Provably-secure masking for side-channel security

Gaëtan Cassiers

COSADE 2024

fnrs
LA LIBERTÉ DE CHERCHER
UCLouvain

Outline

Masking

Masking circuits: the composition challenge

Composable masking

Masking in hardware

Masking

Masking

Replace **every intermediate variable** x in a computation by a sharing

$$(x_1, \ldots, x_n)$$

such that

$$x = x_1 \star \cdots \star x_n$$

in a group $(\mathbb{G}, \star)$. Most often: $\mathbb{F}_{2^k}$ or $\mathbb{Z}_k$.

Masking

Replace **every intermediate variable** x in a computation by a sharing

$$(x_1, \ldots, x_n)$$

such that

$$x = x_1 \star \cdots \star x_n$$

in a group $(\mathbb{G}, \star)$. Most often: $\mathbb{F}_{2^k}$ or $\mathbb{Z}_k$.

Replace **every computation** by operations on the shares:

$$\text{Elementary operation} \rightarrow \text{Gadget}$$

Why does it work?

Statistical security:

Assume $x \in \mathbb{F}_2$, $L = l(x_1) + \cdots + l(x_n)$.

$(n-1)$-th statistical order security:

$$n \geq 2 \Rightarrow \mathbb{E}(L|x=0) = \mathbb{E}(L|x=1)$$
$$n \geq 3 \Rightarrow \mathbb{V}(L|x=0) = \mathbb{V}(L|x=1)$$
$$\cdots$$

Share-recombination security:

For any $S \subsetneq \{1, \ldots, n\}$:

$$(x_i)_{i \in S} \perp\!\!\!\perp x$$

Recombine noisy information on n shares (distribution convolution): $\exp(n)$.

Masking verification

Why not measurement-based evaluation (e.g., TVLA)?

Masking verification

Why not measurement-based evaluation (e.g., TVLA)?

- Practical relevance
- Realistic
- Covers whole execution

$\Rightarrow$ Testing and validation

Masking verification

Why not measurement-based evaluation (e.g., TVLA)?

- Practical relevance
- Realistic
- Covers whole execution

⇒ Testing and validation

- Needs device
- Depends on circuit synthesis & layout
- Depends on evaluator's setup

⇒ Design and pre-silicon validation

Threshold probing model

t-probing security:

Any tuple of t intermediate values in the computation is independent of the secrets.

Requires $t \leq n-1$, generalizes single sharing security. (In this talk: $t = n-1$, for efficiency.)

Often needed and often good enough.

Masking circuits: the composition challenge

Masked addition gadget G_+

Compute

$$z = x + y$$

Inputs:

$$(x_1, \ldots, x_n)$$

$$(y_1, \ldots, y_n)$$

Outputs:

$$(z_1, \ldots, z_n)$$

Such that

$$(z_1 + \cdots + z_n) = (x_1 + \cdots + x_n) + (y_1 + \cdots + y_n)$$

Masked addition gadget G_+

Compute

$$z = x + y$$

Inputs:

$$(x_1, \ldots, x_n)$$

$$(y_1, \ldots, y_n)$$

Outputs:

$$(z_1, \ldots, z_n)$$

Such that

$$(z_1 + \cdots + z_n) = (x_1 + \cdots + x_n) + (y_1 + \cdots + y_n)$$

G_+ algorithm:

$$z_1 = x_1 + y_1$$

$$\vdots$$

$$z_n = x_n + y_n$$

Generalization: $(d-1)$-probing secure sharewise gadget for any affine operation.

Masked multiplication: ISW gadget $G_\times$

Compute

$$z = x \times y$$

Inputs:

$$(x_1, \ldots, x_n)$$

$$(y_1, \ldots, y_n)$$

Outputs:

$$(z_1, \ldots, z_n)$$

Such that

$$(z_1 + \cdots + z_n) = (x_1 + \cdots + x_n) \times (y_1 + \cdots + y_n)$$

Masked multiplication: ISW gadget $G_\times$

Compute

$$z = x \times y$$

Inputs:

$$(x_1, \ldots, x_n)$$

$$(y_1, \ldots, y_n)$$

Outputs:

$$(z_1, \ldots, z_n)$$

Such that

$$(z_1 + \cdots + z_n) = (x_1 + \cdots + x_n) \times (y_1 + \cdots + y_n)$$

Proposal:

$$\begin{aligned} z_1 &= (x_1 \times y_1) + (x_1 \times y_2) + (x_1 \times y_3) \\ z_2 &= (x_2 \times y_1) + (x_2 \times y_2) + (x_2 \times y_3) \\ z_3 &= (x_3 \times y_1) + (x_3 \times y_2) + (x_3 \times y_3) \end{aligned}$$

$$x_1 \times y_1 + x_1 \times y_2 + x_1 \times y_3 = x_1 \times y$$

Masked multiplication: ISW gadget $G_\times$

Compute

$$z = x \times y$$

Inputs:

$$(x_1, \ldots, x_n)$$

$$(y_1, \ldots, y_n)$$

Outputs:

$$(z_1, \ldots, z_n)$$

Such that

$$(z_1 + \cdots + z_n) = (x_1 + \cdots + x_n) \times (y_1 + \cdots + y_n)$$

Proposal:

$$\begin{aligned} z_1 &= (x_1 \times y_1) + (x_1 \times y_2) + (x_1 \times y_3) \\ z_2 &= (x_2 \times y_1) + (x_2 \times y_2) + (x_2 \times y_3) \\ z_3 &= (x_3 \times y_1) + (x_3 \times y_2) + (x_3 \times y_3) \end{aligned}$$

$$x_1 \times y_1 + x_1 \times y_2 + x_1 \times y_3 = x_1 \times y$$

Masked multiplication: ISW gadget $G_\times$

Compute

$$z = x \times y$$

Inputs:

$$(x_1, \ldots, x_n)$$

$$(y_1, \ldots, y_n)$$

Outputs:

$$(z_1, \ldots, z_n)$$

Such that

$$(z_1 + \cdots + z_n) = (x_1 + \cdots + x_n) \times (y_1 + \cdots + y_n)$$

ISW multiplication gadget:

$$z_1 = (x_1 \times y_1) + (x_1 \times y_2 + r_1) + (x_1 \times y_3 + r_2)$$
$$z_2 = (x_2 \times y_1 - r_1) + (x_2 \times y_2) + (x_2 \times y_3 + r_3)$$
$$z_3 = (x_3 \times y_1 - r_2) + (x_3 \times y_2 - r_3) + (x_3 \times y_3)$$

$$x_1 \times y_1 + x_1 \times y_2 + x_1 \times y_3 = x_1 \times y$$

Solution: add randomness r_1, r_2, r_3

A masked AES circuit

In first AES round ($\mathbb{F}_{256}$):

$$a = p + k$$
$$z = \mathrm{Sbox}(a)$$

A masked AES circuit

In first AES round ($\mathbb{F}_{256}$):

$$a = p + k \qquad \textit{Addition gadget}$$

$$c = a^2 \qquad \textit{Linear gadget}$$

$$d = b \otimes c \qquad \textit{Multiplication gadget}$$

$$\vdots$$

A masked AES circuit

In first AES round ($\mathbb{F}_{256}$):

$$
\begin{aligned}
a &= p + k && \textit{Addition gadget} \\
c &= a^2 && \textit{Linear gadget} \\
d &= b \otimes c && \textit{Multiplication gadget} \\
&\vdots
\end{aligned}
$$

Masked design:

$$
\begin{aligned}
(a_1, a_2, a_3) &= G_{\oplus}((p_1, p_2, p_3), (k_1, k_2, k_3)) &&= (p_1 \oplus k_1, p_2 \oplus k_2, p_3 \oplus k_3) \\
(c_1, c_2, c_3) &= G_{\cdot^2}((a_1, a_2, a_3)) &&= (a_1^2, a_2^2, a_3^2) \\
(d_1, d_2, d_3) &= G_{\otimes}((a_1, a_2, a_3), (c_1, c_2, c_3)) &&= G_{\otimes}((a_1, a_2, a_3), (a_1^2, a_2^2, a_3^2))
\end{aligned}
$$

Composition problem

$$(d_1, d_2, d_3) = G_{\otimes}((a_1, a_2, a_3), (a_1^2, a_2^2, a_3^2))$$

$$d_1 = (a_1 \otimes a_1^2 \quad) \oplus (a_1 \otimes a_2^2 \oplus r_1) \oplus (a_1 \otimes a_3^2 \oplus r_2)$$

$$d_2 = (a_2 \otimes a_1^2 \oplus r_1) \oplus (a_2 \otimes a_2^2 \quad) \oplus (a_2 \otimes a_3^2 \oplus r_3)$$

$$d_3 = (a_3 \otimes a_1^2 \oplus r_2) \oplus (a_3 \otimes a_2^2 \oplus r_3) \oplus (a_3 \otimes a_3^2 \quad)$$

Composition problem

$$(d_1, d_2, d_3) = G_{\otimes}((a_1, a_2, a_3), (a_1^2, a_2^2, a_3^2))$$

$$d_1 = (a_1 \otimes a_1^2 \quad) \oplus (a_1 \otimes a_2^2 \oplus r_1) \oplus (a_1 \otimes a_3^2 \oplus r_2)$$
$$d_2 = (a_2 \otimes a_1^2 \oplus r_1) \oplus (a_2 \otimes a_2^2 \quad) \oplus (a_2 \otimes a_3^2 \oplus r_3)$$
$$d_3 = (a_3 \otimes a_1^2 \oplus r_2) \oplus (a_3 \otimes a_2^2 \oplus r_3) \oplus (a_3 \otimes a_3^2 \quad)$$

Not 2-probing secure: probing $a_1 \otimes a_2^2$ and a_3 reveals information about a.

Composition problem

$$(d_1, d_2, d_3) = G_{\otimes}((a_1, a_2, a_3), (a_1^2, a_2^2, a_3^2))$$

$$\begin{aligned}
d_1 &= (a_1 \otimes a_1^2 \quad\;\;) \oplus (a_1 \otimes a_2^2 \oplus r_1) \oplus (a_1 \otimes a_3^2 \oplus r_2) \\
d_2 &= (a_2 \otimes a_1^2 \oplus r_1) \oplus (a_2 \otimes a_2^2 \quad\;\;) \oplus (a_2 \otimes a_3^2 \oplus r_3) \\
d_3 &= (a_3 \otimes a_1^2 \oplus r_2) \oplus (a_3 \otimes a_2^2 \oplus r_3) \oplus (a_3 \otimes a_3^2 \quad\;\;)
\end{aligned}$$

Not 2-probing secure: probing $a_1 \otimes a_2^2$ and a_3 reveals information about a.

$a_1 \otimes a_2^2 = 0$ if $a_1 = 0$ or $a_2 = 0$, that is if $a_1 \in \{0, a \oplus a_3\}$.

Since a_1 is uniform,

$$\Pr[a_1 \otimes a_2^2 = 0 \mid a = a_3] = \frac{1}{2} \Pr[a_1 \otimes a_2^2 = 0 \mid a \neq a_3]$$

and therefore

$$\Pr[a = a_3 \mid a_1 \otimes a_2^2 = 0] \approx \frac{1}{2} \frac{1}{|\mathbb{F}_{256}|}$$

Refreshing to make sharings independent

Fix proposal: Input sharings must be independent, we refresh one of the input sharings.

Refreshing to make sharings independent

Fix proposal: Input sharings must be independent, we refresh one of the input sharings.

Simple refresh gadget:

for $i = 1, \ldots, n-1$ **do**

$r_i \xleftarrow{\$} \mathbb{G}$

$a_i \leftarrow a_i + r_i$

$a_n \leftarrow a_n - r_i$

Refreshing to make sharings independent

Fix proposal: Input sharings must be independent, we refresh one of the input sharings.

Simple refresh gadget:

for $i = 1, \ldots, n-1$ **do**

$r_i \xleftarrow{\$} \mathbb{G}$

$a_i \leftarrow a_i + r_i$

$a_n \leftarrow a_n - r_i$

Back to "AES":

- multiply
  - (a_1, a_2, a_3), and
  - $(b_1, b_2, b_3) = (a_1 \oplus r_1, a_2 \oplus r_2, a_3 \oplus r_1 \oplus r_2)$
- Probes:
  - $(a_1 \oplus r_1) \times a_2^2$
  - $a_3 \oplus r_1 \oplus r_2$

$a_3 + r_1 + r_2$ is independent of a_3.

Refreshing to make sharings independent

Fix proposal: Input sharings must be independent, we refresh one of the input sharings.

Simple refresh gadget:

for $i = 1, \ldots, n-1$ **do**

$r_i \xleftarrow{\$} \mathbb{G}$

$a_i \leftarrow a_i + r_i$

$a_n \leftarrow a_n - r_i$

Back to "AES":

- multiply
  - (a_1, a_2, a_3), and
  - $(b_1, b_2, b_3) = (a_1 \oplus r_1, a_2 \oplus r_2, a_3 \oplus r_1 \oplus r_2)$
- Probes:
  - $(a_1 \oplus r_1) \times a_2^2$
  - $a_3 \oplus r_1$

A new sharing: $(a_1 \oplus r_1, a_2, a_3 \oplus r_1)$.

Composable masking

Simulatability

A set of probes $\mathcal{P}$ in a gadget can be simulated with the input shares $\mathcal{I}$ if there exists an algorithm $\mathcal{S}$ such that $\mathcal{P} \approx \mathcal{S}(\mathcal{I})$.

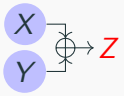
In other words, $\mathcal{P}$ is independent of all inputs except $\mathcal{I}$.

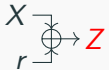
Simulatability

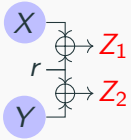
A set of probes $\mathcal{P}$ in a gadget can be simulated with the input shares $\mathcal{I}$ if there exists an algorithm $\mathcal{S}$ such that $\mathcal{P} \approx \mathcal{S}(\mathcal{I})$.

In other words, $\mathcal{P}$ is independent of all inputs except $\mathcal{I}$.

E.g. (randomness \$):





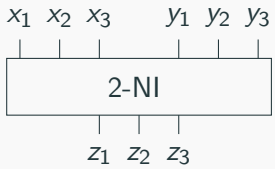


Non-Interference

A gadget G is t-non-interferent (t-NI) if any set of t probes can be simulated with t shares of each input sharing.

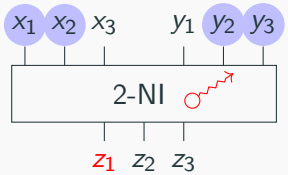
Non-Interference

A gadget G is t-non-interferent (t-NI) if any set of t probes can be simulated with t shares of each input sharing.



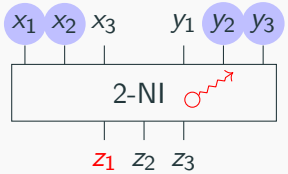
Non-Interference

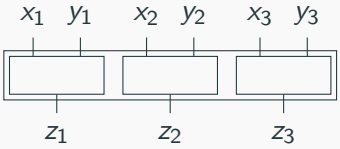
A gadget G is t-non-interferent (t-NI) if any set of t probes can be simulated with t shares of each input sharing.



Non-Interference

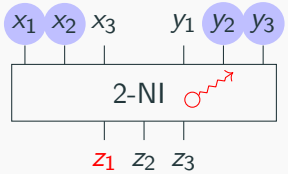
A gadget G is t-non-interferent (t-NI) if any set of t probes can be simulated with t shares of each input sharing.

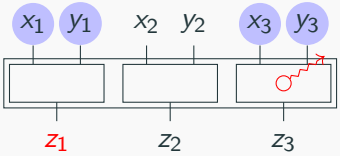




Non-Interference

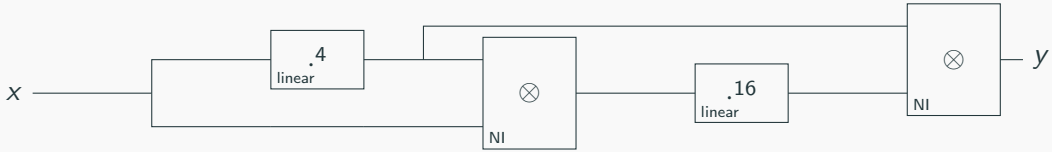
A gadget G is t-non-interferent (t-NI) if any set of t probes can be simulated with t shares of each input sharing.



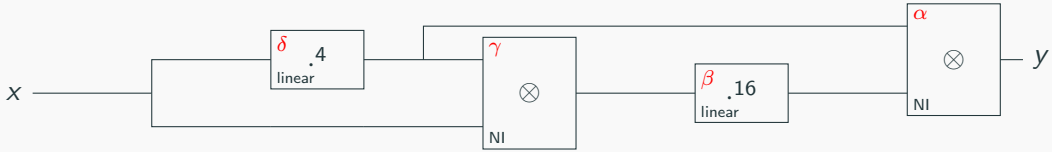


Sharewise gadgets are NI.

Probe propagation

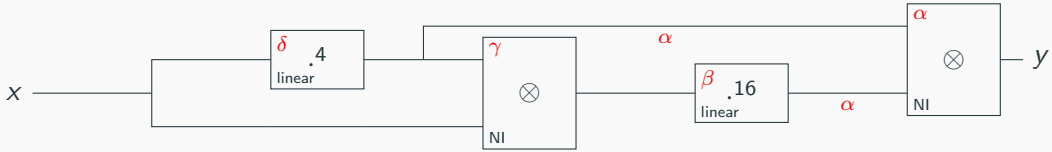


Probe propagation



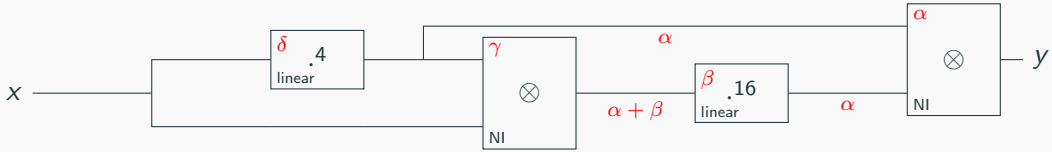
- $\alpha + \beta + \gamma + \delta \leq t$

Probe propagation



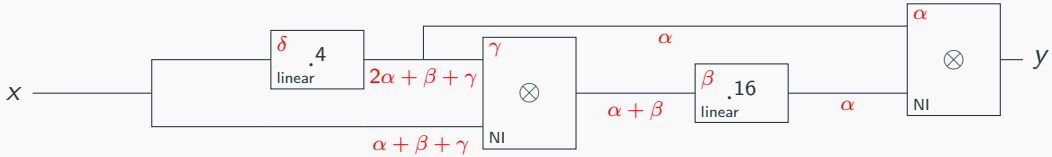
- $\alpha + \beta + \gamma + \delta \leq t$

Probe propagation



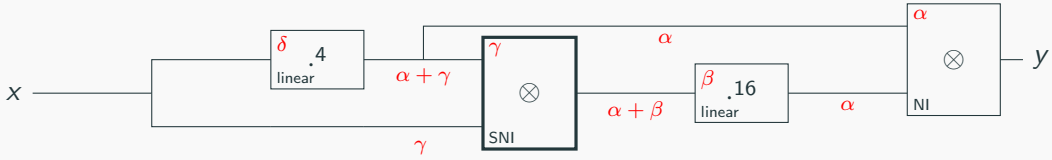
- $\alpha + \beta + \gamma + \delta \leq t$

Probe propagation



- $\alpha + \beta + \gamma + \delta \leq t$
- $2\alpha + \beta + \gamma$ above threshold

Probe propagation



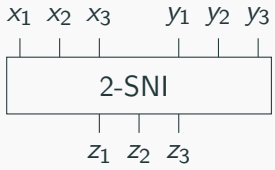
- $\alpha + \beta + \gamma + \delta \leq t$
- $2\alpha + \beta + \gamma$ above threshold
- **Strong** non-interference

Strong non-interference

A gadget G is t-strongly non-interferent (t-SNI) if any set of t_1 internal probes and t_2 output probes can be simulated with t_1 shares of each input sharing ($t_1 + t_2 = t$).

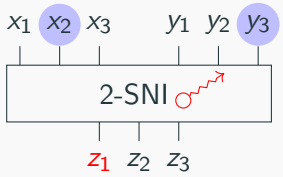
Strong non-interference

A gadget G is t-strongly non-interferent (t-SNI) if any set of t_1 internal probes and t_2 output probes can be simulated with t_1 shares of each input sharing ($t_1 + t_2 = t$).



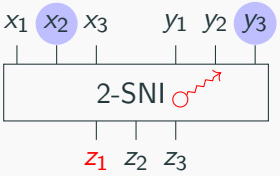
Strong non-interference

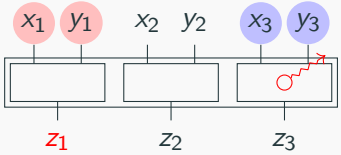
A gadget G is t-strongly non-interferent (t-SNI) if any set of t_1 internal probes and t_2 output probes can be simulated with t_1 shares of each input sharing ($t_1 + t_2 = t$).



Strong non-interference

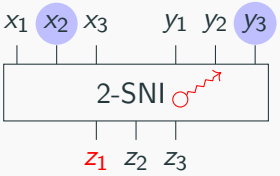
A gadget G is t-strongly non-interferent (t-SNI) if any set of t_1 internal probes and t_2 output probes can be simulated with t_1 shares of each input sharing ($t_1 + t_2 = t$).

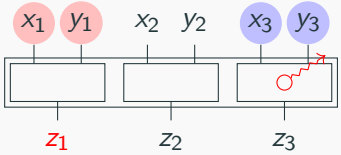




Strong non-interference

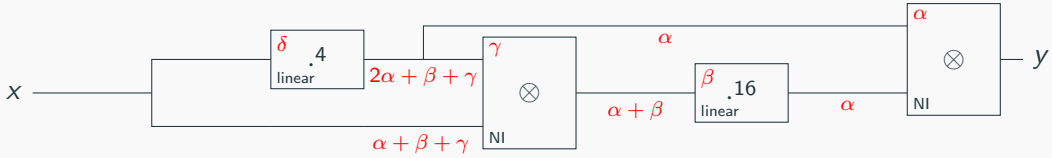
A gadget G is t-strongly non-interferent (t-SNI) if any set of t_1 internal probes and t_2 output probes can be simulated with t_1 shares of each input sharing ($t_1 + t_2 = t$).



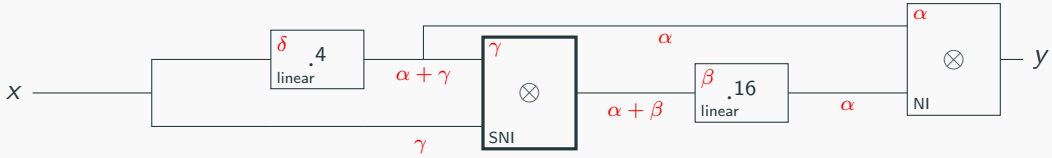


SNI is trivially composable: any composition of t-SNI gadgets is a t-SNI gadget.

Probe propagation with SNI gadgets

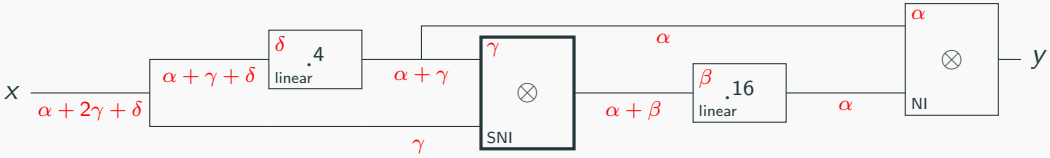


Probe propagation with SNI gadgets



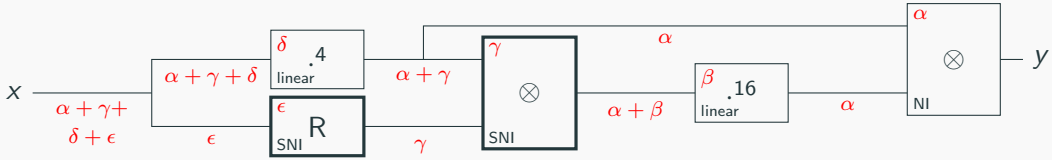
- SNI “stops” probe propagation.

Probe propagation with SNI gadgets



- SNI "stops" probe propagation.

Probe propagation with SNI gadgets



- SNI "stops" probe propagation.
- SNI refreshing for composition with non-SNI gadgets

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$
- $\{z_i, y\}$:

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$
- $\{z_i, y\}$: $\{z_1, y\}$

Simple refresh is not SNI

2 shares simple refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow x_3 - r_1$

$z_3 \leftarrow y - r_2$

Generalization: $\mathcal{O}(n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$
- $\{z_i, y\}$: $\{z_1, y\} = \{x_1 + r_1, x_3 - r_1\}$

SNI refresh

A 2-SNI refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow r_1 + r_2$

$z_3 \leftarrow x_3 - y$

Generalization: $\mathcal{O}(n \log n)$

SNI refresh

A 2-SNI refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow r_1 + r_2$

$z_3 \leftarrow x_3 - y$

Generalization: $\mathcal{O}(n \log n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$
- $\{z_i, y\}$**:**

SNI refresh

A 2-SNI refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow r_1 + r_2$

$z_3 \leftarrow x_3 - y$

Generalization: $\mathcal{O}(n \log n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$
- $\{z_i, y\}$**: depends on at most 1 input**

SNI refresh

A 2-SNI refresh:

$r_1, r_2 \xleftarrow{\$} \mathbb{G}$

$z_1 \leftarrow x_1 + r_1$

$z_2 \leftarrow x_2 + r_2$

$y \leftarrow r_1 + r_2$

$z_3 \leftarrow x_3 - y$

Generalization: $\mathcal{O}(n \log n)$

Possible sets of probes:

- 2 internal probes:
  subset of $\{x_1, x_2, x_3, r_1, r_2, y\}$
  depends on at most 2 inputs
- $\{z_i, z_j\}$: output sharing is uniform
- $\{z_i, r_j\}$: depends on at most 1 input
- $\{x_i, z_j\}$: $z_j \perp\!\!\!\perp x_i$
- $\{z_i, y\}$**: depends on at most 1 input**

Composition theorem: t-NI gadget + t-SNI output refresh $\rightarrow$ t-SNI gadget

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad\quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad\quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad\quad)$$

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad\quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad\quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad\quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. Simulate probes

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad\quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad\quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad\quad)$$

Given a set of probes P:

1. **Select simulation inputs.**
2. Simulate probes

Input selection invariants:

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad\quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad\quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad\quad)$$

Given a set of probes P:

1. **Select simulation inputs.**
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. **Select simulation inputs.**
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad\quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad\quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad\quad)$$

Given a set of probes P:

1. **Select simulation inputs.**
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. **Select simulation inputs.**
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. **Simulate probes**

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. **Simulate probes**

Input selection invariants:

- **For each** x_i **or** y_i **probe:** $\{x_i, y_i\}$**.**
- **For each** $x_i \times y_j$ **or** $x_i \times y_j + r_{i,j}$ **probe:** $\{x_i, y_j\}$**.**
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

- any probe except intermediate sum: trivial

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. **Simulate probes**

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

- any probe except intermediate sum: trivial
- intermediate sum: for each $x_i \times y_j + r_{i,j}$ term:
  - if x_i and y_j are selected: trivial

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. **Simulate probes**

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

- any probe except intermediate sum: trivial
- intermediate sum: for each $x_i \times y_j + r_{i,j}$ term:
  - if x_i and y_j are selected: trivial
  - otherwise: *simulate as fresh random*

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. **Simulate probes**

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

- any probe except intermediate sum: trivial
- intermediate sum: for each $x_i \times y_j + r_{i,j}$ term:
  - if x_i and y_j are selected: trivial
  - otherwise: *simulate as fresh random*

Correctness:

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- **If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.**

Simulation:

- any probe except intermediate sum: trivial
- intermediate sum: for each $x_i \times y_j + r_{i,j}$ term:
  - if x_i and y_j are selected: trivial
  - otherwise: *simulate as fresh random*

Correctness:

- *fresh random* for $x_i \times y_i + r_{i,j}$,

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

- any probe except intermediate sum: trivial
- intermediate sum: for each $x_i \times y_j + r_{i,j}$ term:
  - if x_i and y_j are selected: trivial
  - otherwise: *simulate as fresh random*

Correctness:

- *fresh random* for $x_i \times y_i + r_{i,j}$,
- all others: trivial.

ISW multiplication is NI

Gadget (with $r_{i,j} = -r_{j,i}$):

$$z_1 = (x_1 \times y_1 \quad) + (x_1 \times y_2 + r_{1,2}) + (x_1 \times y_3 + r_{1,3})$$
$$z_2 = (x_2 \times y_1 + r_{2,1}) + (x_2 \times y_2 \quad) + (x_2 \times y_3 + r_{2,3})$$
$$z_3 = (x_3 \times y_1 + r_{3,1}) + (x_3 \times y_2 + r_{3,2}) + (x_3 \times y_3 \quad)$$

Given a set of probes P:

1. Select simulation inputs.
2. Simulate probes

Input selection invariants:

- For each x_i or y_i probe: $\{x_i, y_i\}$.
- For each $x_i \times y_j$ or $x_i \times y_j + r_{i,j}$ probe: $\{x_i, y_j\}$.
- For each probe in the sum for z_i: $\{x_i\}$.
- If any $r_{i,j}/r_{j,i}$ appears in 2 probes, $\{x_i, y_i, x_j, y_j\}$.

Simulation:

- any probe except intermediate sum: trivial
- intermediate sum: for each $x_i \times y_j + r_{i,j}$ term:
  - if x_i and y_j are selected: trivial
  - otherwise: *simulate as fresh random*

Correctness:

- *fresh random* for $x_i \times y_i + r_{i,j}$,
- all others: trivial.

SNI: at most $n - 2$ internal probes, but $n - 1$ randoms in each sum...

Composable masking

Isolation

Share isolation

Linear composition

$$(c_1, c_2, c_3) = (a_1 \oplus b_1, a_2 \oplus b_2, a_3 \oplus b_3)$$
$$(d_1, d_2, d_3) = (c_1^2, c_2^2, c_3^2)$$

Trivially $(d-1)$-probing secure.

Share isolation

Linear composition

$$(c_1, c_2, c_3) = (a_1 \oplus b_1, a_2 \oplus b_2, a_3 \oplus b_3)$$
$$(d_1, d_2, d_3) = (c_1^2, c_2^2, c_3^2)$$

Trivially $(d-1)$-probing secure.

Sharewise implementation

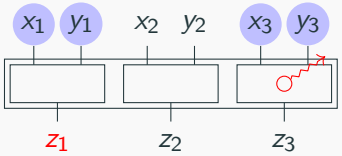
- Implies t-probing security
- Composable
- Only for affine gadgets

Not for non-linear gadgets
$\rightarrow$ Let's simulate it: PINI

PINI: Probe-Isolating Non-Interference

Sharewise gadgets and simulation:

- For each probe, you get the circuit share.



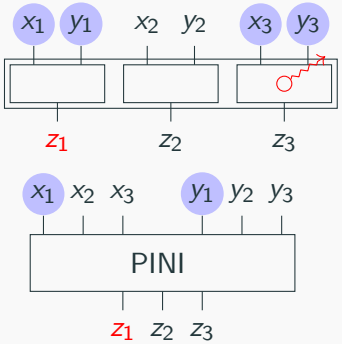
PINI: Probe-Isolating Non-Interference

Sharewise gadgets and simulation:

- For each probe, you get the circuit share.

PINI

- For each **output** probe, you get the circuit share.



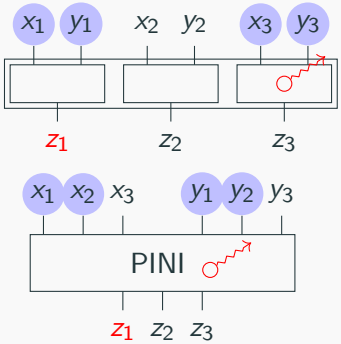
PINI: Probe-Isolating Non-Interference

Sharewise gadgets and simulation:

- For each probe, you get the circuit share.

PINI

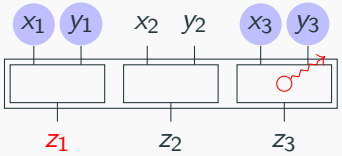
- For each **output** probe, you get the circuit share.
- For each **internal** probe, you **choose** one circuit share.



PINI: Probe-Isolating Non-Interference

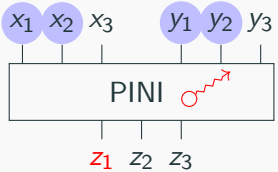
Sharewise gadgets and simulation:

- For each probe, you get the circuit share.



PINI

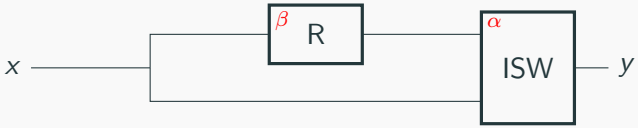
- For each **output** probe, you get the circuit share.
- For each **internal** probe, you **choose** one circuit share.



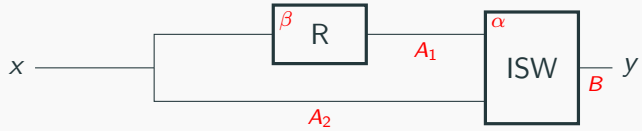
A gadget G is t-probe-isolating non-interferent (t-PINI) if, for any set P of internal probes and any set $B \subset \{1, \ldots, n\}$ such that $|P| + |B| \leq t$, there exists $A \subset \{1, \ldots, n\}$, $|A| = |P|$ such that the probes in P and the output shares with index in B can be simulated with the input shares with index in $A \cup B$.

PINI gadgets are trivially composable.

PINI multiplication with refreshing

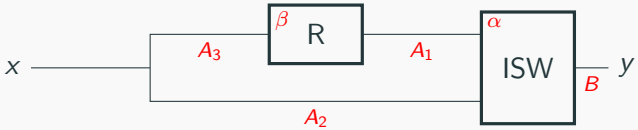


PINI multiplication with refreshing



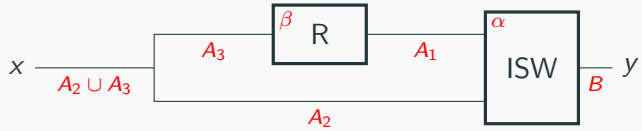
$|A_1| \leq \alpha$, $|A_2| \leq \alpha$

PINI multiplication with refreshing



$|A_1| \leq \alpha$, $|A_2| \leq \alpha$, $|A_3| \leq \beta$

PINI multiplication with refreshing



$$|A_1| \leq \alpha,\ |A_2| \leq \alpha\ ,\ |A_3| \leq \beta \Rightarrow |A_2 \cup A_3| \leq \alpha + \beta = |P|$$

“Native” PINI multiplication

ISW multiplication gadget: not PINI

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

“Native” PINI multiplication

ISW multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

Masked multiplication trick:

$$x \times y + r = (x + 1) \times r + x \times (y - r) = (\cancel{x \times r} + r) + (x \times y - \cancel{x \times r})$$

“Native” PINI multiplication

ISW multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

Masked multiplication trick:

$$x \times y + r = (x + 1) \times r + x \times (y - r)$$

“Native” PINI multiplication

ISW multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

Masked multiplication trick:

$$x \times y + r = (x + 1) \times r + x \times (y - r)$$

“Native” PINI multiplication

ISW multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

Masked multiplication trick:

$$x \times y + r = (x + 1) \times r + x \times (y - r)$$

“Native” PINI multiplication

ISW multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

Masked multiplication trick:

$$x \times y + r = (x + 1) \times r + x \times (y - r)$$

“Native” PINI multiplication

ISW multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 &&) + (x_1 \times y_2 + r_1 &&) + (x_1 \times y_3 + r_2 &&) \\
z_2 &= (x_2 \times y_1 - r_1 &&) + (x_2 \times y_2 &&) + (x_2 \times y_3 + r_3 &&) \\
z_3 &= (x_3 \times y_1 - r_2 &&) + (x_3 \times y_2 - r_3 &&) + (x_3 \times y_3 &&)
\end{aligned}
$$

Masked multiplication trick:

$$x \times y + r = (x + 1) \times r + x \times (y - r)$$

"Native" PINI multiplication

PINI multiplication gadget:

$$
\begin{aligned}
z_1 &= (x_1 \times y_1 \qquad\qquad\qquad\qquad\qquad) + ((x_1 + 1) \otimes r_1 + x_1 \times (y_2 + r_1)) + ((x_1 + 1) \times r_3 + x_1 \times (y_3 - r_2)) \\
z_2 &= ((x_2 - 1) \times r_1 + x_2 \times (y_1 - r_1)) + (x_2 \times y_2 \qquad\qquad\qquad\qquad\qquad) + ((x_2 + 1) \times r_3 + x_2 \times (y_3 - r_3)) \\
z_3 &= ((x_3 - 1) \times r_1 + x_3 \times (y_1 - r_2)) + ((x_3 - 1) \times r_2 + x_3 \times (y_2 - r_3)) + (x_3 \times y_3 \qquad\qquad\qquad\qquad\qquad)
\end{aligned}
$$

Masked multiplication trick:

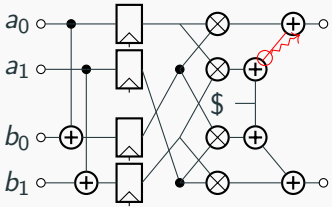
$$x \times y + r = (x + 1) \times r + x \times (y - r)$$

Masking in hardware

Robust probing model

Real-world: CMOS, not arithmetic circuits.

Modeled with extended probes:

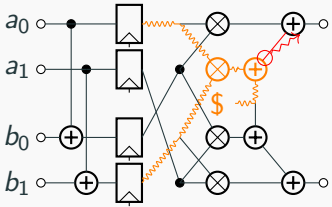


Robust probing model

Real-world: CMOS, not arithmetic circuits.

Modeled with extended probes:

- Glitches
  $\rightarrow$ all inputs of combinationial circuit.

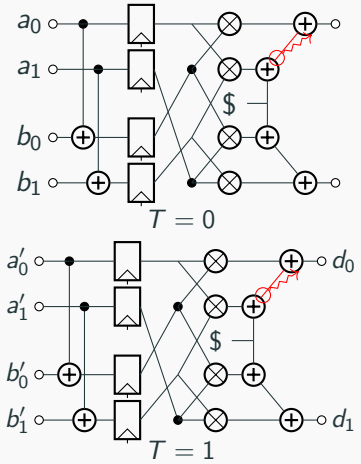


Robust probing model

Real-world: CMOS, not arithmetic circuits.

Modeled with extended probes:

- Glitches
  $\rightarrow$ all inputs of combinationial circuit.
- Transitions
  $\rightarrow$ wire for two consecutive cycles.

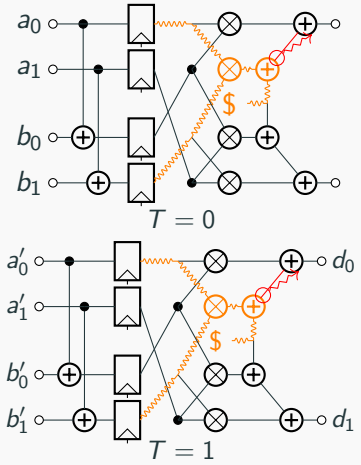


Robust probing model

Real-world: CMOS, not arithmetic circuits.

Modeled with extended probes:

- Glitches
  $\rightarrow$ all inputs of combinationial circuit.
- Transitions
  $\rightarrow$ wire for two consecutive cycles.
- Both
  $\rightarrow$ all inputs for two cycles.

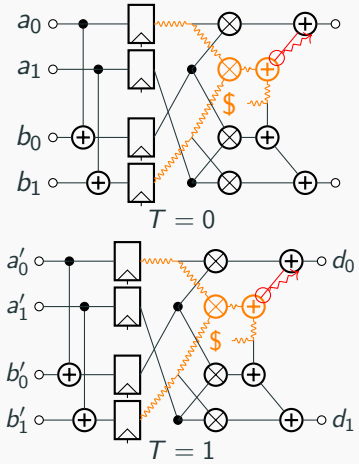


Robust probing model

Real-world: CMOS, not arithmetic circuits.

Modeled with extended probes:

- Glitches
  $\rightarrow$ all inputs of combinationial circuit.
- Transitions
  $\rightarrow$ wire for two consecutive cycles.
- Both
  $\rightarrow$ all inputs for two cycles.



Sharewise circuits are robust: glitches and transitions confined to one circuit share.

Glitch-robustness

Glitches captured by gadget interfaces

- being carried by wires

Glitch-robustness

Glitches captured by gadget interfaces

- being carried by wires

Glitch-robust simulatability

- applies to *NI notions
- glitch-robust version of composition theorems

Glitch-robustness

Glitches captured by gadget interfaces

- being carried by wires

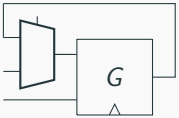
Glitch-robust simulatability

- applies to *NI notions
- glitch-robust version of composition theorems

Where do we need registers?

Transition-robustness: OPINI

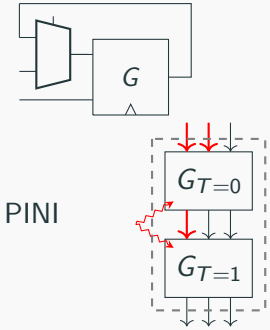
For share-isolating, if you know the input, you can simulate the corresponding output.



Transition-robustness: OPINI

For share-isolating, if you know the input, you can simulate the corresponding output.

- Not for PINI



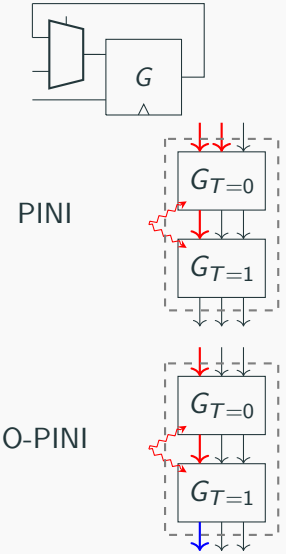
Transition-robustness: OPINI

For share-isolating, if you know the input, you can simulate the corresponding output.

- Not for PINI

OPINI (Output-PINI):

- PINI
- *For each input you know, simulate the corresponding output.*



Transition-robustness: OPINI

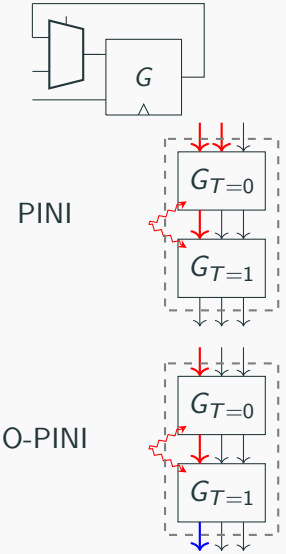
For share-isolating, if you know the input, you can simulate the corresponding output.

- Not for PINI

OPINI (Output-PINI):

- PINI
- *For each input you know, simulate the corresponding output.*

OPINI gadgets trivially compose under transitions.



Transition-robustness: OPINI

For share-isolating, if you know the input, you can simulate the corresponding output.

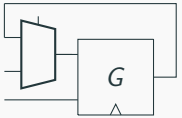
- Not for PINI

OPINI (Output-PINI):

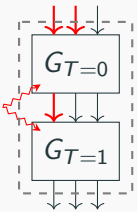
- PINI
- *For each input you know, simulate the corresponding output.*

OPINI gadgets trivially compose under transitions.

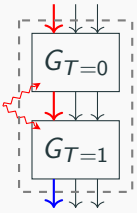
OPINI costs randomness, area and/or latency.



PINI



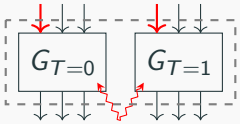
O-PINI



Cheap transition-robustness

Transitions can be worked around:

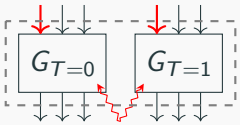
- with transitions, PINI gadgets compose in parallel,



Cheap transition-robustness

Transitions can be worked around:

- with transitions, PINI gadgets compose in parallel,
- pipeline bubbles remove transition leakage.



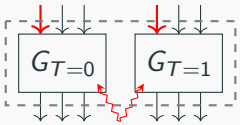
Cheap transition-robustness

Transitions can be worked around:

- with transitions, PINI gadgets compose in parallel,
- pipeline bubbles remove transition leakage.

For cheap transition-robustness:

- Put PINI gadgets in a pipeline
- Insert a bubble between iterations
  - Rare event in many crypto designs
- Use sharewise gadgets freely



Hardware Private Circuits: PINI in hardware

Masked AND gate cost

Gadget	Security	Latency		Area (GE)			Area w/ RNG (GE)		
		x	y	$t=1$	$t=2$	$t=3$	$t=1$	$t=2$	$t=3$
DOM-indep	NI	1	1	35	85	156	74	203	392
HPC1	PINI	2	1	51	127	217	130	324	611
HPC2	PINI	2	1	82	209	393	121	327	629
HPC3	PINI	1	1	69	165	301	148	401	773
HPC4	OPINI	1	1	93	238	451	290	829	1632

- Nangate45 / Yosys / Trivium RNG.
- There optimized variants exist.
- Some gadgets can be extended to larger fields.
- Non-exhaustive list!

SMAesH: SIMPLE-Crypto's Masked AES in Hardware

Masked AES-128 implementation:

- Arbitrary-order with HPC1& HPC3.
- 32-bit architecture
- 86 cycles latency
- Includes key schedule & PRNG (Trivium)
- Open-source:
  `https://github.com/simple-crypto/SMAesH`
- Verified (fullVerif, SILVER)
- CHES23 challenge: 290k traces (first order)

	Area (kGE)		
t	S-box		Full AES
	w/o RNG	w/ RNG	w/ RNG
1	1.9	3.4	24.4
2	4.6	8.3	47.4
3	8.1	15.6	81.2

Version 2 just released (area and latency optimizations).

Conclusion

Composable masking schemes:

- provable security
- automated generation & verification
- good efficiency & flexibility

Conclusion

Composable masking schemes:

- provable security
- automated generation & verification
- good efficiency & flexibility

Challenges:

- tooling
- security beyond threshold probing model
- randomness reduction/re-use