Low Randomness ~~Masking~~ and Shulfifgn:

An Evaluation Using Mutual Information

Kostas Papagiannopoulos
kostaspap88@gmail.com
kpcrypto.net
Radboud University Nijmegen
Digital Security Department
The Netherlands

Radboud Universiteit Nijmegen

connecting innovators

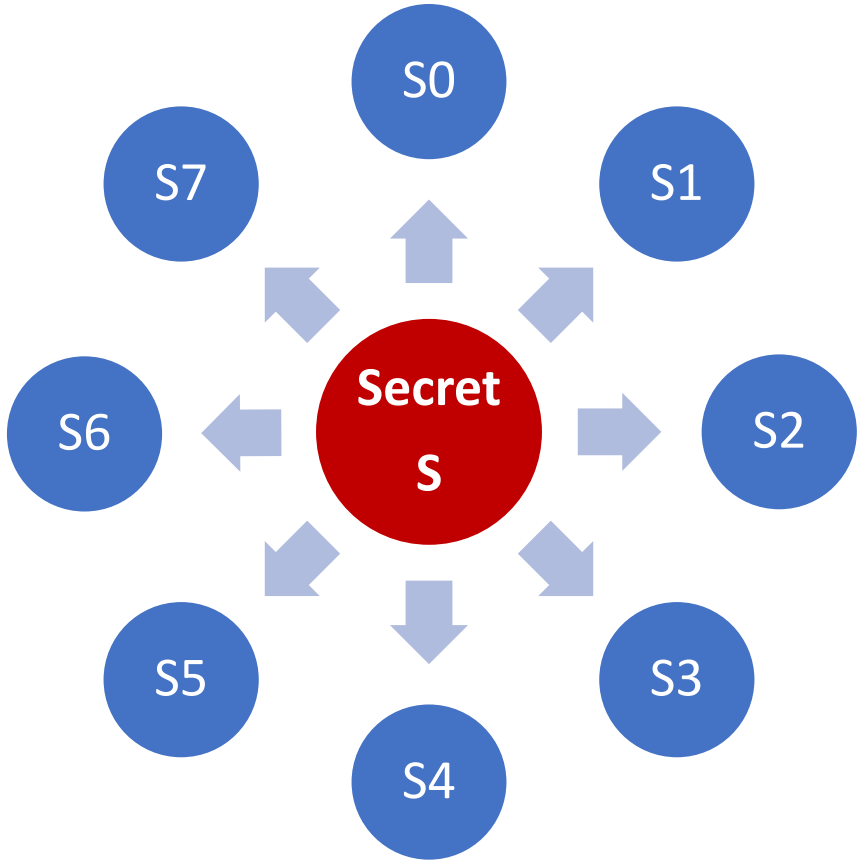
Overview

- Masking, shuffling and the cost of RNG
- New countermeasure variants that recycle randomness
- Pitfalls in formal security and noise amplification

Introduction

Masking and Shuffling Schemes Against Side-Channel Analysis

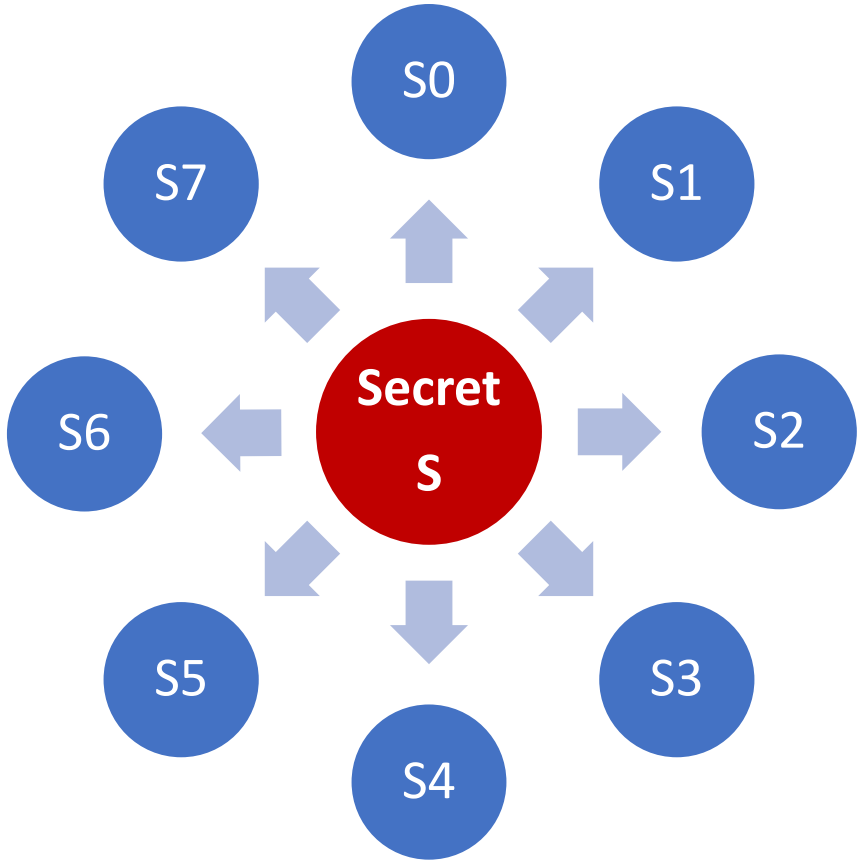
Introduction: Masking Schemes



- One of the most popular countermeasures against SCA
- Forces the adversary to recombine shares
- Performs noise amplification [1]

[1] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks.
[2] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks

Introduction: Masking Schemes

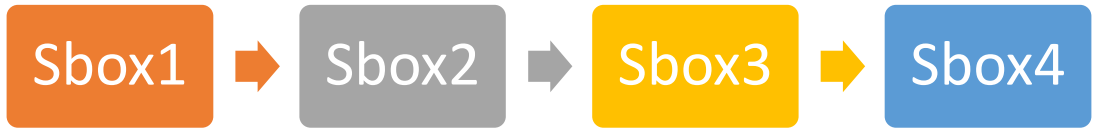


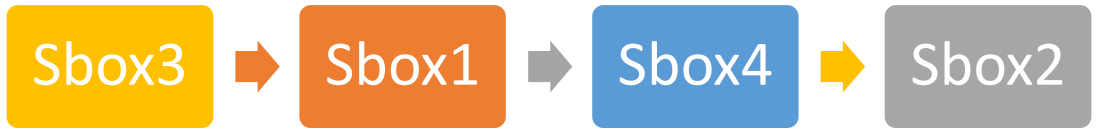
- One of the most popular countermeasures against SCA
- Forces the adversary to recombine shares
- Performs noise amplification [1]

- Computationally demanding in operations and RNG, $O(n^2)$ random elements for ISW multiplication with n shares [2]

[1] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks.
[2] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks

Introduction: Shuffling Schemes

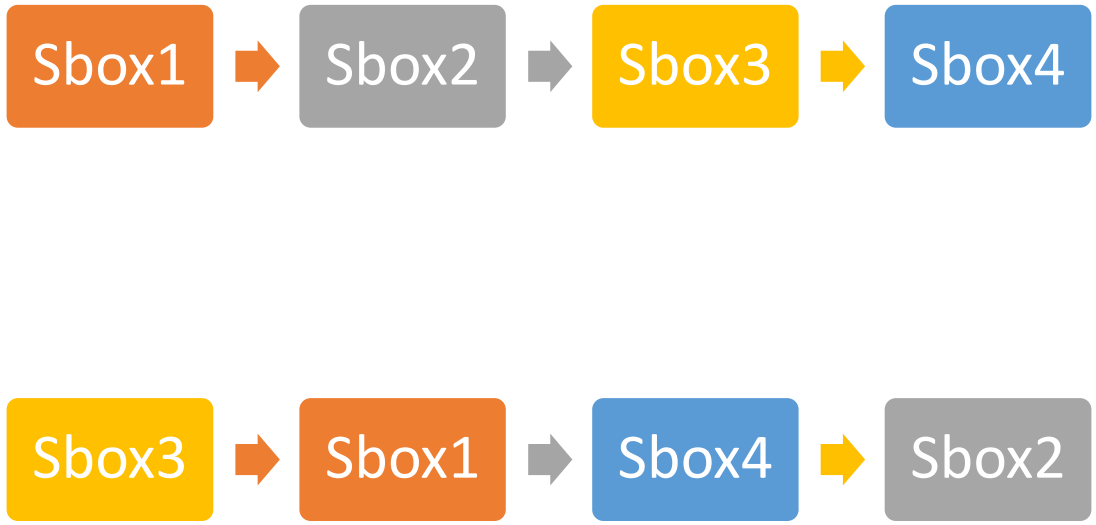




- Widely deployed countermeasure
- Permutes blocks
- Performs noise amplification [3]

[3] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note
[4] Donald E. Knuth. The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms

Introduction: Shuffling Schemes



- Widely deployed countermeasure
- Permutes blocks
- Performs noise amplification [3]

- Computationally demanding in RNG, approx. $\mathrm{k} * \mathrm{ceil}(log_2 \mathrm{k})$ random bits, for k operations shuffled [4]

[3] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note
[4] Donald E. Knuth. The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms

Introduction: RNG Overhead

- The RNG constitutes a considerable performance overhead

[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Introduction: RNG Overhead

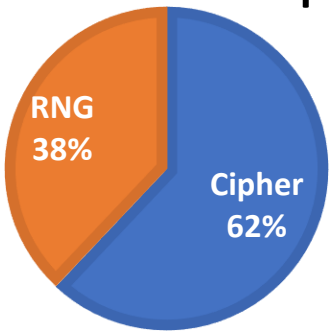
- The RNG constitutes a considerable performance overhead

- 2^{nd}-order AES on AVR pseudoRNG [5]

[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Introduction: RNG Overhead

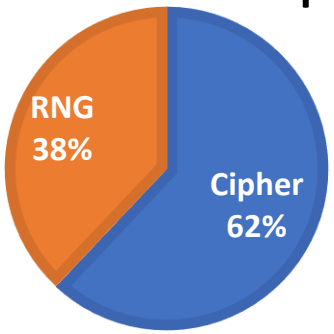
- The RNG constitutes a considerable performance overhead
- 2nd-order AES on AVR pseudoRNG [5]



[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Introduction: RNG Overhead

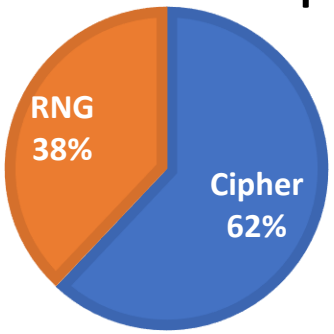
- The RNG constitutes a considerable performance overhead
- 2nd-order AES on AVR pseudoRNG [5]
- 2nd-order PRESENT on ARM Cortex-M4 trueRNG [6]



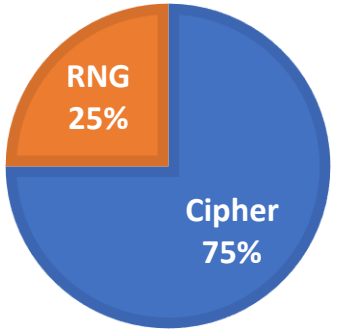
[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Introduction: RNG Overhead

- The RNG constitutes a considerable performance overhead
- 2nd-order AES on AVR pseudoRNG [5]



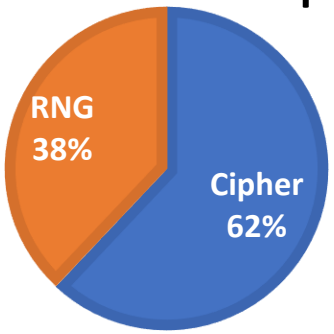
- 2nd-order PRESENT on ARM Cortex-M4 trueRNG [6]



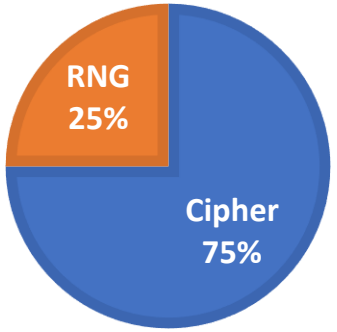
[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Introduction: RNG Overhead

- The RNG constitutes a considerable performance overhead
- 2nd-order AES on AVR pseudoRNG [5]



- 2nd-order PRESENT on ARM Cortex-M4 trueRNG [6]

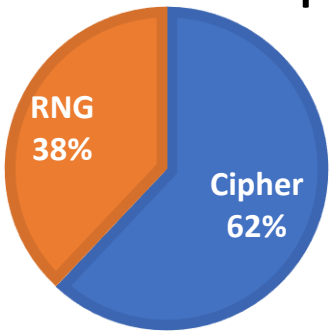


- 4th-order AES on ARM Cortex-A with NEON assembly /dev/urandom [7]

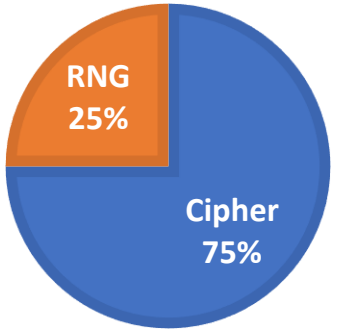
[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Introduction: RNG Overhead

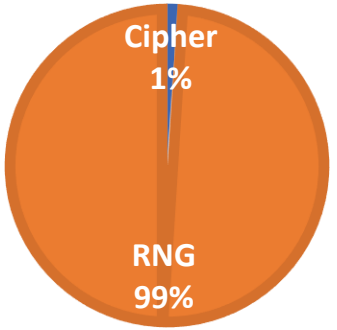
- The RNG constitutes a considerable performance overhead
- 2nd-order AES on AVR pseudoRNG [5]



- 2nd-order PRESENT on ARM Cortex-M4 trueRNG [6]



- 4th-order AES on ARM Cortex-A with NEON assembly /dev/urandom [7]



[5] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, Clara Paglialonga, and François-Xavier Standaert. Consolidating inner product masking
[6] Wouter de Groot, Kostas Papagiannopoulos, Antonio de La Piedra, Erik Schneider and Lejla Batina. Bitsliced masking and arm: Friends or foes?
[7] Benjamin Gregoire and Kostas Papagiannopoulos and Peter Schwabe and Ko Stoffelen . Vectorizing Higher-Order Masking

Recycled Randomness Masking

Reducing the RNG overhead in masking with RRM

RRM: Example

- Assume two 2nd-order secure, independent ISW mult. gadgets $z = xy, c = ab$

$$z_0 = x_0y_0 \oplus w_0 \oplus w_1$$
$$z_1 = x_1y_1 \oplus ((w_0 \oplus x_0y_1) \oplus x_1y_0) \oplus w_2$$
$$z_2 = x_2y_2 \oplus ((w_1 \oplus x_0y_2) \oplus x_2y_0) \oplus ((w_2 \oplus x_1y_2) \oplus x_2y_1)$$

$$c_0 = a_0b_0 \oplus t_0 \oplus t_1$$
$$c_1 = a_1b_1 \oplus ((t_0 \oplus a_0b_1) \oplus a_1b_0) \oplus t_2$$
$$c_2 = a_2b_2 \oplus ((t_1 \oplus a_0b_2) \oplus a_2b_0) \oplus ((t_2 \oplus a_1b_2) \oplus a_2b_1)$$

RRM: Example

- **Recycle some random numbers** from the first to the second gadget

$$z_0 = x_0 y_0 \oplus w_0 \oplus w_1$$
$$z_1 = x_1 y_1 \oplus ((w_0 \oplus x_0 y_1) \oplus x_1 y_0) \oplus w_2$$
$$z_2 = x_2 y_2 \oplus ((w_1 \oplus x_0 y_2) \oplus x_2 y_0) \oplus ((w_2 \oplus x_1 y_2) \oplus x_2 y_1)$$

$$c_0 = a_0 b_0 \oplus w_0 \oplus w_1$$
$$c_1 = a_1 b_1 \oplus ((w_0 \oplus a_0 b_1) \oplus a_1 b_0) \oplus t_2$$
$$c_2 = a_2 b_2 \oplus ((w_1 \oplus a_0 b_2) \oplus a_2 b_0) \oplus ((t_2 \oplus a_1 b_2) \oplus a_2 b_1)$$

Reduced Randomness cost by 2 random numbers

RRM: Example

- **Formal security verification** [8] : the 2-multiplication gadget is 2-NI

$$z_0 = x_0y_0 \oplus w_0 \oplus w_1$$
$$z_1 = x_1y_1 \oplus ((w_0 \oplus x_0y_1) \oplus x_1y_0) \oplus w_2$$
$$z_2 = x_2y_2 \oplus ((w_1 \oplus x_0y_2) \oplus x_2y_0) \oplus ((w_2 \oplus x_1y_2) \oplus x_2y_1)$$

$$c_0 = a_0b_0 \oplus w_0 \oplus w_1$$
$$c_1 = a_1b_1 \oplus ((w_0 \oplus a_0b_1) \oplus a_1b_0) \oplus t_2$$
$$c_2 = a_2b_2 \oplus ((w_1 \oplus a_0b_2) \oplus a_2b_0) \oplus ((t_2 \oplus a_1b_2) \oplus a_2b_1)$$

[8] Jean-Sebastien Coron. Formal Verification of Side-channel Countermeasures via Elementary Circuit Transformations.

RRM: Example

- **Recycle more!**

$z_0 = x_0y_0 \oplus w_0 \oplus w_1$
$z_1 = x_1y_1 \oplus ((w_0 \oplus x_0y_1) \oplus x_1y_0) \oplus w_2$
$z_2 = x_2y_2 \oplus ((w_1 \oplus x_0y_2) \oplus x_2y_0) \oplus ((w_2 \oplus x_1y_2) \oplus x_2y_1)$

$c_0 = a_0b_0 \oplus w_0 \oplus w_1$
$c_1 = a_1b_1 \oplus ((w_0 \oplus a_0b_1) \oplus a_1b_0) \oplus w_2$
$c_2 = a_2b_2 \oplus ((w_1 \oplus a_0b_2) \oplus a_2b_0) \oplus ((w_2 \oplus a_1b_2) \oplus a_2b_1)$

Reduced Randomness cost by 3 random numbers

RRM: Example

- **Formal security verification** : INSECURE, check $z_2 \oplus c_2$

$z_0 = x_0y_0 \oplus w_0 \oplus w_1$
$z_1 = x_1y_1 \oplus ((w_0 \oplus x_0y_1) \oplus x_1y_0) \oplus w_2$
$z_2 = x_2y_2 \oplus ((w_1 \oplus x_0y_2) \oplus x_2y_0) \oplus ((w_2 \oplus x_1y_2) \oplus x_2y_1)$

$c_0 = a_0b_0 \oplus w_0 \oplus w_1$
$c_1 = a_1b_1 \oplus ((w_0 \oplus a_0b_1) \oplus a_1b_0) \oplus w_2$
$c_2 = a_2b_2 \oplus ((w_1 \oplus a_0b_2) \oplus a_2b_0) \oplus ((w_2 \oplus a_1b_2) \oplus a_2b_1)$

- Recycling excessively can hurt probing security even between independent gadgets

RRM: Efficient Gadgets

- Search for 2-multiplication, NI gadgets that recycle randomness

Recycling	2-mult ISW gadgets [2]			2-mult BBP gadgets [9]		
	Security Order			Security Order		
	1	2	3	1	2	3
Yes	1	4	8	1	2	6
No	2	6	12	2	4	8

Randomness Cost Table

[2] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks
[9] Sonia Belaïd, Fabrice Benhamouda, Alain Passelègue, Emmanuel Prouff, Adrian Thillard, and Damien Vergnaud. Randomness complexity of private circuits for multiplication.

RRM: Noise Amplification Pitfall

- **Central Limit Theorem:**
  Let m occurrences of random number, emitting leakages $L_1, L_2, \ldots, L_m \sim N(\mu, \sigma)$

- Averaging leakages gives: $L_{avg} = \frac{1}{m}\sum_{i=1}^{m} L_i \sim N(\mu, \frac{\sigma}{\sqrt{m}})$,
  i.e. exploiting the recycling can de-noise the signal [10]

[10] Alberto Battistello, Jean-Sébastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal side-channel attacks and countermeasures on the ISW masking scheme
[11] François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks.

RRM: Noise Amplification Pitfall

- **Central Limit Theorem:**
  Let m occurrences of random number, emitting leakages $L_1, L_2, \ldots, L_m \sim N(\mu, \sigma)$

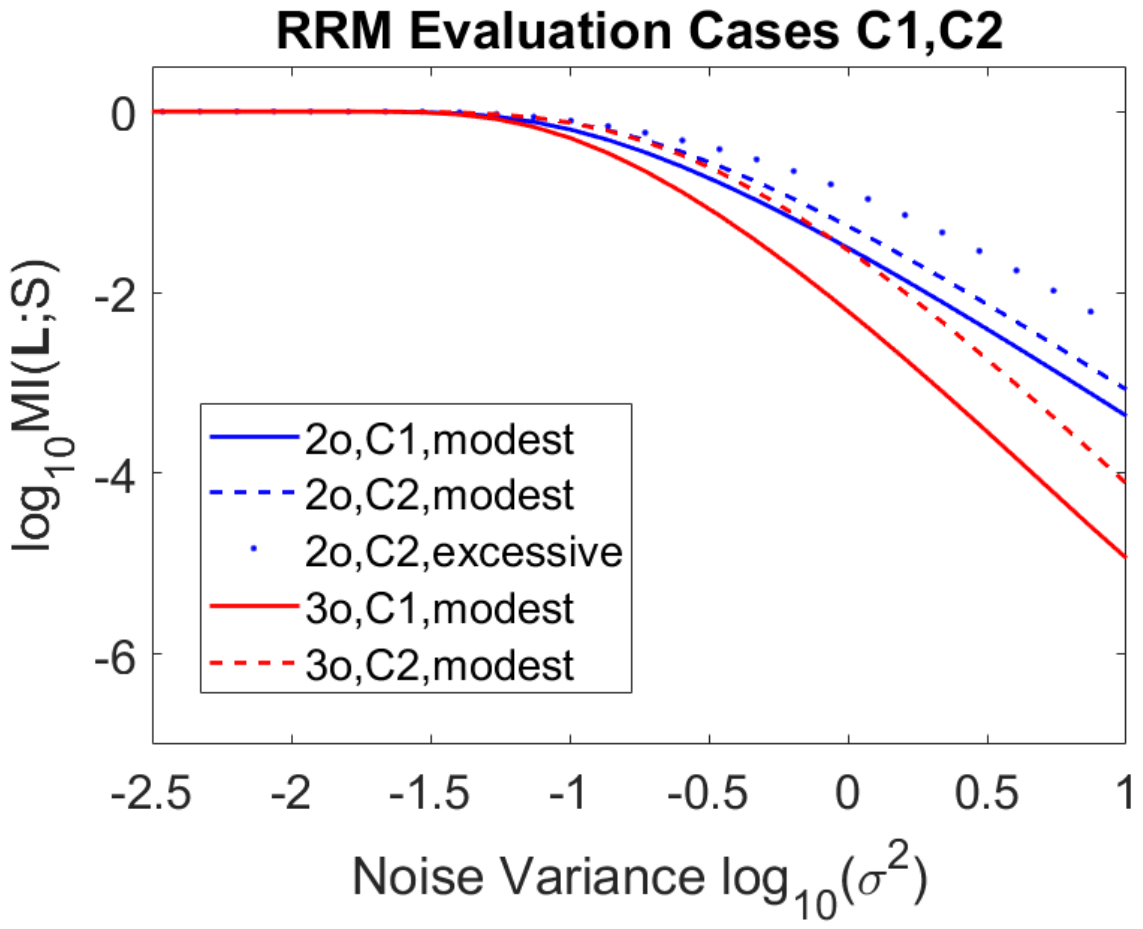
- Averaging leakages gives: $L_{avg} = \frac{1}{m}\sum_{i=1}^{m} L_i \sim N(\mu, \frac{\sigma}{\sqrt{m}})$,
  i.e. exploiting the recycling can de-noise the signal [10]

- Let 2 types of adversaries and we perform an information-theoretic analysis [11]
  C1: naive, doesn't see recycling

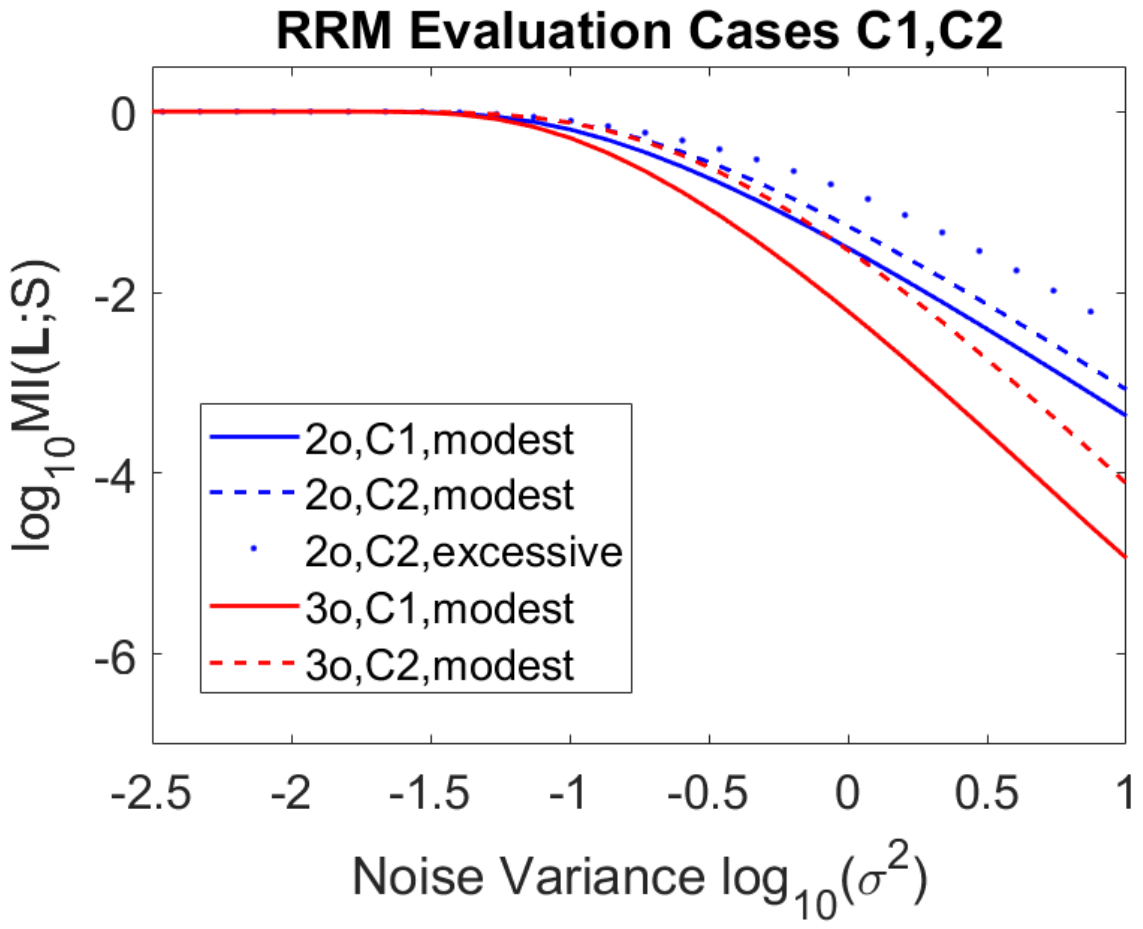
  C2: smart, can see leakages from recycling

[10] Alberto Battistello, Jean-Sébastien Coron, Emmanuel Prouff, and Rina Zeitoun. Horizontal side-channel attacks and countermeasures on the ISW masking scheme
[11] François-Xavier Standaert, Tal Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks.

RRM: Noise Amplification Pitfall

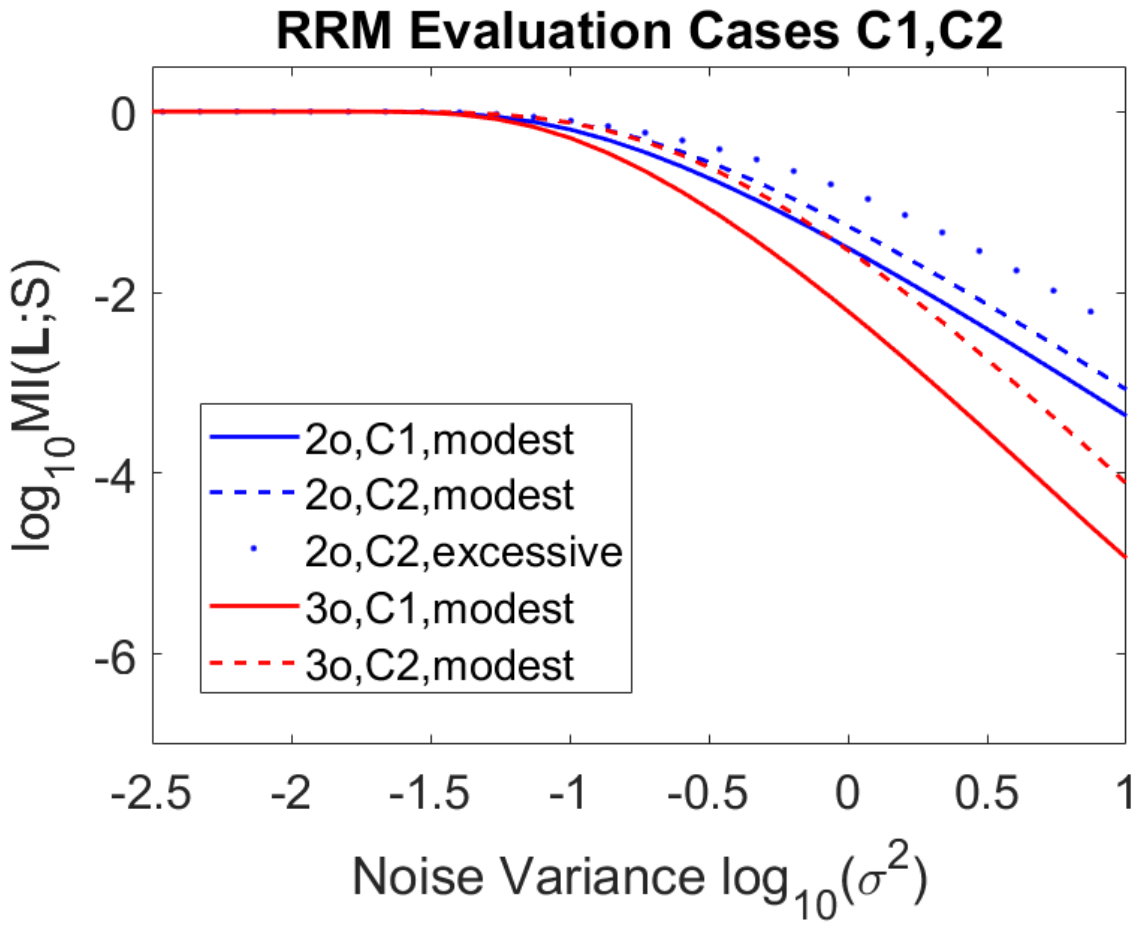


RRM: Noise Amplification Pitfall



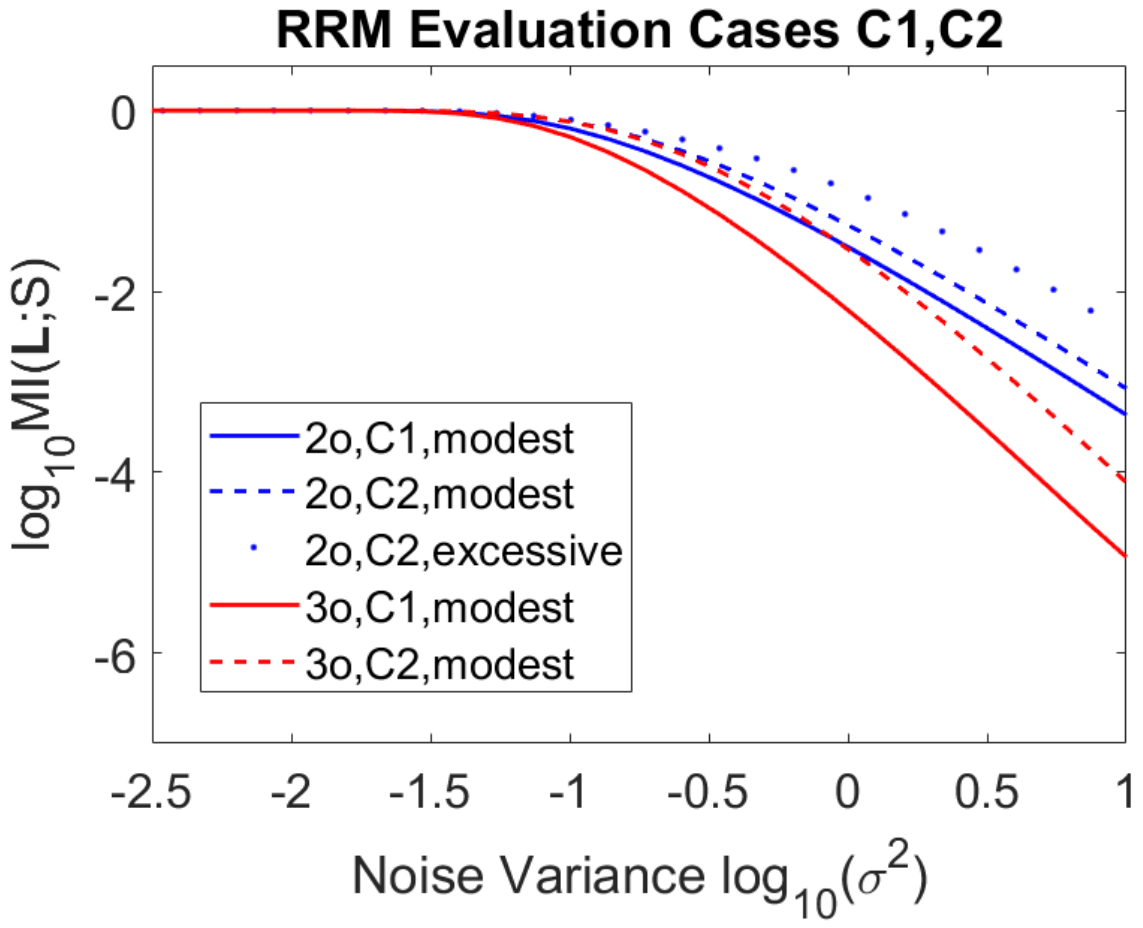
1. The naive adversary C1 cannot take advantage of recycling

RRM: Noise Amplification Pitfall



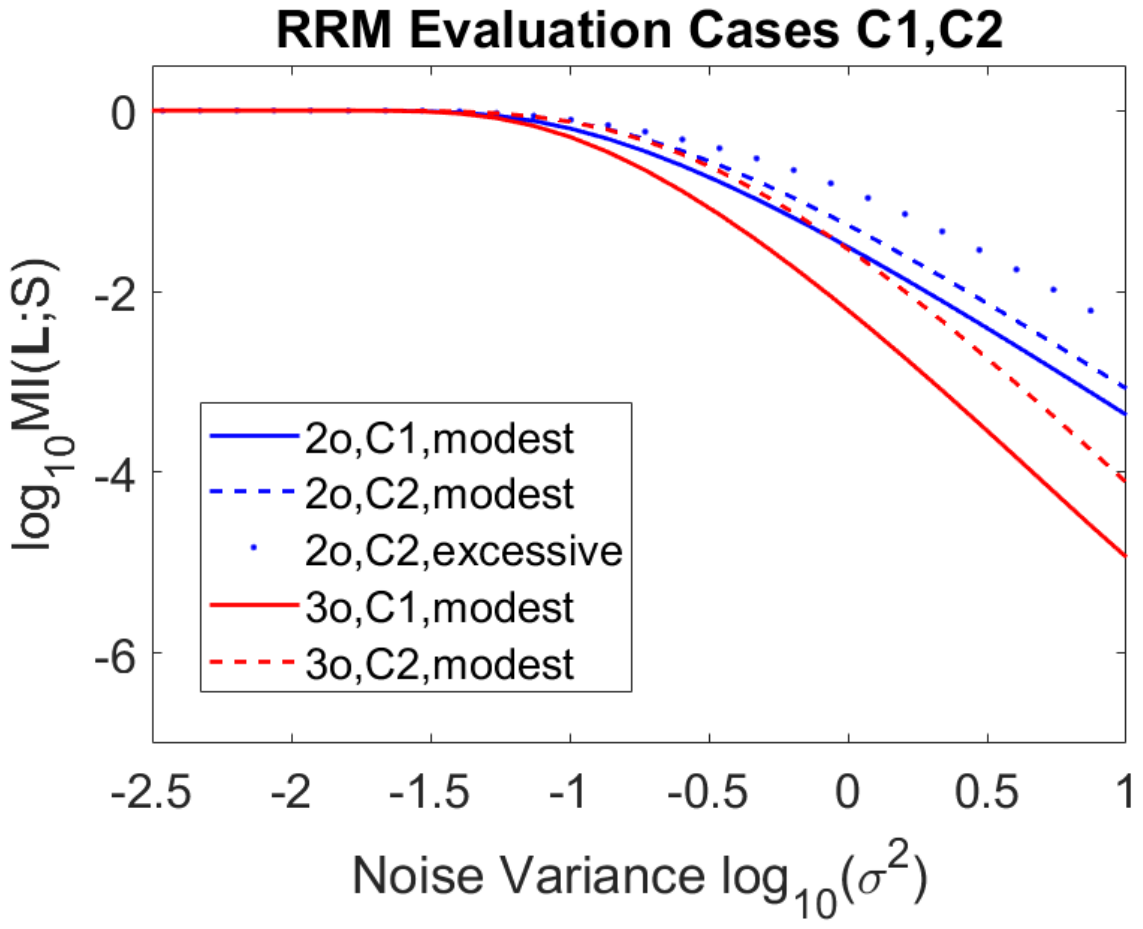
1. The naive adversary C1 cannot take advantage of recycling
2. The smart adversary C2 can shift the curve to the right

RRM: Noise Amplification Pitfall



1. The naive adversary C1 cannot take advantage of recycling
2. The smart adversary C2 can shift the curve to the right
3. Excessive recycling can damage the security

RRM: Noise Amplification Pitfall

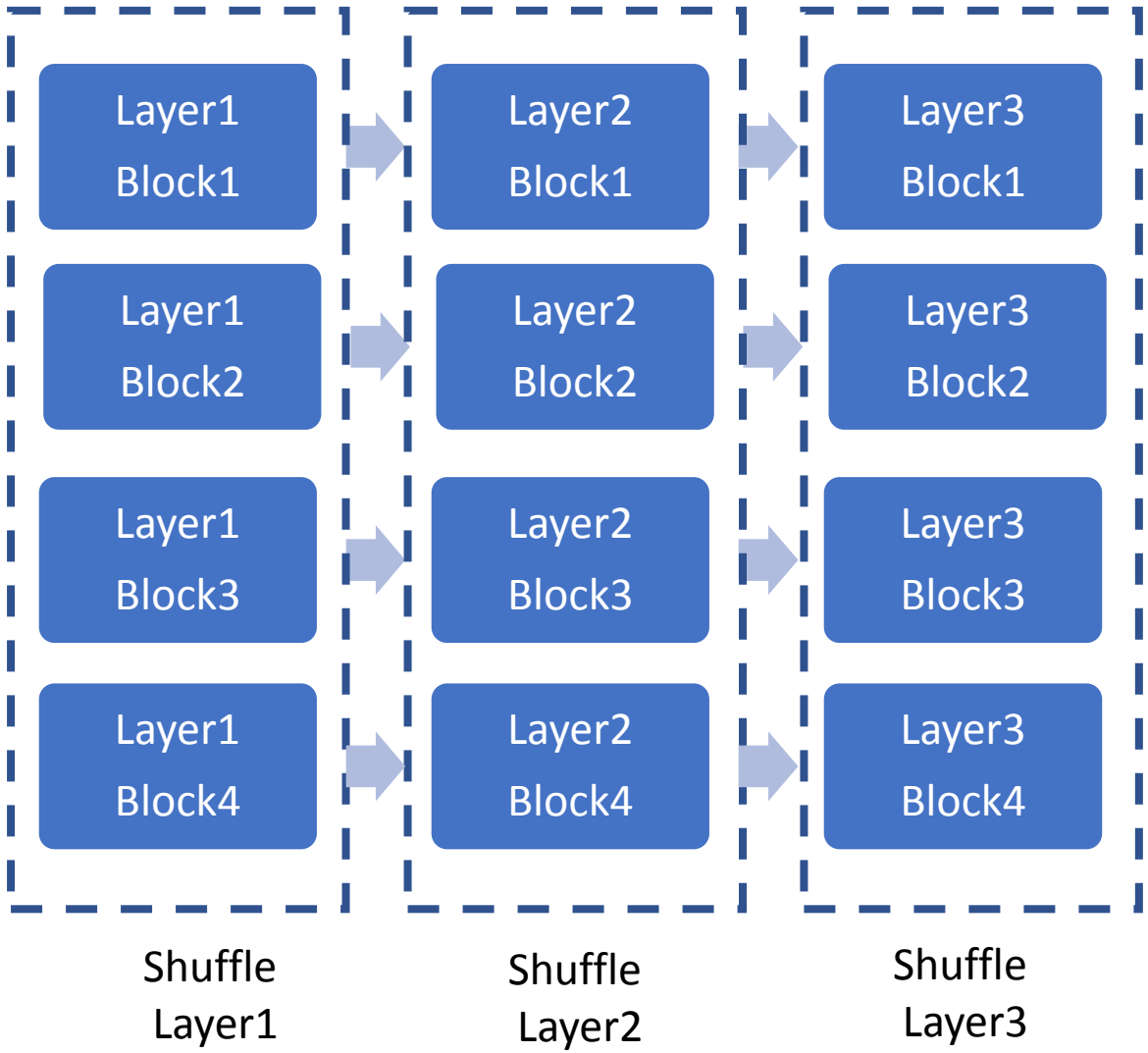


1. The naive adversary C1 cannot take advantage of recycling
2. The smart adversary C2 can shift the curve to the right
3. Excessive recycling can damage the security
4. **RRM is a tradeoff between security and randomness cost**

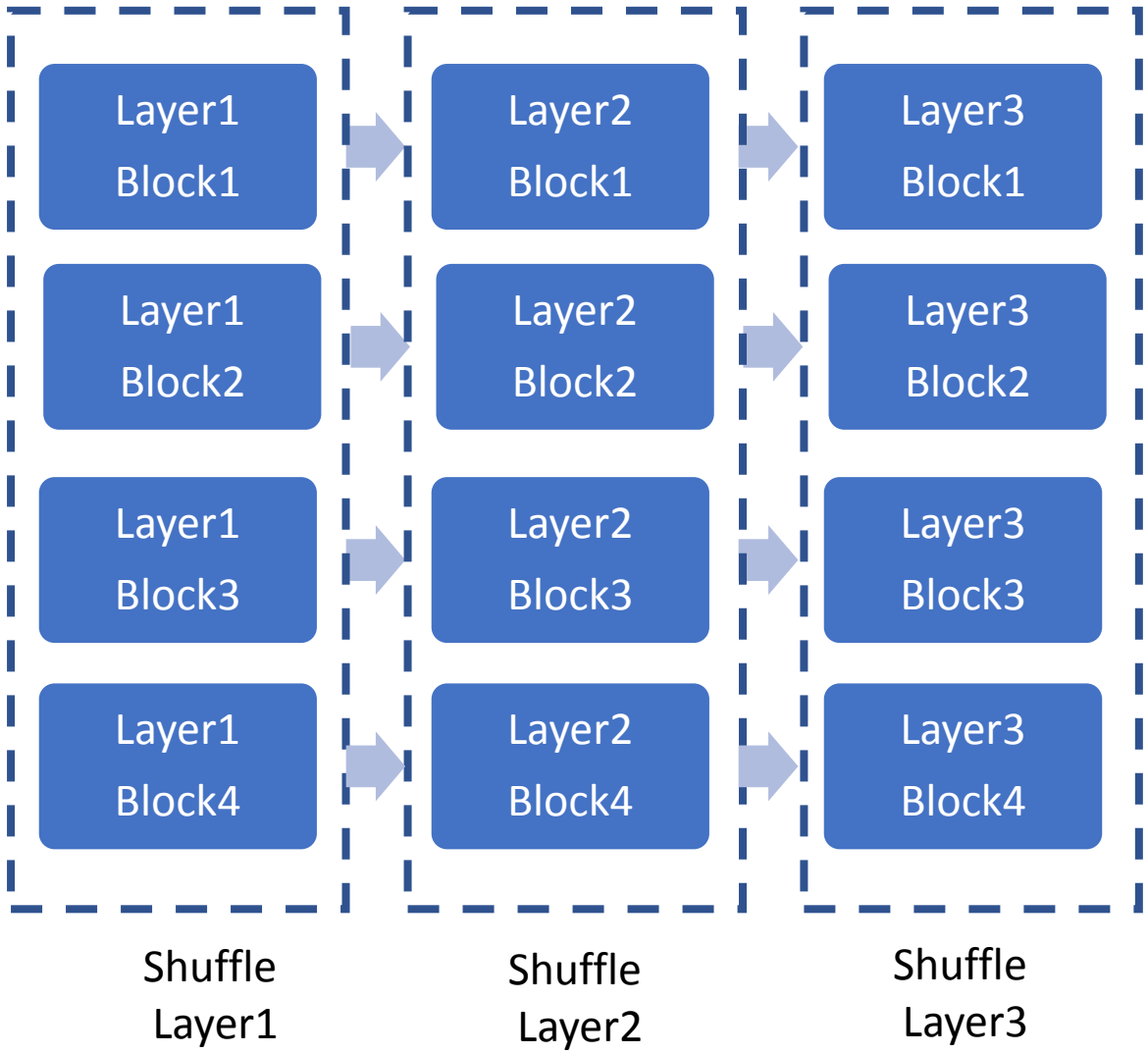
Reduced Randomness Shuffling

Reducing the RNG overhead in shuffling with RRS

RRS: Original Shuffling



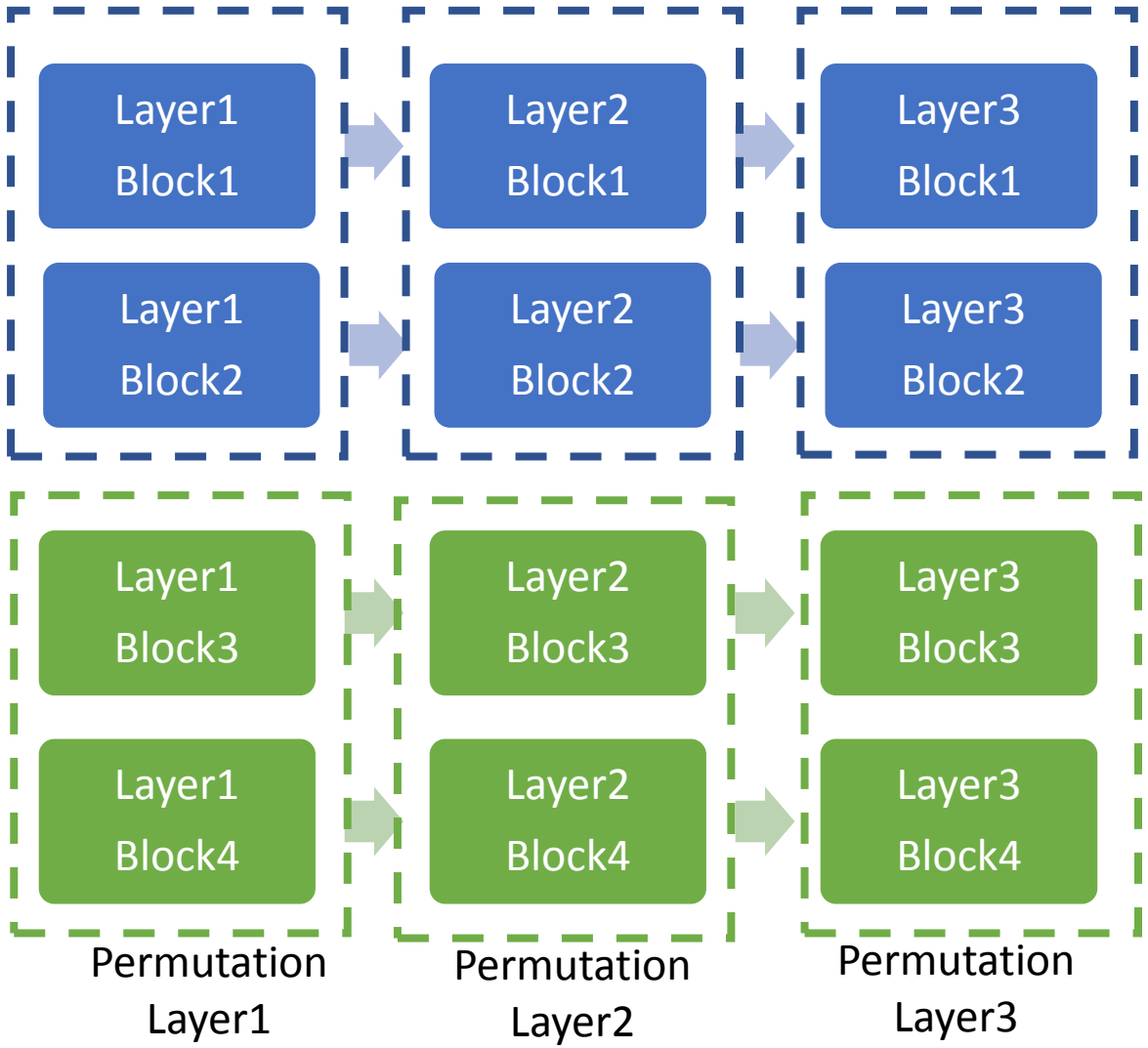
RRS: Original Shuffling



Randomness cost:

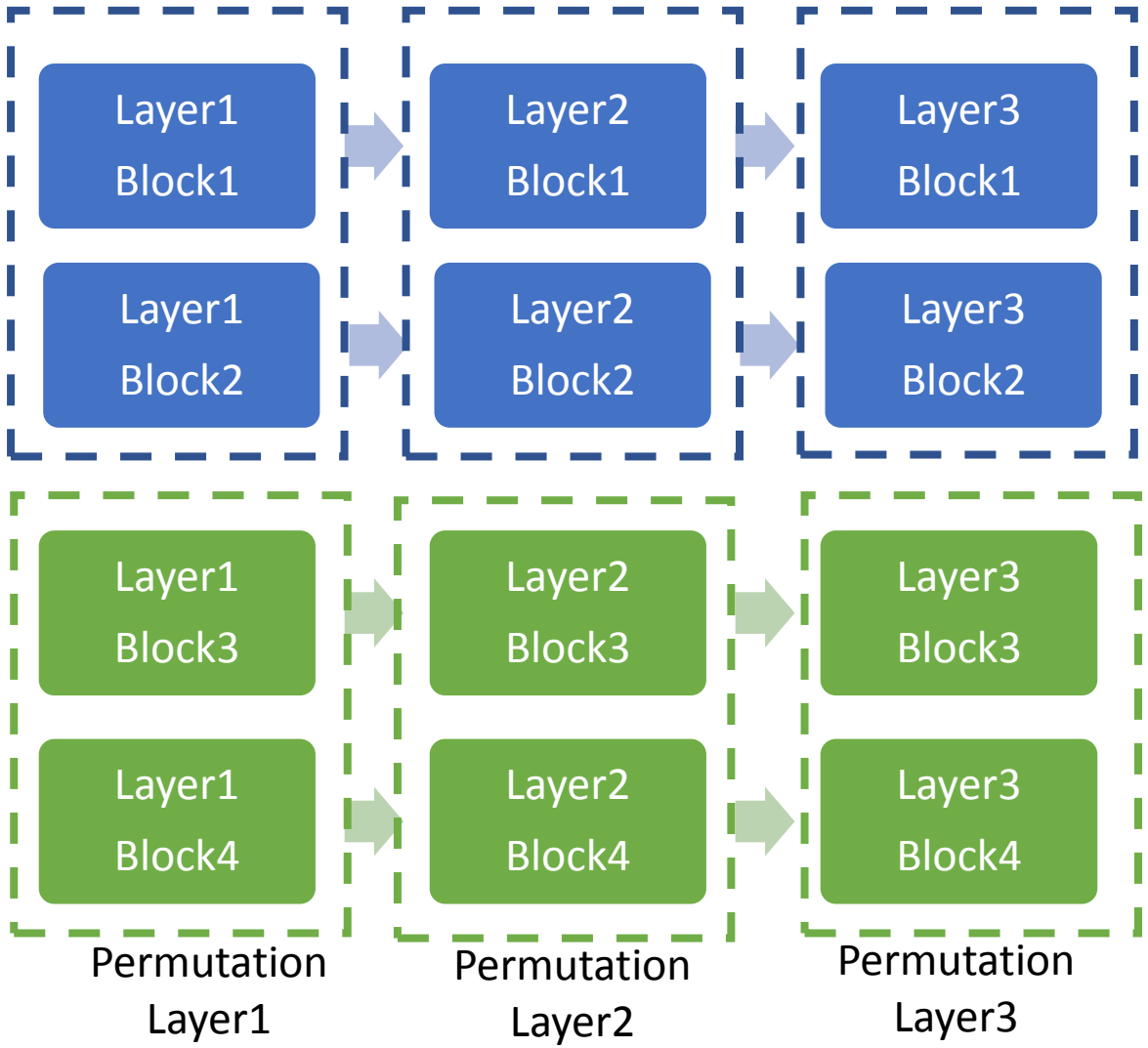
- Shuffle 3 layers independently
- Each layer must shuffle 4 blocks
- $3 * 4 * log_2 4 = 24\ bits$

RRS: Partitioned Shuffling



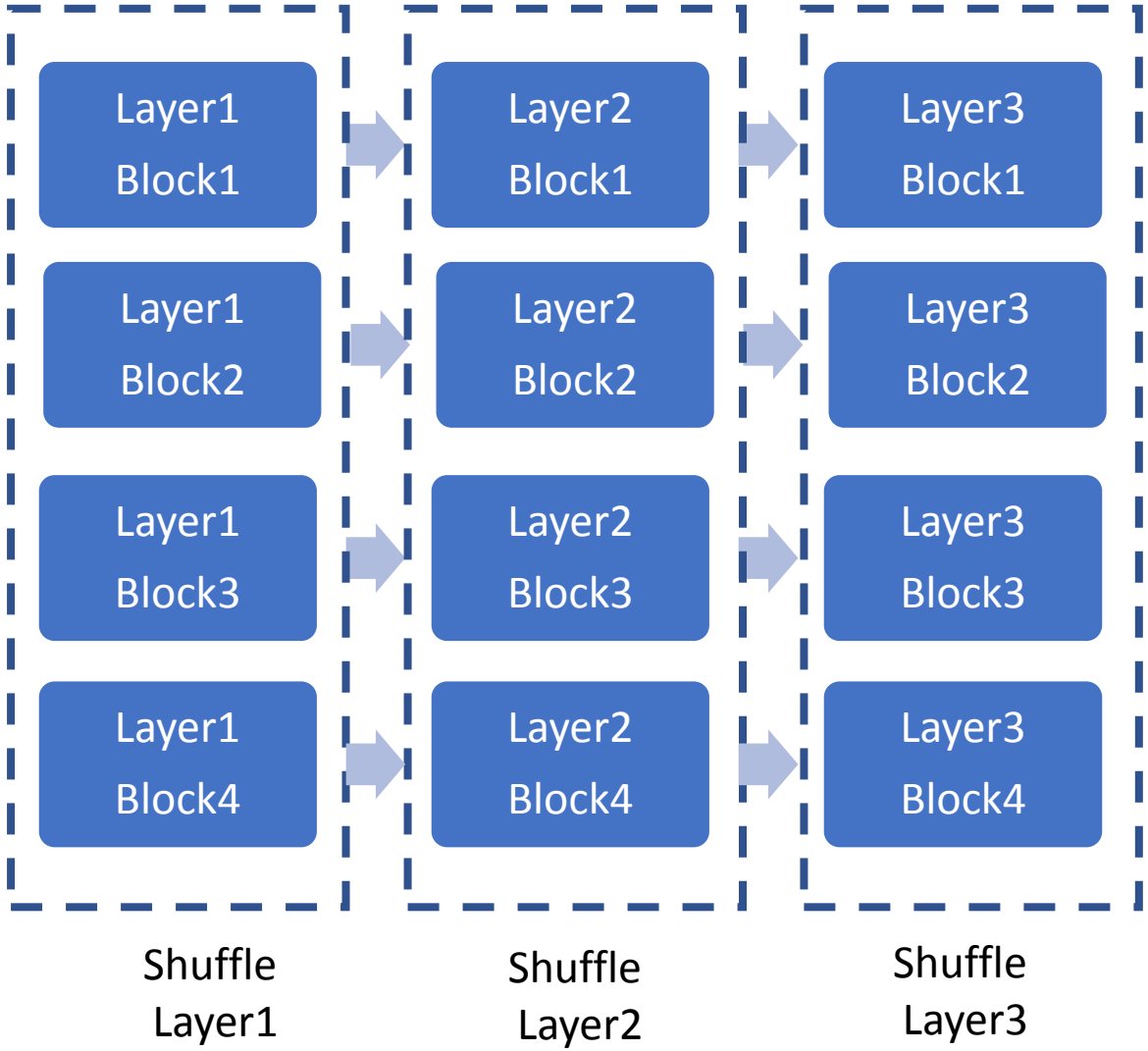
- **Partition** the layers in two, i.e. partition factor $f_p = 2$

RRS: Partitioned Shuffling

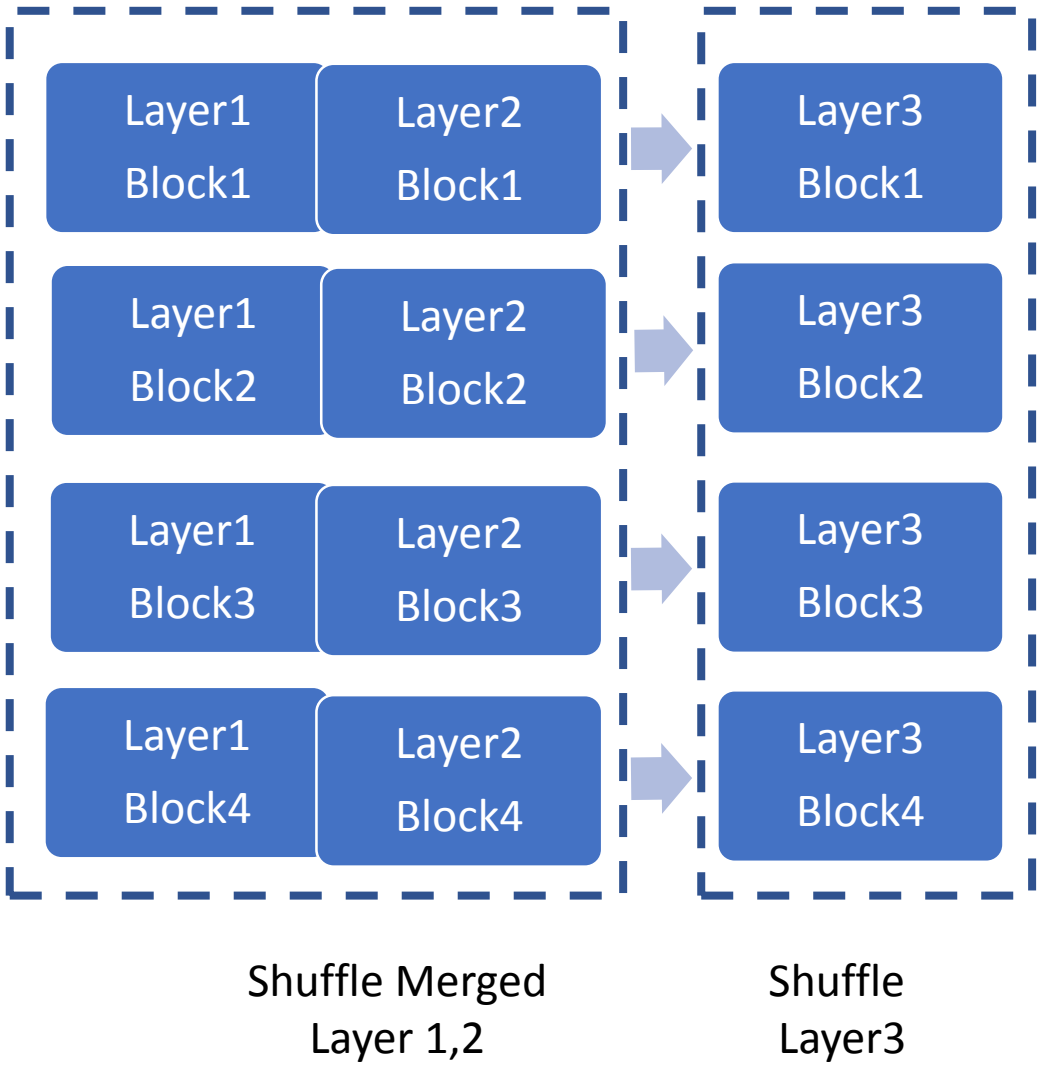


- **Partition** the layers in two, i.e. partition factor $f_p = 2$
- **Randomness cost:**
  - Shuffle 6 layers independently
  - Each partitioned layer shuffles 2 blocks
  - $6 * 2 * log_2 2 = 12\ bits < 24\ bits$

RRS: Original Shuffling

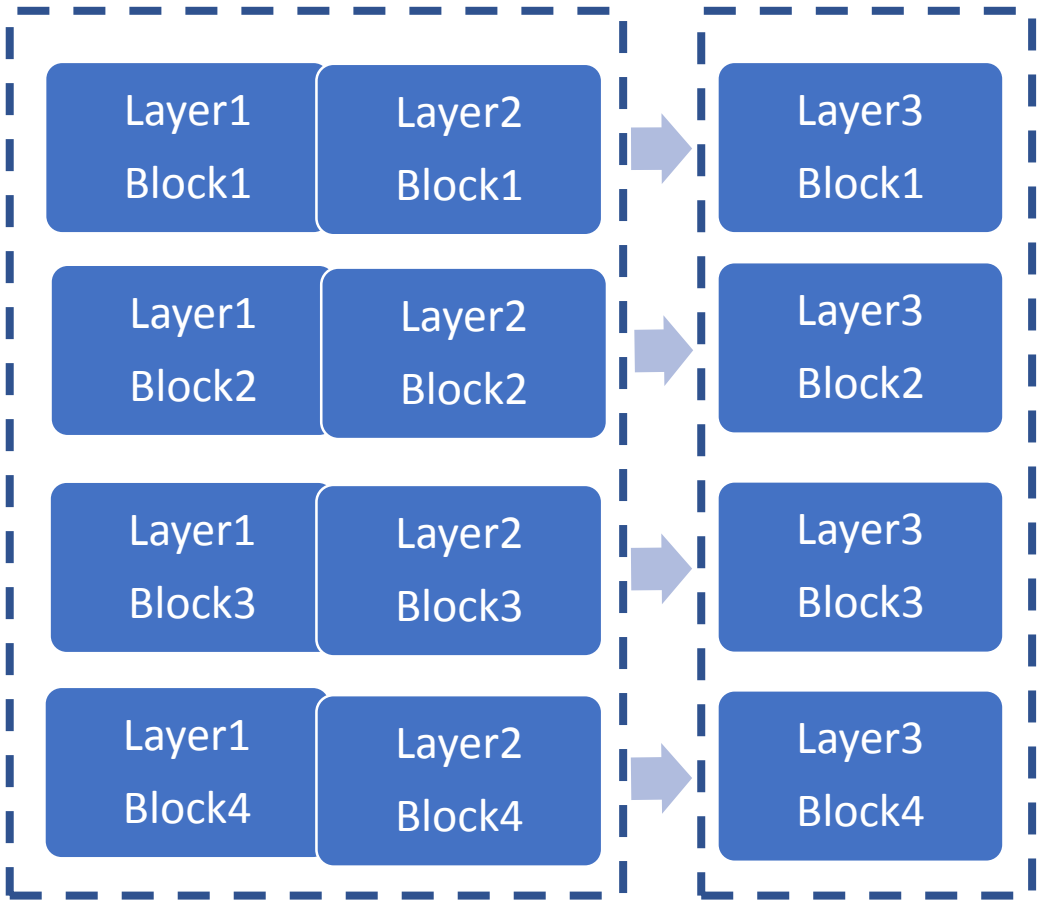


RRS: Merged Shuffling



- **Merge** the 2 layers and shuffle them together, i.e. merge factor $f_m = 2$

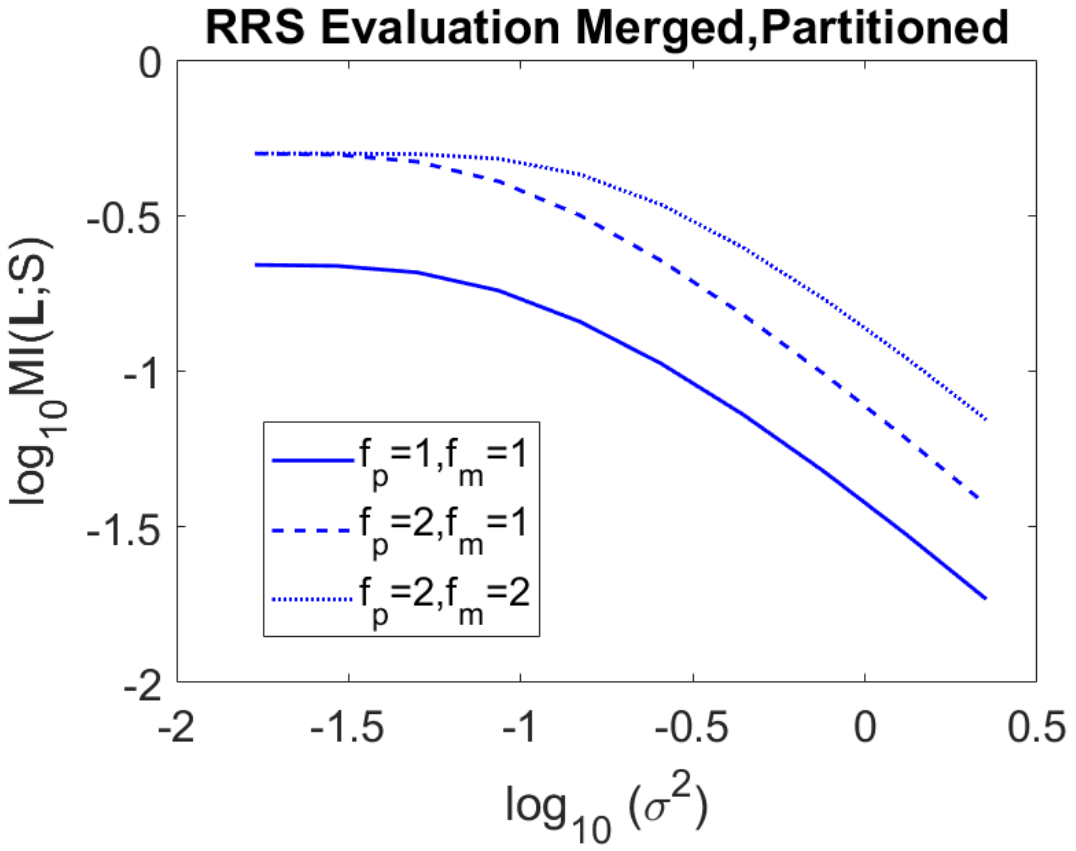
RRS: Merged Shuffling



- **Merge** the 2 layers and shuffle them together, i.e. merge factor $f_m = 2$

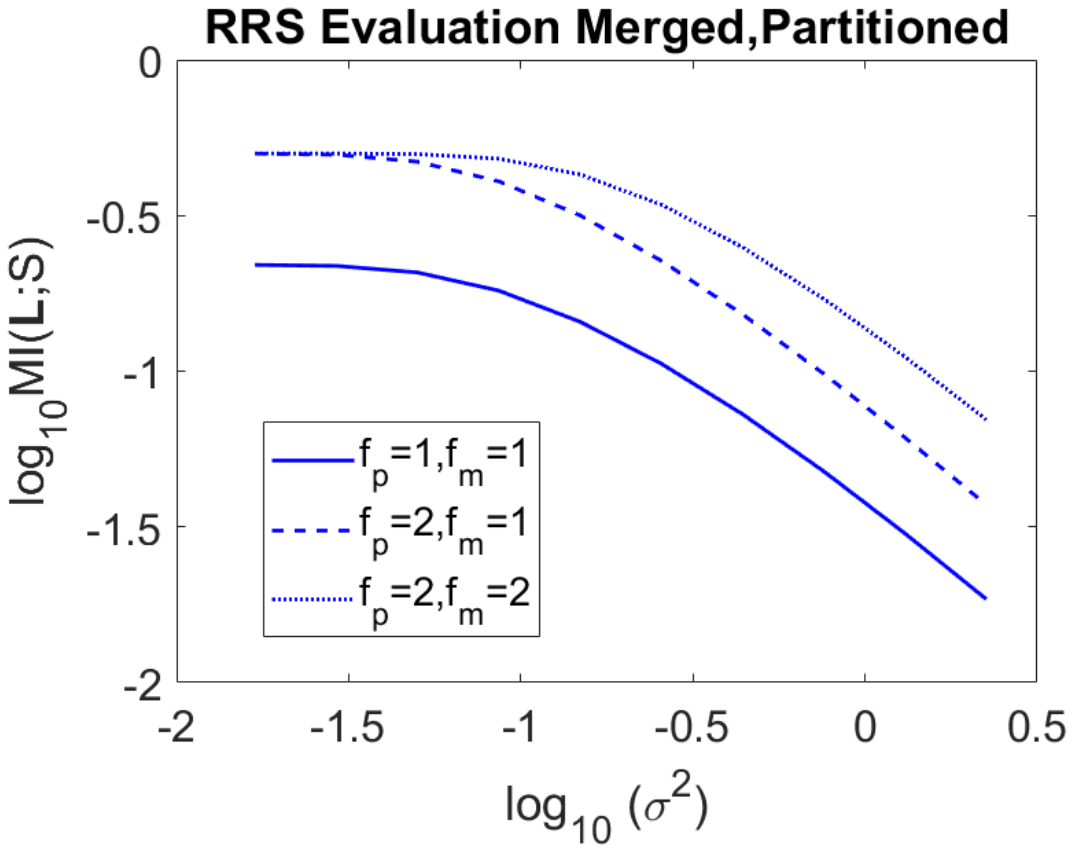
- **Randomness cost:**
  - Shuffle 2 layers:
    - Merged layer 1,2
    - Layer 3
  - Merged layer 1,2 has 4 blocks
  - Non-merged layer 3 has 4 blocks
  - $4 * log_2 4 + 4 * log_2 4 = 16\ bits < 24\ bits$

RRS: Noise Amplification



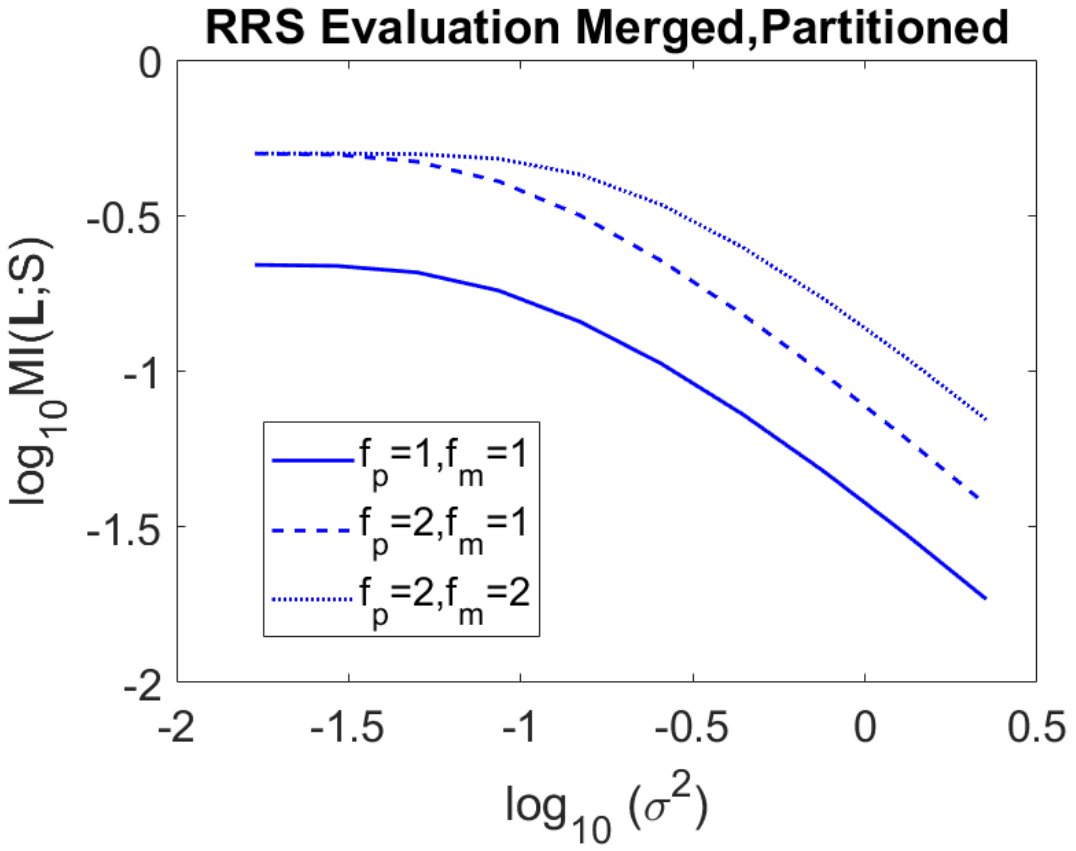
1. Partitioning or merging layers can reduce the randomness cost

RRS: Noise Amplification



1. Partitioning or merging layers can reduce the randomness cost
2. Like RRM it damages the noise amplification stage of shuffling

RRS: Noise Amplification



1. Partitioning or merging layers can reduce the randomness cost
2. Like RRM it damages the noise amplification stage of shuffling
3. **RRS is a tradeoff between security and randomness cost**

Future Directions

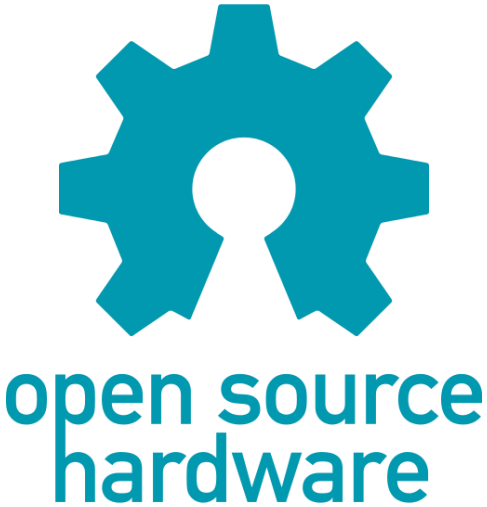
Towards parametric design for side-channel countermeasures

Future Directions: RNG

- We have demonstrated how to reduce the randomness cost in masking and shuffling

Future Directions: RNG

- We have demonstrated how to reduce the randomness cost in masking and shuffling
- Establish the required properties for a generator used in side-channel protection



Future Directions: Parametric Design

- Modern architecture: x^{th}-order masking

Future Directions: Parametric Design

- Modern architecture: x^{th}-order masking
- Parametric architecture: multitude of countermeasure variants to choose from
  e.g. x^{th}-order masking with y recycled random numbers and merged shuffling of z cipher layers

Future Directions: Parametric Design

- Modern architecture: x^{th}-order masking
- Parametric architecture: multitude of countermeasure variants to choose from
  e.g. x^{th}-order masking with y recycled random numbers and merged shuffling of z cipher layers

Athens Tower, 1971

Future Directions: Parametric Design

- Modern architecture: x^{th}-order masking
- Parametric architecture: multitude of countermeasure variants to choose from
  e.g. x^{th}-order masking with y recycled random numbers and merged shuffling of z cipher layers

Athens Tower, 1971

Turning Torso Malmö, 2005