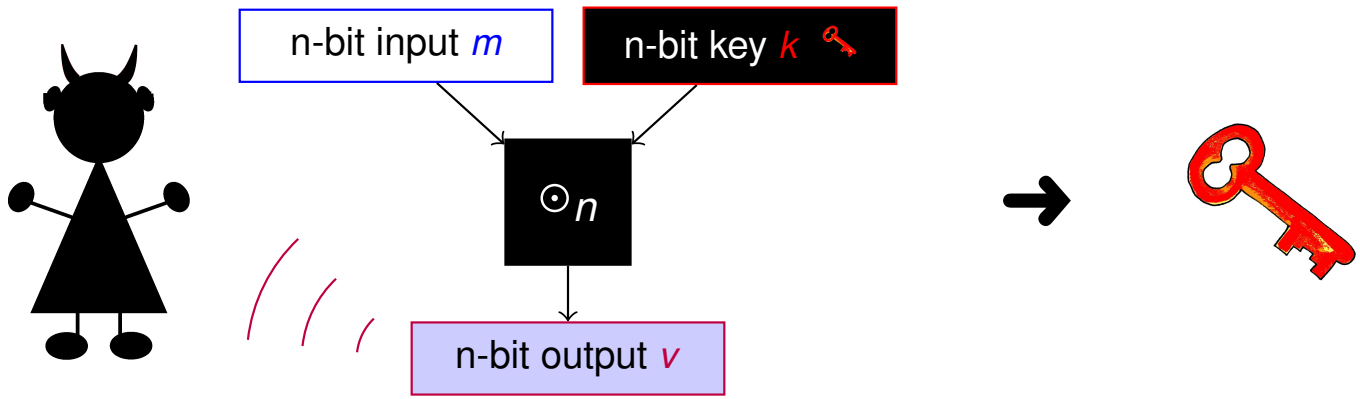
- $m^i \leftarrow \mathrm{GF}(2^n)$
- $\mathscr{L}^i = \mathrm{HW}(v^i) + \varepsilon^i,\ \varepsilon^i \sim \mathcal{N}(0, \sigma^2)$

recover k.



Chosen Contributions

Cryptanalysis: Power-Analysis Attacks

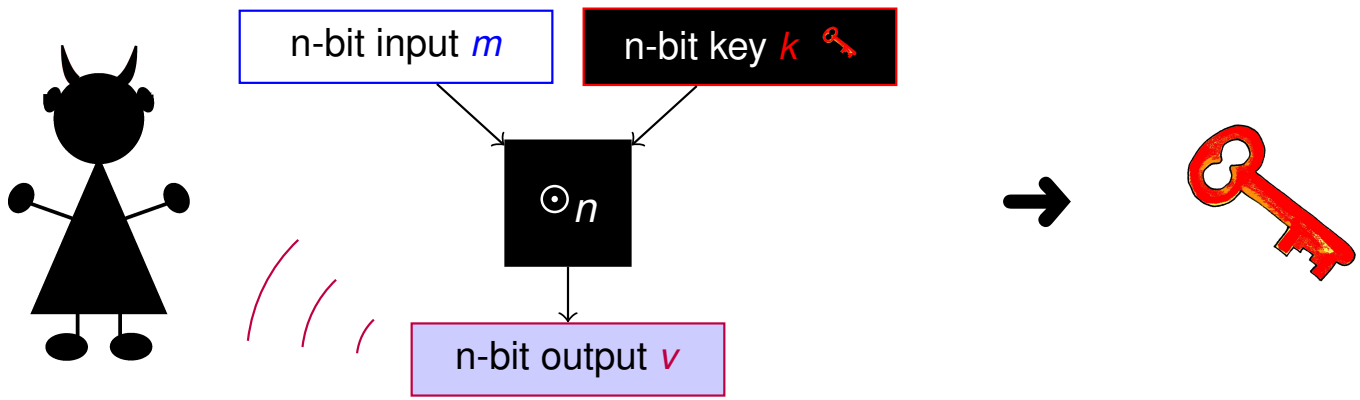


Hidden Multiplier Problem:

1. Successful attack with low noise → S. Belaïd, P-A. Fouque, and B. Gérard — Asiacrypt 2014
2. Improved attack for higher noise → S. Belaïd, J-S. Coron, B. Gérard, P-A. Fouque, J-G. Kammerer, and E. Prouff — CHES 2015

Chosen Contributions

Cryptanalysis: Power-Analysis Attacks



Hidden Multiplier Problem:

1. Successful attack with low noise → S. Belaïd, P-A. Fouque, and B. Gérard — Asiacrypt 2014
2. Improved attack for higher noise → S. Belaïd, J-S. Coron, B. Gérard, P-A. Fouque, J-G. Kammerer, and E. Prouff — CHES 2015

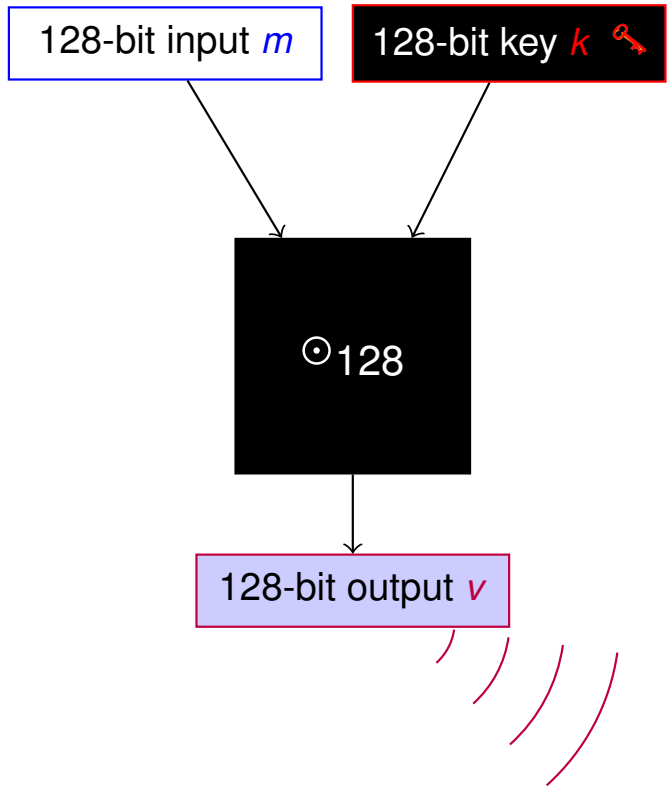
Main Observation

Current Issue: each bit of the 128-bit multiplication's result depends on all the key bits

➜ no divide-and-conquer strategy

Hypotheses:

- leakage of multiplication's outputs $\mathrm{HW}(v) + \varepsilon$
- multiplication in $\mathrm{GF}(2^{128})$



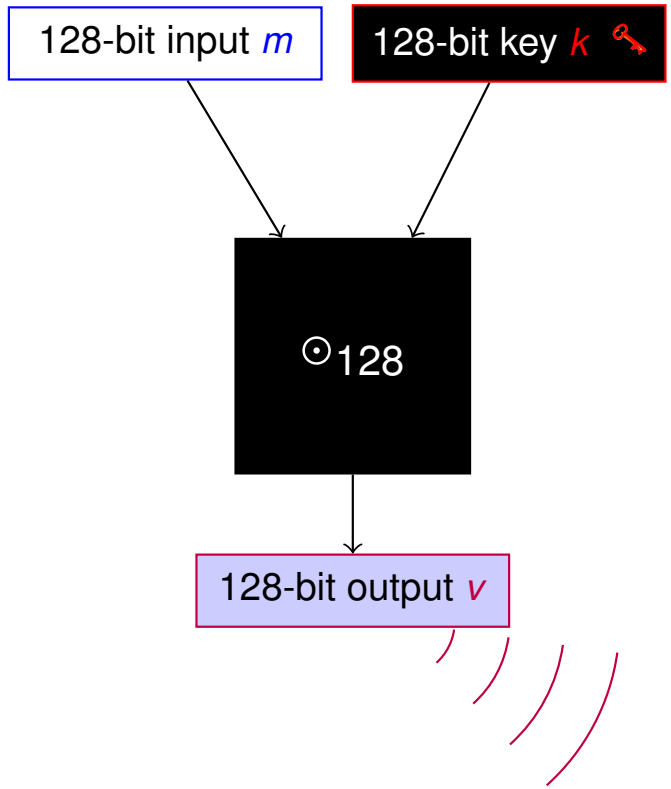
Main Observation

Current Issue: each bit of the 128-bit multiplication's result depends on all the key bits

➜ no divide-and-conquer strategy

Hypotheses:

- leakage of multiplication's outputs $\mathrm{HW}(v) + \varepsilon$
- multiplication in $\mathrm{GF}(2^{128})$



Main observation:
the LSB of a variable's Hamming weight is a linear function of its bits:

$$\mathrm{lsb}_0\left(\mathrm{HW}(v)\right) = \bigoplus_{0 \leqslant i \leqslant 127} v_i = \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j \right) k_i$$

System of Equations

With ℓ Hamming weight values $\{\mathrm{HW}(v^{(i)})\}_{0 \leqslant i < \ell}$, we recover k by solving $\mathscr{S}$:

$$\mathscr{S} = \begin{cases} \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(0)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(0)} \\ \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(1)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(1)} \\ \cdots & \\ \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(\ell-1)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(\ell-1)} \end{cases}$$

System of Equations

With ℓ Hamming weight values $\{\mathrm{HW}(v^{(i)})\}_{0 \leqslant i < \ell}$, we recover k by solving $\mathscr{S}$:

$$\mathscr{S} = \begin{cases} \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(0)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(0)} \\ \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(1)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(1)} \\ \cdots & \\ \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(\ell-1)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(\ell-1)} \end{cases}$$

But in practice, the leakage comes with noise: $\mathscr{L} = \mathrm{HW}(v) + \varepsilon$

$$\mathrm{lsb}_0 \left(\lceil \mathscr{L} \rfloor \right) \quad = \quad \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right) \oplus b_\varepsilon$$

System of Equations

With ℓ Hamming weight values $\{\mathrm{HW}(v^{(i)})\}_{0 \leqslant i < \ell}$, we recover k by solving $\mathscr{S}$:

$$\widehat{\mathscr{S}} = \begin{cases} \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(0)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(0)} \oplus b_\varepsilon^{(0)} \\ \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(1)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(1)} \oplus b_\varepsilon^{(1)} \\ \cdots & \\ \bigoplus_{0 \leqslant i \leqslant 127} \left(\bigoplus_{0 \leqslant j \leqslant 127-i} m_j^{(\ell-1)} \right) & k_i = \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right)^{(\ell-1)} \oplus b_\varepsilon^{(\ell-1)} \end{cases}$$

But in practice, the leakage comes with noise: $\mathscr{L} = \mathrm{HW}(v) + \varepsilon$

$$\mathrm{lsb}_0 \left(\lceil \mathscr{L} \rfloor \right) \quad = \quad \mathrm{lsb}_0 \left(\mathrm{HW}(v) \right) \oplus b_\varepsilon$$

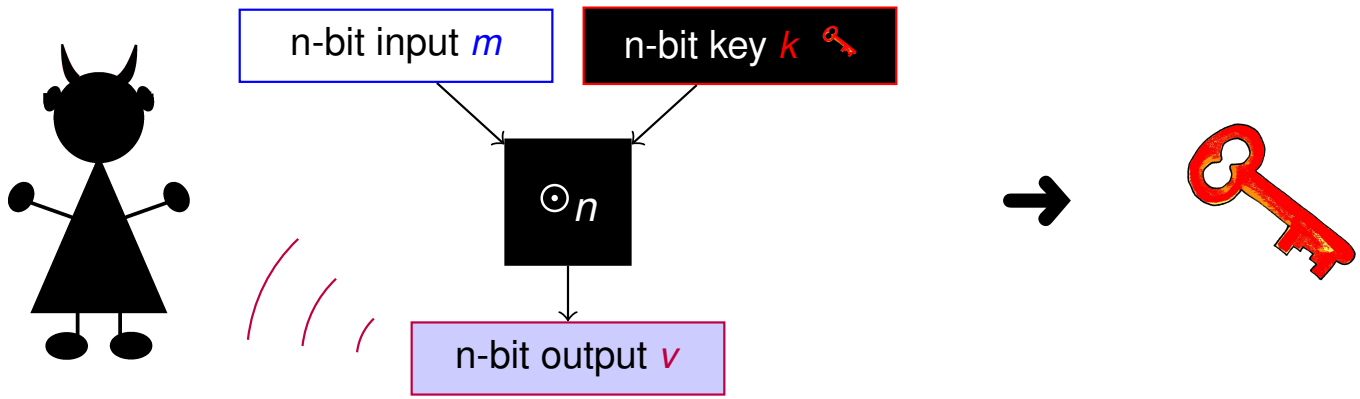
Complexities

Method	Signal-to-Noise Ratio			
	3.200	800	200	128
Naive method $(\mathscr{C}_s, \mathscr{C}_t)$	$(2^9, 2^{21})$	$(2^9, 2^{21})$	$(2^9, 2^{65})$	$(2^9, 2^{107})$
LPN (LF Algo) $(\mathscr{C}_s, \mathscr{C}_t)$	$(2^{12}, 2^{14})$	$(2^{21}, 2^{22})$	$(2^{33}, 2^{34})$	$(2^{49}, 2^{50})$
Linear decoding $(\mathscr{C}_s, \mathscr{C}_t)$	$(2^7, 2^6)$	$(2^7, 2^7)$	$(2^9, 2^{25})$	$(2^{10}, 2^{62})$

$$\text{Signal-to-noise ratio} = \frac{\text{signal variance}}{\text{noise variance}} = \frac{32}{\sigma^2}$$

Chosen Contributions

Cryptanalysis: Power-Analysis Attacks

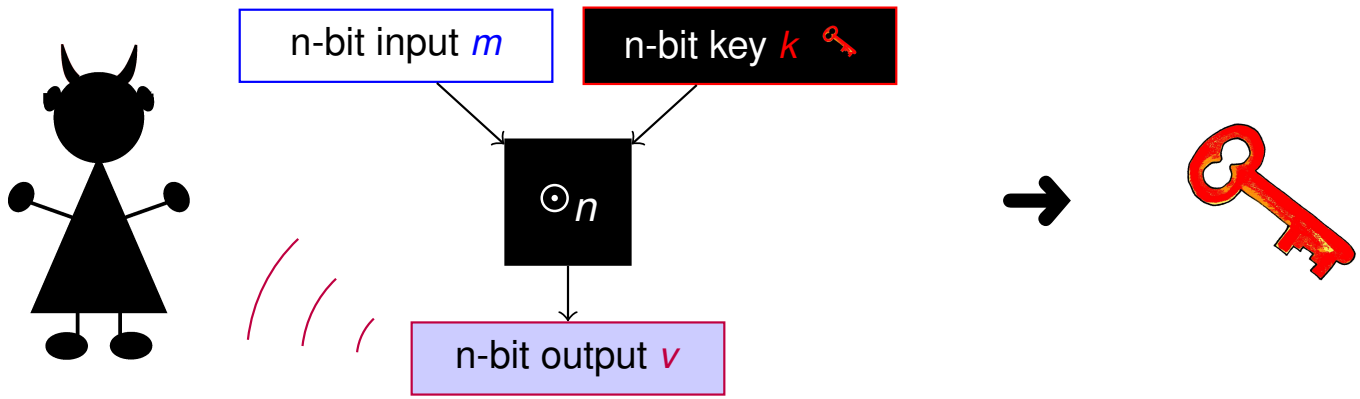


Hidden Multiplier Problem:

1. Successful attack with low noise → S. Belaïd, P-A. Fouque, and B. Gérard — Asiacrypt 2014
2. Improved attack for higher noise → S. Belaïd, J-S. Coron, B. Gérard, P-A. Fouque, J-G. Kammerer, and E. Prouff — CHES 2015

Chosen Contributions

Cryptanalysis: Power-Analysis Attacks



Hidden Multiplier Problem:

1. Successful attack with low noise → S. Belaïd, P-A. Fouque, and B. Gérard — Asiacrypt 2014
2. Improved attack for higher noise → S. Belaïd, J-S. Coron, B. Gérard, P-A. Fouque, J-G. Kammerer, and E. Prouff — CHES 2015

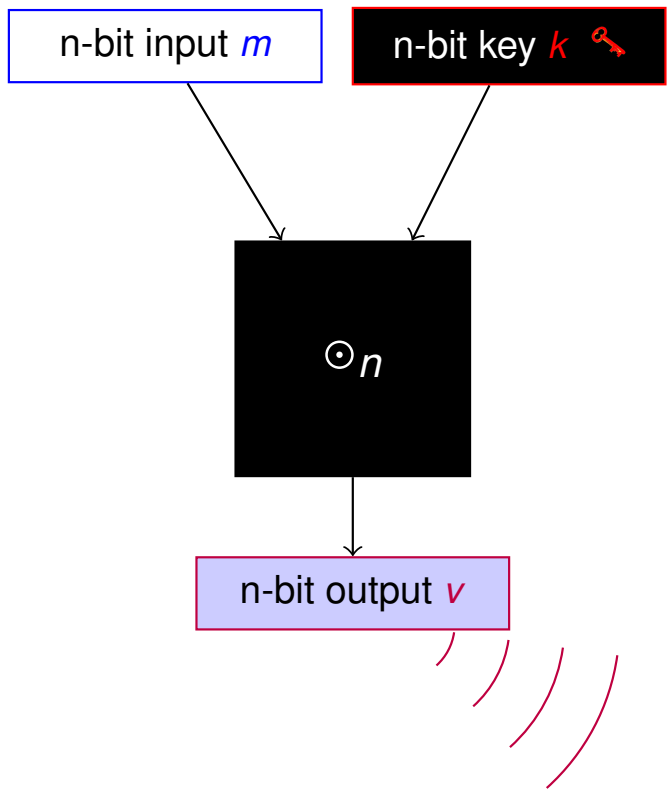
Main Observation

New Attack:

- ➜ filter the multiplication's outputs leakage to extract high and low Hamming weights
- ➜ solve a system with errors

Improvements:

- ✔ more generic
- ✔ less impacted by noise



System of Equations

Reminder:

$$\mathscr{L}(v) = \mathrm{HW}(v) + \varepsilon = \mathrm{HW}(m \odot k) + \varepsilon$$

Extreme cases:

$\mathrm{HW}(v) = 0 \rightarrow v = 0$

$$\begin{cases} v_0 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(0,j)}} m_i \right) k_j = 0 \\ v_1 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(1,j)}} m_i \right) k_j = 0 \\ \vdots \quad \vdots \quad \vdots \\ v_{n-1} = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(n-1,j)}} m_i \right) k_j = 0 \end{cases}$$

$\mathrm{HW}(v) = n \rightarrow v = 2^n - 1$

$$\begin{cases} v_0 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(0,j)}} m_i \right) k_j = 1 \\ v_1 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(1,j)}} m_i \right) k_j = 1 \\ \vdots \quad \vdots \quad \vdots \\ v_{n-1} = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(n-1,j)}} m_i \right) k_j = 1 \end{cases}$$

System of Equations

Reminder:

$$\mathscr{L}(v) = \mathrm{HW}(v) + \varepsilon = \mathrm{HW}(m \odot k) + \varepsilon$$

Usual cases:

$\mathscr{L}(v)$ low ➜ $v \approx 0$

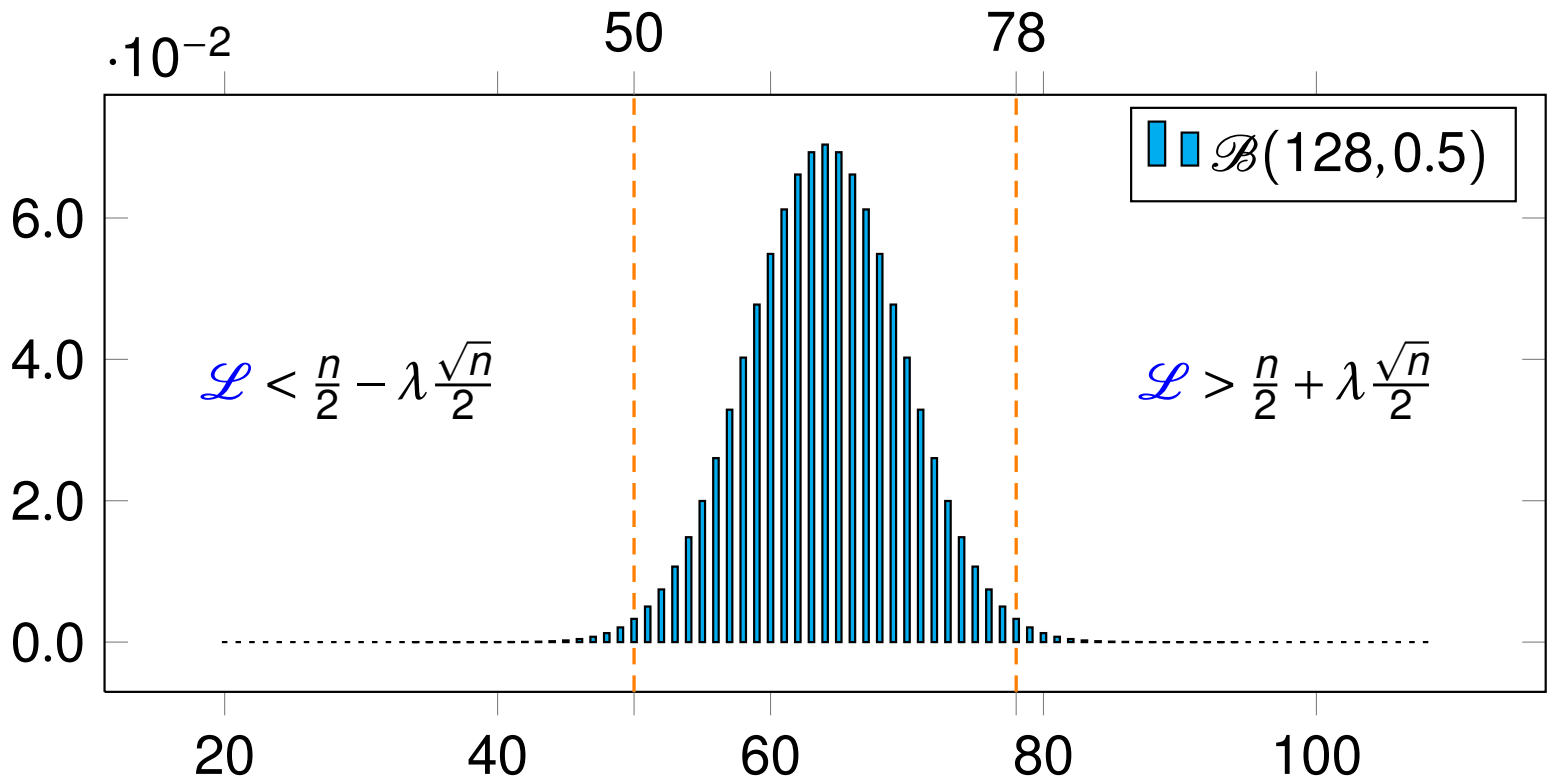
$$\begin{cases} v_0 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(0,j)}} m_i \right) k_j = 0 \\ v_1 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(1,j)}} m_i \right) k_j = 0 \\ \vdots \quad \vdots \quad \vdots \\ v_{n-1} = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(n-1,j)}} m_i \right) k_j = 0 \end{cases}$$

$\mathscr{L}(v)$ high ➜ $v \approx 2^n - 1$

$$\begin{cases} v_0 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(0,j)}} m_i \right) k_j = 1 \\ v_1 = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(1,j)}} m_i \right) k_j = 1 \\ \vdots \quad \vdots \quad \vdots \\ v_{n-1} = \bigoplus_{0 \leqslant j < n} \left(\bigoplus_{i \in I^{(n-1,j)}} m_i \right) k_j = 1 \end{cases}$$

with an error probability p

Filtering



$\left.\begin{array}{l} \text{SNR} = 128 \\ n = 128 \\ \lambda \approx 2.5 \end{array}\right\}$ filtering: 1 trace over 2^5
error probability: $p \approx 0.38$

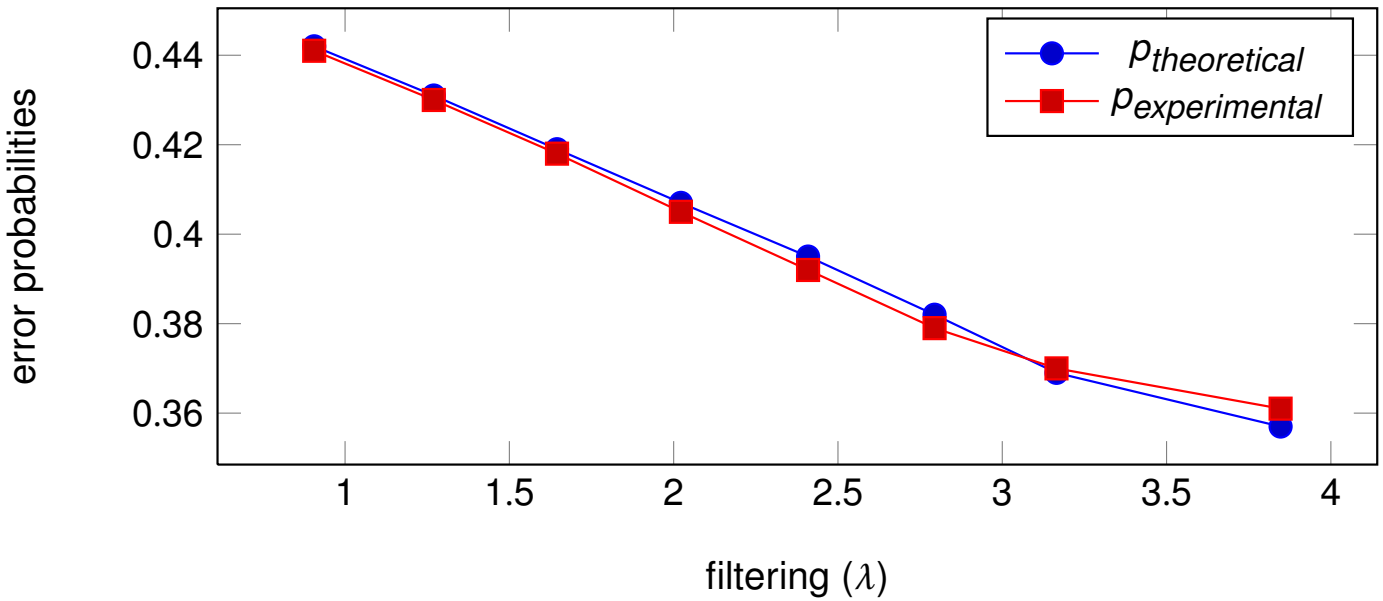
Error Probabilities

$\log_2(1/F(\lambda))$	30	25	20	15	10	5
SNR $= 128$, $\sigma = 0.5$						
λ	6.00	5.46	4.85	4.15	3.29	2.16
p	0.23	0.25	0.28	0.31	0.34	0.39
p [BFG14]	0.31					
SNR $= 8$, $\sigma = 2$						
λ	6.37	5.79	5.14	4.39	3.48	2.28
p	0.25	0.27	0.29	0.32	0.35	0.40
p [BFG14]	> 0.49					
SNR $= 2$, $\sigma = 4$						
λ	7.42	6.73	5.97	5.09	4.03	2.64
p	0.28	0.30	0.32	0.34	0.37	0.41
p [BFG14]	> 0.49					
SNR $= 0.5$, $\sigma = 8$						
λ	10.57	9.58	8.48	7.21	5.71	3.73
p	0.34	0.36	0.37	0.39	0.41	0.44
p [BFG14]	> 0.49					

$$\text{Signal-to-noise ratio} = \frac{\text{signal variance}}{\text{noise variance}} = \frac{32}{\sigma^2}$$

Experiments for $n = 128$

- Filtering on a Virtex 5 - 128 bits ($n = 128$) : SNR $= 8.21$, $\sigma = 7.11$

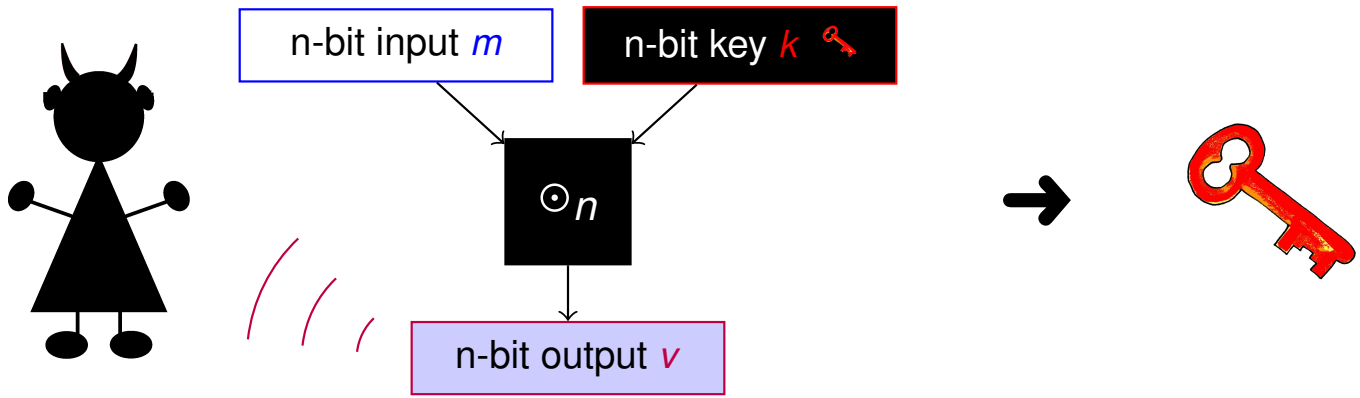


- Expected complexities to recover k with 2^{20} consumption traces

trade-offs (time , memory)	$\left(2^{59.31}, 2^{27.00}\right)$
	$\left(2^{51.68}, 2^{36.00}\right)$
	$\left(2^{50.00}, 2^{44.00}\right)$

Conclusion on the Multiplication Cryptanalysis

Cryptanalysis: Power-Analysis Attacks



Summary

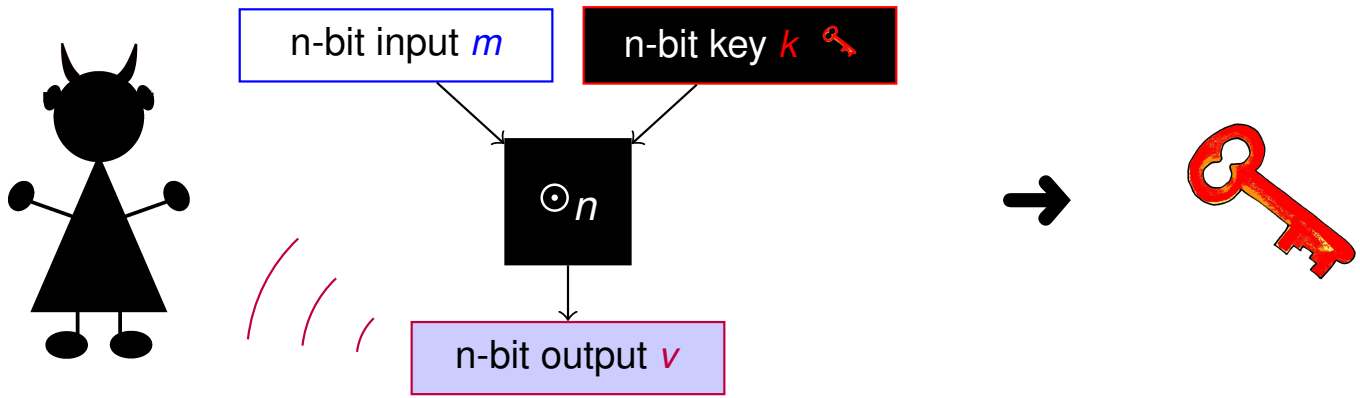
- ✔ successful attacks on multiplications from the output's leakage
- ✔ practical for $n = 128$ (use cases: AES-GCM, re-keying)

Further Work

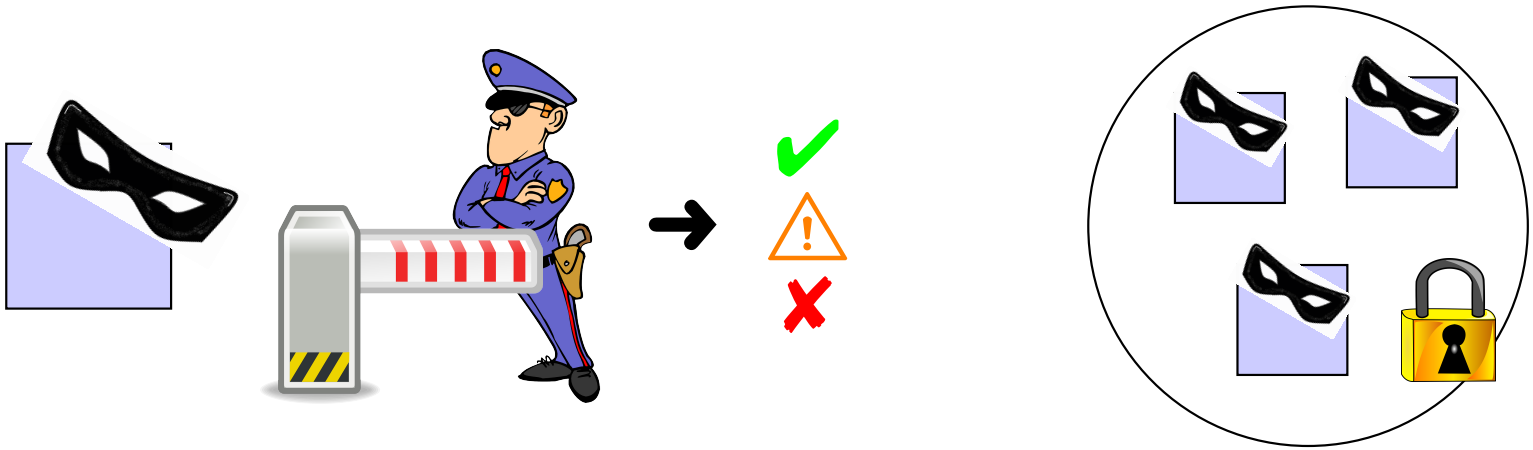
- ➜ application of similar attacks on other primitives
- ➜ deeper analysis of LPN techniques in the context of side-channel analysis

Chosen Contributions

Cryptanalysis: Power-Analysis Attacks



Cryptography: countermeasures against Power-Analysis Attacks



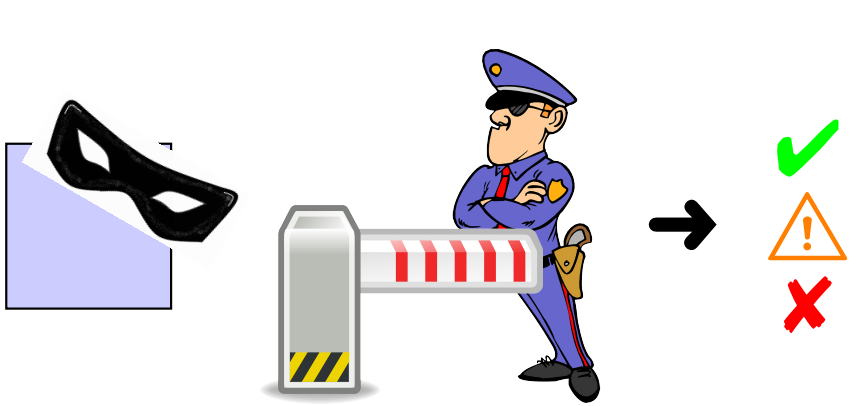
Chosen Contributions

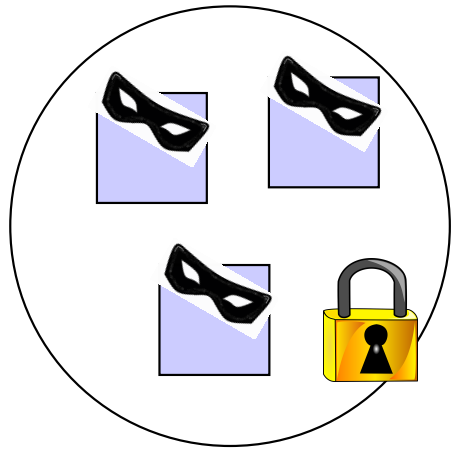
Masking and t-probing model:

1. Verified Proofs of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, B. Grégoire, and P-Y. Strub. Eurocrypt 2015

2. Composition of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, and B. Grégoire. ePrint 2015

Cryptography: countermeasures against Power-Analysis Attacks



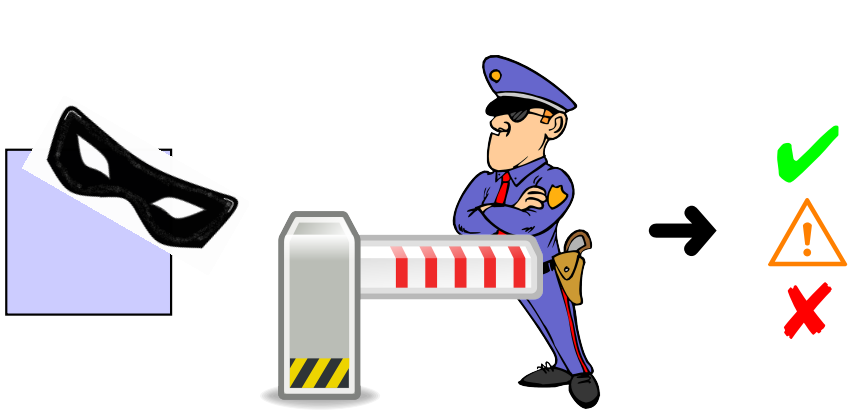


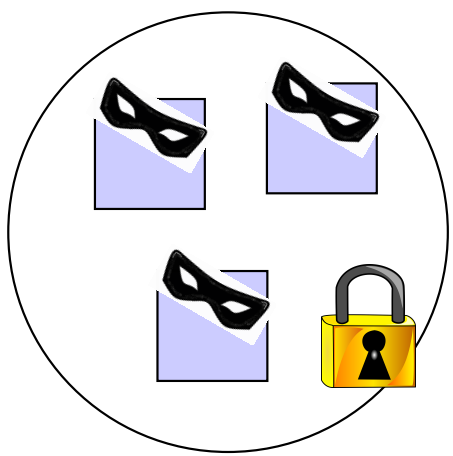
Chosen Contributions

Masking and t-probing model:

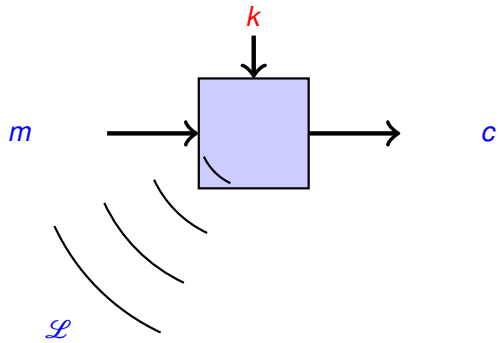
1. Verified Proofs of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, B. Grégoire, and P-Y. Strub. Eurocrypt 2015
2. Composition of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, and B. Grégoire. ePrint 2015

Cryptography: countermeasures against Power-Analysis Attacks





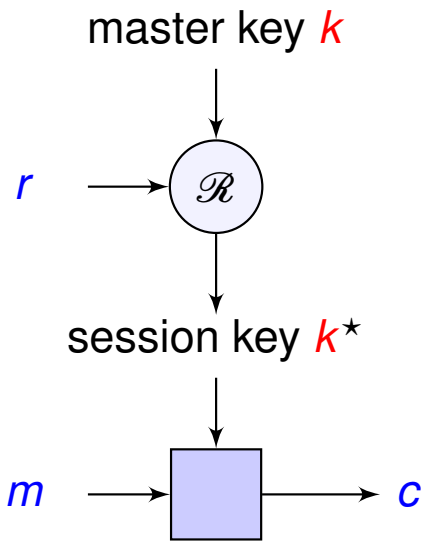
Countermeasures against Power-Analysis Attacks



Problem: leakage $\mathscr{L}$ is key-dependent

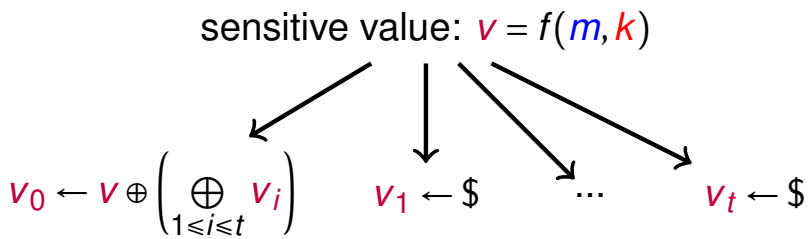
Fresh Re-keying

Idea: regularly change k



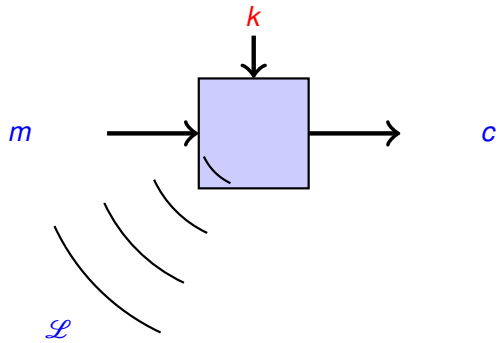
Masking

Idea: make leakage $\mathscr{L}$ random



➜ each t-uple of v_i is independent from v

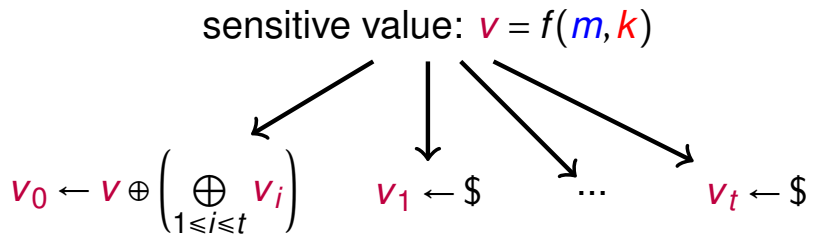
Countermeasures against Power-Analysis Attacks



Problem: leakage $\mathscr{L}$ is key-dependent

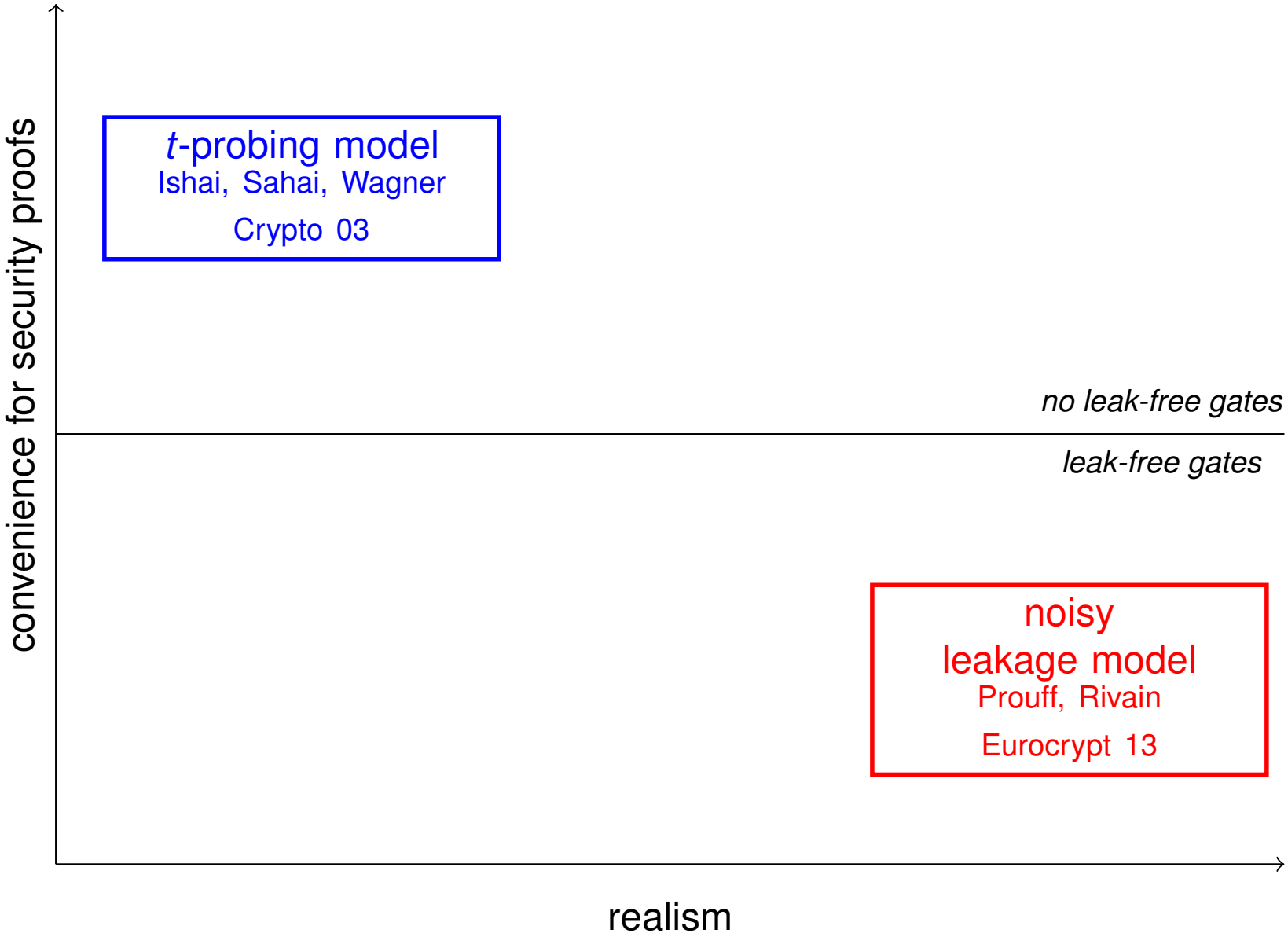
Masking

Idea: make leakage $\mathscr{L}$ random

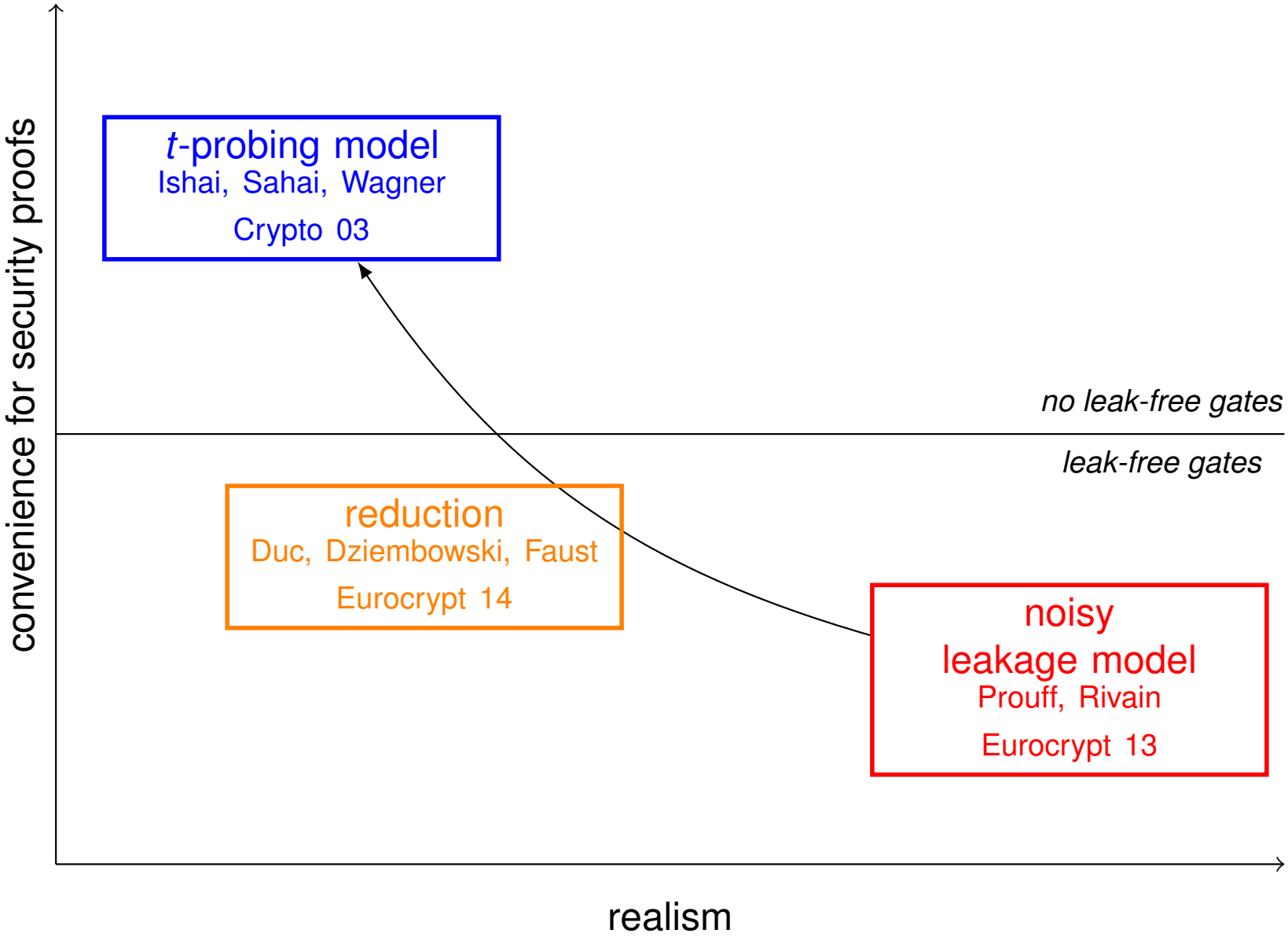


➜ each t-uple of v_i is independent from v

Security of Masked Programs: Leakage Model



Security of Masked Programs: Leakage Model



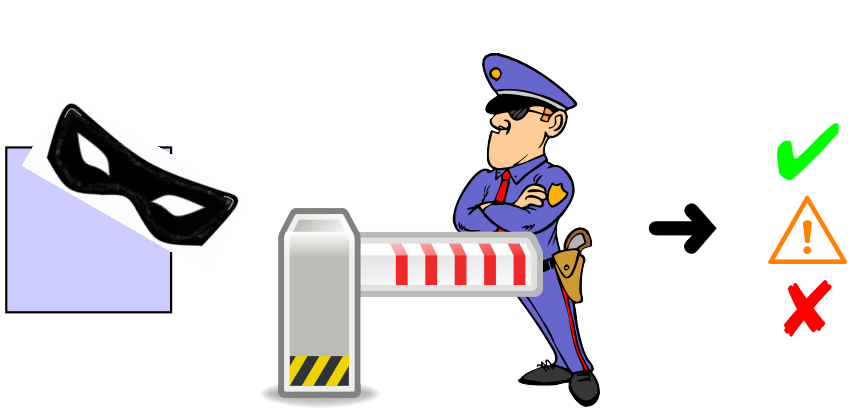
Chosen Contributions

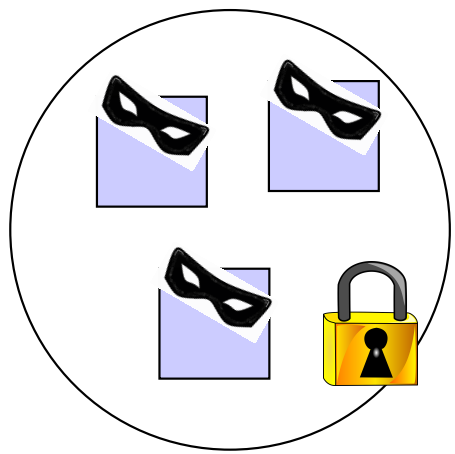
Masking and t-probing model:

1. Verified Proofs of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, B. Grégoire, and P-Y. Strub. Eurocrypt 2015

2. Composition of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, and B. Grégoire. ePrint 2015

Cryptography: countermeasures against Power-Analysis Attacks



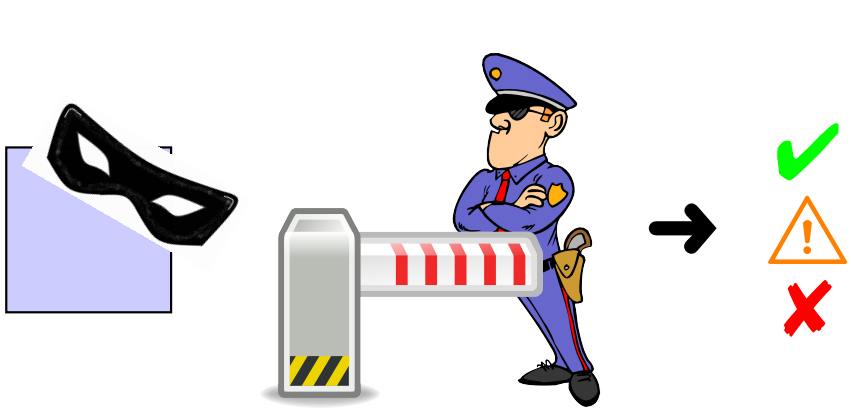


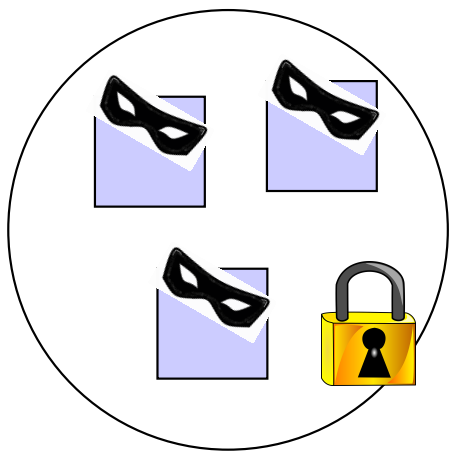
Chosen Contributions

Masking and t-probing model:

1. Verified Proofs of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, B. Grégoire, and P-Y. Strub — Eurocrypt 2015
2. Composition of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, and B. Grégoire — ePrint 2015

Cryptography: countermeasures against Power-Analysis Attacks





Security in the t-probing model

1. show that a t-uple is independent from the secret
2. test all the possible t-uples

function Ex-t3(x_1, x_2, x_3, x_4, c):

(* $x_1, x_2, x_3 = \$$ *)
(* $x_4 = x + x_1 + x_2 + x_3$ *)

$r_1 \leftarrow \$$
$r_2 \leftarrow \$$
$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$
$t_1 \leftarrow x_2 + r_1$
$t_2 \leftarrow (x_2 + r_1) + x_3$
$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$
$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

Security in the t-probing model

1. show that a t-uple is independent from the secret
2. test all the possible t-uples

function Ex-t3(x_1, x_2, x_3, x_4, c):

(* $x_1, x_2, x_3 = \$$ *)
(* $x_4 = x + x_1 + x_2 + x_3$ *)

$r_1 \leftarrow \$$
$r_2 \leftarrow \$$
$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$
$t_1 \leftarrow x_2 + r_1$
$t_2 \leftarrow (x_2 + r_1) + x_3$
$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$
$y_4 \leftarrow c + r_2$
return (y_1, y_2, y_3, y_4)

1. independent from the secret? ✔

Security in the t-probing model

1. show that a t-uple is independent from the secret
2. test all the possible t-uples

function Ex-t3(x_1, x_2, x_3, x_4, c):

(* $x_1, x_2, x_3 = \$$ *)
(* $x_4 = x + x_1 + x_2 + x_3$ *)

$r_1 \leftarrow \$$
$r_2 \leftarrow \$$
$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$
$t_1 \leftarrow x_2 + r_1$
$t_2 \leftarrow (x_2 + r_1) + x_3$
$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$
$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

1. independent from the secret?

✘

Security in the t-probing model

1. show that a t-uple is independent from the secret
2. test all the possible t-uples

function Ex-t3(x_1, x_2, x_3, x_4, c):

(* $x_1, x_2, x_3 = \$$ *)
(* $x_4 = x + x_1 + x_2 + x_3$ *)

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

1. independent from the secret?

?

Security in the t-probing model

1. show that a t-uple is independent from the secret
2. test all the possible t-uples

function Ex-t3(x_1, x_2, x_3, x_4, c):

(* $x_1, x_2, x_3 = \$$ *)
(* $x_4 = x + x_1 + x_2 + x_3$ *)

$r_1 \leftarrow \$$
$r_2 \leftarrow \$$
$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$
$t_1 \leftarrow x_2 + r_1$
$t_2 \leftarrow (x_2 + r_1) + x_3$
$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$
$y_4 \leftarrow c + r_2$
return(y_1, y_2, y_3, y_4)

1. independent from the secret?

✘ many mistakes

Security in the t-probing model

1. show that a t-uple is independent from the secret
2. test all the possible t-uples

function Ex-t3(x_1, x_2, x_3, x_4, c):

(* $x_1, x_2, x_3 = \$$ *)
(* $x_4 = x + x_1 + x_2 + x_3$ *)

$r_1 \leftarrow \$$
$r_2 \leftarrow \$$
$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$
$t_1 \leftarrow x_2 + r_1$
$t_2 \leftarrow (x_2 + r_1) + x_3$
$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$
$y_4 \leftarrow c + r_2$
return(y_1, y_2, y_3, y_4)

1. independent from the secret?

✘ many mistakes

2. test 286 3-uples

✘ missing cases
✘ inefficient

1. Show that a t-uple is independent from the secret

Inputs: t intermediate variables, $b \leftarrow \text{true}$

(Rule 1) secret variables?

- yes ➜ (Rule 2)
- no ➜ ✔

(Rule 2) an expression v is invertible in the only occurrence of a random r?

- yes ➜ $v \leftarrow r$; (Rule 1)
- no ➜ (Rule 3)

(Rule 3) is flag $b = \text{true}$?

- yes ➜ simplify; $b \leftarrow \text{false}$; (Rule 1)
- no ➜ ✘

function $\text{Ex-t3}(x_1, x_2, x_3, x_4, c)$:

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

$\text{return}(y_1, y_2, y_3, y_4)$

✔ ➜ distribution independent from the secret

✘ ➜ might be used for an attack

1. Show that a t-uple is independent from the secret

Inputs: t intermediate variables, $b \leftarrow \text{true}$

(Rule 1) secret variables?

- yes ➜ (Rule 2)
- no ➜ ✔

(Rule 2) an expression v is invertible in the only occurrence of a random r?

- yes ➜ $v \leftarrow r$; (Rule 1)
- no ➜ (Rule 3)

(Rule 3) is flag $b = \text{true}$?

- yes ➜ simplify; $b \leftarrow \text{false}$; (Rule 1)
- no ➜ ✘

function Ex-t3(x_1, x_2, x_3, x_4, c):

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

✔ ➜ distribution independent from the secret

✘ ➜ might be used for an attack

1. Show that a t-uple is independent from the secret

Inputs: t intermediate variables, $b \leftarrow \text{true}$

(Rule 1) secret variables?

yes ➜ (Rule 2)

no ➜ ✔

(Rule 2) an expression v is invertible in the only occurrence of a random r?

yes ➜ $v \leftarrow r$; (Rule 1)

no ➜ (Rule 3)

(Rule 3) is flag $b = \text{true}$?

yes ➜ simplify; $b \leftarrow \text{false}$; (Rule 1)

no ➜ ✘

function Ex-t3(x_1, x_2, x_3, x_4, c):

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

✔ ➜ distribution independent from the secret

✘ ➜ might be used for an attack

1. Show that a t-uple is independent from the secret

Inputs: t intermediate variables, $b \leftarrow \text{true}$

(Rule 1) secret variables?

- yes ➜ (Rule 2)
- no ➜ ✔

(Rule 2) an expression v is invertible in the only occurrence of a random r?

- yes ➜ $v \leftarrow r$; (Rule 1)
- no ➜ (Rule 3)

(Rule 3) is flag $b = \text{true}$?

- yes ➜ simplify; $b \leftarrow \text{false}$; (Rule 1)
- no ➜ ✘

function $\text{Ex-t3}(x_1, x_2, x_3, x_4, c)$:

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow x_3$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

$\text{return}(y_1, y_2, y_3, y_4)$

✔ ➜ distribution independent from the secret

✘ ➜ might be used for an attack

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

New Idea: proofs for sets of more than t variables

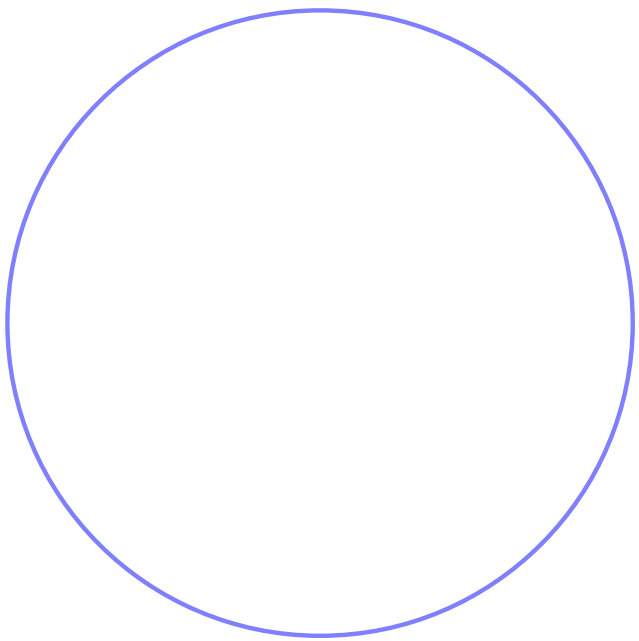
- find larger sets which cover all the intermediate variables is a hard problem
- two algorithms efficient in practice

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

New Idea: proofs for sets of more than t variables

- find larger sets which cover all the intermediate variables is a hard problem
- two algorithms efficient in practice



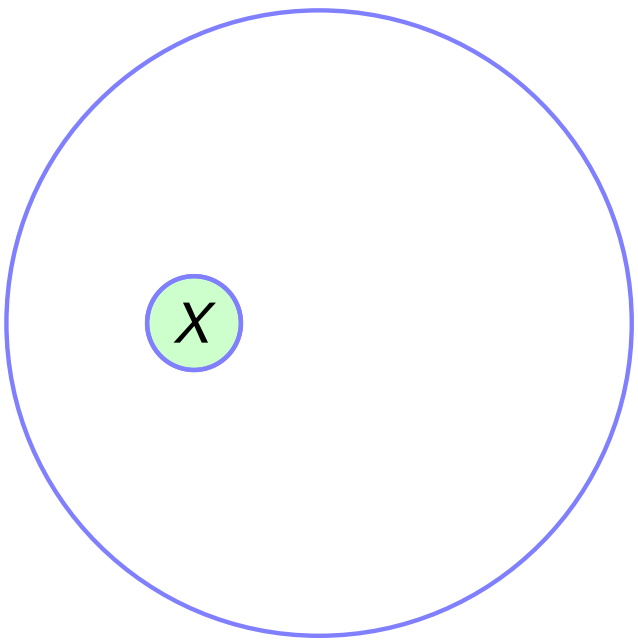
Algorithm 1:

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

New Idea: proofs for sets of more than t variables

- find larger sets which cover all the intermediate variables is a hard problem
- two algorithms efficient in practice



Algorithm 1:

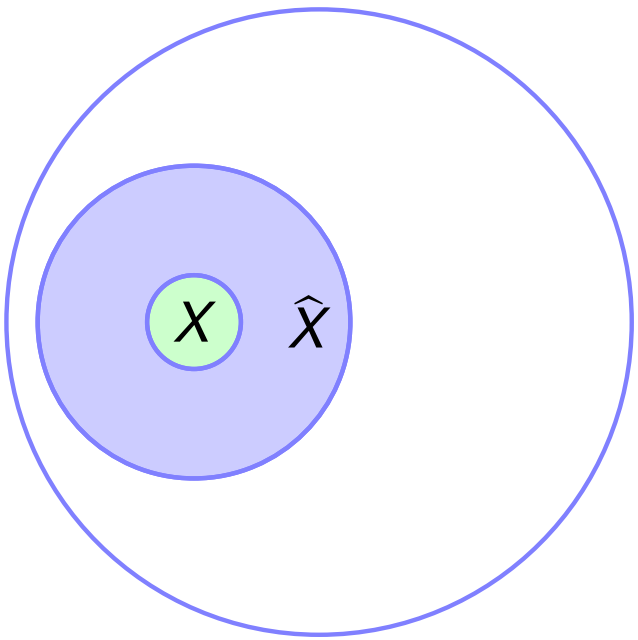
1. select $X = (t \text{ variables})$ and prove its independence

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

New Idea: proofs for sets of more than t variables

- find larger sets which cover all the intermediate variables is a hard problem
- two algorithms efficient in practice



Algorithm 1:

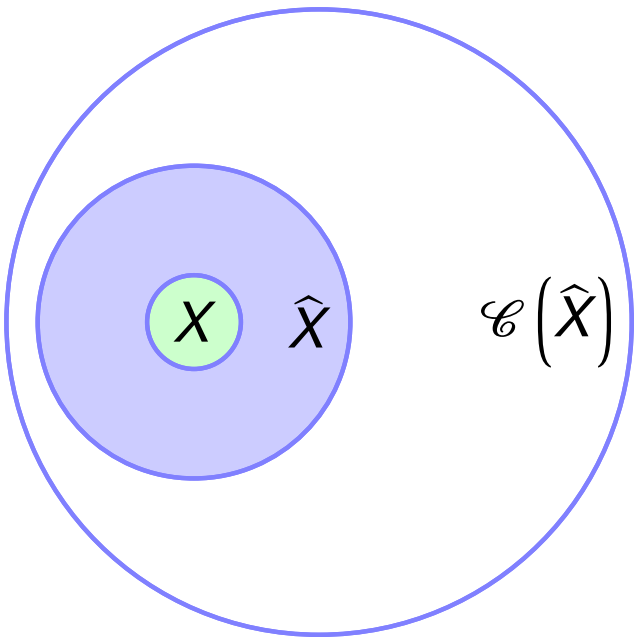
1. select $X = (t \text{ variables})$ and prove its independence
2. extend X to $\widehat{X}$ with more observations but still independence

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

New Idea: proofs for sets of more than t variables

- find larger sets which cover all the intermediate variables is a hard problem
- two algorithms efficient in practice



Algorithm 1:

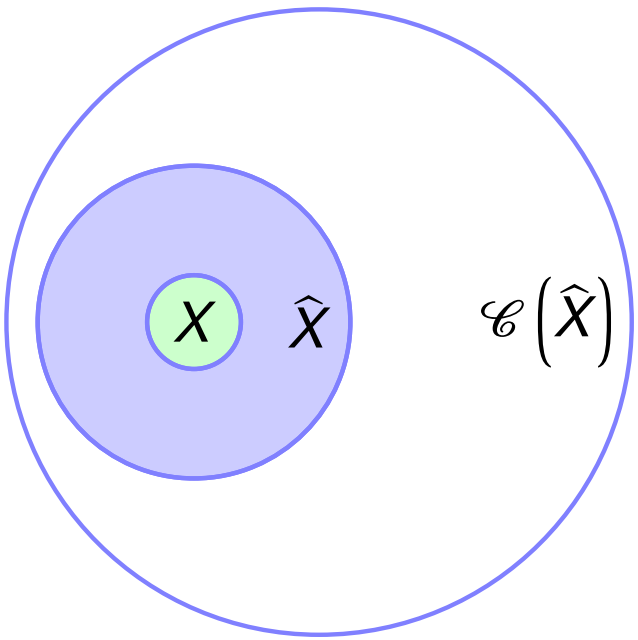
1. select $X = (t$ variables) and prove its independence
2. extend X to $\hat{X}$ with more observations but still independence
3. recursively descend in set $\mathscr{C}\left(\hat{X}\right)$

2. Extension to All Possible Sets

Problem: n intermediate variables ➜ $\binom{n}{t}$ proofs

New Idea: proofs for sets of more than t variables

- find larger sets which cover all the intermediate variables is a hard problem
- two algorithms efficient in practice



Algorithm 1:

1. select $X = (t \text{ variables})$ and prove its independence
2. extend X to $\widehat{X}$ with more observations but still independence
3. recursively descend in set $\mathscr{C}\left(\widehat{X}\right)$
4. merge $\widehat{X}$ and $\mathscr{C}\left(\widehat{X}\right)$ once they are processed separately.

2. Extension to All Possible Sets: Example

function $\text{Ex-t3}(x_1, x_2, x_3, x_4, c)$:

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

2. Extension to All Possible Sets: Example

X: ✔

function Ex-t3(x_1, x_2, x_3, x_4, c):

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

2. Extension to All Possible Sets: Example

$\widehat{X}$: ✔

function Ex-t3(x_1, x_2, x_3, x_4, c):

$r_1 \leftarrow \$$

$r_2 \leftarrow \$$

$y_1 \leftarrow x_1 + r_1$

$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$t_1 \leftarrow x_2 + r_1$

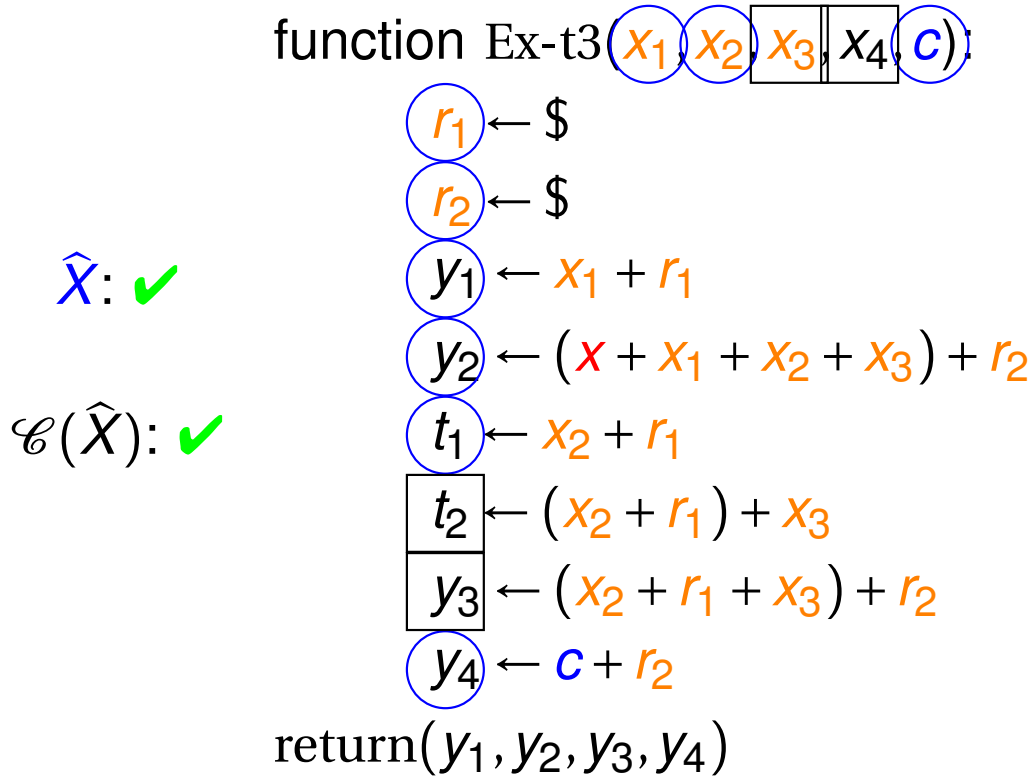
$t_2 \leftarrow (x_2 + r_1) + x_3$

$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$

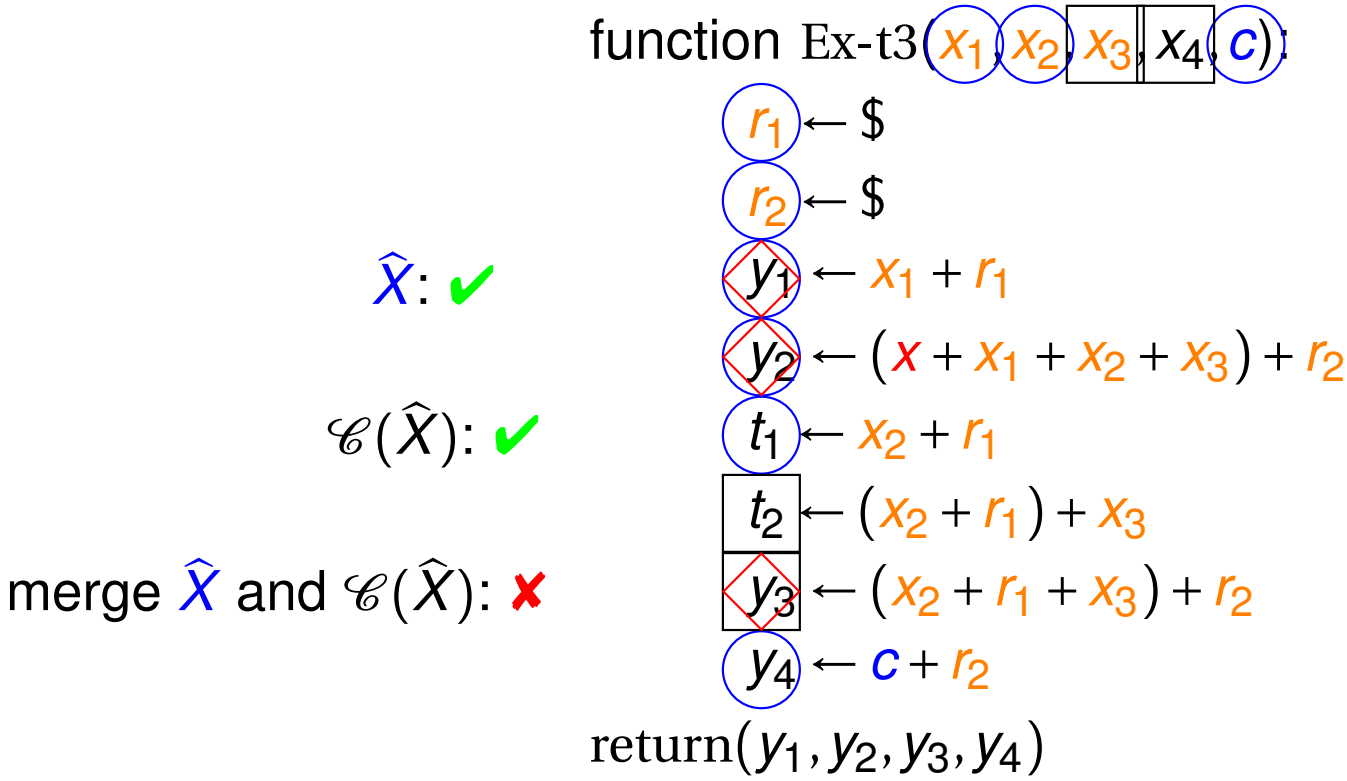
$y_4 \leftarrow c + r_2$

return(y_1, y_2, y_3, y_4)

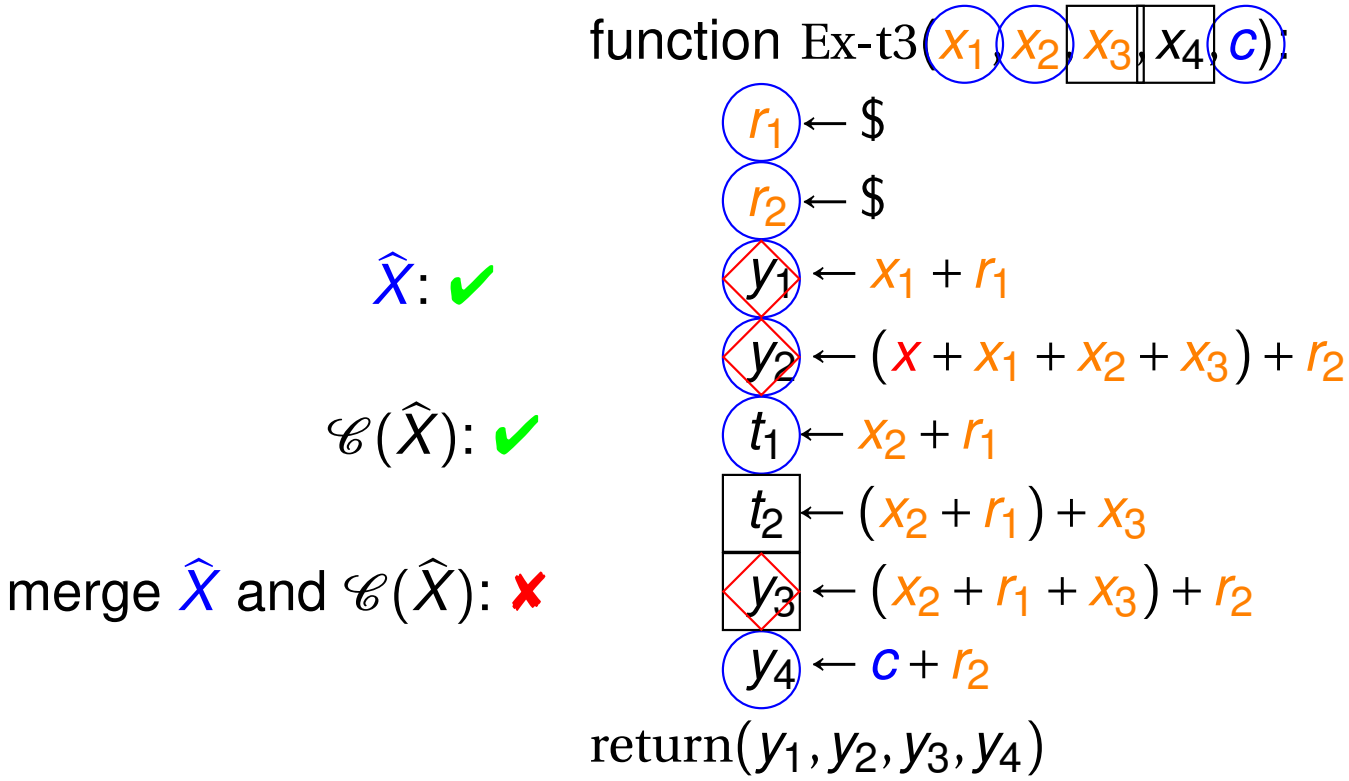
2. Extension to All Possible Sets: Example



2. Extension to All Possible Sets: Example



2. Extension to All Possible Sets: Example



➜ 207 proofs instead of 286

Application to the Sbox [CPRR13, Algorithm 4]

Method	# tuples	Security	Complexity # sets	Complexity time*
First-Order Masking				
naive	63	✔	63	0.001s
Alg. 1			17	0.001s
Alg. 2			17	0.001s
Second-Order Masking				
naive	12,561	✔	12,561	0.180s
Alg. 1			851	0.046s
Alg. 2			619	0.029s
Third-Order Masking				
naive	4,499,950	✔	4,499,950	140.642s
Alg. 1			68,492	9.923s
Alg. 2			33,075	3.894s
Fourth-Order Masking				
naive	2,277,036,685	✔	-	unpractical
Alg. 1			8,852,144	2959.770s
Alg. 2			3,343,587	879.235s

*run on a headless VM with a dual core (only one core is used in the computation) 64-bit processor clocked at 2GHz

Benchmarks

Reference	Target	# tuples	Security	Complexity # sets	Complexity time (s)
First-Order Masking					
FSE13	full AES	17,206	✔	3,342	128
MAC-SHA3	full Keccak-f	13,466	✔	5,421	405
Second-Order Masking					
RSA06	Sbox	1,188,111	✔	4,104	1.649
CHES10	Sbox	7,140	1^{st}-order flaws (2)	866	0.045
CHES10	AES KS	23,041,866	✔	771,263	340,745
FSE13	2 rnds AES	25,429,146	✔	511,865	1,295
FSE13	4 rnds AES	109,571,806	✔	2,317,593	40,169
Third-Order Masking					
RSA06	Sbox	2,057,067,320	3^{rd}-order flaws (98,176)	2,013,070	695
FSE13	Sbox(4)	4,499,950	✔	33,075	3.894
FSE13	Sbox(5)	4,499,950	✔	39,613	5.036
Fourth-Order Masking					
FSE13	Sbox (4)	2,277,036,685	✔	3,343,587	879
Fifth-Order Masking					
CHES10	⊙	216,071,394	✔	856,147	45

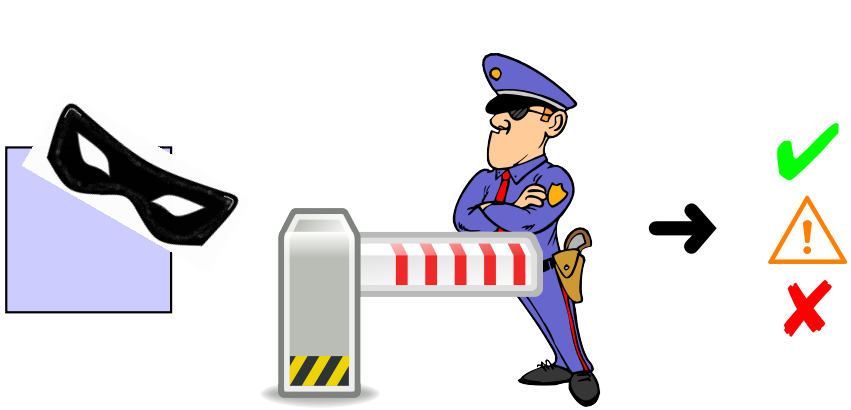
Chosen Contributions

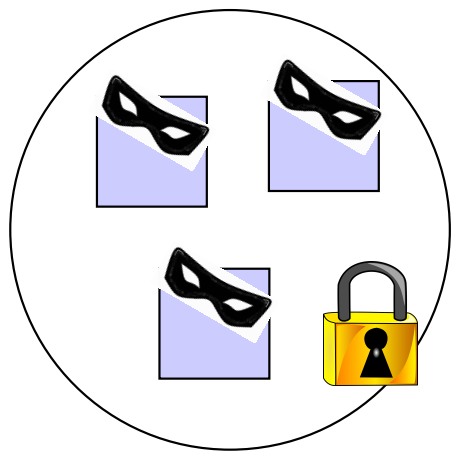
Masking and t-probing model:

1. Verified Proofs of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, B. Grégoire, and P-Y. Strub. Eurocrypt 2015

2. Composition of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, and B. Grégoire. ePrint 2015

Cryptography: countermeasures against Power-Analysis Attacks





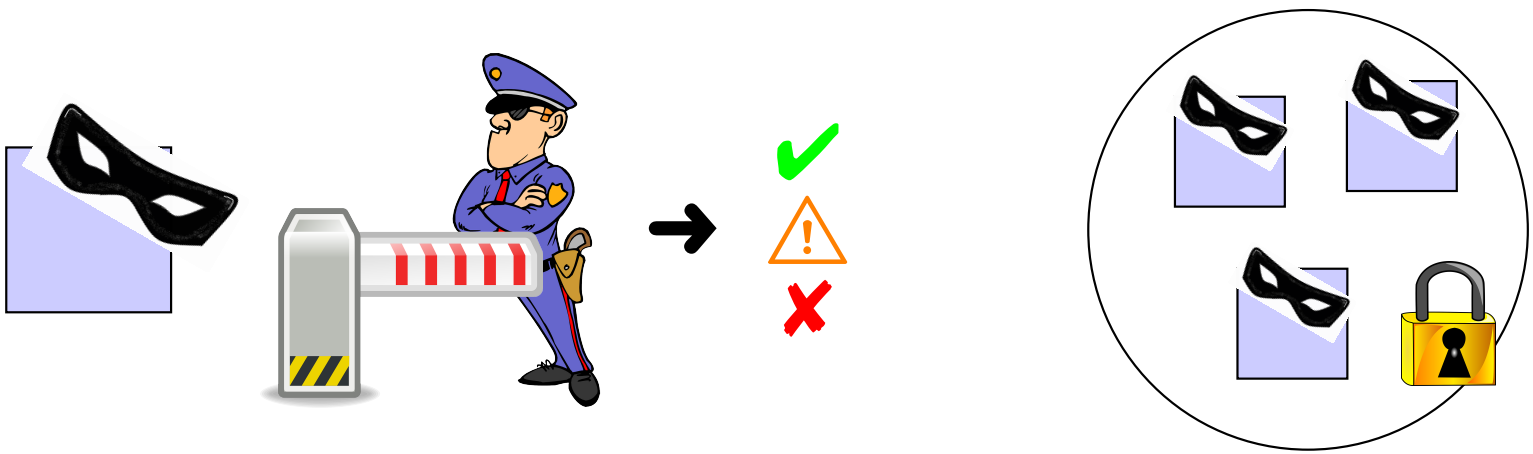
Chosen Contributions

Masking and t-probing model:

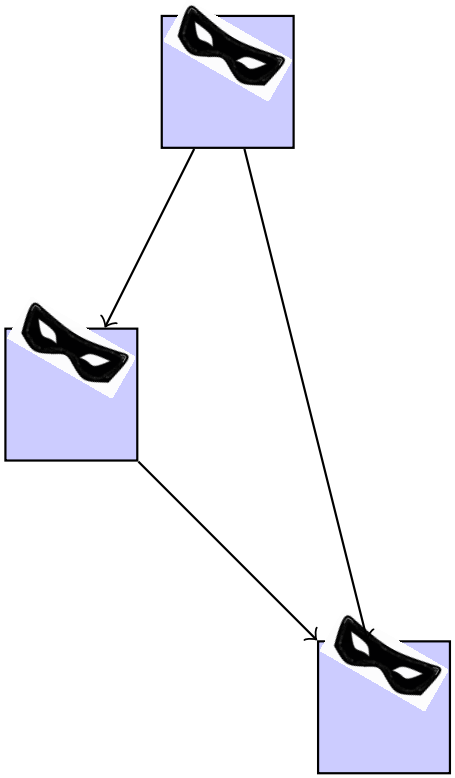
1. Verified Proofs of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, B. Grégoire, and P-Y. Strub. Eurocrypt 2015

2. Composition of Higher-Order Masking → G. Barthe, S. Belaïd, F. Dupressoir, P-A. Fouque, and B. Grégoire. ePrint 2015

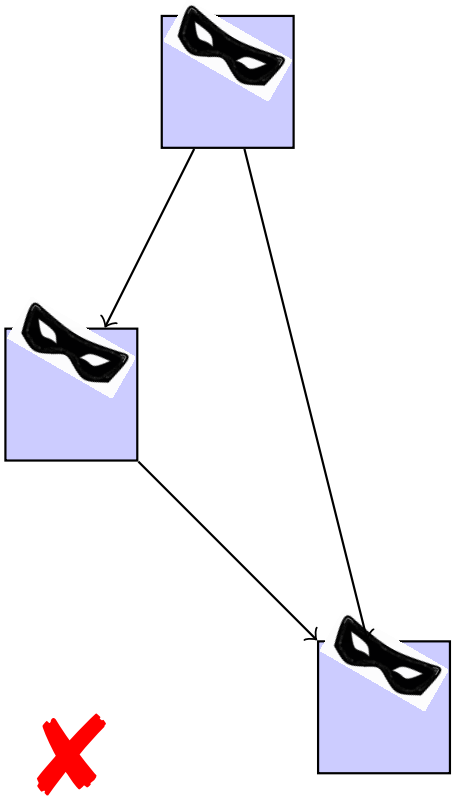
Cryptography: countermeasures against Power-Analysis Attacks



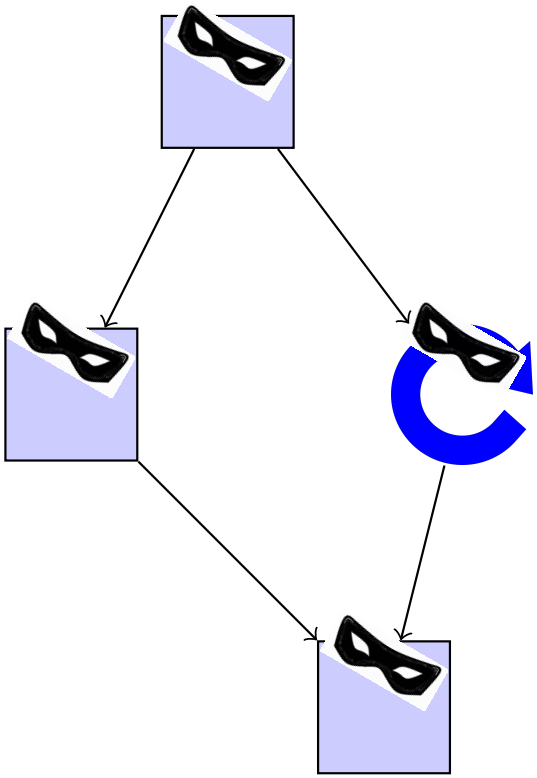
Current Issues in Composition

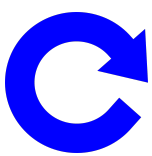


Current Issues in Composition



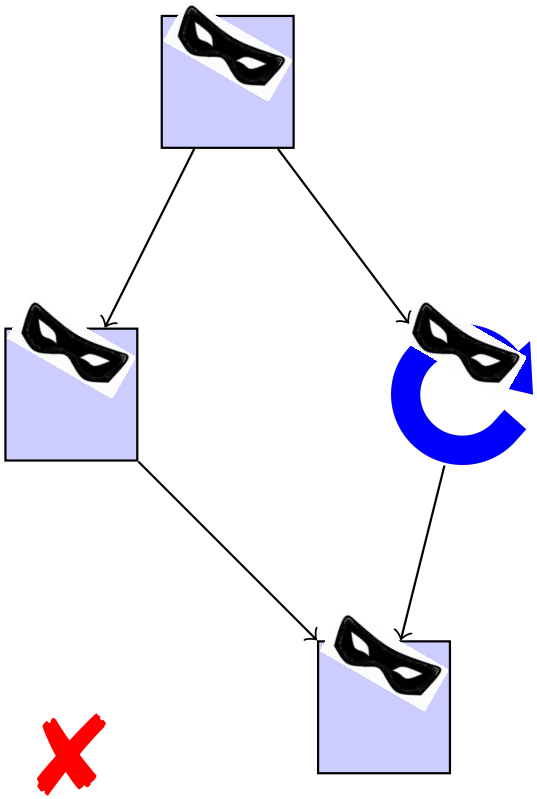
Current Issues in Composition

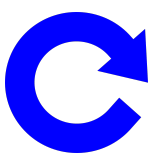




A refresh algorithm takes as input a sharing $(x_i)_{i\geq 0}$ of x and returns a new sharing $(x'_i)_{i\geq 0}$ of x such that $(x_i)_{i\geq 1}$ and $(x'_i)_{i\geq 1}$ are mutually independent.

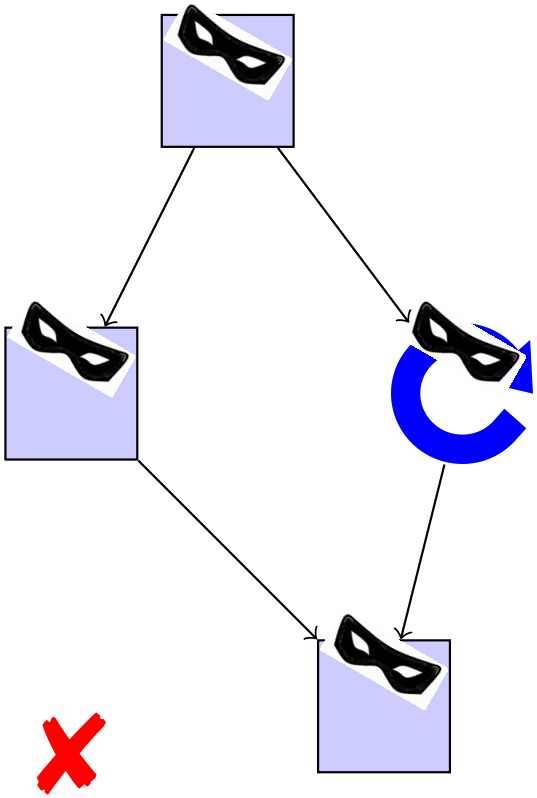
Current Issues in Composition

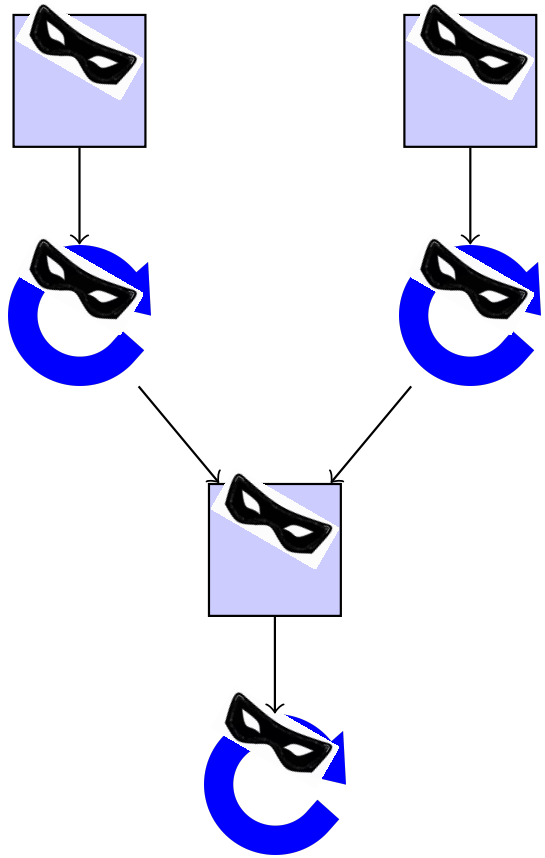


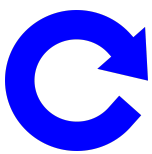


A refresh algorithm takes as input a sharing $(x_i)_{i\geq 0}$ of x and returns a new sharing $(x'_i)_{i\geq 0}$ of x such that $(x_i)_{i\geq 1}$ and $(x'_i)_{i\geq 1}$ are mutually independent.

Current Issues in Composition







A refresh algorithm takes as input a sharing $(x_i)_{i\geq 0}$ of x and returns a new sharing $(x'_i)_{i\geq 0}$ of x such that $(x_i)_{i\geq 1}$ and $(x'_i)_{i\geq 1}$ are mutually independent.

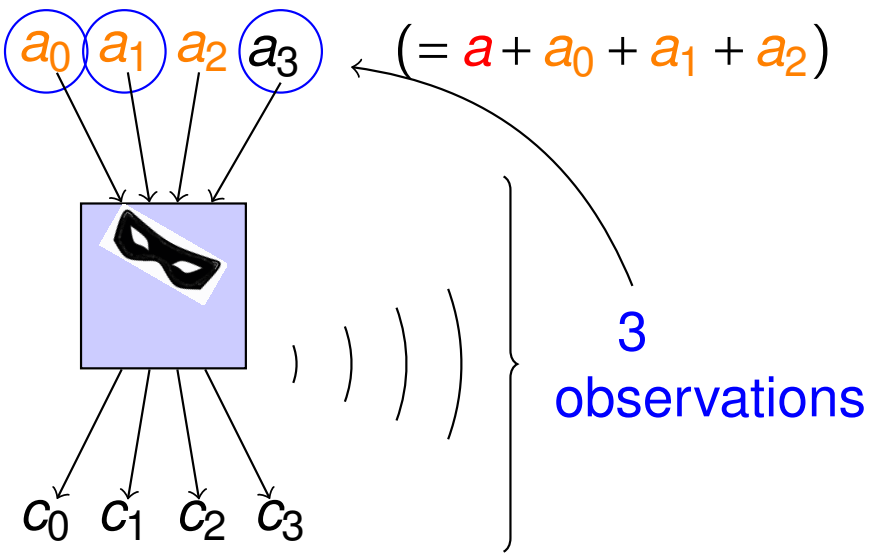
Security properties in the t-probing model

if t is fixed: show that any set of t intermediate variables is independent from the secret

Security properties in the t-probing model

if t is fixed: show that any set of t intermediate variables is independent from the secret

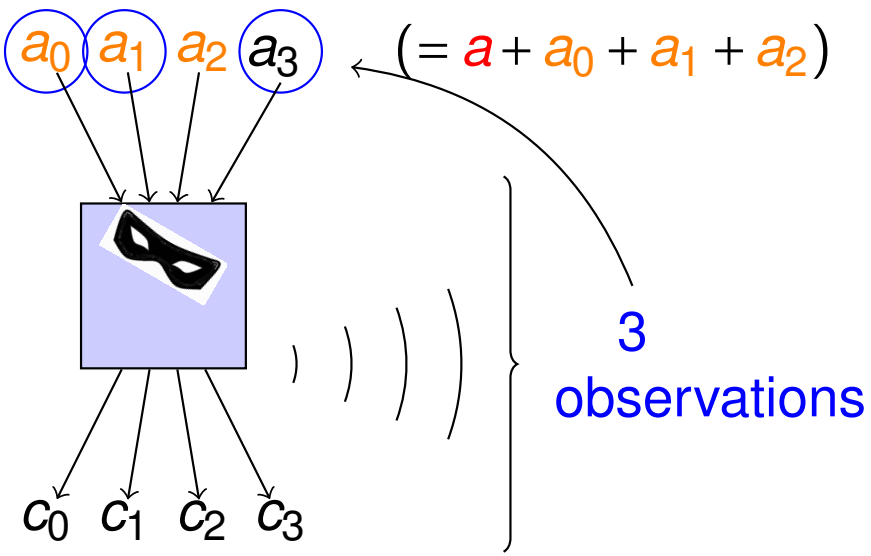
if t is not fixed: show that any set of t intermediate variables can be simulated with at most t shares of each input



Security properties in the t-probing model

if t is fixed: show that any set of t intermediate variables is independent from the secret

if t is not fixed: show that any set of t intermediate variables can be simulated with at most t shares of each input



function Linear-function-t$(a_0, ..., a_i, ...a_t)$:

for $i = 0$ to t

$c_i \leftarrow f(a_i)$

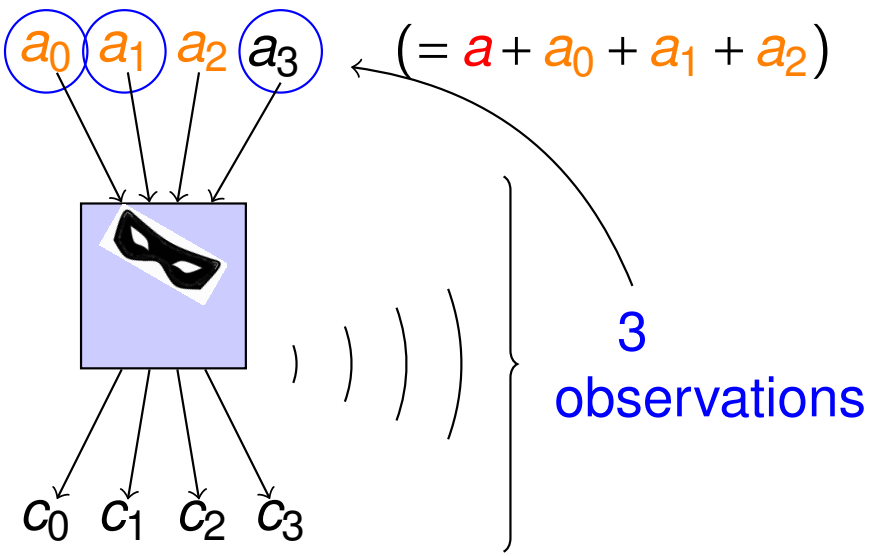
return $(c_0, ..., c_i, ..., c_t)$

➜ straightforward for linear functions

Security properties in the t-probing model

if t is fixed: show that any set of t intermediate variables is independent from the secret

if t is not fixed: show that any set of t intermediate variables can be simulated with at most t shares of each input



function Linear-function-t($a_0, ..., a_i, ...a_t$):

for $i = 0$ to t

$c_i \leftarrow f(a_i)$

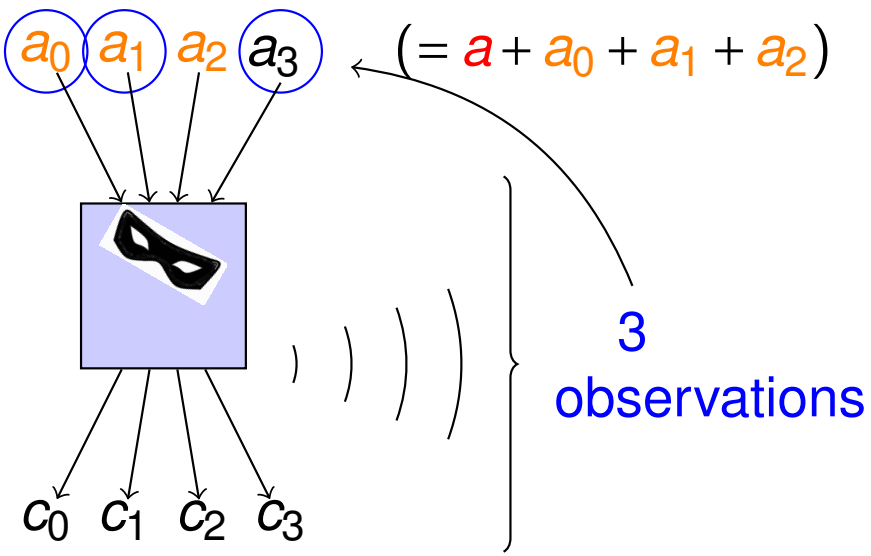
return $(c_0, ..., c_i, ..., c_t)$

➜ straightforward for linear functions

Security properties in the t-probing model

if t is fixed: show that any set of t intermediate variables is independent from the secret

if t is not fixed: show that any set of t intermediate variables can be simulated with at most t shares of each input



function Linear-function-t($a_0, ..., a_i, ...a_t$):

for $i = 0$ to t

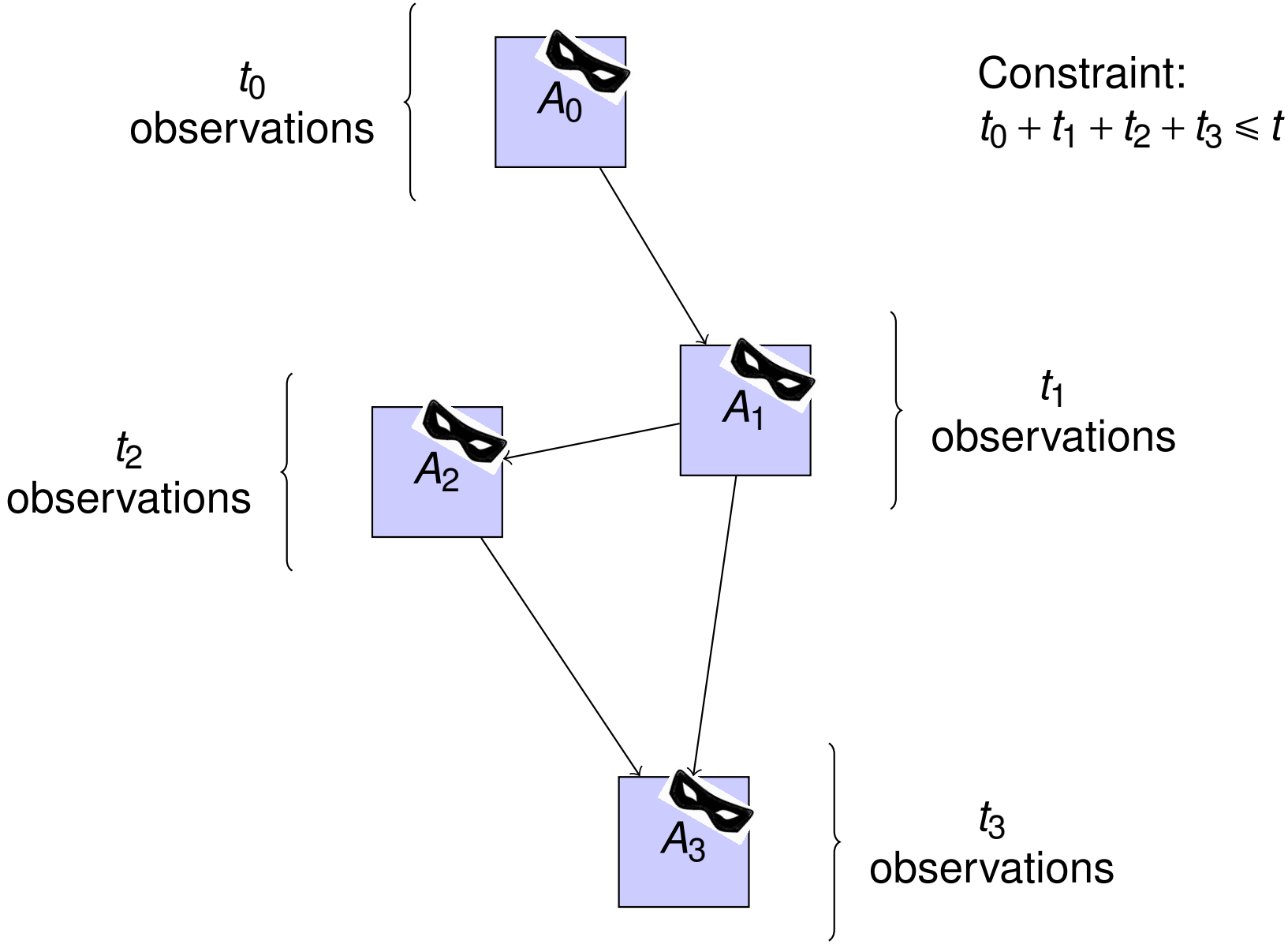
$c_i \leftarrow f(a_i)$

return $(c_0, ..., c_i, ..., c_t)$

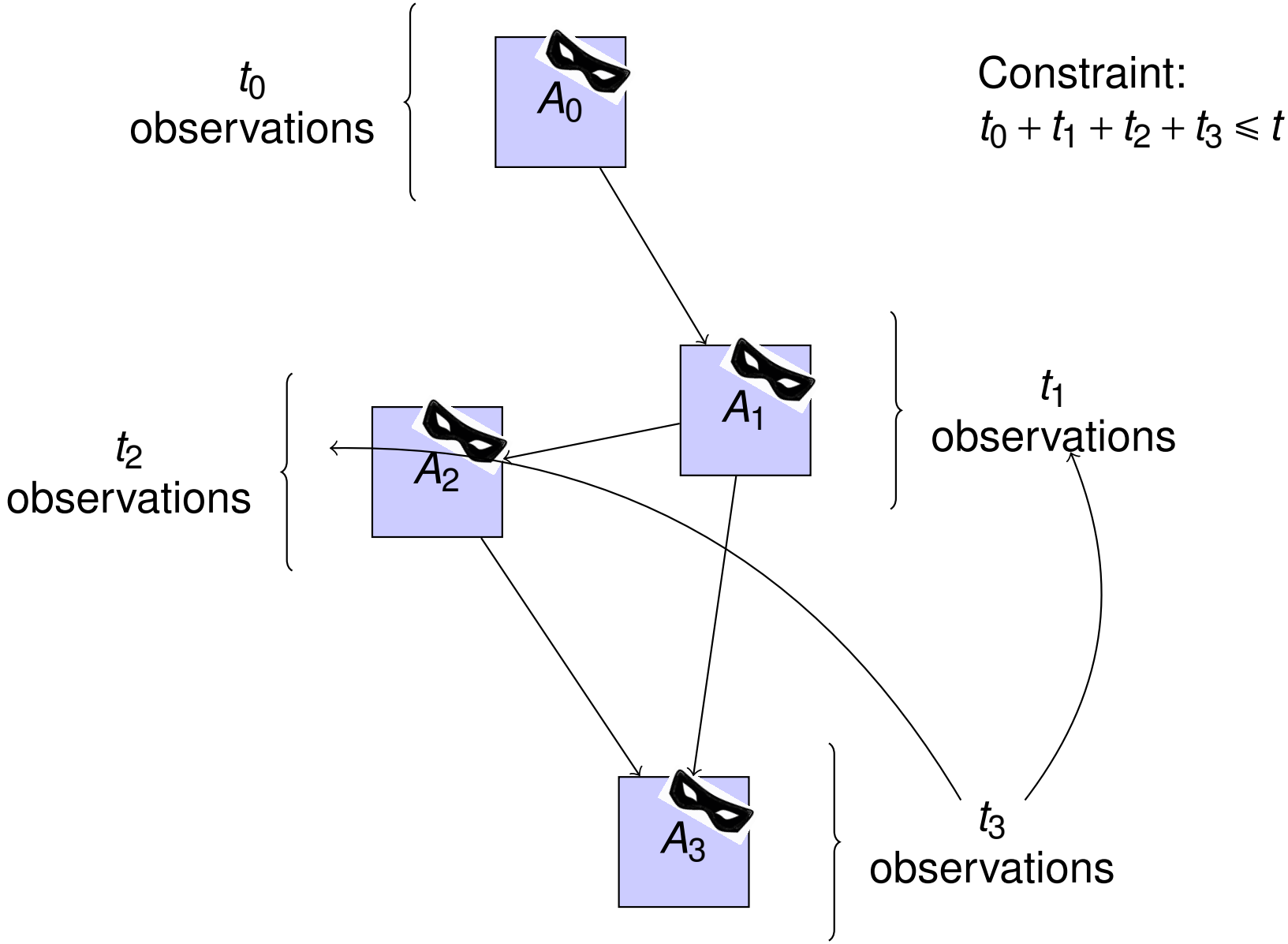
➜ straightforward for linear functions

➜ formal proofs with EasyCrypt and pen-and paper proofs for small non-linear functions

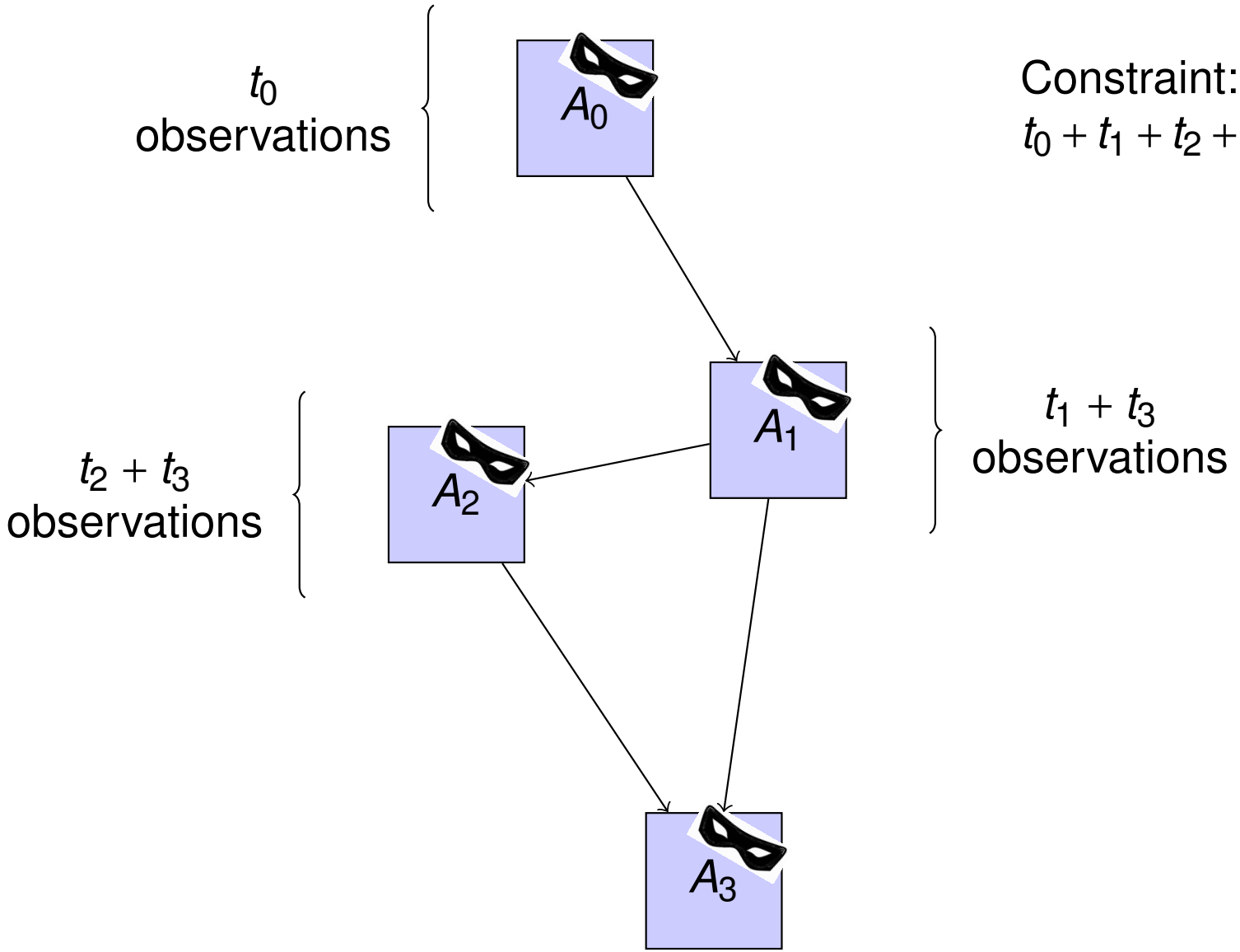
Current Issues



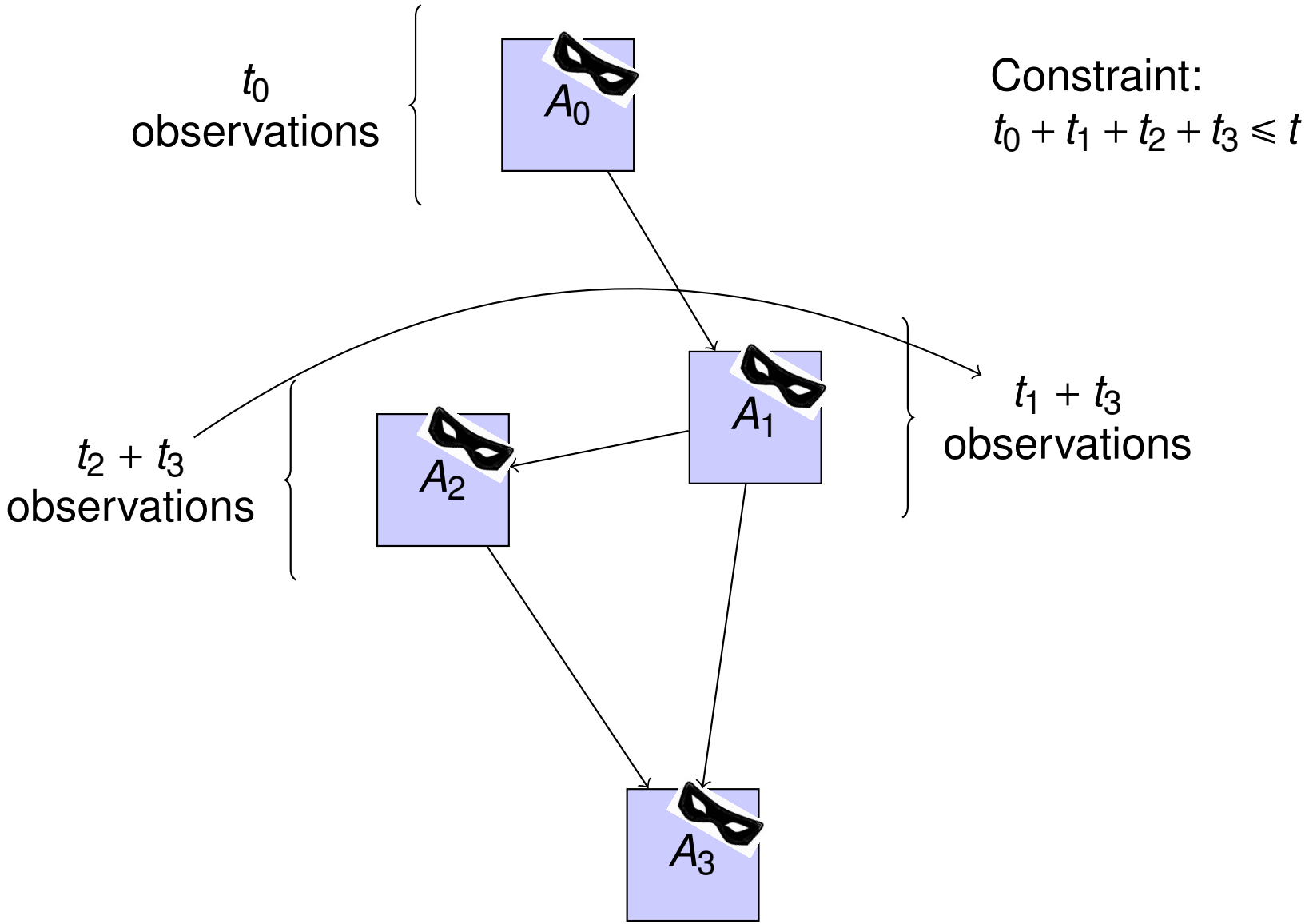
Current Issues



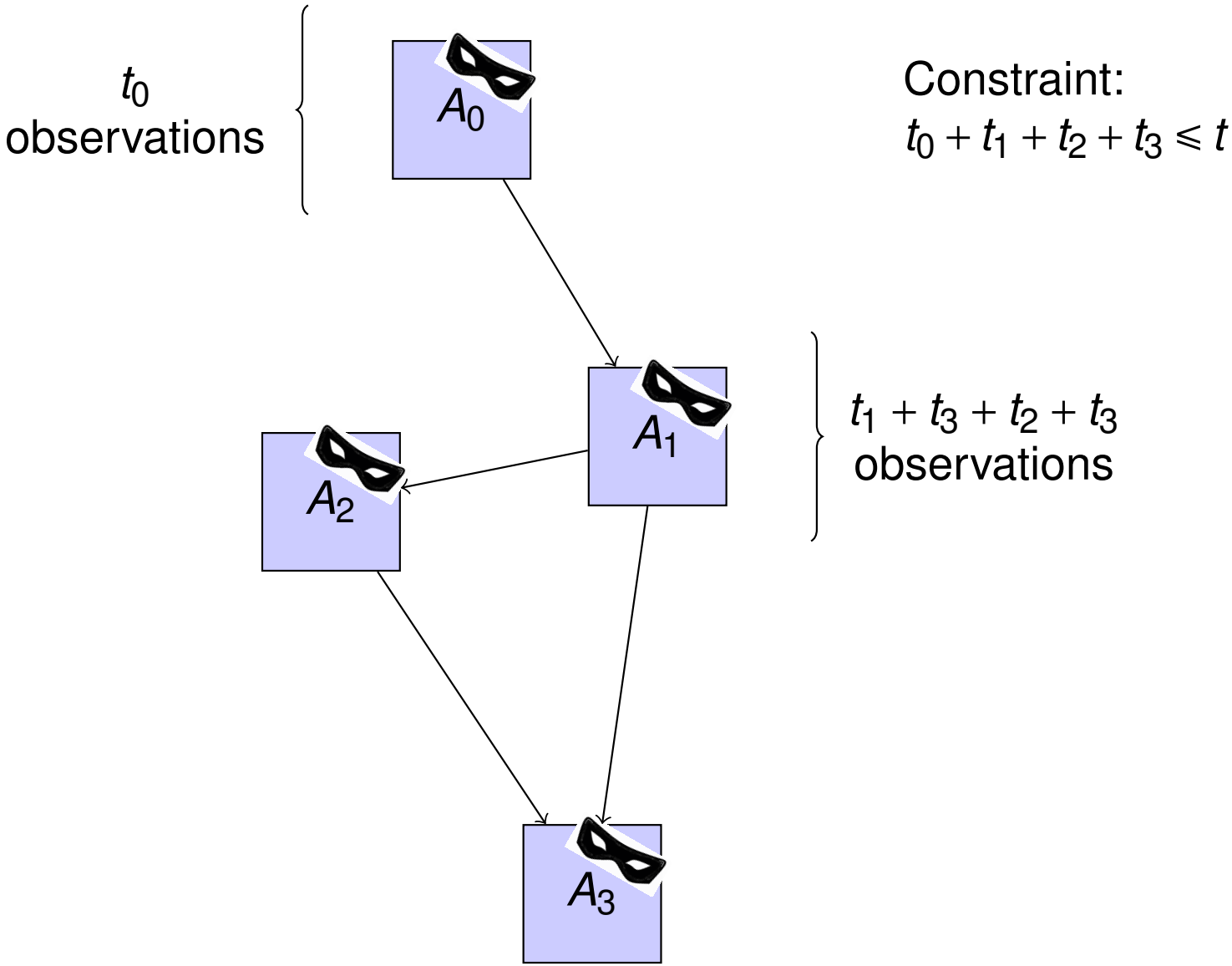
Current Issues



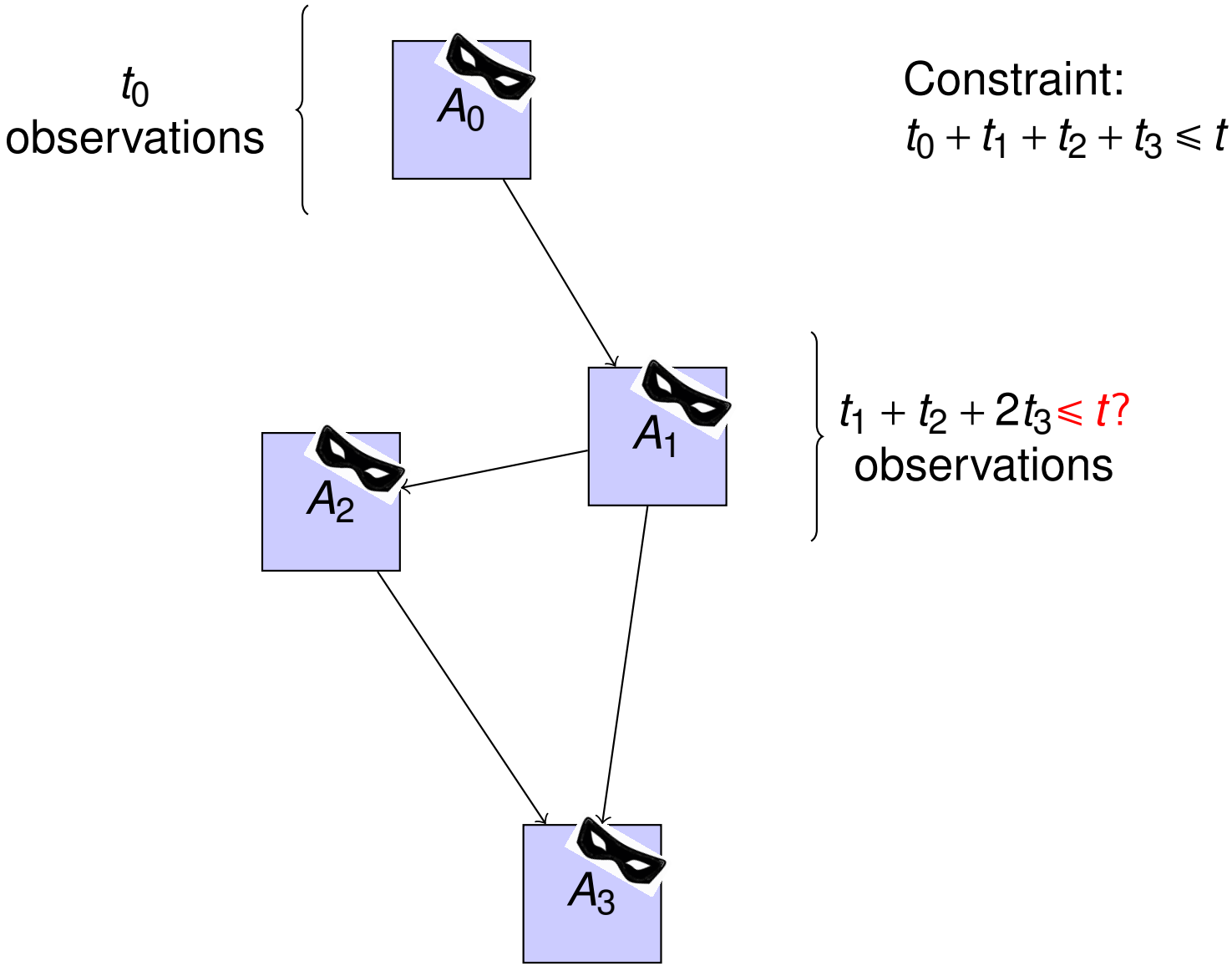
Current Issues



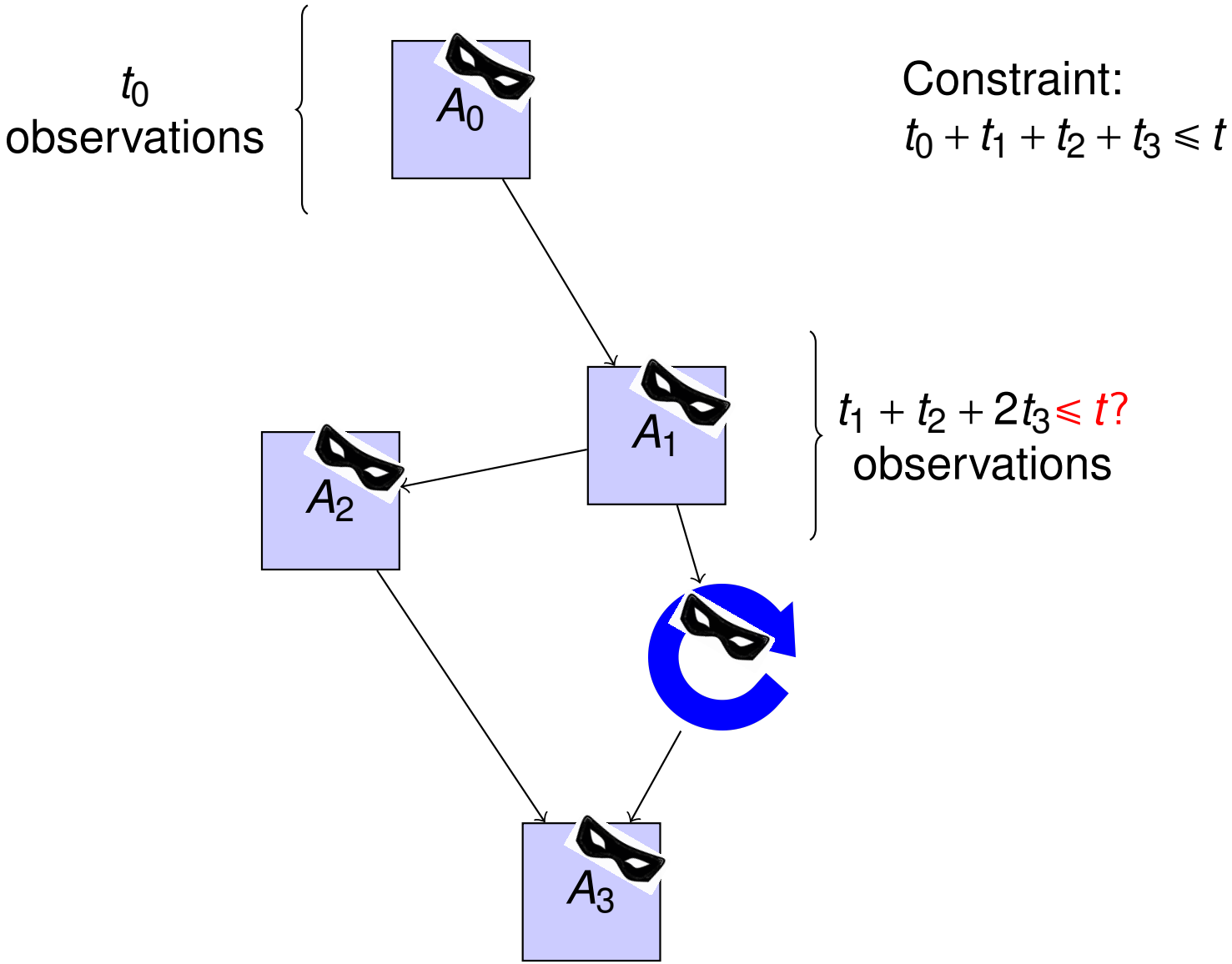
Current Issues



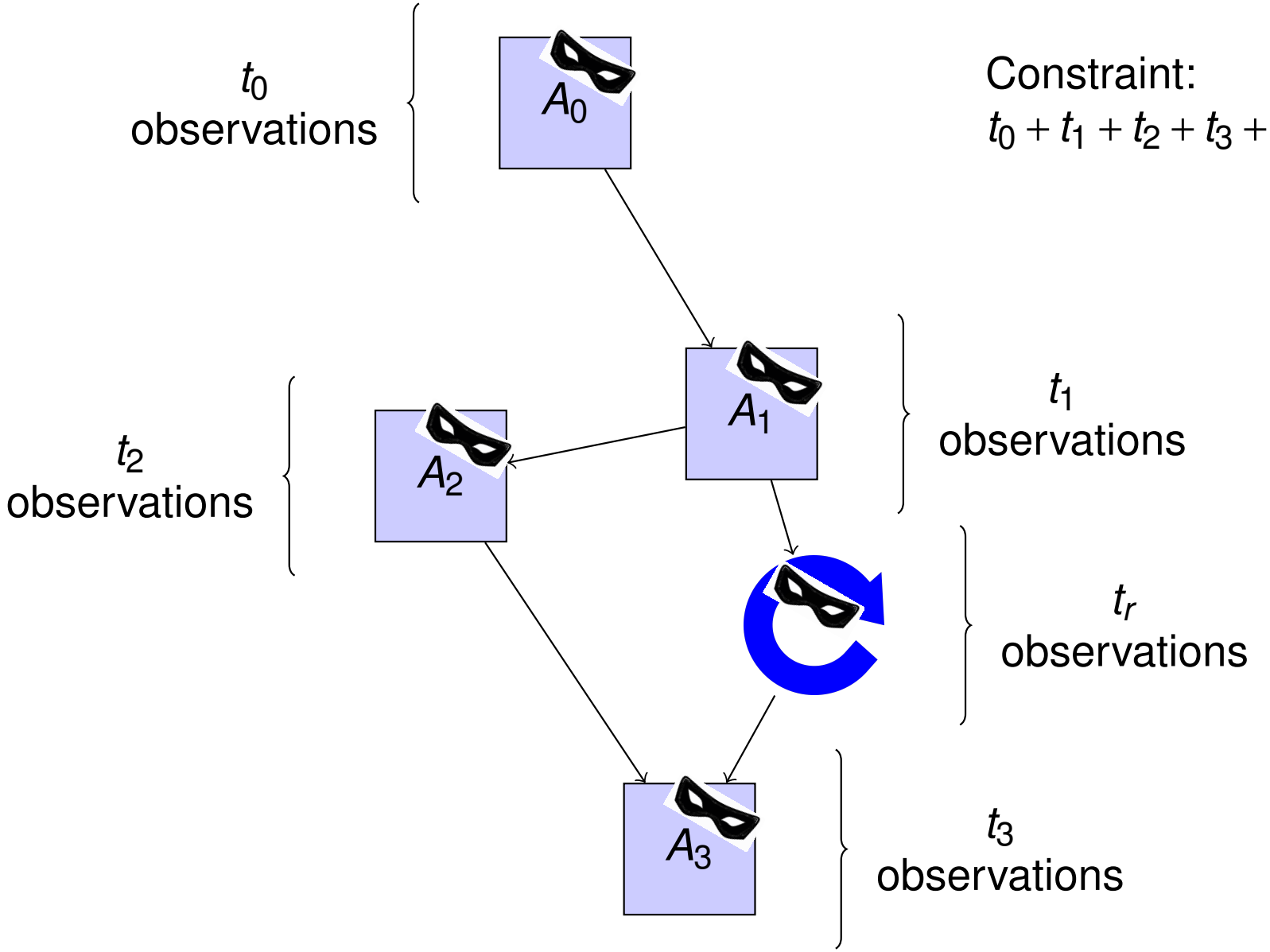
Current Issues



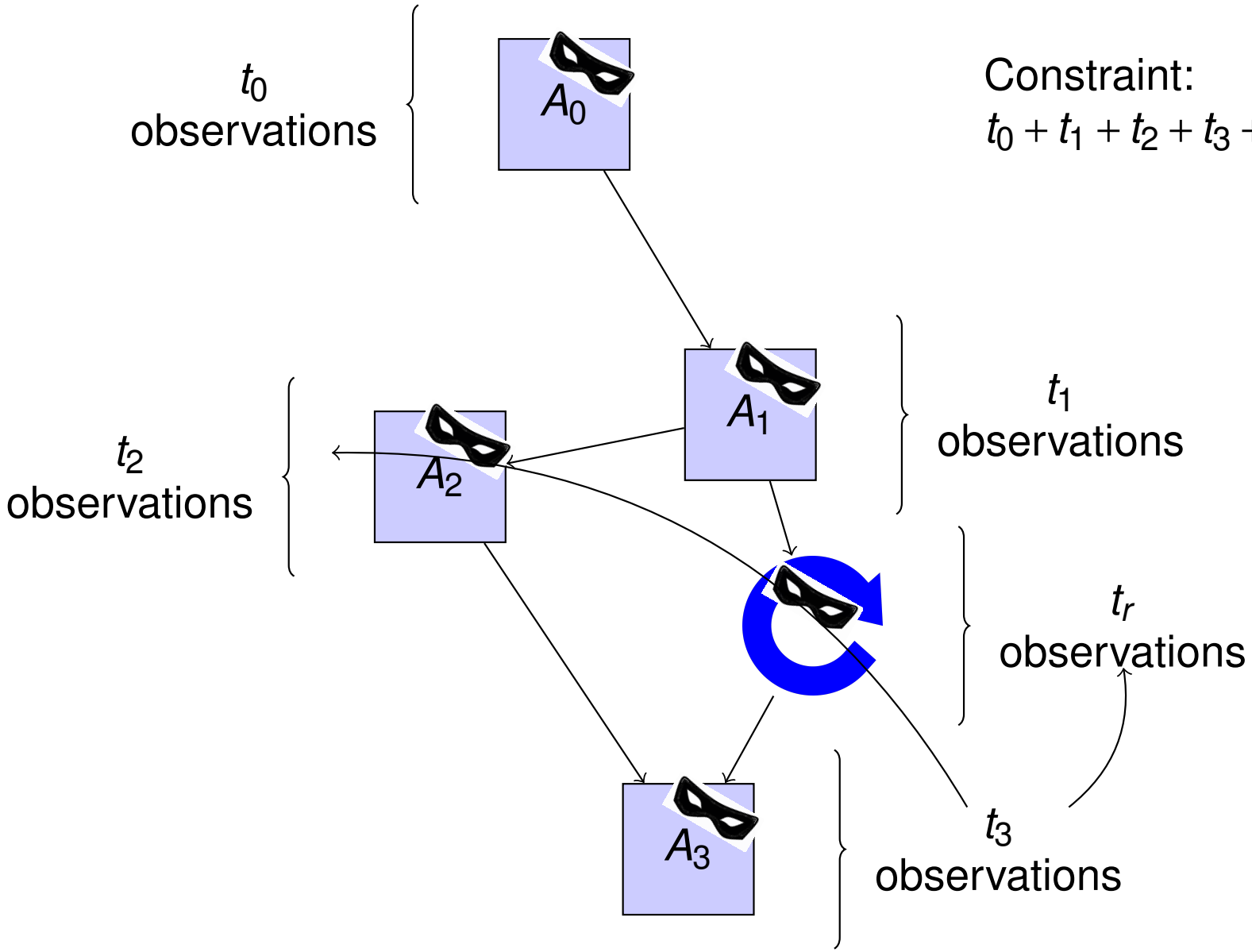
Current Issues



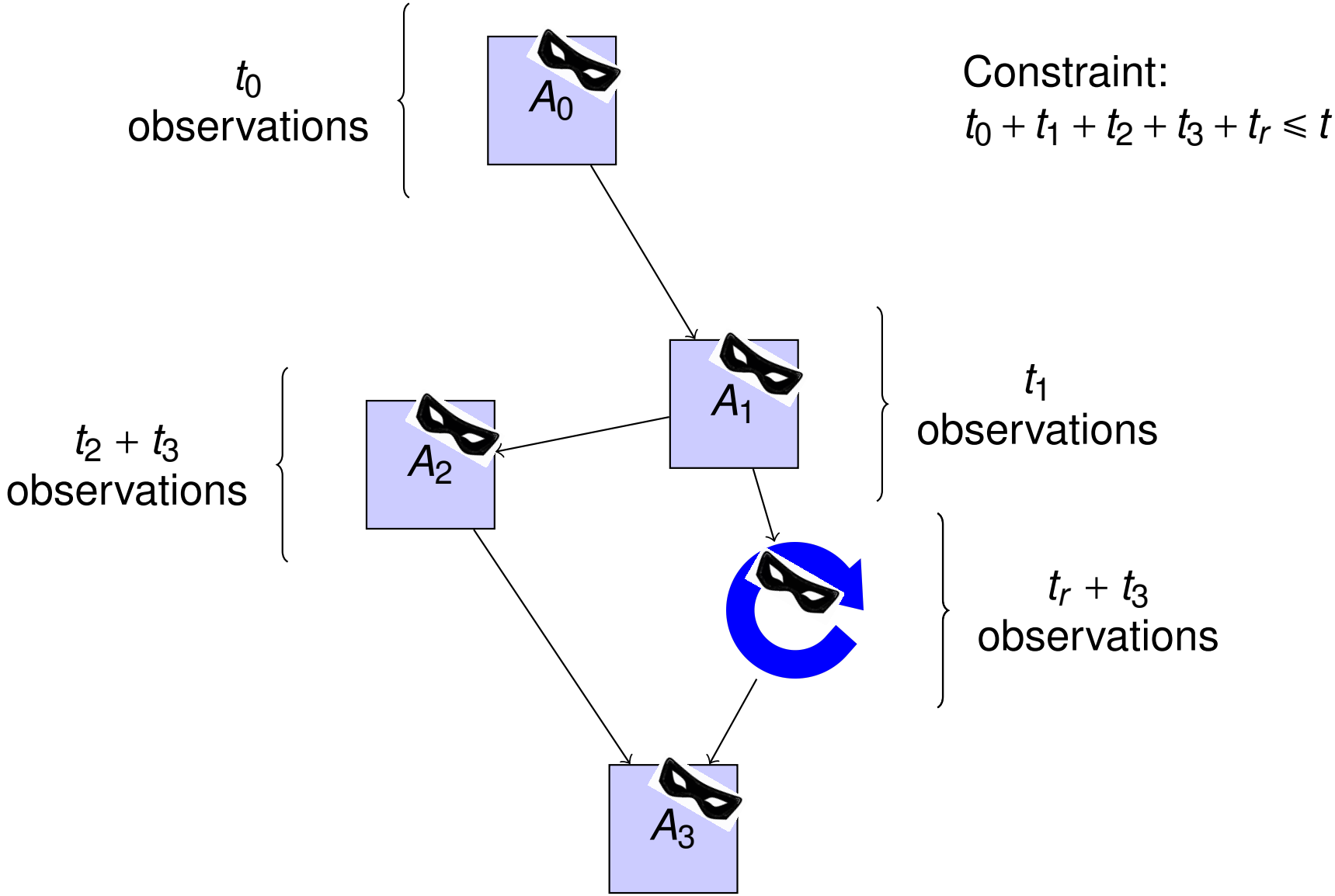
Current Issues



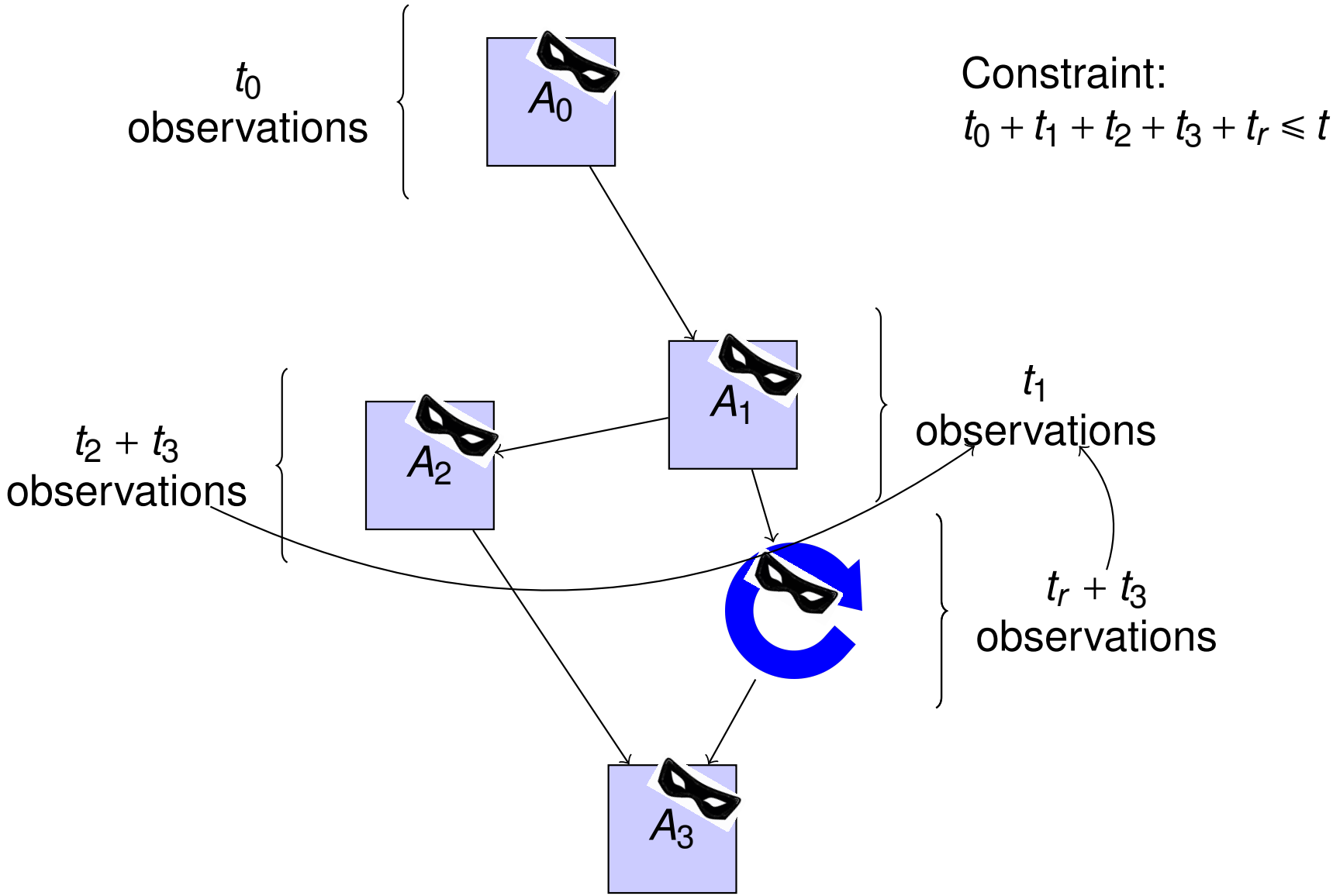
Current Issues



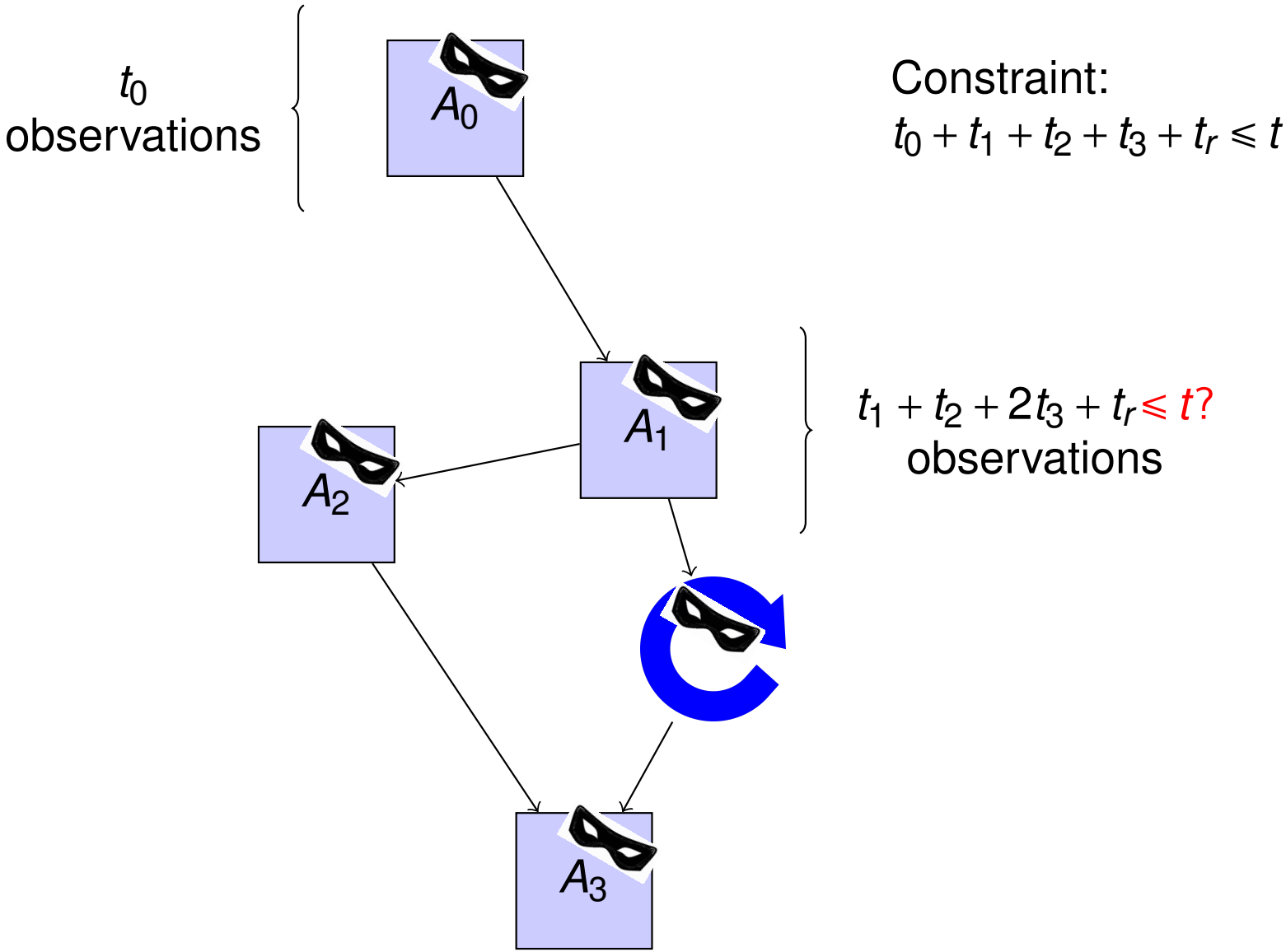
Current Issues



Current Issues



Current Issues



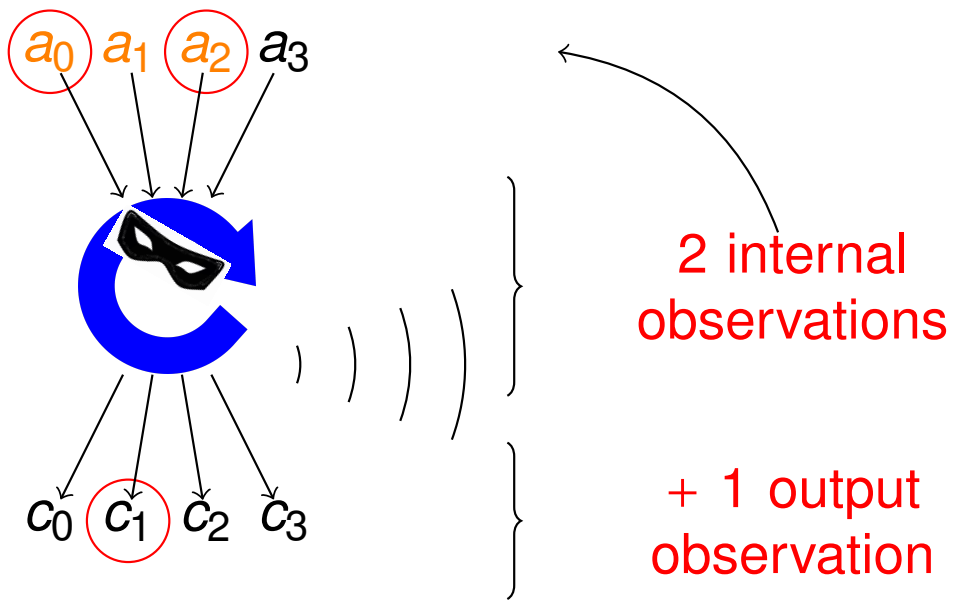
Stronger security property for Refresh

Strong Non-Interference in the t-probing model:

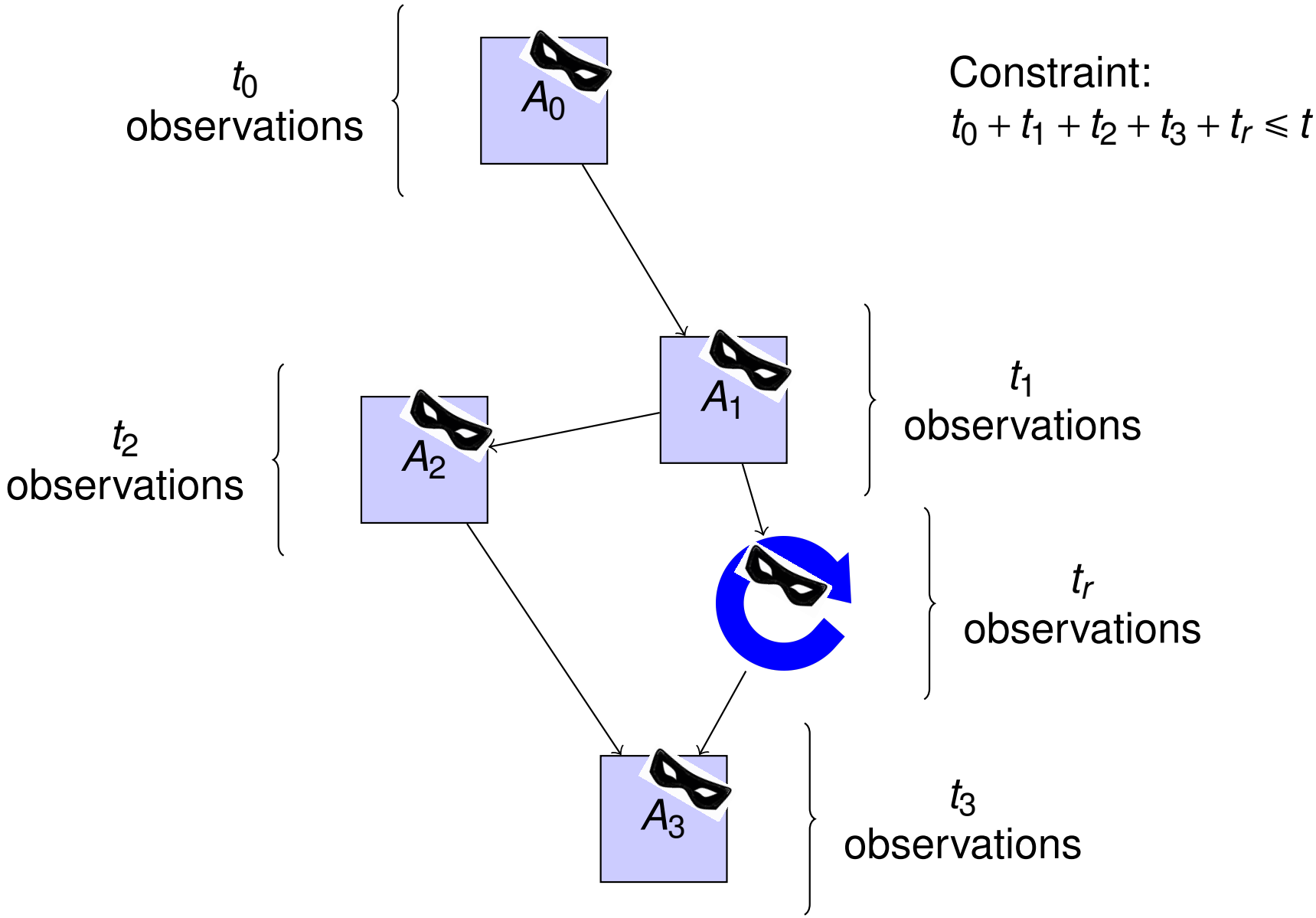
if t is not fixed: show that any set of t intermediate variables with

- t_1 on internal variables
- $t_2 = t - t_1$ on the outputs

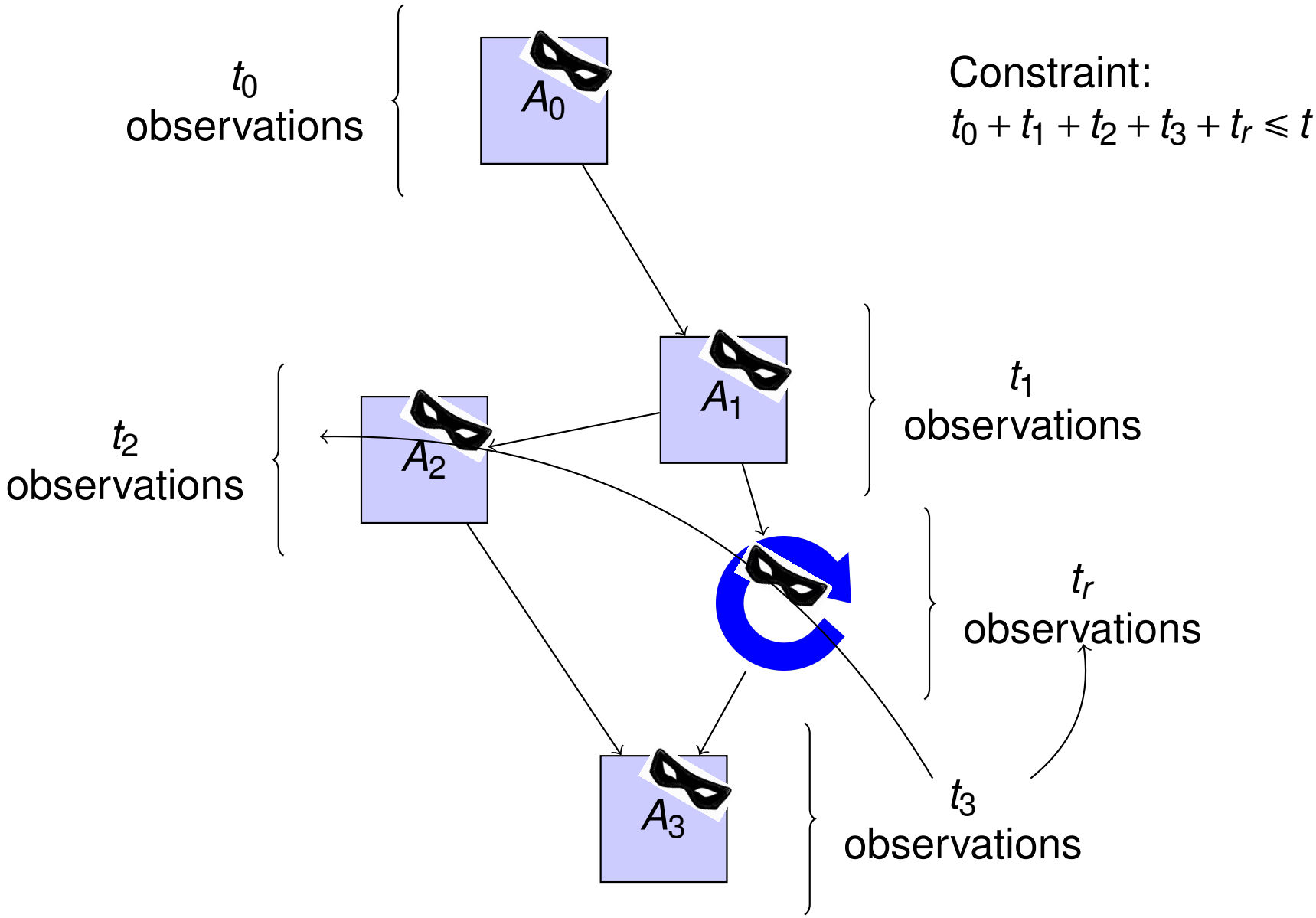
can be simulated with at most t_1 shares of each input



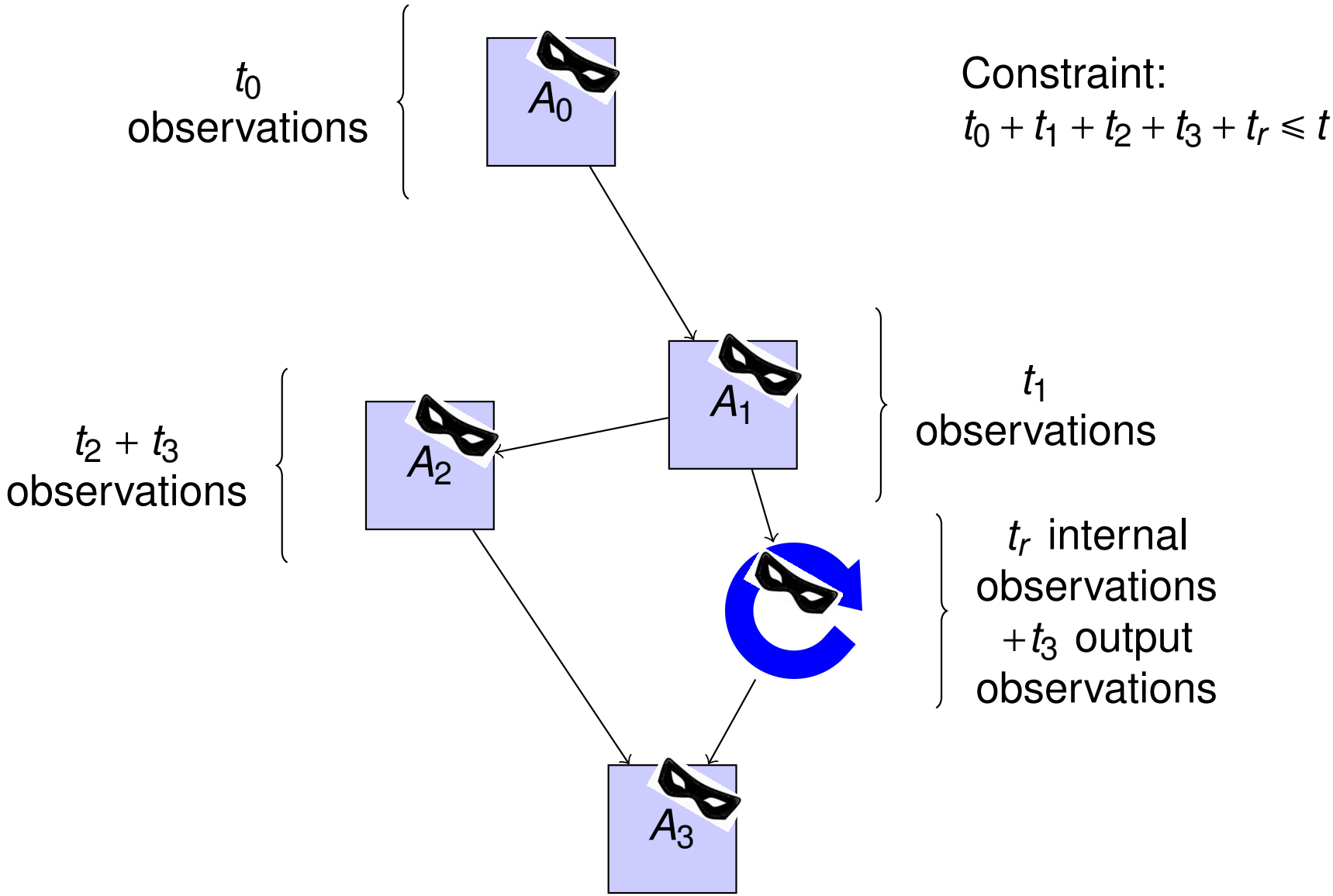
Secure Composition



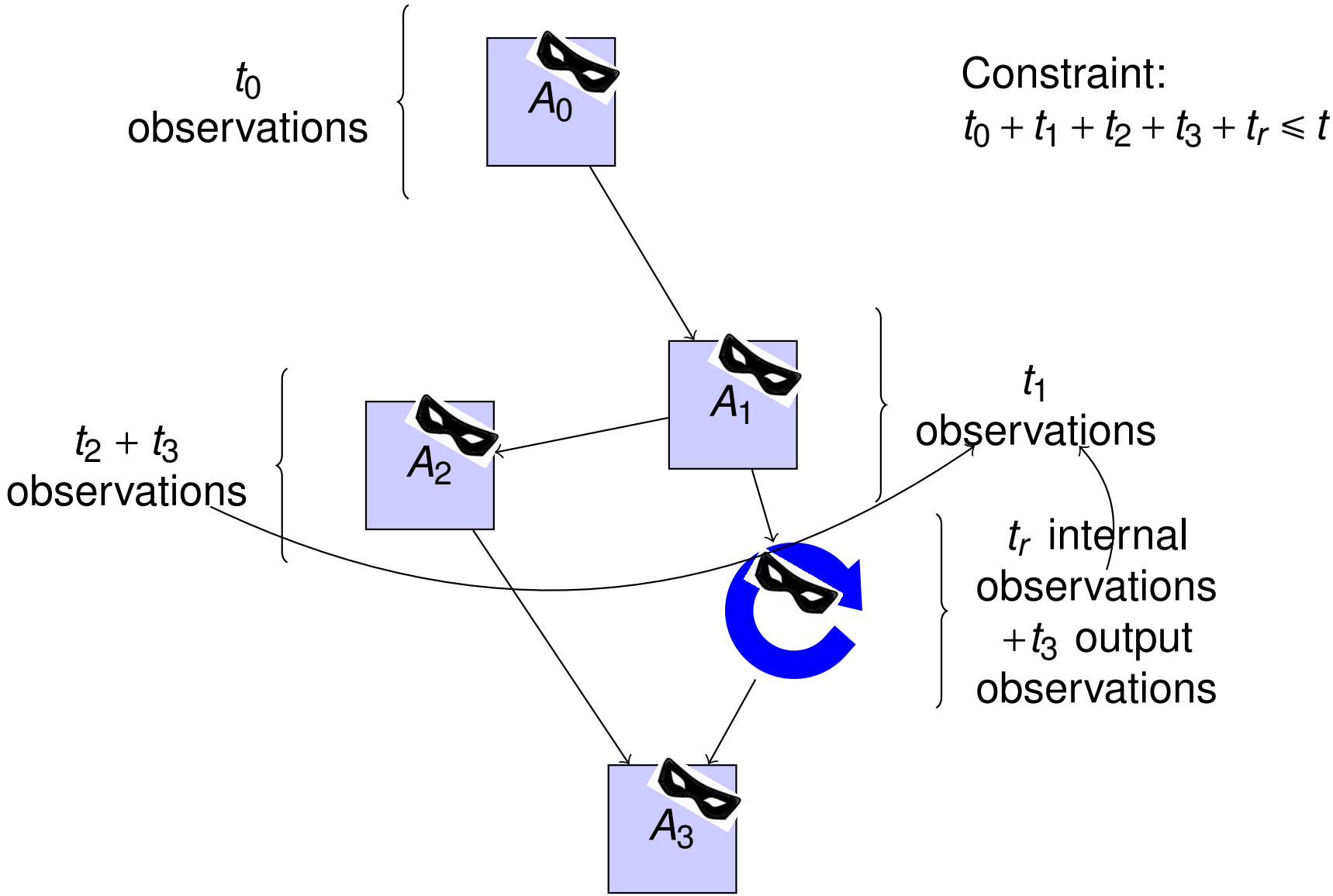
Secure Composition



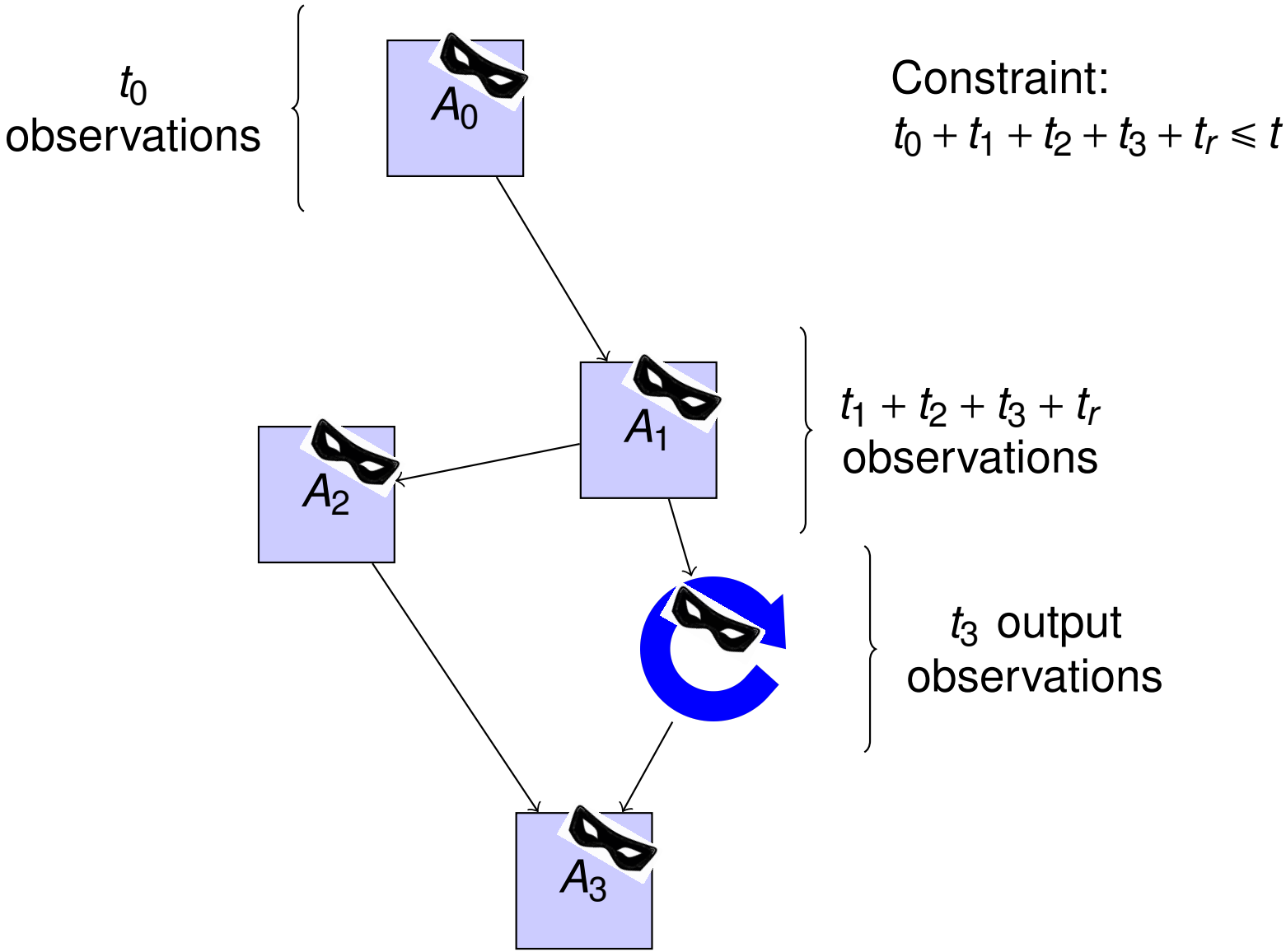
Secure Composition



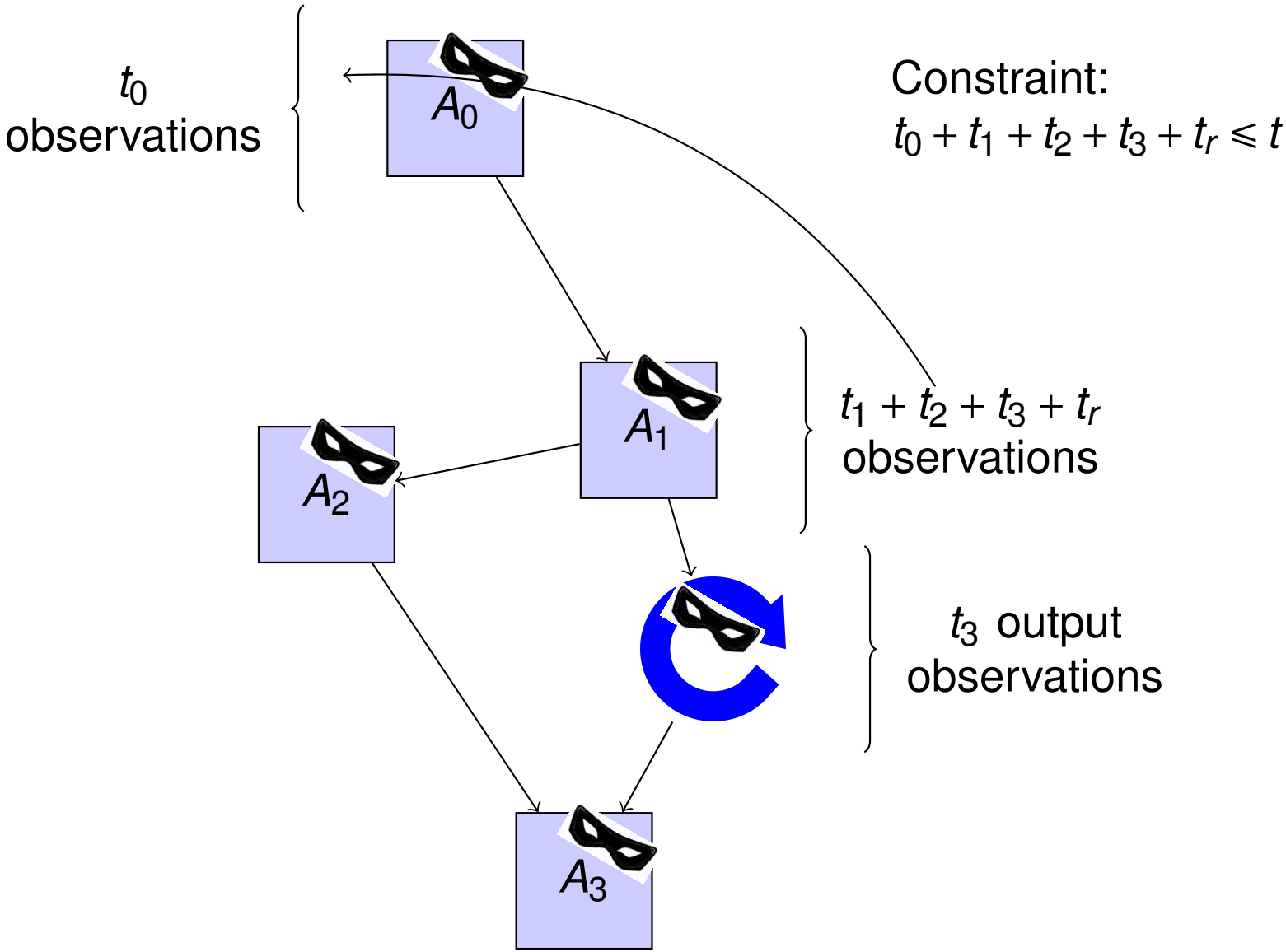
Secure Composition



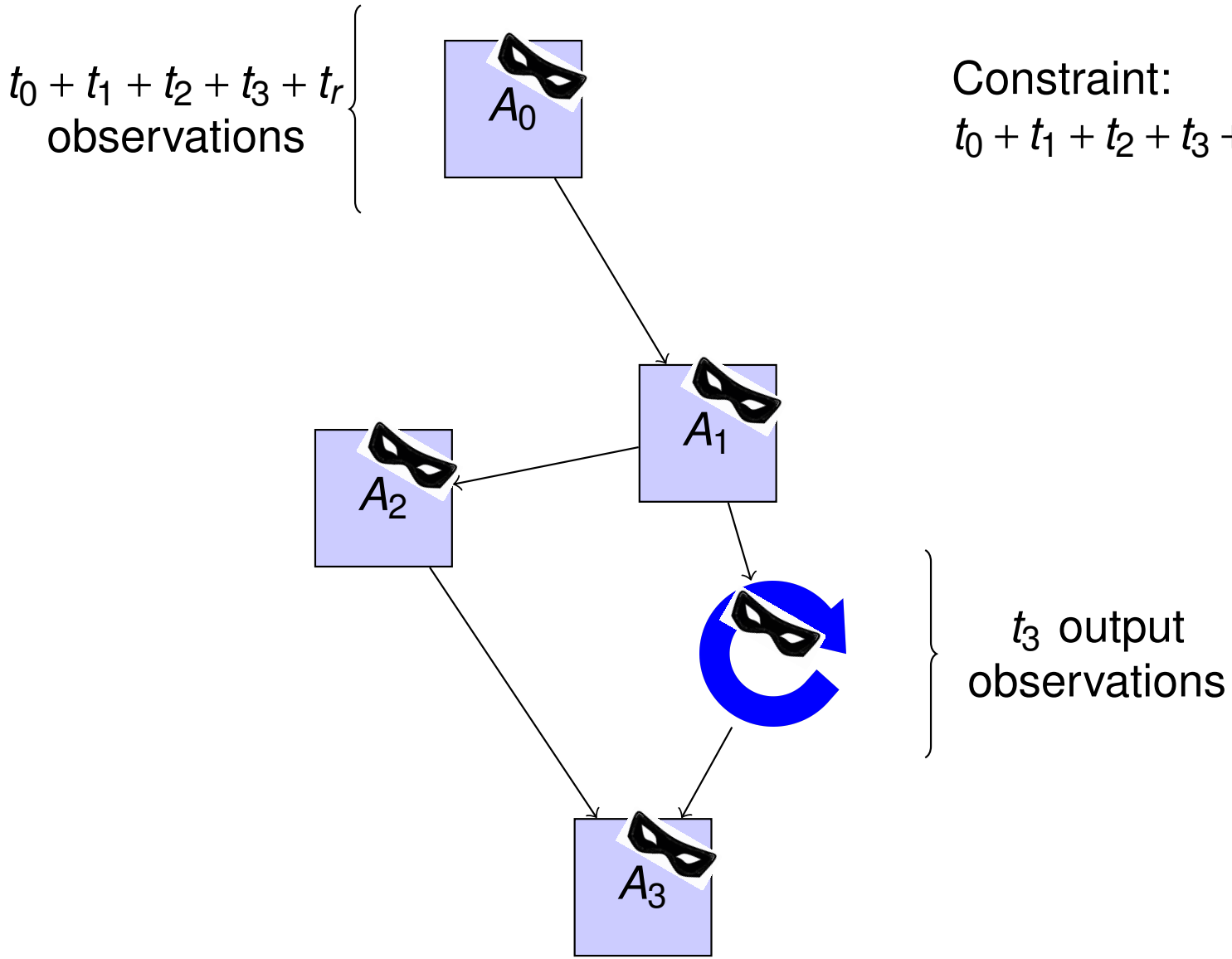
Secure Composition



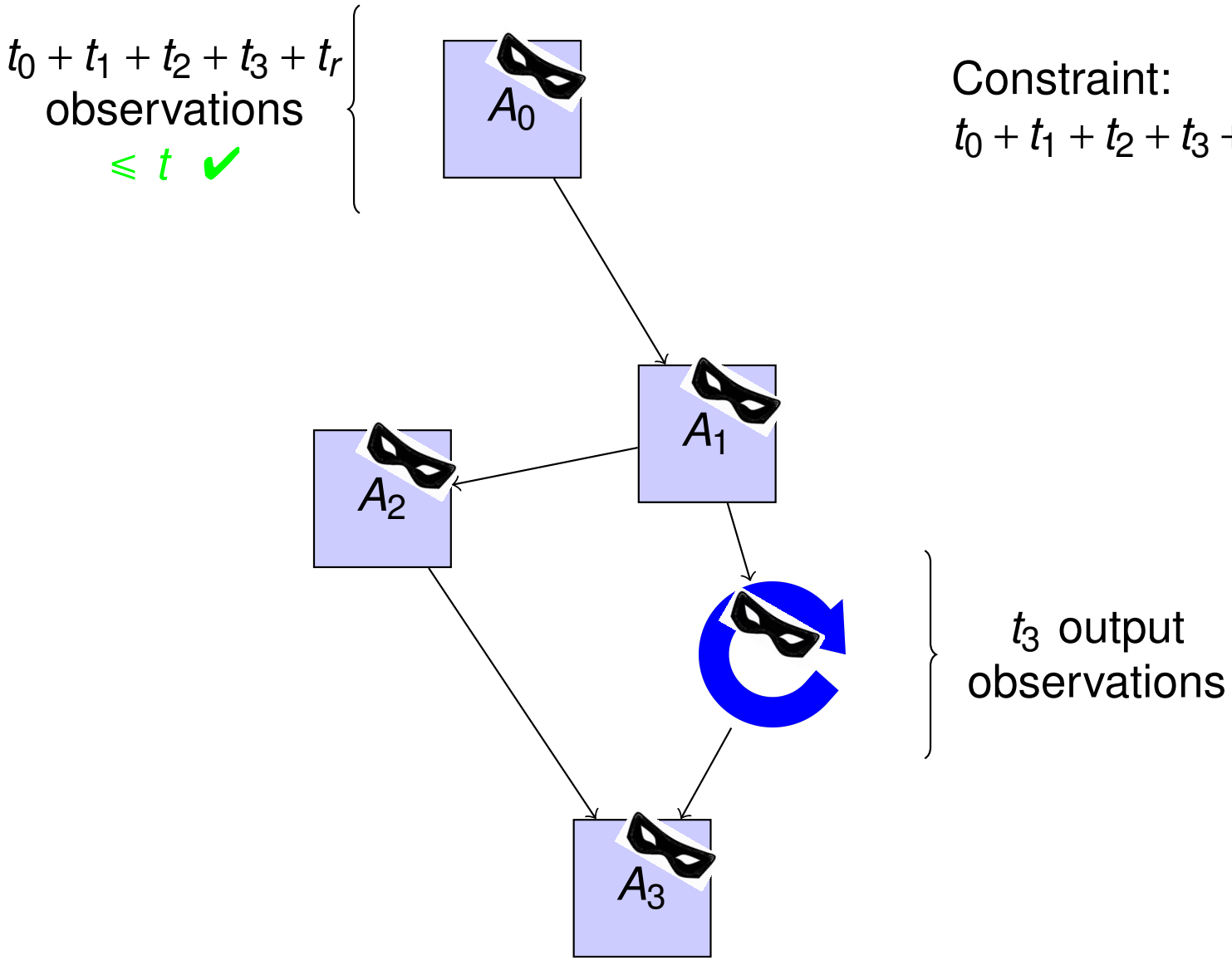
Secure Composition



Secure Composition



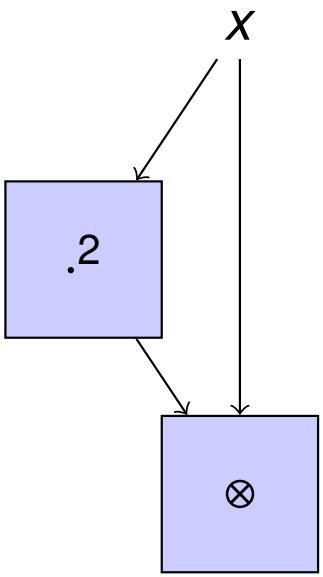
Secure Composition



Secure Composition

Automatic tool for C-based algorithms

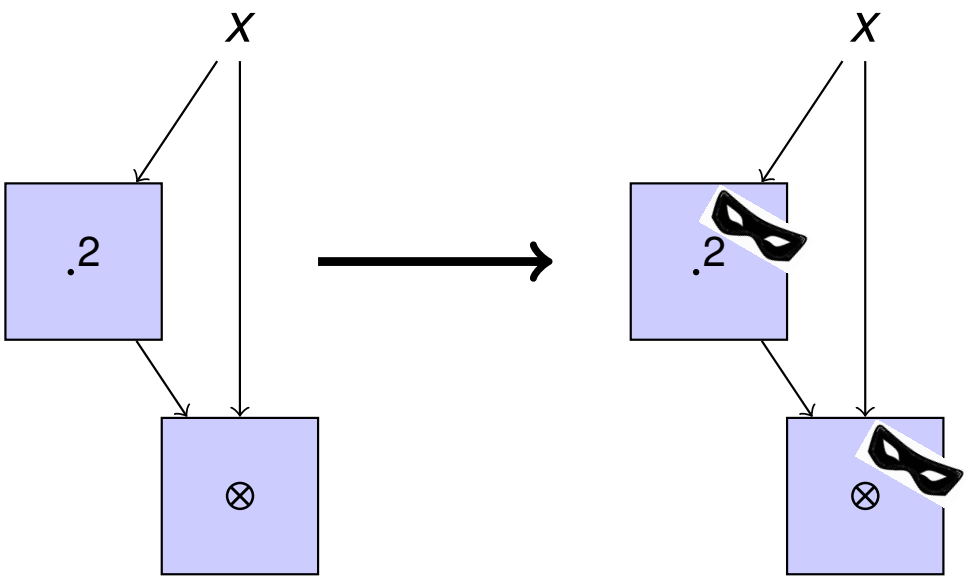
- unprotected algorithm ➜ higher-order masked algorithm
- example for AES S-box



Secure Composition

Automatic tool for C-based algorithms

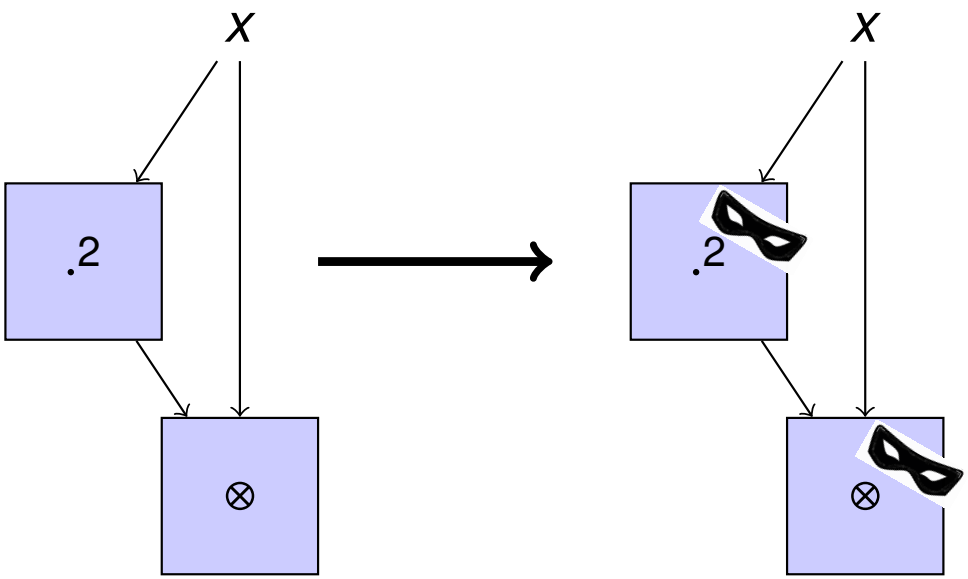
- unprotected algorithm ➜ higher-order masked algorithm
- example for AES S-box



Secure Composition

Automatic tool for C-based algorithms

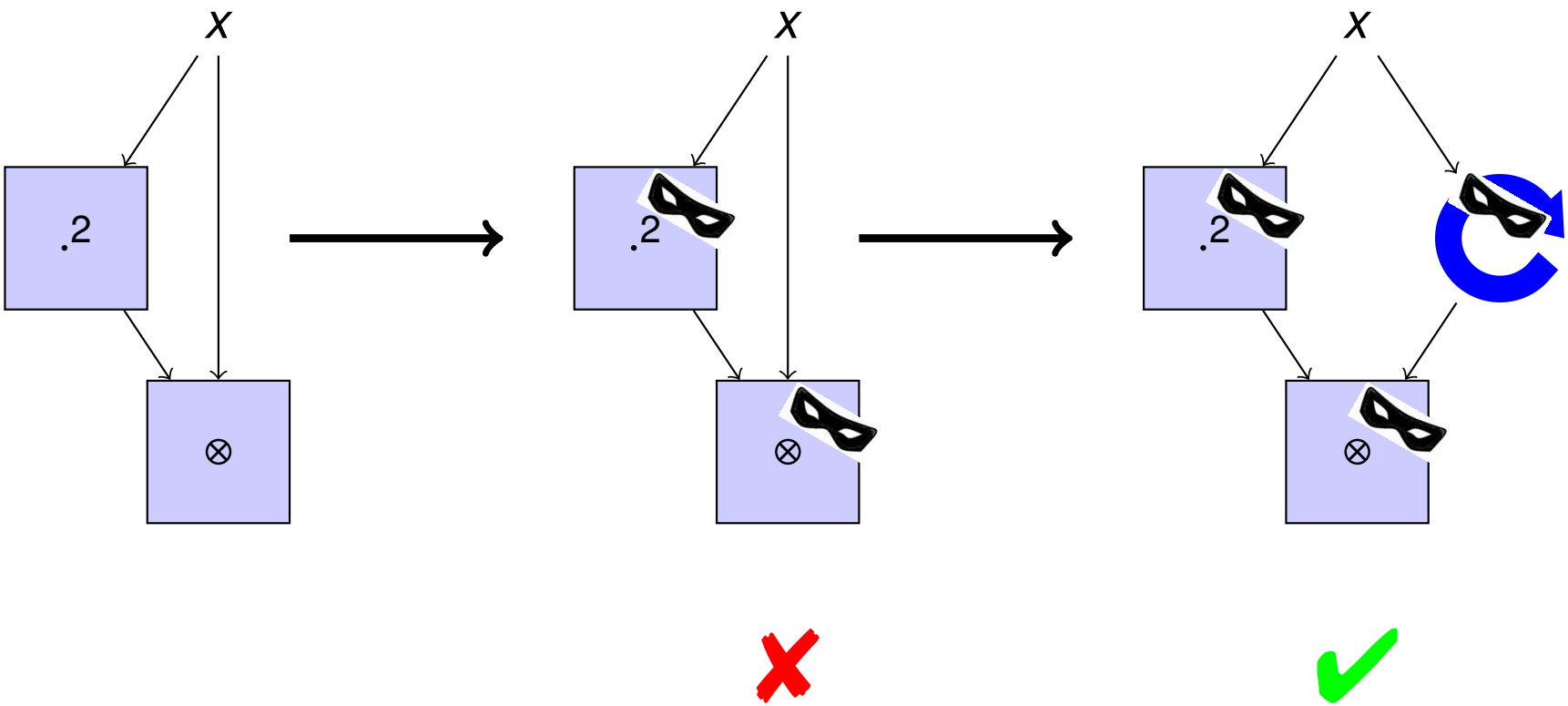
- unprotected algorithm ➜ higher-order masked algorithm
- example for AES S-box



Secure Composition

Automatic tool for C-based algorithms

- unprotected algorithm ➜ higher-order masked algorithm
- example for AES S-box



Some Results

Resource usage statistics for generating masked algorithms (at any order) from some unmasked implementations[1]

Scheme	# Refresh	Time	Memory
AES ($\odot$)	2	0.09s	4Mo
AES ($x \odot g(x)$)	0	0.05s	4Mo
Keccak with Refresh	0	121.20	456Mo
Keccak	600	2728.00s	22870Mo
Simon	67	0.38s	15Mo
Speck	61	6.22s	38Mo

[1] On a Intel(R) Xeon(R) CPU E5-2667 0 @ 2.90GHz with 64Go of memory running Linux (Fedora)

Some Results

Resource usage statistics for generating masked algorithms (at any order) from some unmasked implementations[1]

Scheme	# Refresh	Time	Memory
AES ($\odot$)	2	0.09s	4Mo
AES ($x \odot g(x)$)	0	0.05s	4Mo
Keccak with Refresh	0	121.20s	456Mo
Keccak	600	2728.00s	22870Mo
Simon	67	0.38s	15Mo
Speck	61	6.22s	38Mo

[1] On a Intel(R) Xeon(R) CPU E5-2667 0 @ 2.90GHz with 64Go of memory running Linux (Fedora)

Conclusion on Higher-Order Masking

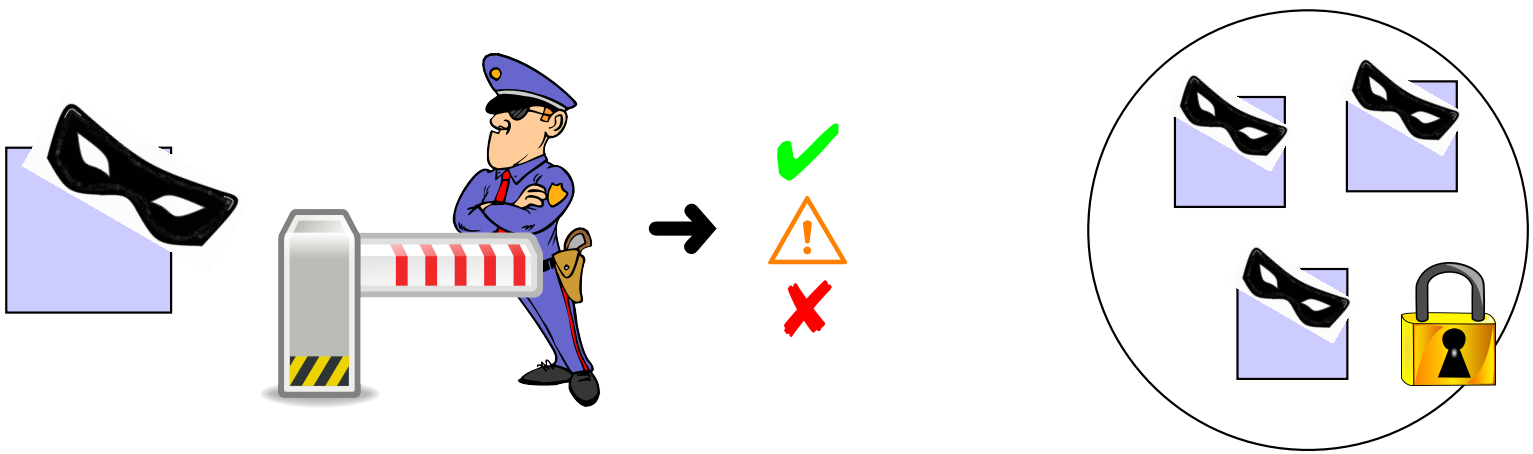
Summary

- ✔ verification of higher-order masking schemes
- ✔ efficient and proven composition
- ✔ two automatic tools

Further Work

- ➜ extend the verification to higher orders using composition
- ➜ integrate transition/glitch-based model
- ➜ build practical experiments for both attacks and new countermeasures

Cryptography: countermeasures against Power-Analysis Attacks



Conclusion

Cryptanalysis: Power-Analysis Attacks

- ➜ investigate the LPN algorithms in the context of power-analysis attacks
- ➜ analyze the operation modes

Cryptography: countermeasures against Power-Analysis Attacks

- ➜ implement and evaluate our countermeasures on real devices (software and hardware)
- ➜ make verifications and compositions as practical as possible
- ➜ use the characterization of a device as a leakage model

Publications


Sonia Belaïd, Luk Bettale, Emmanuelle Dottax, Laurie Genelle, and Franck Rondepierre.
Differential Power Analysis of HMAC SHA-2 in the Hamming weight model. SECRYPT 2013.


Michel Abdalla, Sonia Belaïd, and Pierre-Alain Fouque.
Leakage-resilient symmetric encryption via re-keying. CHES 2013.


Sonia Belaïd, Pierre-Alain Fouque, and Benoît Gérard.
Side-channel analysis of multiplications in GF(2128) - application to AES-GCM. ASIACRYPT 2014.


Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Pierre-Yves Strub.
Verified proofs of higher-order masking. EUROCRYPT 2015.


Sonia Belaïd, Jean-Sébastien Coron, Benoît Gérard, Pierre-Alain Fouque, Jean-Gabriel Kammerer, and Emmanuel Prouff.
Improved Side-Channel Analysis of Finite-Field Multiplication. CHES 2015.


Michel Abdalla, Sonia Belaïd, David Pointcheval, Sylvain Ruhault, and Damien Vergnaud.
Robust pseudo-random number generators with input secure against side-channel attacks. ACNS 2015.


Sonia Belaïd, Fabrizio De Santis, Johann Heyszl, Stefan Mangard, Marcel Medwed, Jörn-Marc Schmidt, François-Xavier Standaert, and Stefan Tillich.
Towards fresh re-keying with leakage-resilient PRFs: cipher design principles and analysis. Journal of Cryptographic Engineering 2014.


Sonia Belaïd, Vincent Grosso, and François-Xavier Standaert.
Masking and leakage-resilient primitives: One, the other(s) or both? Cryptography and Communications 2015.


Sonia Belaïd, Luk Bettale, Emmanuelle Dottax, Laurie Genelle, and Franck Rondepierre.
Differential Power Analysis of HMAC SHA-1 and HMAC SHA-2 in the Hamming weight model. E-Business and Telecommunications - International Joint Conference, ICETE 2014, Revised Selected Papers, Springer, 2015.