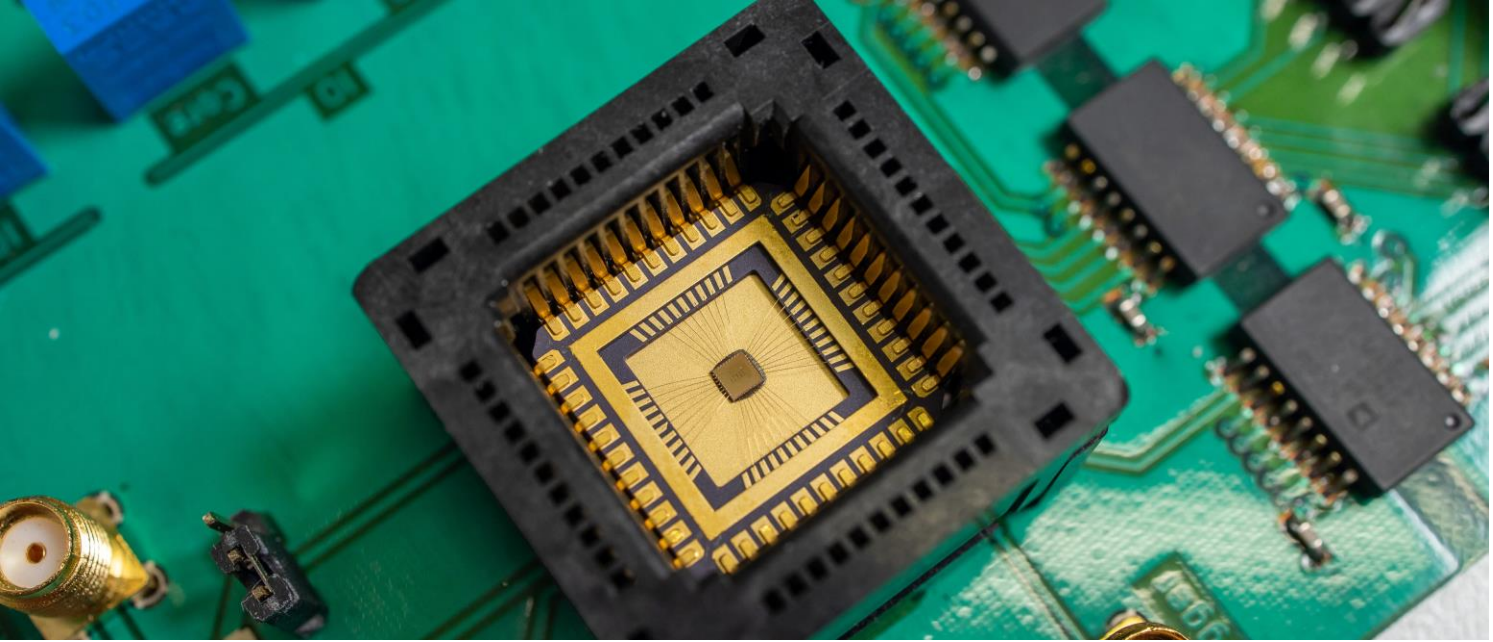


RUB IMPLEMENTATION SECURITY

RUHR-UNIVERSITÄT BOCHUM

Generic Hardware Private Circuits: Towards Automated Generation of Composable Secure Gadgets

David Knichel, Pascal Sasdrich and Amir Moradi

CHES, September 2022, Leuven, Belgium

CASA Cyber Security in the Age of Large-Scale Adversaries

DFG

STATUS QUO

What do we have?

Masking on an algorithmic Level

- Requires high expertise
- Prone to errors
- Often no formal security proof
- Leads to optimized designs

Gadget-based Masking

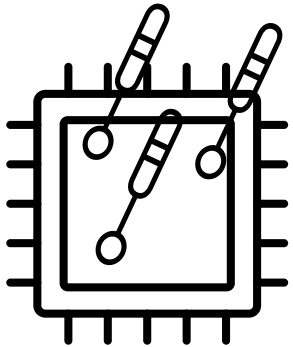
- Based on composability notions
- Can be automated
- Leads to provable secure designs
- Usually introduces higher overhead

ADVERSARY MODEL

What can an SCA adversary observe?

The ISW d-probing model

- Offers high abstraction
- Existing reduction into the *Noisy Leakage Model* (which is close to the real world)
- Extension to HW: robust d-probing Model



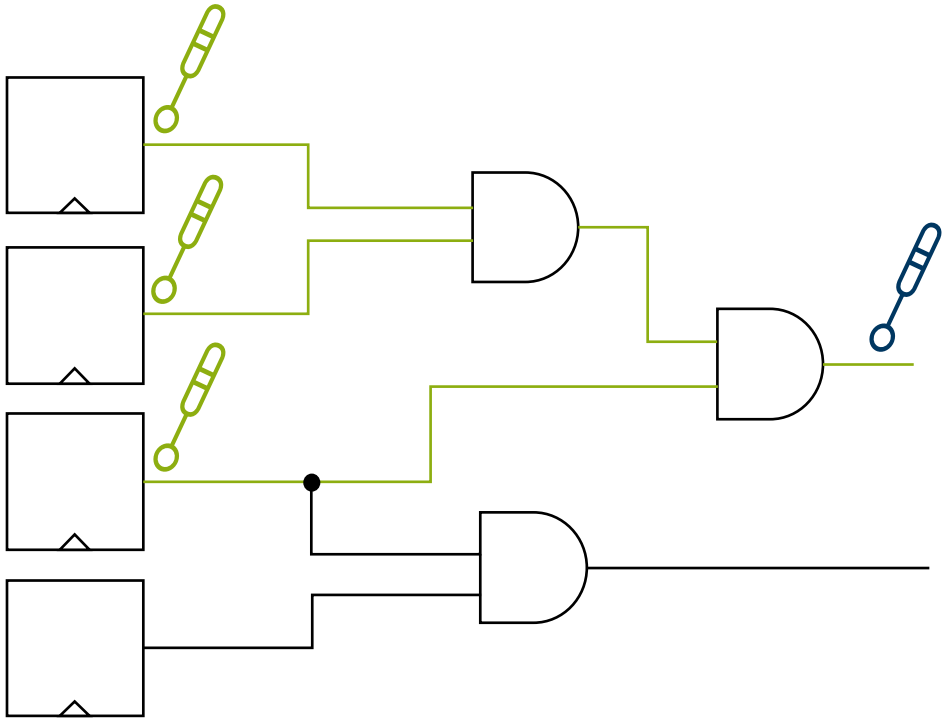
ADVERSARY MODEL

Glitch-extended robust probing model

Standard probe

⇩

Extended probe

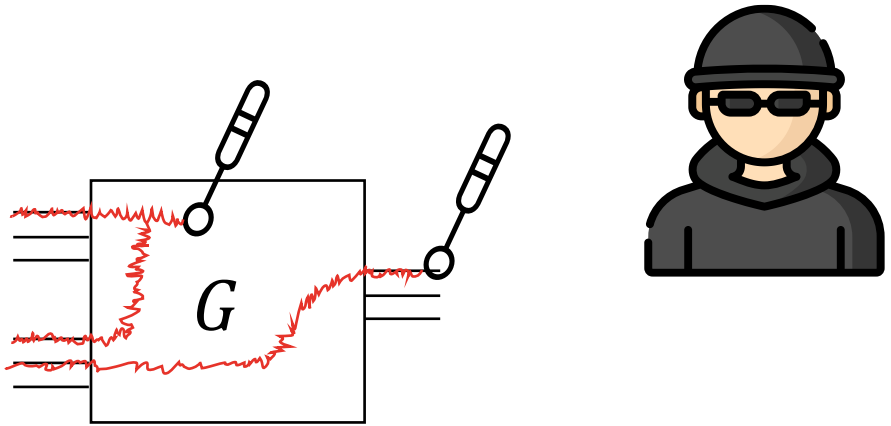


COMPOSABILITY

How to construct composable sub-circuits?

Composability Notions

- Restrict *Probe Propagation* within input-output boundaries of a sub-circuit

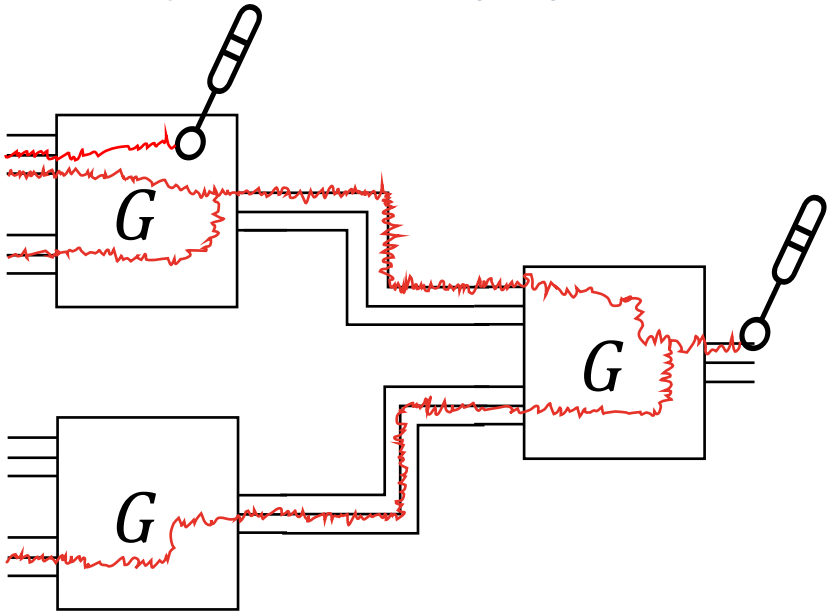


COMPOSABILITY

Implication to the overall circuit

Composed Circuit

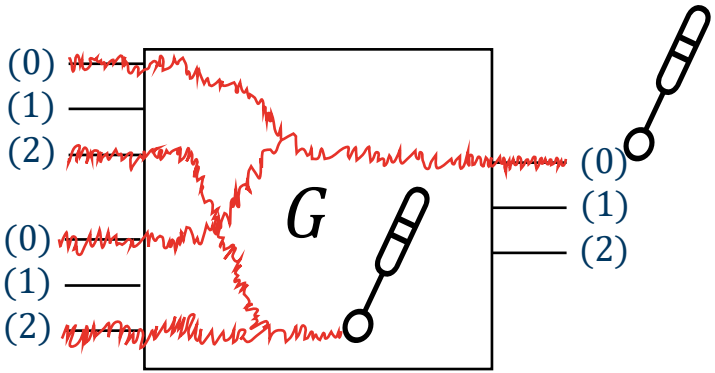
- Composability notion allows arguing about probe propagation within the overall circuit



PROBE-ISOLATING NON-INTERFERENCE (PINI)

How to restrict probe propagation?

Output-Input propagation only within same domain



PINI GADGETS

Status Quo

Gadgets fulfilling the PINI notion are typically realizing atomic non-linear gates

- AND, OR, NAND, NOR, ...

This introduces randomness and latency overhead per gadget

- Results in high overall overhead w.r.t. latency and randomness requirements

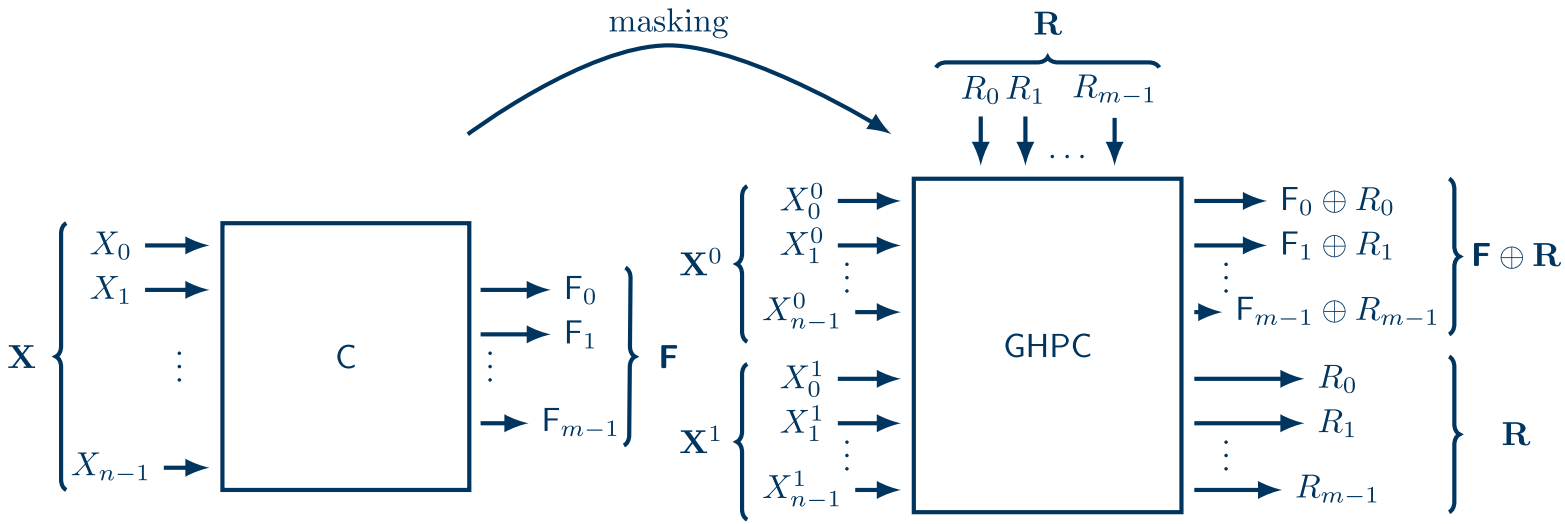
Can we build PINI gadgets with wider functionality to reduce randomness requirements and latency?

YES WE CAN!

Generic Hardware Private Circuits

We can transform any vectorial Boolean function into a 1-PINI-composable Gadget!

- Two register stages
- One bit randomness per coordinate function



BUT HOW?

Shannon Decomposition

$$F(X_0, X_1, \dots, X_{n-1}) = X_i \cdot \underbrace{F(X_0, \dots, X_{i-1}, 1, X_i, \dots, X_{n-1})}_{} \oplus \overline{X_i} \cdot \underbrace{F(X_0, \dots, X_{i-1}, 0, X_{i+1}, \dots, X_{n-1})}_{}$$

Shannon Cofactors

Observation: Only one Cofactor selected depending on X_i → Multiplexer with X_i as select signal

GHPC

An example always helps

$$F(X,Y,Z) = XYZ \oplus XZ \oplus Y$$

Direct Sharing: $F_m = (X^0 \oplus X^1)(Y^0 \oplus Y^1)(Z^0 \oplus Z^1) \oplus (X^0 \oplus X^1)(Z^0 \oplus Z^1) \oplus (Y^0 \oplus Y^1)$

Shannon Decomposition + blinding:

$$F^0$$

$$= \left[\overline{X^0}\,\overline{Y^0}\,\overline{Z^0}\left[\,X^1Y^1Z^1 \oplus X^1Z^1 \oplus Y^1 \oplus R\right]\right] \oplus \left[\overline{X^0}\,\overline{Y^0}\,Z^0\left[\,X^1Y^1\overline{Z^1} \oplus X^1\,\overline{Z^1} \oplus Y^1 \oplus R\right]\right]$$

$$\oplus \left[\overline{X^0}\,Y^0\overline{Z^0}\left[\,X^1\,\overline{Y^1}\,Z^1 \oplus X^1Z^1 \oplus \overline{Y^1} \oplus R\right]\right] \oplus \ldots$$

$$F^1 = R$$

Now the brackets correspond to the registers needed.

PROOF SKETCH

Why is this 1-PINI-composable?

$$\left[\overline{X^0}\,\overline{Y^0}\,\overline{Z^0}\left[X^1Y^1Z^1 \oplus X^1Z^1 \oplus Y^1 \oplus R\right]\right] \oplus \left[\overline{X^0}\,\overline{Y^0}\,Z^0\left[X^1Y^1\overline{Z^1} \oplus X^1\,\overline{Z^1} \oplus Y^1 \oplus R\right]\right]$$

$$\oplus \left[\overline{X^0}\,Y^0\overline{Z^0}\left[X^1\,\overline{Y^1}\,Z^1 \oplus X^1Z^1 \oplus \overline{Y^1} \oplus R\right]\right] \oplus \ \ldots$$

Probe placed on output (1) will reveal only R → Propagates in no share domain.

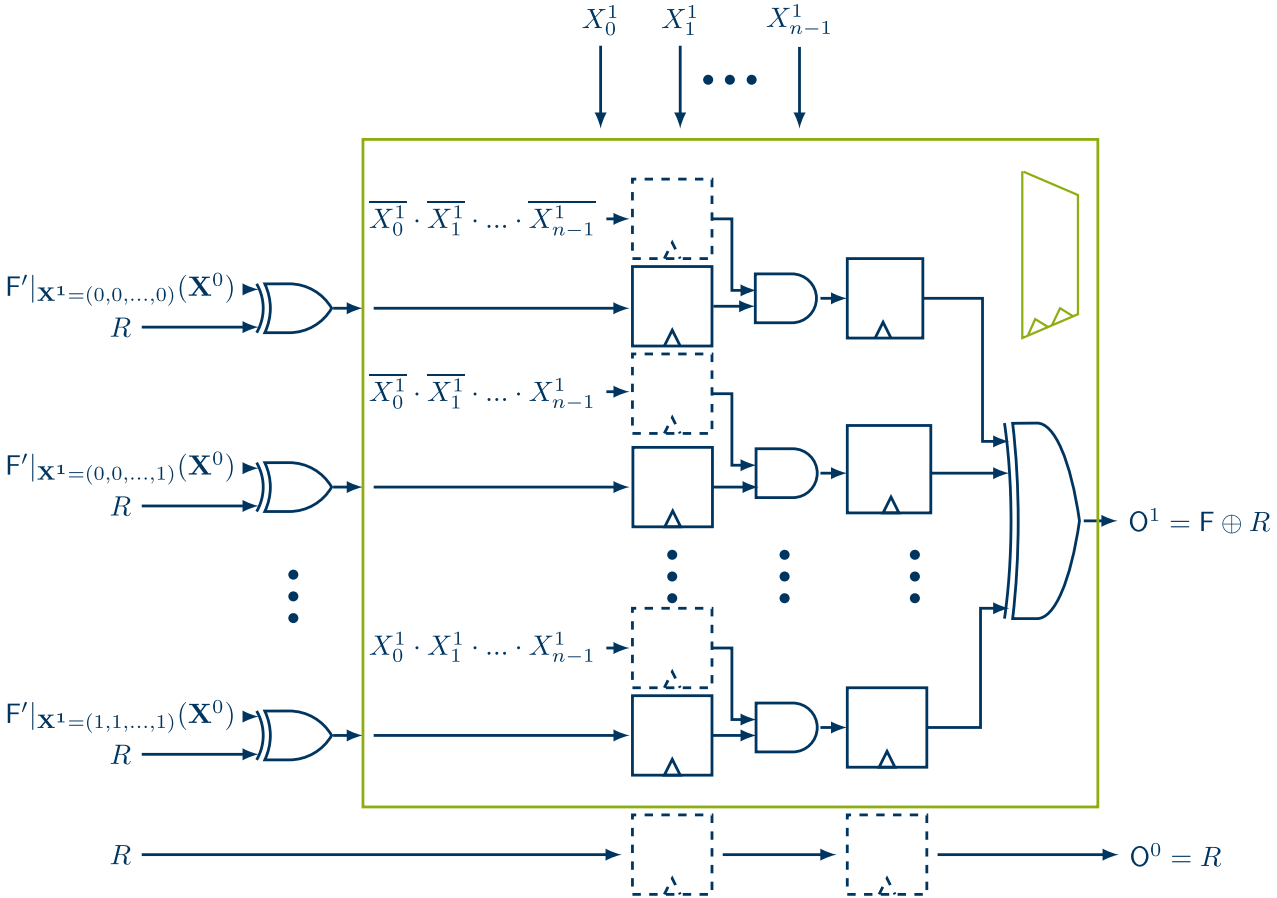
Probe placed on output (0) will reveal only one cofactor blinded by R (depending on X^0, Y^0, Z^0). → Propagates only into share domain 0.

Any internal probe on a single (cofactor + select)- term will only propagate into share domain 0 because the cofactor is blinded.

Any other term only includes either share domain 0 or share domain 1.

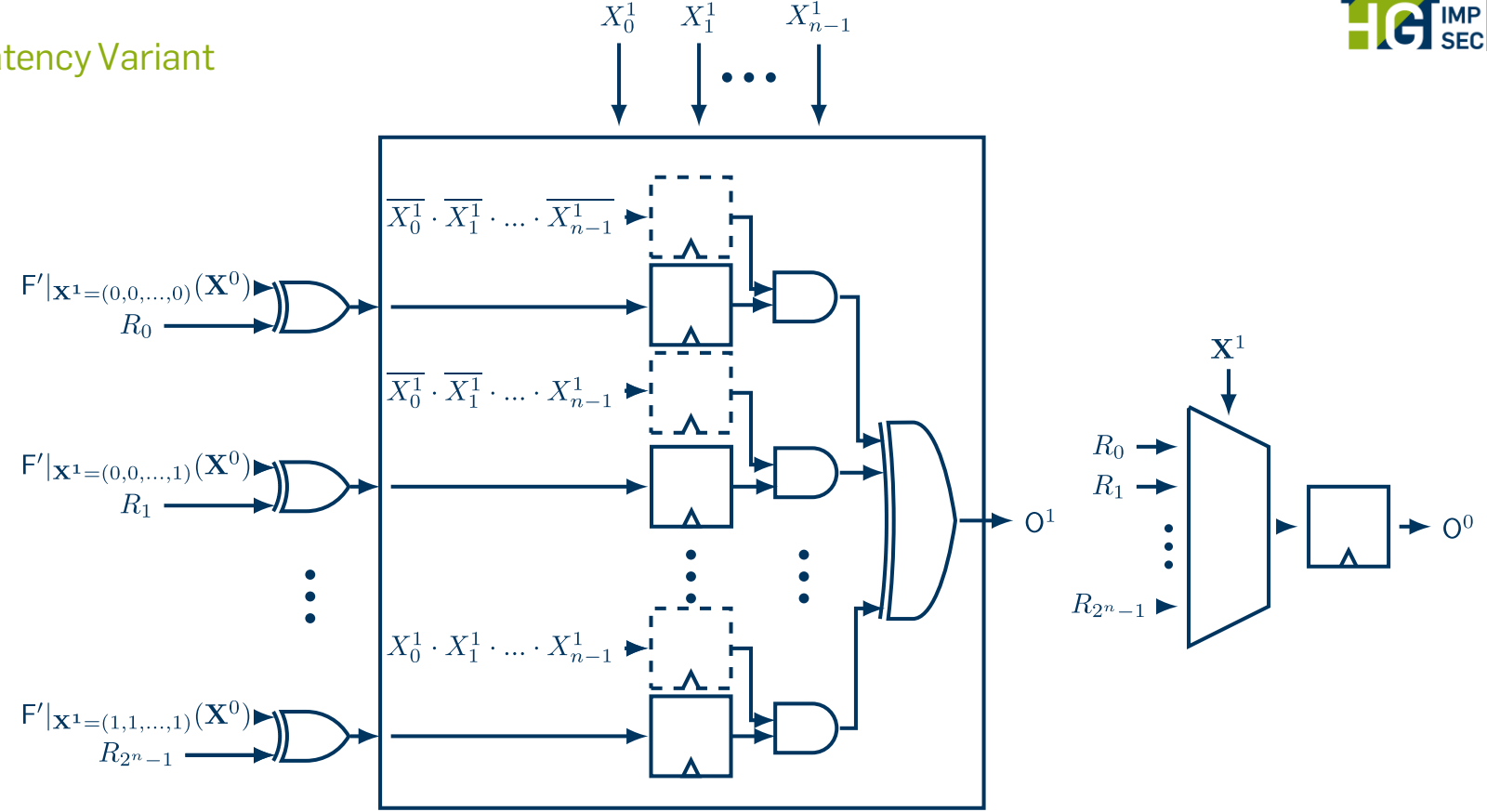
GHPC

Design


X_0^1 X_1^1 X_{n-1}^1
$\overline{X_0^1} \cdot \overline{X_1^1} \cdot \ldots \cdot \overline{X_{n-1}^1}$
$\overline{X_0^1} \cdot \overline{X_1^1} \cdot \ldots \cdot X_{n-1}^1$
$X_0^1 \cdot X_1^1 \cdot \ldots \cdot X_{n-1}^1$
$\mathsf{F}'|_{\mathbf{X}^1=(0,0,\ldots,0)}(\mathbf{X}^0)$
$\mathsf{F}'|_{\mathbf{X}^1=(0,0,\ldots,1)}(\mathbf{X}^0)$
$\mathsf{F}'|_{\mathbf{X}^1=(1,1,\ldots,1)}(\mathbf{X}^0)$
R
$\mathsf{O}^1 = \mathsf{F} \oplus R$
$\mathsf{O}^0 = R$

GHPC

Low-Latency Variant



CONCLUSION

Let's sum up

GHPC enables systematic construction of 1-PINI gadgets with any functionality

- Can be automated simply based on the functional description of the circuit

Significantly reduces randomness and latency requirements to existing 1-PINI gadgets.

- At the cost of a higher area requirement of the masked circuit itself

GHPC gadgets are trivially composable with other PINI-composable gadgets

- Trivially integrable into automated masking tools like AGEMA
- Enables dynamic trade-off between randomness, area and latency requirements of a design

Extension to higher orders is non-trivial

Thanks!

Any Questions?

david.knichel@rub.de