Radboud University

Probing Secure Composability Without Fresh Randomness: Theory and Application to Ascon

Vahid Jahandideh, Bart Mennink, Lejla Batina
Radboud University (The Netherlands)

CHES 2025
September 14–18, 2025

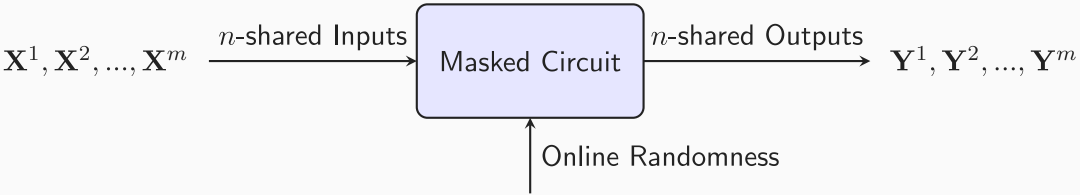
- Side-channel attacks (SCAs) threaten lightweight cryptography.

Motivation

- Side-channel attacks (SCAs) threaten lightweight cryptography.
- Masking is the main countermeasure but requires costly fresh (online) randomness.

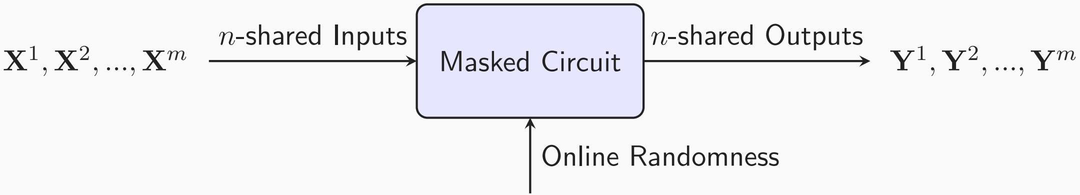
Motivation

- Side-channel attacks (SCAs) threaten lightweight cryptography.
- Masking is the main countermeasure but requires costly fresh (online) randomness.



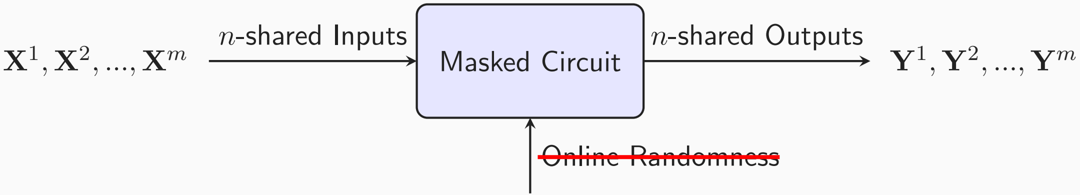
Motivation

- Side-channel attacks (SCAs) threaten lightweight cryptography.
- Masking is the main countermeasure but requires costly fresh (online) randomness.
- Resource-constrained devices (IoT, embedded) struggle with randomness generation.



Motivation

- Side-channel attacks (SCAs) threaten lightweight cryptography.
- Masking is the main countermeasure but requires costly fresh (online) randomness.
- Resource-constrained devices (IoT, embedded) struggle with randomness generation.
- **Question:** Can we achieve probing security without online randomness?

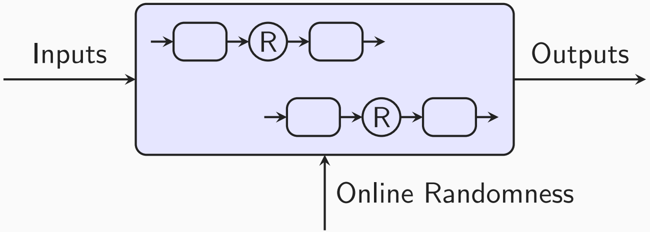


Research Question

- Standard composition requires refresh gadgets $\Rightarrow$ needs randomness.

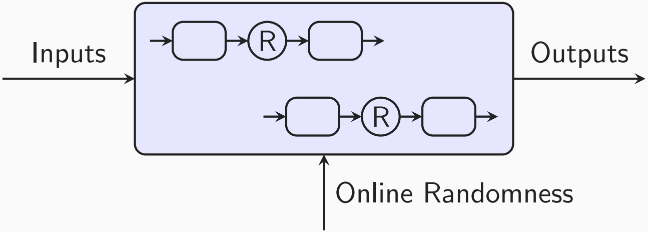
Research Question

- Standard composition requires refresh gadgets $\Rightarrow$ needs randomness.

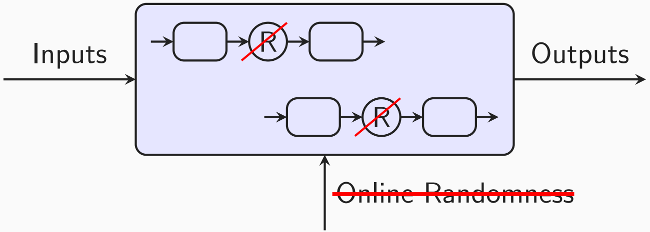


Research Question

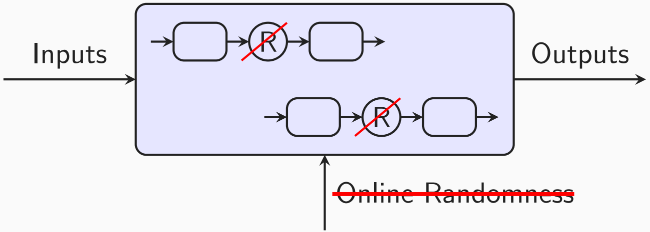
- Standard composition requires refresh gadgets $\Rightarrow$ needs randomness.
- Deterministic masking: rely only on offline randomness (input shares).



- Standard composition requires refresh gadgets $\Rightarrow$ needs randomness.
- Deterministic masking: rely only on offline randomness (input shares).



- Standard composition requires refresh gadgets $\Rightarrow$ needs randomness.
- Deterministic masking: rely only on offline randomness (input shares).
- Our goal: Prove & implement composability without fresh randomness.

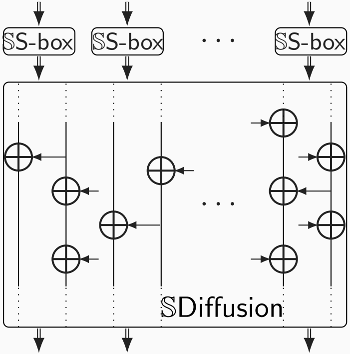


Key Idea

- Use the diffusion layer of round-based ciphers as an implicit refresh.

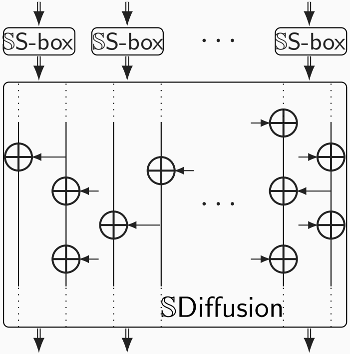
Key Idea

- Use the diffusion layer of round-based ciphers as an implicit refresh.



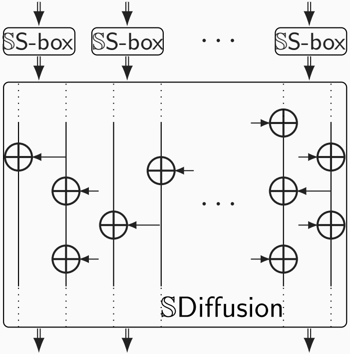
Key Idea

- Use the diffusion layer of round-based ciphers as an implicit refresh.
- Diffusion ensures adversary probes across rounds remain independent.



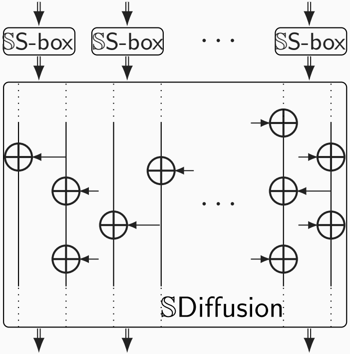
Key Idea

- Use the diffusion layer of round-based ciphers as an implicit refresh.
- Diffusion ensures adversary probes across rounds remain independent.
- Branch number $\mathcal{B}_{l\text{-box}}$ quantifies independence guarantee.



Key Idea

- Use the diffusion layer of round-based ciphers as an implicit refresh.
- Diffusion ensures adversary probes across rounds remain independent.
- Branch number $\mathcal{B}_{l\text{-box}}$ quantifies independence guarantee.
- Security order derives from gadget security $d_{\mathbb{S}\text{S-box}}$ and $\mathcal{B}_{l\text{-box}}$.



Our Contributions

- Theory: **Composition theorem** for probing-secure deterministic masking.

- Theory: **Composition theorem** for probing-secure deterministic masking.
- Simulation-based reasoning adapted to deterministic setting.

Our Contributions

- Theory: **Composition theorem** for probing-secure deterministic masking.
- Simulation-based reasoning adapted to deterministic setting.

- Formal results:

Our Contributions

- Theory: **Composition theorem** for probing-secure deterministic masking.
- Simulation-based reasoning adapted to deterministic setting.

- Formal results:
    - First-order composability proof.

Our Contributions

- Theory: **Composition theorem** for probing-secure deterministic masking.
- Simulation-based reasoning adapted to deterministic setting.

- Formal results:
  - First-order composability proof.
  - Higher-order security via branch number.

Our Contributions

- Theory: **Composition theorem** for probing-secure deterministic masking.
- Simulation-based reasoning adapted to deterministic setting.

- Formal results:
  - First-order composability proof.
  - Higher-order security via branch number.

- Practice: Implementation on Ascon (NIST LWC standard).

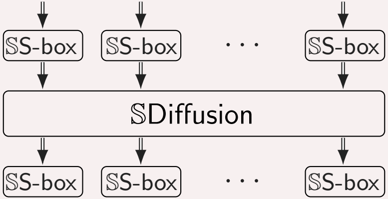
Composition Theorem (Theorem 1)

Theorem (simplified)

Consider two $\mathbb{S}$S-box layers separated by a $\mathbb{S}$Diffusion layer.

Theorem (simplified)

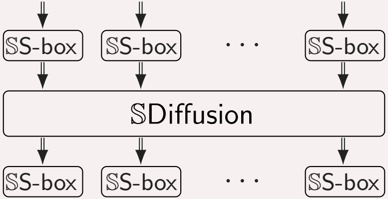
Consider two $\mathbb{S}$S-box layers separated by a $\mathbb{S}$Diffusion layer.



Composition Theorem (Theorem 1)

Theorem (simplified)

Consider two $\mathbb{S}$S-box layers separated by a $\mathbb{S}$Diffusion layer.

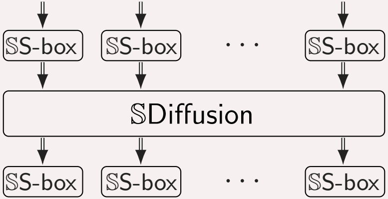


If each $\mathbb{S}$S-box is $d_{\mathbb{S}\text{S-box}}$-probing secure and diffusion has box branch number $\mathcal{B}_{l\text{-box}}$, then the combined structure achieves security order:

Composition Theorem (Theorem 1)

Theorem (simplified)

Consider two $\mathbb{S}$S-box layers separated by a $\mathbb{S}$Diffusion layer.



If each $\mathbb{S}$S-box is $d_{\mathbb{S}\text{S-box}}$-probing secure and diffusion has box branch number $\mathcal{B}_{l\text{-box}}$, then the combined structure achieves security order:

$$\min\{d_{\mathbb{S}\text{S-box}},\ (\mathcal{B}_{l\text{-box}} - 1)\} \leq d_{\text{comb}} \leq d_{\mathbb{S}\text{S-box}}.$$

Box Linear Branch Number

- Consider a simple linear mapping on $\mathbf{S} \in \{0,1\}^{15}$:

$$\mathbf{S}' \leftarrow \mathbf{S} \oplus (\mathbf{S} \lll 1) \oplus (\mathbf{S} \lll 3),$$

where $\lll w$ denotes a w-bit circular left shift.

Box Linear Branch Number

- Consider a simple linear mapping on $\mathbf{S} \in \{0,1\}^{15}$:

$$\mathbf{S}' \leftarrow \mathbf{S} \oplus (\mathbf{S} \lll 1) \oplus (\mathbf{S} \lll 3),$$

where $\lll w$ denotes a w-bit circular left shift.

- For instance:

$$S'[0] = S[0] \oplus S[14] \oplus S[12].$$

Box Linear Branch Number

- Consider a simple linear mapping on $\mathbf{S} \in \{0,1\}^{15}$:

$$\mathbf{S}' \leftarrow \mathbf{S} \oplus (\mathbf{S} \lll 1) \oplus (\mathbf{S} \lll 3),$$

  where $\lll w$ denotes a w-bit circular left shift.
- For instance:

$$S'[0] = S[0] \oplus S[14] \oplus S[12].$$

- Any parity relation between inputs and outputs will involve at least 4 terms.

Box Linear Branch Number

- Consider a simple linear mapping on $\mathbf{S} \in \{0,1\}^{15}$:

$$\mathbf{S}' \leftarrow \mathbf{S} \oplus (\mathbf{S} \lll 1) \oplus (\mathbf{S} \lll 3),$$

  where $\lll w$ denotes a w-bit circular left shift.
- For instance:

$$S'[0] = S[0] \oplus S[14] \oplus S[12].$$

- Any parity relation between inputs and outputs will involve at least 4 terms.
- With fewer than 4 probes, no variable outside the probed set can leak.

- Consider a simple linear mapping on $\mathbf{S} \in \{0,1\}^{15}$:

$$\mathbf{S}' \leftarrow \mathbf{S} \oplus (\mathbf{S} \lll 1) \oplus (\mathbf{S} \lll 3),$$

  where $\lll w$ denotes a w-bit circular left shift.
- For instance:

$$S'[0] = S[0] \oplus S[14] \oplus S[12].$$

- Any parity relation between inputs and outputs will involve at least 4 terms.
- With fewer than 4 probes, no variable outside the probed set can leak.

Definition: Box Linear Branch Number

Let $\mathbf{M}$ be a $b \times b$ binary matrix representing a linear diffusion layer, and let S denote its b-bit input. The box linear branch number $\mathcal{B}_{l\text{-box}}$ is defined as:

$$\mathcal{B}_{l\text{-box}} = \min_{S \neq 0^b} \left[\#\text{active S-boxes}(S) + \#\text{active S-boxes}(S\mathbf{M}^\top) \right],$$

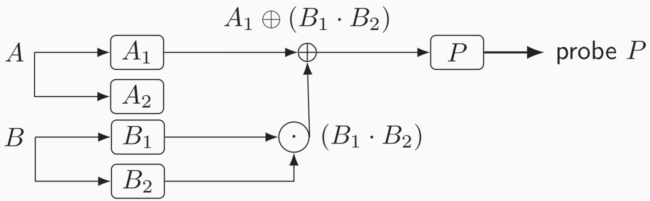
where an S-box is considered active if any of its input bits is non-zero.

Protection using Input Shares

- Offline randomness embedded in the input shares can *blind* probes on intermediate values.

- Offline randomness embedded in the input shares can *blind* probes on intermediate values.
- Example ($n = 2$): Let $\{A_1, A_2\}$ and $\{B_1, B_2\}$ be the shares of secrets A and B. Consider the probe

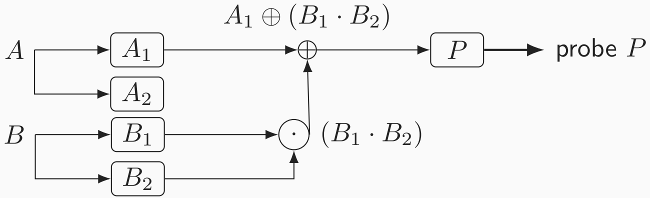
$$P = A_1 \oplus (B_1 \cdot B_2).$$



Protection using Input Shares

- Offline randomness embedded in the input shares can *blind* probes on intermediate values.
- Example ($n = 2$): Let $\{A_1, A_2\}$ and $\{B_1, B_2\}$ be the shares of secrets A and B. Consider the probe

$$P = A_1 \oplus (B_1 \cdot B_2).$$



- In the typical probe-propagation framework, both shares $\{B_1, B_2\}$ are required, and P would be identified as *secret dependent*.

Protection using Input Shares

- Offline randomness embedded in the input shares can *blind* probes on intermediate values.
- Example ($n = 2$): Let $\{A_1, A_2\}$ and $\{B_1, B_2\}$ be the shares of secrets A and B. Consider the probe

$$P = A_1 \oplus (B_1 \cdot B_2).$$

- In the typical probe-propagation framework, both shares $\{B_1, B_2\}$ are required, and P would be identified as *secret dependent*.

Protection using Input Shares

- Offline randomness embedded in the input shares can *blind* probes on intermediate values.
- Example ($n = 2$): Let $\{A_1, A_2\}$ and $\{B_1, B_2\}$ be the shares of secrets A and B. Consider the probe

$$P = A_1 \oplus (B_1 \cdot B_2).$$

- In the typical probe-propagation framework, both shares $\{B_1, B_2\}$ are required, and P would be identified as *secret dependent*.
- However, direct verification shows that P is in fact independent of the secrets A and B.

Protection using Input Shares

- Offline randomness embedded in the input shares can *blind* probes on intermediate values.
- Example ($n = 2$): Let $\{A_1, A_2\}$ and $\{B_1, B_2\}$ be the shares of secrets A and B. Consider the probe

$$P = A_1 \oplus (B_1 \cdot B_2).$$

- In the typical probe-propagation framework, both shares $\{B_1, B_2\}$ are required, and P would be identified as *secret dependent*.
- However, direct verification shows that P is in fact independent of the secrets A and B.
- $\Rightarrow$ Simulation rules are efficient, but not always *general*.

Key Requirement

The input shares must be mutually independent.

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

This Work (Deterministic Composition)

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

- *Scope:* Gadget-level design.

This Work (Deterministic Composition)

- *Scope:* Circuit-/round-level.

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

- *Scope:* Gadget-level design.
- *Randomness:* Higher-order TI typically needs **online refresh randomness**.

This Work (Deterministic Composition)

- *Scope:* Circuit-/round-level.
- *Randomness:* **No online randomness** — rely only on initial (offline) sharing.

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

- *Scope:* Gadget-level design.
- *Randomness:* Higher-order TI typically needs **online refresh randomness**.
- *Composition:* Secure composition relies on explicit **refresh gadgets**.

This Work (Deterministic Composition)

- *Scope:* Circuit-/round-level.
- *Randomness:* **No online randomness** — rely only on initial (offline) sharing.
- *Composition:* Diffusion layer acts as an **implicit refresh** across rounds.

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

- *Scope:* Gadget-level design.
- *Randomness:* Higher-order TI typically needs **online refresh randomness**.
- *Composition:* Secure composition relies on explicit **refresh gadgets**.
- *Proof tooling:* Often tied to SNI/PINI-style simulation rules.

This Work (Deterministic Composition)

- *Scope:* Circuit-/round-level.
- *Randomness:* **No online randomness** — rely only on initial (offline) sharing.
- *Composition:* Diffusion layer acts as an **implicit refresh** across rounds.
- *Security bound:* $d_{\text{comb}} \in \left[\min\left\{d_{\mathbb{S}\text{S-box}},\ (\mathcal{B}_{l\text{-box}} - 1)\right\},\ d_{\mathbb{S}\text{S-box}}\right]$.

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

- *Scope:* Gadget-level design.
- *Randomness:* Higher-order TI typically needs **online refresh randomness**.
- *Composition:* Secure composition relies on explicit **refresh gadgets**.
- *Proof tooling:* Often tied to SNI/PINI-style simulation rules.
- *Trade-off:* More shares and/or refresh cost $\Rightarrow$ randomness and performance overhead.

This Work (Deterministic Composition)

- *Scope:* Circuit-/round-level.
- *Randomness:* **No online randomness** — rely only on initial (offline) sharing.
- *Composition:* Diffusion layer acts as an **implicit refresh** across rounds.
- *Security bound:* $d_{\text{comb}} \in \left[\min\left\{d_{\mathbb{S}\text{S-box}},\ (\mathcal{B}_{l\text{-box}} - 1)\right\},\ d_{\mathbb{S}\text{S-box}}\right]$.
- *Practicality:* Simpler randomness pipeline; compatible with deterministic $\mathbb{S}$S-box.

Comparison with Threshold Implementation (TI)

Threshold Implementation (TI)

- *Scope:* Gadget-level design.
- *Randomness:* Higher-order TI typically needs **online refresh randomness**.
- *Composition:* Secure composition relies on explicit **refresh gadgets**.
- *Proof tooling:* Often tied to SNI/PINI-style simulation rules.
- *Trade-off:* More shares and/or refresh cost ⇒ randomness and performance overhead.

This Work (Deterministic Composition)

- *Scope:* Circuit-/round-level.
- *Randomness:* **No online randomness** — rely only on initial (offline) sharing.
- *Composition:* Diffusion layer acts as an **implicit refresh** across rounds.
- *Security bound:* $d_{\text{comb}} \in \left[\min\{d_{\mathbb{S}\text{S-box}}, (\mathcal{B}_{l\text{-box}} - 1)\}, d_{\mathbb{S}\text{S-box}}\right]$.
- *Practicality:* Simpler randomness pipeline; compatible with deterministic $\mathbb{S}$S-box.

Takeaway

TI composes via *refresh gadgets* (costing online randomness), whereas our approach composes via the cipher's *diffusion layer* (no online randomness).

- Ascon: NIST LWC standard; bricklayer design with diffusion-friendly rounds.

Case Study: Ascon (1/2)

- Ascon: NIST LWC standard; bricklayer design with diffusion-friendly rounds.
- Box linear branch number: $\mathcal{B}_{l\text{-box}} = 4$.

Case Study: Ascon (1/2)

- Ascon: NIST LWC standard; bricklayer design with diffusion-friendly rounds.
- Box linear branch number: $\mathcal{B}_{l\text{-box}} = 4$.
- Composition bound: $d_{\text{comb}} \in \left[\min\left\{d_{\mathbb{S}\text{S-box}},\ (\mathcal{B}_{l\text{-box}} - 1)\right\},\ d_{\mathbb{S}\text{S-box}}\right] \Rightarrow$ up to $d = 3$ if gadgets allow.

Case Study: Ascon (1/2)

- Ascon: NIST LWC standard; bricklayer design with diffusion-friendly rounds.
- Box linear branch number: $\mathcal{B}_{l\text{-box}} = 4$.
- Composition bound: $d_{\text{comb}} \in \left[\min\left\{d_{\mathbb{S}\text{S-box}},\ (\mathcal{B}_{l\text{-box}} - 1)\right\},\ d_{\mathbb{S}\text{S-box}}\right] \Rightarrow$ up to $d = 3$ if gadgets allow.
- We implement first- and second-order deterministic masked Ascon.

Case Study: Ascon (1/2)

- Ascon: NIST LWC standard; bricklayer design with diffusion-friendly rounds.
- Box linear branch number: $\mathcal{B}_{l\text{-box}} = 4$.
- Composition bound: $d_{\text{comb}} \in \left[\min\left\{d_{\mathbb{S}\text{S-box}},\ (\mathcal{B}_{l\text{-box}} - 1)\right\},\ d_{\mathbb{S}\text{S-box}}\right] \Rightarrow$ up to $d = 3$ if gadgets allow.
- We implement first- and second-order deterministic masked Ascon.
- Recent work extends this approach to *third-order* masked Ascon [JSB25].

Case Study: Ascon (2/2)

- For the second-order $\mathbb{S}$S-box, we observe a useful separation:

- For the second-order $\mathbb{S}$S-box, we observe a useful separation:

- Using this separation, we also **directly** verified second-order probing security.

- For the second-order $\mathbb{S}$S-box, we observe a useful separation:

n-shared inputs → Secrets map **non-linearly** / Shares (for fixed secrets) map **linearly** → n-shared outputs

- Using this separation, we also **directly** verified second-order probing security.
- Each input share Y to round r can be expressed in terms of the initial input shares X_i:

$$Y = \left(\bigoplus_{i \in \mathcal{I}} X_i \right) \oplus \left(\bigoplus_{j \in \mathcal{J}} X_j \cdot f_j(S) \right),$$

where $\mathcal{I}, \mathcal{J}$ are disjoint index sets, and each $f_j(S)$ is a Boolean function of the native state S.

Case Study: Ascon (2/2)

- For the second-order $\mathbb{S}$S-box, we observe a useful separation:

n-shared inputs → [Secrets map **non-linearly** → ; Shares (for fixed secrets) map **linearly** →] → n-shared outputs

- Using this separation, we also **directly** verified second-order probing security.
- Each input share Y to round r can be expressed in terms of the initial input shares X_i:

$$Y = \left(\bigoplus_{i \in \mathcal{I}} X_i \right) \oplus \left(\bigoplus_{j \in \mathcal{J}} X_j \cdot f_j(S) \right),$$

  where $\mathcal{I}, \mathcal{J}$ are disjoint index sets, and each $f_j(S)$ is a Boolean function of the native state S.
- This highlights how input shares inherently contribute to probing security.

Experimental Results

- Implemented masked Ascon with deterministic gadgets.

Experimental Results

- Implemented masked Ascon with deterministic gadgets.
- Verified with > 1M traces using T-tests.

- Implemented masked Ascon with deterministic gadgets.
- Verified with > 1M traces using T-tests.
- Passed both first- and second-order leakage tests.

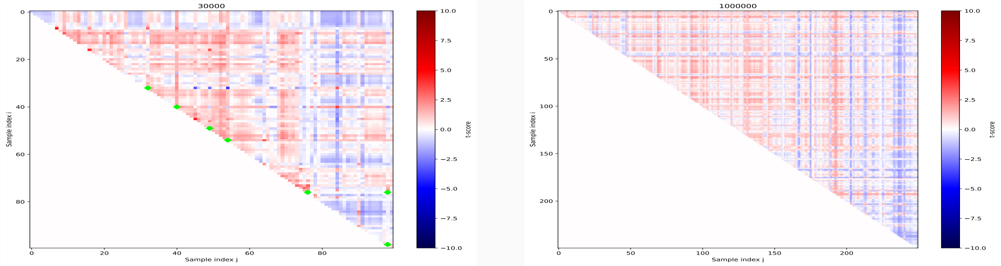


Figure: Second-order T-test results. Left: $n = 2$ shares ($d = 1$ protection), with green spots indicating values outside threshold. Right: $n = 3$ shares ($d = 2$ protection), showing no leakage. X- and Y-axes correspond to selected sample indices.

Experimental Results

- Implemented masked Ascon with deterministic gadgets.
- Verified with > 1M traces using T-tests.
- Passed both first- and second-order leakage tests.

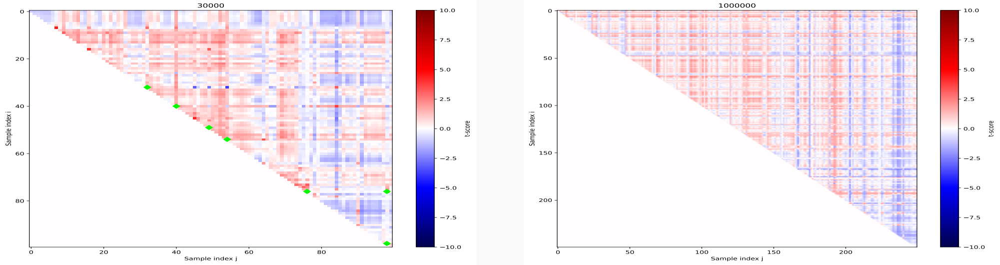


Figure: Second-order T-test results. Left: $n = 2$ shares ($d = 1$ protection), with green spots indicating values outside threshold. Right: $n = 3$ shares ($d = 2$ protection), showing no leakage. X- and Y-axes correspond to selected sample indices.

- Significant randomness savings compared to standard masking.

Conclusion

- Fresh randomness is not strictly necessary for secure masking.

Conclusion

- Fresh randomness is not strictly necessary for secure masking.
- Diffusion layer acts as implicit refresh.

Conclusion

- Fresh randomness is not strictly necessary for secure masking.
- Diffusion layer acts as implicit refresh.
- Our framework proves and demonstrates composability without online randomness.

Conclusion

- Fresh randomness is not strictly necessary for secure masking.
- Diffusion layer acts as implicit refresh.
- Our framework proves and demonstrates composability without online randomness.
- Opens path for resource-efficient, provably secure masking.

Conclusion

- Fresh randomness is not strictly necessary for secure masking.
- Diffusion layer acts as implicit refresh.
- Our framework proves and demonstrates composability without online randomness.
- Opens path for resource-efficient, provably secure masking.

Thank you for your attention!

Code: `https://github.com/vahid-jahandideh/deterministic-masking-ascon`