On the Use of Masking to Defeat Power-Analysis Attacks

ENS Paris Crypto Day

February 16, 2016

Presented by Sonia Belaïd
Outline

Power-Analysis Attacks

Masking Countermeasure

  Leakage Models

  Security in the probing model

Construction of Secure Masking Schemes - Composition
→ Black-box cryptanalysis
→ Side-channel analysis
Black-box cryptanalysis: \( \mathcal{A} \leftarrow (m_i, c_i) \)

Side-Channel Analysis
→ Black-box cryptanalysis

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

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

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

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

Side-Channel Analysis: $\mathcal{A} \leftarrow (m_i, c_i, \mathcal{L}_i)$
A Power-Analysis Attack against AES-128

Figure: Consumption trace of a full AES-128 from the DPA Contest v2
A Power-Analysis Attack against AES-128

Figure: Consumption trace of a full AES-128 from the DPA Contest v2
A Power-Analysis Attack against AES-128

128-bit input $m$

\[ k_0 \xrightarrow{\oplus} S-box \xrightarrow{\oplus} 8\text{-bit } v \xrightarrow{\oplus} f(v) + \epsilon \]

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
A Power-Analysis Attack against AES-128

128-bit input $m$

8 bits

$K_0 \oplus$

S-box

8-bit $v$

$f(v) + \epsilon$

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
A Power-Analysis Attack against AES-128

128-bit input $m$

8 bits

$\oplus$

$S$-box

8-bit $v$

$f(v) + \epsilon$

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
Algorithmic Countermeasures

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

Two main algorithmic solutions:

- **Fresh Re-keying**: regularly change $k$
- **Masking**: make leakage $\mathcal{L}$ random
Fresh Re-keying

Idea: regularly change $k$

master key $k$

$r$

$R$

session key $k^*$

$m$

$c$
Idea: make leakage $L$ random

sensitive value: $v = f(m, k)$

$v_0 \leftarrow v \oplus \bigoplus_{1 \leq i \leq t} v_i$

$v_1 \leftarrow \$

$\ldots$

$v_t \leftarrow \$

$\implies$ each $t$-uple of $(v_i)_i$ is independent from $v$
Outline

Power-Analysis Attacks

Masking Countermeasure
  Leakage Models
  Security in the probing model
  Construction of Secure Masking Schemes - Composition
Current Research on Masking

Masking

- [EC:DDF14] Unifying Leakage Models: From Probing Attacks to Noisy Leakage
- [C:ISW03] Private Circuits: Securing Hardware against Probing Attacks
- [CHES:RP10] Provably Secure Higher-Order Masking of AES
- [EC:BBDFGS15] formal proofs of masking schemes
- [ePrint:BBDFG15] generation of formally proven masking schemes at any order
- [EC:BBPPTV16] improvement of the randomness complexity for some multiplications
Current Research on Masking

Masking

Security

Efficiency

[C:ISW03] Private Circuits: Securing Hardware against Probing Attacks
[CHES:RP10] Provably Secure Higher-Order Masking of AES
[EC:DDF14] Unifying Leakage Models: From Probing Attacks to Noisy Leakage
[EC:BPPTV16] Improvement of the Randomness Complexity for Some Multiplications
Current Research on Masking

- **Masking**
  - Security
    - Realism: leakage models
  - Proofs: formal proofs of security
  - Efficiency
Current Research on Masking


[EC:DDF14] Unifying Leakage Models: From Probing Attacks to Noisy Leakage


...
Current Research on Masking

Masking

Security

Efficiency

Realism
leakage models

- [EC:DDF14] Unifying Leakage Models: From Probing Attacks to Noisy Leakage

Proofs
formal proofs of security

- [C:ISW03] Private Circuits: Securing Hardware against Probing Attacks
- [CHES:RP10] Provably Secure Higher-Order Masking of AES
- [EC:BPPTV16] Improvement of the randomness complexity for some multiplications
Current Research on Masking

Masking

Security

Realism
leakage models


[EC:DDF14] Unifying Leakage Models: From Probing Attacks to Noisy Leakage


...
Current Research on Masking

Masking

Security

Realism
leakage models

[EC:DDF14] Unifying Leakage Models: From Probing Attacks to Noisy Leakage
...

Proofs
formal proofs of security

[C:ISW03] Private Circuits: Securing Hardware against Probing Attacks
[CHES:RP10] Provably Secure Higher-Order Masking of AES
[EC:BBDFGS15] formal proofs of masking schemes
[ePrint:BBDFG15] generation of formally proven masking schemes at any order

Efficiency
Current Research on Masking

Masking

Security

Realism
- leakage models
  - [EC:PR13]: Masking against Side-Channel Attacks: A Formal Security Proof
  - [EC:DDF14]: Unifying Leakage Models: From Probing Attacks to Noisy Leakage
  - [EC:DFS15]: Making Masking Security Proofs Concrete - Or How to Evaluate the Security of Any Leaking Device

Proofs
- formal proofs of security
  - [C:ISW03]: Private Circuits: Securing Hardware against Probing Attacks
  - [CHES:RP10]: Provably Secure Higher-Order Masking of AES
  - [FSE:CPRR13]: Higher-Order Side Channel Security and Mask Refreshing
  - [EC:BBPPTV16]: Improvement of the randomness complexity for some multiplications

Efficiency

...
Outline

Power-Analysis Attacks

Masking Countermeasure

Leakage Models

Security in the probing model

Construction of Secure Masking Schemes - Composition
Power-Analysis Attacks on Masking Schemes

First-order masking

\[ \mathcal{G}(\mathcal{L}(v + m), \mathcal{L}(m)) \] to the predictions on \( v \)
Power-Analysis Attacks on Masking Schemes

$3^{rd}$-order masking

$\mathcal{C}(\mathcal{L}(v + m_1), \mathcal{L}(m_2), \mathcal{L}(m_3), \mathcal{L}(m_1 + m_2 + m_3))$ to the predictions on $v$
Security of Masked Programs: Leakage Model

- t-probing model
  - Ishai, Sahai, Wagner
  - Crypto 03

- no leak-free gates
- leak-free gates

- noisy leakage model
  - Prouff, Rivain
  - Eurocrypt 13
Security of Masked Programs: Leakage Model

- **t-probing model**
  - Ishai, Sahai, Wagner
  - Crypto 03

- **reduction**
  - Duc, Dziembowski, Faust
  - Eurocrypt 14

- **noisy leakage model**
  - Prouff, Rivain
  - Eurocrypt 13

- ```no leak-free gates```

- ```leak-free gates```
Outline

Power-Analysis Attacks

Masking Countermeasure

- Leakage Models
- Security in the probing model
- Construction of Secure Masking Schemes - Composition
Security in the \( t \)-probing model

\( t \)-probing model assumptions:

- only one variable is leaking at a time
- the attacker can get the exact value of at most \( t \) variables

\( \rightarrow \) show that all the \( t \)-uples are independent from the secret
Security in the $t$-probing model

$v$: randomly generated variable
$c$: known constant
$x$: secret variable

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

\( v \): randomly generated variable

\( c \): known constant

\( x \): secret variable

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

\[
\begin{align*}
(* & \; x_1, x_2, x_3 = \$ \; *) \\
(* & \; x_4 = x + x_1 + x_2 + x_3 \; *) \\
\end{align*}
\]

\[
\begin{align*}
\text{r}_1 & \leftarrow \$ \\
\text{r}_2 & \leftarrow \$ \\
y_1 & \leftarrow x_1 + \text{r}_1 \\
y_2 & \leftarrow (x + x_1 + x_2 + x_3) + \text{r}_2 \\
t_1 & \leftarrow x_2 + \text{r}_1 \\
t_2 & \leftarrow (x_2 + \text{r}_1) + x_3 \\
y_3 & \leftarrow (x_2 + \text{r}_1 + x_3) + \text{r}_2 \\
y_4 & \leftarrow c + \text{r}_2 \\
\end{align*}
\]

1. independent from the secret?
Security in the $t$-probing model

$v$: randomly generated variable
$c$: known constant
$x$: secret variable

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

(* $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

- $v$: randomly generated variable
- $c$: known constant
- $x$: secret variable

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

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

$r_1 \leftarrow \_\_\_$
$r_2 \leftarrow \_\_\_$

1. independent from the secret?

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

Return $(y_1, y_2, y_3, y_4)$
Security in the $t$-probing model

$v$: randomly generated variable
$c$: known constant
$x$: secret variable

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

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

$r_1 \leftarrow \$
$r_2 \leftarrow \$

1. independent from the secret?

$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$

$x$ many mistakes

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

$v$: randomly generated variable
$c$: known constant
$x$: secret variable

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
Security in the $t$-probing model

Contributions:

1. new algorithm to decide whether a $t$-uple is independent from the secret
   - no false positive
   - more efficient than existing works
2. new algorithm to enumerate all the $t$-uples
   - more efficient than existing works

Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Pierre-Yves Strub.

Verified proofs of higher-order masking. EUROCRYPT 2015.
1. Show that a $t$-uple is independent from the secret

Inputs: $t$ intermediate variables, $b \leftarrow$ true

(Rule 1) secret variables?
yes $\rightarrow$ (Rule 2)
no $\rightarrow$ ✓

(Rule 2) an expression $v$ is invertible in the only occurrence of a random $r$?
yes $\rightarrow$ $v \leftarrow r$; (Rule 1)
no $\rightarrow$ (Rule 3)

(Rule 3) is flag $b = true$?
yes $\rightarrow$ simplify; $b \leftarrow$ false; (Rule 1)
no $\rightarrow$ ✗

✓ $\rightarrow$ distribution independent from the secret
✗ $\rightarrow$ might be used for an attack

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($y_1, y_2, y_3, y_4$)
1. Show that a $t$-uple is independent from the secret inputs

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

(Rule 1) secret variables?
- yes $\rightarrow$ (Rule 2)
- no $\rightarrow$ ✓

(Rule 2) an expression $v$ is invertible in the only occurrence of a random $r$?
- yes $\rightarrow$ $v \leftarrow r$; (Rule 1)
- no $\rightarrow$ (Rule 3)

(Rule 3) is flag $b = \text{true}$?
- yes $\rightarrow$ simplify; $b \leftarrow \text{false}$; (Rule 1)
- no $\rightarrow$ ✗

✓ $\rightarrow$ distribution independent from the secret
✗ $\rightarrow$ 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 $\rightarrow$ (Rule 2)
- no $\rightarrow$ ✓

(Rule 2) an expression $v$ is invertible in the only occurrence of a random $r$?
- yes $\rightarrow$ $v \leftarrow r$; (Rule 1)
- no $\rightarrow$ (Rule 3)

(Rule 3) is flag $b = \text{true}$?
- yes $\rightarrow$ simplify; $b \leftarrow \text{false}$; (Rule 1)
- no $\rightarrow$ x

✓ $\rightarrow$ distribution independent from the secret
x $\rightarrow$ might be used for an attack
1. Show that a \( t \)-uple is independent from the secret inputs.

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

(Rule 1) secret variables?
yes  \( \rightarrow \) (Rule 2)
no  \( \rightarrow \) ✓

(Rule 2) an expression \( v \) is invertible in the only occurrence of a random \( r \)?
yes  \( \rightarrow \) \( v \leftarrow r \); (Rule 1)
no  \( \rightarrow \) (Rule 3)

(Rule 3) is flag \( b = \text{true} \)?
yes  \( \rightarrow \) simplify; \( b \leftarrow \text{false} \); (Rule 1)
no  \( \rightarrow \) ✗

✓  \( \rightarrow \) distribution independent from the secret
✗  \( \rightarrow \) might be used for an attack

function \( \text{Ex-t3}(x_1, x_2, x_3, x_4, c) \):
\[
\begin{align*}
r_1 & \leftarrow x_1 + r_1 \\
r_2 & \leftarrow x_3 \\
y_1 & \leftarrow x_1 + r_1 \\
y_2 & \leftarrow x_3 \\
t_1 & \leftarrow x_2 + r_1 \\
t_2 & \leftarrow (x_2 + r_1 + x_3) \\
y_3 & \leftarrow (x_2 + r_1 + x_3) + r_2 \\
y_4 & \leftarrow c + r_2 \\
\end{align*}
\]
return \((y_1, y_2, y_3, y_4)\)
2. Extension to All Possible Sets

**Problem:** \( n \) intermediate variables \( \rightarrow \binom{n}{t} \) proofs
2. Extension to All Possible Sets

**Problem:** $n$ intermediate variables $\Rightarrow \binom{n}{t}$ 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 $\rightarrow \binom{n}{t}$ 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:

Algorithm 1:
2. Extension to All Possible Sets

Problem: $n$ intermediate variables $\rightarrow \binom{n}{t}$ 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. Extension to All Possible Sets

Problem: \( n \) intermediate variables \( \rightarrow \binom{n}{t} \) 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 \( \hat{X} \) with more observations but still independence
2. Extension to All Possible Sets

Problem: $n$ intermediate variables $\Rightarrow \binom{n}{t}$ 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 $\hat{X}$ with more observations but still independence
3. recursively descend in set $C(\hat{X})$
2. Extension to All Possible Sets

**Problem:** $n$ intermediate variables $\Rightarrow \binom{n}{t}$ 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 $\mathcal{C} (\hat{X})$
4. merge $\hat{X}$ and $\mathcal{C} (\hat{X})$ 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

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

\[
\begin{align*}
  r_1 &\leftarrow \$ \\
  r_2 &\leftarrow \$
\end{align*}
\]

\[
\begin{align*}
  y_1 &\leftarrow x_1 + r_1 \\
  y_2 &\leftarrow (x + x_1 + x_2 + x_3) + r_2 \\
  t_1 &\leftarrow x_2 + r_1 \\
  t_2 &\leftarrow (x_2 + r_1) + x_3 \\
  y_3 &\leftarrow (x_2 + r_1 + x_3) + r_2 \\
  y_4 &\leftarrow c + r_2
\end{align*}
\]

return \((y_1, y_2, y_3, y_4)\)
2. Extension to All Possible Sets: Example

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

\[\begin{align*}
  r_1 & \leftarrow \$ \\
r_2 & \leftarrow \$
\end{align*}\]

\[\begin{align*}
y_1 & \leftarrow x_1 + r_1 \\
y_2 & \leftarrow (x + x_1 + x_2 + x_3) + r_2 \\
t_1 & \leftarrow x_2 + r_1 \\
t_2 & \leftarrow (x_2 + r_1) + x_3 \\
y_3 & \leftarrow (x_2 + r_1 + x_3) + r_2 \\
y_4 & \leftarrow c + r_2
\end{align*}\]

return\((y_1, y_2, y_3, y_4)\)
2. Extension to All Possible Sets: Example

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

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

\[
\begin{align*}
  r_1 & \leftarrow \$ \\
  r_2 & \leftarrow \$
\end{align*}
\]

\(\hat{X}: \checkmark\)

\[
\begin{align*}
  y_1 & \leftarrow x_1 + r_1 \\
  y_2 & \leftarrow (x + x_1 + x_2 + x_3) + r_2 \\
  t_1 & \leftarrow x_2 + r_1 \\
  t_2 & \leftarrow (x_2 + r_1) + x_3 \\
  y_3 & \leftarrow (x_2 + r_1 + x_3) + r_2 \\
  y_4 & \leftarrow c + r_2
\end{align*}
\]

\(C(\hat{X}): \checkmark\)

merge \(\hat{X}\) and \(C(\hat{X})\): \xmark

return \((y_1, y_2, y_3, y_4)\)
2. Extension to All Possible Sets: Example

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

$\begin{array}{l}
r_1 \leftarrow$
$r_2 \leftarrow$
$y_1 \leftarrow x_1 + r_1$
$y_2 \leftarrow (x + x_1 + x_2 + x_3) + r_2$
$t_1 \leftarrow x_2 + r_1$
$t_2 \leftarrow (x_2 + r_1) + x_3$
$y_3 \leftarrow (x_2 + r_1 + x_3) + r_2$
$y_4 \leftarrow c + r_2$
\end{array}$

return $(y_1, y_2, y_3, y_4)$

$\Rightarrow$ 207 proofs instead of 286
# Application to the Sbox [CPRR13, Algorithm 4]

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

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

<table>
<thead>
<tr>
<th>Reference</th>
<th>Target</th>
<th># tuples</th>
<th>Security</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td># sets</td>
</tr>
<tr>
<td><strong>First-Order Masking</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSE13</td>
<td>full AES</td>
<td>17,206</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>MAC-SHA3</td>
<td>full Keccak-f</td>
<td>13,466</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td><strong>Second-Order Masking</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSA06</td>
<td>Sbox</td>
<td>1,188,111</td>
<td>✓</td>
<td>1st-order flaws (2)</td>
</tr>
<tr>
<td>CHES10</td>
<td>Sbox</td>
<td>7,140</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>CHES10</td>
<td>AES KS</td>
<td>23,041,866</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>FSE13</td>
<td>2 rnds AES</td>
<td>25,429,146</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>FSE13</td>
<td>4 rnds AES</td>
<td>109,571,806</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td><strong>Third-Order Masking</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSA06</td>
<td>Sbox</td>
<td>2,057,067,320</td>
<td>✓</td>
<td>3rd-order flaws (98,176)</td>
</tr>
<tr>
<td>FSE13</td>
<td>Sbox(4)</td>
<td>4,499,950</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>FSE13</td>
<td>Sbox(5)</td>
<td>4,499,950</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td><strong>Fourth-Order Masking</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FSE13</td>
<td>Sbox (4)</td>
<td>2,277,036,685</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td><strong>Fifth-Order Masking</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHES10</td>
<td>⊙</td>
<td>216,071,394</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
Outline

Power-Analysis Attacks

Masking Countermeasure

Leakage Models

Security in the probing model

Construction of Secure Masking Schemes - 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 \((x_i)_{i \geq 0}\) and \((x'_i)_{i \geq 0}\) are mutually independent.
Current Issues in Composition

A refresh algorithm takes as input a sharing $x_i \geq 0$ and returns a new sharing $x'_i \geq 0$ of $x$ such that $x_i \geq 1$ and $x'_i \geq 1$ are mutually independent.
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.
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.
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.
Composition in the $t$-probing model

Contributions:

1. new algorithm to verify the security of compositions
   - formal security
   - any order
2. compiler to build a higher-order secure scheme from any C implementation
   - efficient
   - any order

Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, and Benjamin Grégoire.
Compositional Verification of Higher-Order Masking Application to a Verifying Masking Compiler. ePrint 2015.
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

\[
\begin{align*}
\text{a}_0 & \quad \text{a}_1 \\
\text{a}_2 & \quad \text{a}_3 \\
\end{align*}
\]

\[
\begin{align*}
(= \text{a} + \text{a}_0 + \text{a}_1 + \text{a}_2)
\end{align*}
\]

\[
\begin{align*}
\text{c}_0 & \quad \text{c}_1 \\
\text{c}_2 & \quad \text{c}_3 \\
\end{align*}
\]

3 observations
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 $(c_0,...,c_i,...,c_t)$

$\rightarrow$ 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

\[
\begin{align*}
\text{function } \text{Linear-function-}t(a_0,...,a_i,...,a_t): \\
\text{for } i &= 0 \text{ to } t \\
\quad c_i &= f(a_i) \\
\text{return } (c_0,...,c_i,...,c_t)
\end{align*}
\]

→ 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

```plaintext
a_0 a_1 a_2 a_3
\Rightarrow (a + a_0 + a_1 + a_2)

\Rightarrow \text{observations}

\Rightarrow \text{straightforward for linear functions}

\Rightarrow \text{formal proofs with EasyCrypt and pen-and-paper proofs for small non-linear functions}
```

function Linear-function-t($a_0,...,a_i,...,a_t$):
for $i = 0$ to $t$
\[ c_i = f(a_i) \]
return $(c_0,...,c_i,...,c_t)$
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 \leq t \]
Current Issues

\[
\begin{align*}
&\text{Constraint: } t_0 + t_1 + t_2 + t_3 \leq t \\
&t_0 \text{ observations} \\
&t_1 \text{ observations} \\
&t_2 \text{ observations} \\
&t_3 \text{ observations}
\end{align*}
\]
Current Issues

Constraint:

\[ t_0 + t_1 + t_2 + t_3 \leq t \]
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 \leq t \]
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 \leq t \]

\[ t_0 \text{ observations} \]

\[ t_1 + t_3 + t_2 + t_3 \text{ observations} \]
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 \leq t \]

\[ t_1 + t_2 + 2t_3 \leq t \]
Current Issues

$t_0$ observations

\[
\begin{align*}
A_0 \\
A_1 \\
A_2 \\
A_3
\end{align*}
\]

Constraint:
\[t_0 + t_1 + t_2 + t_3 \leq t\]

\[t_1 + t_2 + 2t_3 \leq t?\] observations
Current Issues

Constraint:

\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

- \( t_0 \) observations
- \( t_2 \) observations
- \( t_1 \) observations
- \( t_3 \) observations
- \( t_r \) observations
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

\[
\begin{align*}
A_0 & \quad t_0 \text{ observations} \\
A_1 & \quad t_1 \text{ observations} \\
A_2 & \quad t_2 + t_3 \text{ observations} \\
A_3 & \quad t_r + t_3 \text{ observations}
\end{align*}
\]
Current Issues

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

- \( t_0 \) observations
- \( t_2 + t_3 \) observations
- \( t_1 \) observations
- \( t_r + t_3 \) observations
Current Issues

Constraint: $t_0 + t_1 + t_2 + t_3 + t_r \leq t$

$t_0$ observations

$A_0$

$A_1$

$t_1 + t_2 + 2t_3 + t_r \leq t$?

observations

$t_1$ observations

$A_2$

$t_2$ observations

$A_3$
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

\[
\begin{align*}
&\quad a_0 \quad a_1 \quad a_2 \quad a_3 \\
&\quad c_0 \quad c_1 \quad c_2 \quad c_3
\end{align*}
\]

2 internal observations

+ 1 output observation
Secure Composition

$t_0$ observations

$t_2$ observations

$t_1$ observations

$t_r$ observations

$t_3$ output observations

Constraint:
$t_0 + t_1 + t_2 + t_3 + t_r \leq t$
Secure Composition

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]
Secure Composition

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

- \( t_0 \) observations
- \( A_0 \)
- \( t_2 + t_3 \) observations
- \( A_2 \)
- \( t_1 \) observations
- \( A_1 \)
- \( t_r \) internal observations
- \( t_3 \) output observations
- \( A_3 \)
Secure Composition

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]
Secure Composition

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

- \( t_0 \) observations
- \( A_0 \)
- \( t_1 + t_2 + t_3 + t_r \) observations
- \( A_1 \)
- \( t_3 \) output observations
- \( A_2 \)
- \( A_3 \)
Secure Composition

$t_0$ observations

Constraint:
$t_0 + t_1 + t_2 + t_3 + t_r \leq t$

$t_1 + t_2 + t_3 + t_r$ observations

$t_3$ output observations
Secure Composition

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

Observations:
- \( A_0 \) with \( t_0 + t_1 + t_2 + t_3 + t_r \) observations
- \( A_1 \)
- \( A_2 \)
- \( A_3 \)

Output Observations:
- \( t_3 \) output observations
Secure Composition

\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]

Constraint:
\[ t_0 + t_1 + t_2 + t_3 + t_r \leq t \]
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

\[ x \cdot 2 \oplus x \cdot 2 \]
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\(^1\)

<table>
<thead>
<tr>
<th>Scheme</th>
<th># Refresh</th>
<th>Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES (⊙)</td>
<td>2/Sbox</td>
<td>0.09s</td>
<td>4Mo</td>
</tr>
<tr>
<td>AES (x⊙g(x))</td>
<td>0</td>
<td>0.05s</td>
<td>4Mo</td>
</tr>
<tr>
<td>Keccak with Refresh</td>
<td>0</td>
<td>121.20</td>
<td>456Mo</td>
</tr>
<tr>
<td>Keccak</td>
<td>600</td>
<td>2728.00s</td>
<td>22870Mo</td>
</tr>
<tr>
<td>Simon</td>
<td>67</td>
<td>0.38s</td>
<td>15Mo</td>
</tr>
<tr>
<td>Speck</td>
<td>61</td>
<td>6.22s</td>
<td>38Mo</td>
</tr>
</tbody>
</table>

\(^1\)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\(^1\)

<table>
<thead>
<tr>
<th>Scheme</th>
<th># Refresh</th>
<th>Time</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES ((\odot))</td>
<td>2/Sbox</td>
<td>0.09s</td>
<td>4Mo</td>
</tr>
<tr>
<td>AES ((x \odot g(x)))</td>
<td>0</td>
<td>0.05s</td>
<td>4Mo</td>
</tr>
<tr>
<td>Keccak with Refresh</td>
<td>0</td>
<td>121.20s</td>
<td>456Mo</td>
</tr>
<tr>
<td>Keccak</td>
<td>600</td>
<td>2728.00s</td>
<td>22870Mo</td>
</tr>
<tr>
<td>Simon</td>
<td>67</td>
<td>0.38s</td>
<td>15Mo</td>
</tr>
<tr>
<td>Speck</td>
<td>61</td>
<td>6.22s</td>
<td>38Mo</td>
</tr>
</tbody>
</table>

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

Masking

Security

Realism
models close enough to the reality

Proofs
formal proofs of security

[EC15] formal proofs of masking schemes

[ePrint15] generation of formally proven masking schemes at any order

Efficiency

[EC16] improvement of the randomness complexity for some multiplications

§ extend the verification to higher orders using composition

§ integrate transition/glitch-based model

§ build practical experiments for both attacks and new countermeasures

§ still reduce the randomness in multiplications
Conclusion

Masking

Security

Realism
models close enough to the reality

Proofs
formal proofs of security

[EC15] formal proofs of masking schemes

[ePrint15] generation of formally proven masking schemes at any order

→ extend the verification to higher orders using composition

Efficiency

[EC16] improvement of the randomness complexity for some multiplications

§ extend the verification to higher orders using composition

§ integrate transition/glitch-based model

§ build practical experiments for both attacks and new countermeasures

§ still reduce the randomness in multiplications
Conclusion

Masking

- Security
  - Realism: models close enough to the reality
  - Proofs: formal proofs of security
    - [EC15]: formal proofs of masking schemes
    - [ePrint15]: generation of formally proven masking schemes at any order
      - extend the verification to higher orders using composition
      - integrate transition/glitch-based model

- Efficiency
  - [EC16]: improvement of the randomness complexity for some multiplications
Conclusion

Masking

Security

- Realism
  - models close enough to the reality

- Proofs
  - formal proofs of security
    - [EC15] formal proofs of masking schemes
    - [ePrint15] generation of formally proven masking schemes at any order
      - extend the verification to higher orders using composition
      - integrate transition/glitch-based model
      - build practical experiments for both attacks and new countermeasures

Efficiency

- [EC16] improvement of the randomness complexity for some multiplications
Conclusion

Masking

Security

Realism
models close enough
to the reality

Proofs
formal proofs of security

[EC15] formal proofs of masking schemes

[ePrint15] generation of formally proven masking schemes at any order

→ extend the verification to higher orders using composition

→ integrate transition/glitch-based model

→ build practical experiments for both attacks and new countermeasures

Efficiency

[EC16] improvement of the randomness complexity for some multiplications

→ still reduce the randomness in multiplications