SoK: Pseudorandom Generation for Masked Cryptographic Implementation

Rei Ueno[1], Kazuhiko Minematsu[2] and Naofumi Homma[3]

[1] Kyoto University, Yoshidahommachi, Sakyo-ku, Kyoto-shi, Kyoto 606–8501, Japan
ueno.rei.2e@kyoto-u.ac.jp
NEC Secure System Platform Laboratories, 1753 Shimonumabe, Nakahara, Kawasaki, Kanagawa 211–8666, Japan
k-minematsu@nec.com
[2] Tohoku University, 2–1–1 Katahira, Aoba-ku, Sendai-shi, Miyagi, 980–8577, Japan
naofumi.homma.c8@tohoku.ac.jp

Abstract. This paper investigates pseudorandom generation in the context of masked cryptographic implementation. Although masking and pseudorandom generators (PRGs) have been distinctly studied for a long time, there has been little study on how randomness should be generated in masked implementation. The lack of analysis on mask-bits generators obscures the practical security of masked cryptographic implementation, potentially confusing practitioners (*e.g.*, designer, implementer, and evaluator) about how to realize them securely. This paper provides a novel viewpoint and comprehensive analyses by developing three new models, each of which corresponds to a practical scenario: leakage assessment, experimental evaluation of attack success rate or key lifetime, and deployment. We reveal what PRG properties are required. In particular, we support a long-held belief/folklore through proof and experiment: *for the output of PRG for masking, cryptographic security (i.e., unpredictability) is not necessary.* In addition, we thoroughly investigate the practical security of PRGs within the masking context. We conclude this paper with some recommendations for practitioners and a proposal of leakage-resilient method offering comparative performance.

Keywords: Masking · Pseudorandom Generation · Side-channel attack

1 Introduction

1.1 Background

Masking is a promising countermeasure against side-channel attacks (SCAs) on cryptographic implementation [CJRR99]. Since the introduction of differential power analysis (DPA) *et al.* [KJJ99], both the theoretical and practical aspects of masking have been studied extensively. Assuming that masking is correctly implemented, its formal security has been proven; that is, a polynomial increase of masking order yields an exponential increase of the number of traces for a successful attack [CJRR99, IUH22a, MRS23, BCG+23, UIM+26]. Additionally, numerous tools/methodologies for leakage detection, design automation, and open-source implementations have been developed (*e.g.*, [Rep16, UHMA17, UHMA21, BBC+19, KSM20, SCS+21, SSB+21, KMMS22, BMRT22]). Masked implementation should carefully account for physical defaults—glitch, coupling, and microarchitectural leakage—that may degrade the effective security order due to an unintended cross-share interaction [MPO05, RSVC+11, BGG+14, dCBG+17, DCEM18, FGMDP+18, GMPO20, MKSM22]; however, some masking schemes (*e.g.*, threshold implementation [NRS11, GD23]), masked designs, and tools addressed this issue [SCS+21,

SSB+21, RBN+15, MPW21, ZMM23, ZM24]. These studies make sound masked implementations highly available in practice.

Masked cryptographic implementations always require randomness. At least, randomness is mandatory to decompose the input into shares. Moreover, many practical masked implementations require (large amounts of) fresh randomness during computation for masking gadgets and/or mask refreshing [ISW03, NRS11, KSM21, CPRR14]. Therefore, a pseudorandom generator (PRG) is a fundamental component of masked implementation, as essential as masked circuit[1]. Common PRG choices in previous implementation include a linear feedback shift register (LFSR) [UHA17], (reduced-round) block cipher (BC)-based extendable output function (XOF) [DCRB+16], and stream cipher [DMRB18, CMM+24].

However, their validity remains unclear due to a lack of detailed analysis. Discussions on the required security and properties of PRGs within the masking context are limited, whereas only the requirement that the generated mask should be uniform over the field/ring of the masking and cryptosystem is clearly understood. In masked implementation, the PRG output is not observable to the SCA attacker, while the PRG security has been evaluated generally in the plain model, assuming that the attacker can observe it. Alternatively, only the leakages of PRG are available in SCA. Nevertheless, the leakage resilience of PRGs have not been studied extensively in this context (Only Pietzrak [Pie09] and Cassiers *et al.* [CMM+24] analyzed cases that attacker cannot observe PRG outputs but only observes the PRG leakages). Furthermore, current leakage assessment methodologies do not capture the aspects of mask-bit generation, which implies that there is currently no way to validate the practical security of masked implementations *experimentally* with regard to the mask-bit generator. Establishing a theoretical foundation of masking randomness generation is essential for practice of masked cryptographic implementation.

1.2 Our contributions

In this study, we refer to pseudorandom bits for masking as *mask bits*. We present a systematic analysis of PRGs for mask-bit generation and a leakage-resilient (LR) mask-bit generator based on leveled implementation.

Adversarial modeling. We establish three models—black-box, gray-box, and white-box models—which correspond to practical scenarios: leakage assessment, SCA success rate (SR) and key-lifetime evaluation [UHIM23], and practical deployment, respectively. Our three modelings constitute the first analytical framework to evaluate the mask-bit generation. Our three models are designed such that stronger adversaries strictly include weaker ones, which enables a stage-by-stage analysis of PRG for secure mask-bit generation.

Three-stage analysis. We analyze three models in a stage-by-stage manner to identify the PRG properties to securely generate mask bits. In particular, we support a long-held belief/folklore with a proof and experiment: cryptographic security of PRG (*i.e.*, unpredictability) is unnecessary for secure mask-bit generation[2]. Instead, PRG output should be *high-dimensional uniform equidistribution*, which is easily generated by non-cryptographic PRGs (*e.g.*, LFSR). Additionally, in white-box model (*i.e.*, practical deployment), physical leakage resilience is essential.

New LR mask-bit generator. Based on our analyses, we propose an efficient LR mask-bit generation method that offers comparative performance with existing ones. It follows a concept of *leveled implementation* [BBC+20] for the leakage resilience.

[1] As in literature, we call an implementation of cryptographic function *circuit* for both hardware and software, because they are frequently implemented by logic operations.

[2] For example, in [CMM+24], this proposition was assumed to be true without proof. This has been an important open-problem.

Brief summary. Table 1 summarizes the major PRGs and our proposals for mask-bit generation for practical deployment (See Section 6.4 for details). Existing PRGs are difficult to achieve both practical implementation cost and provable leakage resilience; however, some PRGs may have a sufficiently practical security and performance. In Table 1, "No" and "Limited" in known SCA column mean that a simple power analysis (SPA) would be possible in principle but difficult in practice, as its success would be limited to only specific implementation and good signal-to-noise ratio (SNR). "Practical" indicates that SCA would be practical in major settings.

1.3 Related studies

In [DP08], Dziembowski and Pietrzak presented an LR stream cipher (DP), which achieves indistinguishability of the output from random string under bounded leakage. DP was followed by several LR stream ciphers and PRGs to improve its efficiency [Pie09, YSPY10, DP10]. LR-PRG/stream ciphers may be suitable to mask-bit generation from the security viewpoint, whereas their performance is impractical.

In [ISW03, Section 6], Ishai *et al.* described how to construct a d-probing-secure PRG, which comprising a linear composition of d PRGs. Further, in [IKL+13], Ishai *et al.* presented a *robust PRG* and showed a trivial construction with a linear composition of PRG and an improved construction based on an expander graph. Trivial and improved robust PRGs construct d-probing secure circuit with a cost of $O(d^4)$ and $O(d^{3+\epsilon})$ of true random number generator (TRNG) calls, respectively. In [CGZ20], Coron *et al.* improved the cost of TRNG calls to $O(d^2)$ based on a *multiple PRG* strategy (CGZ). However, the PRG(s) that should be used in the multiple PRG strategy was not discussed, as the PRG(s) are considered black-box and leak-free. Again, the leakage and SCA feasibility on PRGs have rarely been evaluated in the masking context. As we analyze them and reveal how to realize the PRG(s), our results would incorporate CGZ [CGZ20].

In [CMM+24], Cassiers *et al.* discussed hardware masking and claimed a vulnerability of LFSR in generating mask bits, and in [GSH+26], Guinet *et al.* generalized the claim *via* an in-depth analysis. Because the XOR of some bits of LFSR output is fixed to 0 owing to the construction of feedback polynomial, mask bits should be used while avoiding such a mask cancellation. They also demonstrated state-recovery SCA on LFSR for mask-bit generation as a major case study, which was in our white-box model. Yet, neither its generality nor formal security analysis was discussed. Besides, they evaluated the performance of PRGs for hardware masking. As our main focus is the security aspect of mask-bit generation, readers interested in PRG performance are advised to see [CMM+24].

This paper focuses on PRGs that produce mask-bits with no explicit bias, which is a trivial requirement for masked circuits and security proofs. In [Uen26], Ueno investigated how to exploit explicit mask-bit biases and how such biases degrade the SCA security.

1.4 Paper organization

Section 2 introduces the fundamentals of masking. Section 3 describes the modelings of masked implementation and attacks on them. Sections 4 to 6 analyze and discuss SCA on masked implementation in the black-, gray- and white-box models, respectively, and Section 5 includes an experimental validation of our claim. Section 7 provides our recommendations of PRGs to generate mask-bit for practitioners based on our analyses and proposes a novel LR primitive. Finally, Section 8 concludes this study.

Table 1: Major symmetric primitives for mask-bit generation and our proposals (bold) under white-box SCA

Primitive	Typical instance	Cost	Leakage resilience	Known SCA
Plain LFSR	Galois LFSR, Fibonacci LFSR	Low	No	Practical [JD06, BMV07, SJB20, MMS+23, CMM+24]
Improved LFSR	Mersenne twister [MN98], Xorshift [Mar03]	Low	No	Probably limited†
Stream cipher	Trivium [DC06, DCP08]	Low	No	Limited [KDB+22]
LR-PRG/stream cipher	DP [DP08]	Very high	Yes	No (SPA only)
(Reduced-round) block cipher	CTR_DRBG [BK12], counter-based XOF	Medium	No	Practical [Jaf07, DM19a]
	OFB-based XOF	Medium	No	No (SPA only)
One-way hash	Hash_DRBG [BK12], counter-based XOF	Medium	No	Probably practical (same as CTR_DRBG)
	HMAC_DRBG [BK12], OFB-based XOF	Medium	No	Limited [KPP20] (SPA only)
Permutation	Sponge-based XOF [BDH+23, DEMS21]	Medium	Somehow‡	Limited [KPP20] (SPA only)
(LR-)PRF-based XOF	GGM [GGM86], FPS [FPS12], MSJ [MSJ12]	Very high	Yes	No (SPA only)
	Sum of permutation [CLM19], truncated BC [BKR98]	Same as each XOF instance with block cipher		
Robust PRG	IKL+ [IKL+13]	Very high	d-probing	$(d+1)$-probing
Multiple PRG	CGZ [CGZ20]	Low	d-probing*	$(d+1)$-probing
Leveled mask-bit generator	**Ascon + LFSR/stream cipher**	**Low**	**Yes**	**No (SPA only)**
Masked stream cipher/XOF	**Masking without fresh randomness**	**High**	**First order**	**Second-order blind SCA/SPA**

† SCA on LFSR would be applicable to them in principle [JD06, BMV07, SJB20, MMS+23], but it might be difficult to practical parallelized implementations.
‡ Leakage resilience of sponge/duplex-based encryption is discussed in, *e.g.*, [DMP22, BM24, BBC+20].
* Security for masked circuit, but not for distinct PRG (See Section 6.2).

2 Preliminaries

2.1 Notations

Uppercase and lowercase letters (*e.g.*, X and x) denote a random variable and its realization, respectively, unless otherwise defined. $\mathbb{E}_X$ and p_X denote the expectation and probability mass/density function of X, respectively. We denote sequence of n random variables $X_1, X_2, \dots, X_n$ by X^n. We write a keyed function as $F(K, \cdot) = F_K(\cdot)$.

2.2 Masking

Masking is a major countermeasure against SCA [CJRR99, MOP07]. In a masked implementation, each secret variable is decomposed into *shares* over an additive group defining the arithmetic operations of cryptosystem. In a d-th order masking over binary fields, a secret variable z is represented using $d+1$ shares $a_0, a_1, \dots, a_d$ such that

$$z = a_0 \oplus a_1 \oplus \cdots \oplus a_d.$$

This is called *Boolean masking*. For simplicity, we consider Boolean masking, which does not compromise the generality of our claims[3].

Masked linear operation is trivially implemented by duplicating the circuit for each share because for any linear function L and $z = a_0 \oplus a_1 \oplus \cdots \oplus a_d$, it holds that $L(z) = L(a_0) \oplus L(a_1) \oplus \cdots \oplus L(a_d)$. In contrast, it is non-trivial to implement masked nonlinear circuit in a secure manner. For example, based on an Ishai–Sahai–Wagner (ISW) masking scheme [ISW03], a masked AND gate for shared input $a = (a_0, a_1, \dots, a_d)$ and $b = (b_0, b_1, \dots, b_d)$ computes its output $c = (c_0, c_1, \dots, c_d)$ as follows: (i) draw random bits $m_{i,j}$ for $0 \le i < j \le d$ as fresh randomness, (ii) compute intermediate values $m_{j,i}$ as

$$m_{j,i} = \left(m_{i,j} \oplus a_i b_j\right) \oplus a_j b_i,$$

and (iii) compute the output share as

$$c_i = a_i b_i \oplus \bigoplus_{i \neq j} m_{i,j},$$

for each i. The security of this circuit is guaranteed as any d combination of probes does not reveal original (*i.e.*, non-shared) secret values. The ISW masking is followed by many masking schemes, such as (software) TI [NRS11, GD23], DOM [GMK16], UMA [GM17], and gadget-based masking [BBD+16], for improved efficiency, enhanced security, composability, and resolving security degradation by physical defaults causing a cross-share interaction (*e.g.*, glitch [MPO05], coupling [dCBG+17], and microarchitectural leakage [GMPO20, GD23]).

2.3 Probing attack models and notions

Attack models and security notions have been developed to soundly define d-th order masking and verification of masked implementation. In the seminal study of [ISW03], Ishai *et al.* presented the *d-probing model*, representing an attacker using d probes on the circuit. A masked circuit is d-probing secure if an adversary with a black-box access to the

[3] Our claim would hold for other advanced masking schemes, such as multiplicative [GT03, DMRB18], inner-product [DF12, BFGV12], code-based [WMCS20, CGC+21, WCG+22], and polynomial-interpolation maskings [GM11]. Note that, for maskings over non-binary fields/ring (*i.e.*, arithmetic masking [Gou01, DFS16, MMMS23]), the uniformity is defined over their field/ring, which has been used for masking of lattice-based cryptography [OSPG18, BPO+20, CGMZ22b, DHP+22, BGR+21, CGMZ22a, ABC+23b, CGTZ23b, CGTZ23a, ABC+23a, dPPRS23].

circuit can simulate a joint distribution of any d tuple of intermediate values; or simply and intuitively, any combination of d (non-adaptive) probes on the circuit recovers no secret variable [RP10]. The probing model has been extended in multiple aspects for more efficient and secure masking [BBD+15, BBD+16, CS20, RBN+15, FGMDP+18].

These probing-related models/notions are useful for defining a sound d-th order masking and verifying its implementation. Meanwhile, these models/notions do not directly represent realistic non-invasive SCAs, such as DPA, correlation power analysis (CPA) [BCO04], and deep-learning based SCA (DL-SCA) [MPP16, MDP20]. Fortunately, a security reduction to realistic model, namely noisy-leakage model, is known [DDF14]. Thus, masked circuit based on these models/notations are currently promising for d-th order security.

In contrast to masked circuit, mask-bit generation has been rarely discussed. Only a few studies [ISW03, IKL+13, CGZ20] showed constructions of d-probing secure PRG; however, we mention a pitfall of probing model to evaluate SCA security of mask-bit generators in Section 6.2.

2.4 Security proof of masking

The above models/notions define d-th order masking and SCAs. As a sound security scheme, the success rate (SR) on d-th order masked implementation should exponentially decrease by increasing d polynomially because an attacker on d-th order masked implementation is supposed to perform a $(d+1)$-th order SCA. The first proof is shown by Chari *et al.* [CJRR99] for a single-bit DPA, assuming that the measurement noise is Gaussian and sufficiently large. It was followed and extended by several studies [PR13, DFS15, IUH22a, MRS23, BCG+23, UIM+26] to relax and generalize the conditions. Some recent proofs were based on an upper-bound of SR by a mutual information (*i.e.*, leakage amplitude or SNR of side-channel measurement) [dCGRP19], which gave proofs for general SCAs under practical conditions about noise.

These bounds hold assuming that the mask-bit generator is not attacked and the generated mask bits are ideally random. In practice, we should consider how to generate the random numbers with regard to SCA. However, there has been neither experimental strategy (through probing attack notions or leakage assessment) nor theoretical foundation to validate the practical security of masked implementations with a real-world (but non-ideal) PRG. This study solves the gap between theory and practice in mask-bit generation.

3 Modeling of SCA on masking

3.1 Modeling of masked implementation

Let n_f, n_g, n_s, and n_k be the bit lengths of internal state, generated mask bits per state, initial seed, and secret key, respectively. Figure 1 illustrates the abstract of masked implementation of encryption E_K with a PRG, where $\tilde{E}_K^M$ denotes a masked circuit of E_K that satisfies $\tilde{E}_K^M(T) = E_K(T)$ for any mask value M, key K, and plaintext T (C is ciphertext). The PRG consists of a state updating function F and output function G with secret keys K_F and K_G, respectively (keys can be *null*), and it has an n_f-bit internal state S computed from an n_s-bit initial seed. Let S_j be the j-th state, and M_j be the j-th output (*i.e.*, mask value). Given an n_s-bit seed, the PRG initializes the n_f-bit internal state as S_0. The PRG updates the internal state as $S_{j+1} = F_{K_F}(S_j)$, while it generates an n_g-bit mask value as $M_j = G_{K_G}(S_j)$.

Example 3.1 (LFSR). The state updating function F is given by $S_{j+1} = \mathrm{double}(S_j)$, where double is the doubling over $\mathrm{GF}(2^{n_f})$ (n_f is the LFSR bit length), and the output

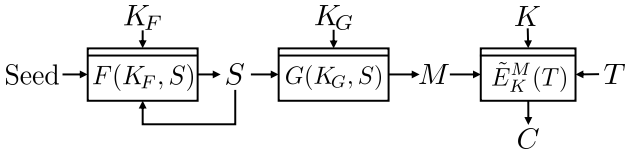


Figure 1: A model for masked implementation comprising masked circuit $\tilde{E}_K^M$ and PRG (F_{K_F}, G_{K_G}).

function G is MSB(S_j) (the most significant bit of S_j). Neither K_F nor K_G is given $(K_F = K_G = null)$.

Example 3.2 (Counter-based XOF). Let H and E_{K_G} be a one-way hash function and BC with a key K_G, respectively. The initial state is given by an n_s-bit seed padded with an $(n_f - n_s)$-bit zero $(n_s < n_f)$. F is given by a counter as $S_{j+1} = S_j + 1$ and G is $H(S_j)$ or $E_{K_G}(S_j)$.

Remark 1 (Bit length of M and unrolling F and G). In practice, the bit length of M (*i.e.*, n_g) depends on the masked circuit implementation and the required number of mask bits per clock cycle. To generate a sufficient amount of mask bits, the PRG should repeatedly call F and G multiple times (or should be implemented in parallel), As in [CMM+24], PRGs can be *unrolled* for higher throughput, implying an expansion of F and G. However, this has little impact on our black- and gray-box analyses because it does not change the outputs essentially, whereas SCA resistance of the implementation would depend on unrolling (related to white-box model).

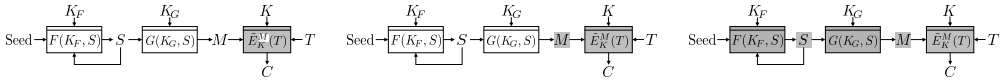
Remark 2 (Correlated randomness). For initial decomposition and refreshing, we need *correlated randomness* [AFL+16]; namely, d' random numbers $R_0, R_1, \ldots, R_{d'}$ satisfying $R_0 \oplus R_1 \oplus \cdots \oplus R_{d'} = 0$, where d' is the number of output shares. The initial decomposition of Z is given as $(A_0, A_1, \ldots, A_d) = (R_0 \oplus Z, R_1, R_2, \ldots, R_d)$ to satisfy the masking correctness: $Z = A_0 \oplus A_1 \oplus \cdots \oplus A_d$. Refreshing is addition of shared zero to achieve uniformity while preserving the secret value. A typical way to generate such correlated randomness is to generate d random numbers $M_1, M_2, \ldots, M_{d'}$ and then to give $R_i = M_i$ for $1 \le i \le d'$ and $R_0 = \bigoplus_{i=1}^{d} M_i$ (If $d = 1$, then $(R_0, R_1) = (M_1, M_1)$) or ring refreshing [BBD+16] realized as $R_i = R_i \oplus R_{i+1 \bmod d'}$.

3.2 Algorithmic leakage

If PRG is inappropriate or flawed as mask-bit generator owing to its algorithmic nature, masked circuit is broken. We identify two such algorithmic leakages, which would involve the gray-box model.

Short frequency. If the attacker knows the frequency of PRG output, the attacker can trivially obtain traces with an identical mask and perform a first-order SCA using them. In practice, the mask-bit frequency should be sufficiently large so that such an attack gets infeasible, even for partial bits.

Brute-force. A first-order SCA trivially works using the mask bits as additional input(s) of the selection function, if the attacker can guess the mask bits with probability 1. Hence, the PRG to generate mask bits should have a sufficiently long bit length of secret key and/or seed, such that M is computationally hard-to-guess. Further, the state updating function should sufficiently diffuse the state to avoid a brute-force of partial state, which is discussed in Section 5.



(a) Black box (b) Gray box (c) White box

Figure 2: Adversarial models for masked cryptographic implementation, where gray parts (boundedly) leak to attacker.

3.3 Adversarial models

Let $\boldsymbol{X} \in \mathcal{X} \subset \mathbb{R}^l$ be side-channel trace, where l denotes the number of sample points. The SCA attacker always knows plaintext T and/or ciphertext C as in common SCAs, observes n traces $\boldsymbol{X}^n$ from the target device, and uses $(T, \boldsymbol{X})^n$ (or $(C, \boldsymbol{X})^n$) for key-recovery.

In SCA on masked implementation, the attacker observes the leakages from the d-th order masked circuit and/or the mask-bit generator. The attacker does not directly observe the mask bits (*i.e.*, PRG outputs) as in common masked implementation. As well, the attacker knows neither the seed nor secret key of PRG. Depending on the availability of leakage, we present three adversarial models of black-, gray-, and white-box (weaker to stronger in this order). Figure 2 displays the three models, where gray parts are components whose leakage is available by the attacker. Note that the black- and gray-box models are designed to analyze the SCA/leakage assessment with *different* order (lower or higher), but their perspectives are definitely different. In this paper, the black-box model discusses the properties required for *PRG outputs* (Section 4), while the gray-box model discusses the *key lifetime* and the *PRG algorithmic leakage* (Section 5).

Black-box model (Figure 2a). The attacker has an access to only the leakage of the masked circuit $\tilde{E}_K^M$, but does not have access to leakages of mask bits M and mask-bit generators. As the relation to practical SCAs (*e.g.*, CPA) or leakage assessment (*e.g.*, TVLA [SM15]), the black-box attackers are defined using a limitation of *leakage and selection function* which the attacker can use; namely, the attacker is allowed to use only a selection and leakage function with *no input of M (or not more than d shares among $A_0, A_1, \ldots, A_d$)*. For example, although the leakage of first-order masked implementation is frequently considered as $\mathrm{HW}(A_0)$ and $\mathrm{HW}(A_1)$ with noise (HW denotes the Hamming weight), the black-box attacker is not allowed to take both A_0 and A_1 (or M) for the leakage and selection function. From the perspective of probing models, the black-box attacker can use one or up to d probe(s) on $\tilde{E}_K^M$, but cannot set the probe(s) on the mask-bit generator (*i.e.*, F, G, and S) nor generated mask M. Recall that a goal of leakage assessment is to verify the non-existence of higher-order leakage than masking order, where the TVLA order (or hypothetical adversary) utilizes up to d-th order leakage (or d probes). Thus, this model corresponds to up-to d-th order TVLA/SCA attacker on d-th order masked implementation and practical leakage assessment, ignoring the physical and algorithmic leakage of mask-bit generator. Hence, if masked circuit has no flaw, the success of black-box SCA and the existence of leakage depend on PRG output; thus, this model is useful for discussing the properties of M (*i.e.*, PRG output) required for masking apart from its leakage.

Gray-box model (Figure 2b). The attacker has an access to the leakages of M in addition to $\tilde{E}_K^M$, but still does not have access to the leakages of F and G. In the probing-related models, the gray-box attack corresponds to $d+1$ probing on the d-th order masked circuit, while the attacker is not allowed to set probe(s) on the mask-bit generator. In practice, this model corresponds to security evaluation against higher-order SCAs, which do not utilize the PRG leakage as in the literature. Probing-related models/notions state that

d-th order masked implementation is not secure against $(d+1)$-probing attacks. However, practical adversary may take any d unboundedly; hence, it is important to consider the difficulty/feasibility of $(d+1)$-th order SCAs on d-th order masked implementation (typically as SR metric [SMY09] and key lifetime [UHIM23]), apart from probing-related models. Additionally, if M leaks due to the algorithmic nature of PRG (*i.e.*, algorithmic leakage), the gray-box attacker may exploit it for mask guessing attack. A concrete example of such mask guessing attack with an algorithmic leakage is described in Section 5.3. Thus, the gray-box model is useful for discussing the masked circuit security quantitatively (*i.e.*, SR and key-lifetime) and the algorithmic leakage.

White-box model (Figure 2c). A white-box attacker has an access to everywhere in the masked implementation, including F and G. The attacker can perform SCA on the mask-bit generator to retrieve its internal state and/or secret key. SCA without information of input nor output is sometimes called *blind SCA* [LDLL14, CR17]. If they are retrieved, the attacker can guess the mask bits with probability 1; accordingly, the security order is degraded to zero as the attacker can perform first order SCA using guessed mask bits. This implies that, in the white-box model, the security of the entire masked implementation depends on the weaker of the PRG or the masked circuit. Therefore, it is important for masked implementation to evaluate physical leakage of PRG and assess its (in)vulnerability in practice. We comprehensively analyze the SCA security of major PRGs in the context of mask-bit generation in Section 6.

Remark 3 (Usage of PRG leakage without state/key recovery). White-box attacker may use leakage for a better guess of each share even if the attacker completely recovers the PRG state/key; however, we focus the PRG state/key recovery because we only consider the security order for simplicity. In other words, even using such a better guess, the key recovery of masked circuit still requires d-th order SCA in Section 5.2 unless a complete state/key recovery.

3.4 Summary of our modeling and analyses: Take away messages

Let $\mathcal{B}$, $\mathcal{G}$, and $\mathcal{W}$ be sets of all black-, gray-, and white-box adversaries, respectively. Our models are designed such that $\mathcal{B} \supset \mathcal{G} \supset \mathcal{W}$; namely, the set of black-box (resp. gray-box) adversaries include the set of gray-box (resp. white-box) adversaries, as in Figure 2. This indicates that the proposition which holds in the black-box (resp. gray-box) model also holds in the gray-box (resp. white-box) model, because gray-box (resp. white-box) adversaries imply black-box (resp. gray-box) adversaries. Our modeling enables a stage-by-stage analysis of PRG properties for secure mask-bit generation.

Black-box model. It is defined such that the attacker does not use leakages of mask values (*i.e.*, part of traces related to mask values), which is to discuss the required property of PRG outputs irrespective to its leakage. This corresponds to the leakage assessment, where evaluator wants to assess the leakage of masked circuit.

Gray-box model. Here, the leakage of mask bits (but not PRG) and potential attacks on masked circuit are discussed. The gray-box attacker performs a higher-order SCA on the masked circuit. In addition, the attacker may exploit the algorithmic leakage if inappropriate mask-bit generator is used, which enables to a mask-guessing first-order SCA (*i.e.*, cancel the masking effect). This corresponds to the evaluation of resistance against higher-order SCA.

White-box model. Here, the PRG leakage is discussed. The attacker performs key/state-recovery SCA on PRG; its success enables first-order SCA with mask guessing. This

Table 2: Summary of three models and analyses

Model	Discussion point(s)	Scenario	Required properties
Black	PRG output	Leakage assessment	Statistical uniformity
Gray	Higher-order SCA and AL of PRG	SR and key-lifetime evaluations	Statistical uniformity AL is unexploitable
White	Physical leakage of PRG	Practical deployment	Statistical uniformity AL is unexploitable Leakage resilience

corresponds to practical deployment, where any parts of implementation can be attacked. As the goal of key/state-recovery SCA on PRG is to mount a first-order mask-guessing SCA as well as gray-box SCA, the white-box SCA utilizes the physical leakage instead of algorithmic leakage if the algorithmic leakage is not exploitable.

Table 2 summarizes the discussion points, corresponding practical scenarios, and PRG properties required for secure mask-bit generation in each model (discussed in Sections 4 to 6), where AL means algorithmic leakage. In all situations, PRG does not need cryptographic security (*i.e.*, unpredictability) for secure mask-bit generation, but a statistical uniformity (precisely, a high-dimensional uniform equidistribution) is sufficient, which can be achieved by, for example, LFSR. Cryptographic PRGs are also sufficient, as unpredictability implies statistical uniformity [Yao82]. In gray- and white-box scenarios, the PRG additionally requires resistance to algorithmic or/and physical leakages to ensure the security.

4 Analysis of black-box adversary

4.1 Our key observation

If the masked circuit has no flaws and is soundly implemented (even with regard to physical defaults), the security order is degraded only if mask bits is inappropriate. As a trivial example, F and G are an identity function (*i.e.*, G outputs a constant value) or the output of G is highly biased far from uniform, in which the attacker would successfully recover the secret key solely by a first-order SCA, implying that the security order is degraded to zero no matter how large d is[4]. By contrast, it has been rarely discussed the properties that are required for mask-bit generation to achieve a secure masked implementation (*e.g.*, statistical randomness, computational unpredictability, or unreproducibility).

In [ISW03], Ishai *et al.* mentioned that random bits are generated by a PRG, but did not mention the property that is required for the security of the ISW private circuit. In addition, in [NRS11], Nikova *et al.* mentioned that the shares and mask should satisfy *uniformity* (*i.e.*, the occurrence probability of any value of shares is an identical constant independent of the secret value) and *statistical independence*, but did not mention whether the mask should be generated randomly. Furthermore, the security proof of masking [DFS15, IUH22a, MRS23, BCG+23, BCGR24] also assumed that shares are uniformly distributed and shares and leakages are independent. Some of these studies would only mention that uniformity and independence are mandatory for the security of masked circuit; however, *(pseudo or true) randomness*, *unpredictability*, or *unreproducibility* sometime has not been utilized to develop masking scheme or prove the security. In fact, unpredictability is not necessary for generating secure mask bits. To prove it, we introduce Lemma 1 as our key observation, which states that the order of traces is meaningless for SCA and leakage assessment.

[4]Note that the black-box attacker cannot leverage the knowledge of PRG output directly, as it is not allowed to take M as the input of selection function. Nevertheless, in this example, the first-order SCA is likely to be successful and leakage assessment would detect a first-order leakage.

Lemma 1. *Given an attack dataset of side-channel traces and plaintexts, the result of common leakage assessment methods and SCAs is invariant under reordering of pairs of traces and plaintexts.*

Proof. This lemma follows from the fact that SCAs and leakage assessments utilize *sufficient statistics* such as empirical means, (co)variance, and related quantities (*e.g.*, likelihood), which considers the order of samples meaningless to represent the random variables. See Appendix A for the full proof. □

4.2 Unnecessity of unpredictability

For a masked implementation, the trace order corresponds to the order of mask values (*i.e.*, how the mask bits are generated), as the j-th execution of masked circuit, from which attacker observes $(T_j, \boldsymbol{X}_j)$ $(1 \le j \le n)$, uses the j-th mask value M_j. Lemma 1 implies that *the order of generated mask bits/value makes no sense for assessments and SCAs if empirical distribution of mask values is identical.* To be concrete, we introduce Theorem 1.

Theorem 1. *Let* $\{ (T_j, \boldsymbol{X}_j, M_{j'}) \mid 1 \le j \le n \}$ *be an attack dataset consisting of* n *plaintexts,* n *traces, and* n *mask values, where the* j*-th observation with* $(T_j, \boldsymbol{X}_j)$ *used mask value* $M_{j'}$*. For any permutation* $\pi_n : \{1, 2, \dots, n\} \to \{1, 2, \dots, n\}$*, SR of SCA using* $\{ (T_j, \boldsymbol{X}_j, M_{\pi_n(j)}) \mid 1 \le j \le n \}$ *is same as that using* $\{ (T_j, \boldsymbol{X}_j, M_j) \mid 1 \le j \le n \}$*.*

Proof. As we said, SCA and leakage assessment utilize sufficient statistic such as empirical means, unbiased empirical (co)variance, and likelihood. Hence, to prove Theorem 1, it suffices to show that the expectation of these sufficient statistics is invariant to π_n. Let f be a measurable function taking $\boldsymbol{X}$ (and T) as input, representing, for example, combining function in higher-order SCA. Its empirical mean with n traces conditioned by mask values $M^n = (M_0, M_1, \dots, M_{n-1})$ is expressed by

$$\mathbb{E}_{T^n, \boldsymbol{X}^n, M^n} \widehat{\mu}[f(T, \boldsymbol{X} \mid M)] = \frac{1}{n} \sum_j \mathbb{E}_{T, \boldsymbol{X}, M} f(T_j, \boldsymbol{X}_j \mid M_j).$$

According to the definition of expectation, for any j, $\mathbb{E}_{T, \boldsymbol{X}, M} f(T_j, \boldsymbol{X}_j \mid M_j)$ is given by

$$\int \sum_{m_j} p_M(m_j) \sum_{t_j} p_{T, \boldsymbol{X}, M}(t_j, \boldsymbol{x}_j \mid m_j) f(t_j, \boldsymbol{x}_j \mid m_j) \, \mathrm{d}\boldsymbol{x}_j.$$

In contrast, when the j-th mask value is given by $M_{\pi_n(j)}$, the empirical mean is expressed by $\mathbb{E}_{T, \boldsymbol{X}, M} f(T_j, \boldsymbol{X}_j \mid M_{\pi_n(j)})$, where m_j is solely replaced with $m_{\pi_n(j)}$. This replacement has no impact on the expectation, as we take the summation about m_j or $m_{\pi_n(j)}$. Thus, it holds $\mathbb{E}_{T, \boldsymbol{X}, M} f(T_j, \boldsymbol{X}_j \mid M_{\pi_n(j)}) = \mathbb{E}_{T, \boldsymbol{X}, M} f(T_j, \boldsymbol{X}_j \mid M_j)$.

According to Lemma 1, the SCA/leakage assessment result is invariant to reordering. Because $\mathbb{E}_{T, \boldsymbol{X}, M} f(T_j, \boldsymbol{X}_j \mid M_{\pi_n(j)}) = \mathbb{E}_{T, \boldsymbol{X}, M} f(T_{\pi_n^{-1}(j)}, \boldsymbol{X}_{\pi_n^{-1}}(j) \mid M_j)$ holds, in expectation, the result is identical for $\{ (T_j, \boldsymbol{X}_j, M_{\pi_n(j)}) \mid 1 \le j \le n \}$ and $\{ (T_j, \boldsymbol{X}_j, M_j) \mid 1 \le j \le n \}$, which implies that the SRs are identical. □

Using Theorem 1, we demonstrate that cryptographic security (*i.e.*, unpredictability) is unnecessary for secure mask-bit generation by constructing an extreme example. Assume that M_j takes a value from $\mathcal{M} = \{0, 1, \dots, n-1\}$ and mask-bit generator generates n different values during n cryptographic executions, where $n = 2^{n_g}$ (n_g is the bit length of M). Let π_n be a random permutation over $\{0, 1, \dots, n-1\}$. Consider two mask-bit generators with an upcounter and a cryptographic PRG using π_n, where the j-th bit mask value associated with (T_j, X_j) is $M_j = j$ and $\pi_n(j)$, respectively. The upcounter is not a cryptographically secure PRG. However, both methods have an identical empirical distribution uniform over $\{0, 1, \dots, n-1\}$. According to Theorem 1, the results of SCA are

identical for both mask-bit generators based on upcounter and cryptographic PRG, as there exists a permutation $j \mapsto \pi_n(j)$ of Theorem 1. Thus, as far as $M^n = (M_1, M_2, \dots, M_n)$ is uniformly distributed over $\mathcal{M}^n$, the SCA SR on masked circuit is identical regardless of the order of mask values.

Because empirical sufficient statistics like mean play an essential role in SCA/assessment algorithms, it would be important for output of mask-bit generator that the *empirical distribution* of mask bits/values is uniform. It is indeed natural, because, for example, TVLA evaluates the difference of empirical means two datasets. If the empirical distribution is identical, then the empirical mean is also identical. In summary, cryptographic security is unnecessary and non-cryptographic PRG is acceptable for mask-bit generation because the order of generated mask bits/value does not make a difference. In Section 5.5, we demonstrate an experimental second-order SCA on masked implementation with different mask-bit generators, which validates our claim in all the black-, gray-, and white-box settings; that is, the order of generated mask values has little impact on SCA result *on average* (*i.e.*, SR).

Remark 4 (Upcounter renders masked circuit insecure *in practice*). We showed the case of upcounter as a counter-intuitive example for a theoretical explanation. In practice, the empirical distribution of generated values is not uniform over $\{0,1\}^{n_g}$ because 2^{n_g} (*i.e.*, frequency) is usually far greater than the number of traces n, implying that upper bits in M are hardly changed. This causes an algorithmic leakage exploitable by a gray-box attacker discussed in Section 5.3.

4.3 PRGs outputting secure mask bits

For generating statistical randomness yielding a uniform empirical distribution, we should consider methods that generate a high-dimensional uniform equidistribution. Namely, for secure mask-bit generation, each bit/value of M should be unbiased (*i.e.*, uniformly equidistributed in one-dimension) as well as M should have neither autocorrelation nor bias in multiple bits; namely, it should be high-dimensional uniform equidistribution. Although it is impossible to generate *truly* uniform equidistribution in any finite dimension (even for cryptographic PRG), a fairly high-dimensional uniform equidistribution PRG has been commonly used, which would be sufficient for secure mask-bit generation. An example would be improved LFSRs like Mersenne twister [MN98], output of which is proven equidistributed uniformly over k-dimensional space with a 32-bit accuracy for all $1 \le k \le 623$. In addition, the outputs of an n_g-bit plain LFSR can be equidistributed uniformly in at most $(n_g - 1)$-dimension with one-bit accuracy, which implies that LFSRs with a sufficiently large n_g would be sufficient for mask-bit generator. Cryptographic PRGs (*e.g.*, stream cipher and XOF) would be also appropriate, as unpredictability implies a high-dimensional uniform equidistribution. In contrast, the linear congruential generator in the standard library would be inappropriate, because it is not equidistributed uniformly in higher than one-dimension.

Relation to [CMM+24, GSH+26]. Cassiers *et al.* claimed that an LFSR-based mask-bit generator can lead to an insecurity owing to its linearity and Guinet *et al.* generalized and clarified the claim. Namely, according to the LFSR feedback polynomial, there exist mask bits that satisfy $M_{j_1} \oplus M_{j_2} \oplus \cdots \oplus M_{j_c} = 0$ with probability 1, where M_{j_b} is the j_b-th bit of generated mask. If $M_{j_1}, M_{j_2}, \dots, M_{j_c}$ are XORed to obtain a correlated pseudorandomness, the resultant bit is fixed to zero, leading to an exploitable leakage. This implies that, in this case, the mask bits are not uniformly equidistributed even in one dimension owing to the construction of refreshing gadgets and defining polynomial. To ensure the security, a correlated pseudorandomness should be uniformly equidistributed with a care of linearity.

5 Analysis of gray-box adversary

5.1 Notations and graphical model

We introduce notations with a graphical model, which is an extension of a communication channel model representing SCA on masked implementation in [IUH22a]. Figure 3 shows the graphical model, where gray random variables are plaintext or leakage observable by the attacker. Note that this model does not represent an actual implementation; it represents relation among random variables (for example, Z does not appear in actual implementation). Here, random variables are defined as follows.

- $T^n = (T_1, T_2, \dots, T_n)$: (Partial) plaintext or ciphertext observable to attacker, where n is the number of traces.
- K: (Partial) secret key of masked circuit.
- $Z^n = (Z_1, Z_2, \dots, Z_n)$: Secret intermediate value calculated from K and T using a selection function. We may write $Z^{(k)}$ to specify the value of k. For example, $Z^{(k)} = \mathrm{Sbox}(Z \oplus k)$ in some typical SCAs on AES.
- $M^n = (M_1, M_2, \dots, M_n)$: Generated mask bits.
- A_i $(0 \le i \le d)$: Shares of Z, where d is the order.
- $\boldsymbol{L}_i$ $(0 \le i \le d)$: Leakage of A_i. We assume that leakages are independent, as in previous literature.
- $\boldsymbol{X}^n = (\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_n)$: Side-channel trace, each of which comprises leakages as $\boldsymbol{X} = (\boldsymbol{L}_i)_{i=0}^d$ or $\boldsymbol{X} = \sum_{i=0}^d \boldsymbol{L}_i$.
- $\hat{K}$: Estimation of (partial) secret key by SCA.

This graphical model represents that the secret variable is function of secret key K and plaintext/ciphertext T and side-channel trace $\boldsymbol{X}$ depends on Z. The plate of n represents that n traces are available for the attacker. In a masked implementation, Z is decomposed into shares $A_0, A_1, \dots, A_d$. Here, $Z \to A_0 \leftarrow A^d$ holds because the share decomposition is regarded as $(A_0, A_1, \dots, A_d) = (Z \oplus R_0, R_1, \dots, R_d)$, where $R_0, R_1, \dots, R_d$ are correlated randomness, and $A_i \to A_0$ represents the randomness correlation as $R_0 = R_1 \oplus R_2 \oplus \cdots \oplus R_d$ holds. Note that the graphical model represents the relation among random variables, but does not represent any actual implementation. In other words, this model is valid regardless of actual implementation as long as the relation of random variables holds, which maintains the generality of the model. Leakages $\boldsymbol{L}_0, \boldsymbol{L}_1, \dots, \boldsymbol{L}_d$ (where $\boldsymbol{X} = (\boldsymbol{L}_0, \boldsymbol{L}_1, \dots, \boldsymbol{L}_d)$) correspond to $A_0, A_1, \dots, A_d$, respectively, and are assumed independent. Attacker estimates K as $\hat{K}$ from T^n and $\boldsymbol{X}^n$.

5.2 Optimal gray-box SCA

To discuss the capability and limitation of gray-box adversary, we introduce *optimal distinguisher* [HRG14, IUH22b]. Let $(T^n, \boldsymbol{X}^n)$ be an attack dataset. Distinguisher $D(T^n, \boldsymbol{X}^n)$ is a function that returns an estimated secret key $\hat{K}$ using the attack dataset. The SR of n-trace SCA is defined as $\mathrm{SR}_n(D) = \mathbb{E}_K \Pr(K = D(T^n, \boldsymbol{X}^n))$ [IUH22b], where Pr is the probability measure. Optimal distinguisher maximizes the SR.

Definition 1 (Optimal distinguisher [HRG14, IUH22b])**.** A distinguisher D_{opt} is optimal if its SR is highest; namely, $\mathrm{SR}_n(D_{\mathrm{opt}}) = \sup_D \mathrm{SR}_n(D)$.

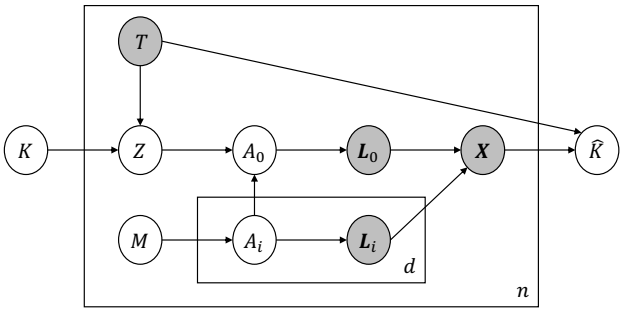


Figure 3: Graphical model representing gray-box SCA.

We introduce an optimal distinguisher used in DL-SCA[5]. Namely, a distinguisher using log-likelihood of secret key, defined as

$$D_{\mathrm{opt}}(T^n, \boldsymbol{X}^n) = \arg\max_{k \in \mathcal{K}} \sum_j \log p_{Z|\boldsymbol{X}}(Z_j^{(k)} \mid \boldsymbol{X}_j), \tag{1}$$

is proven to be optimal [IUH21], where $Z_j^{(k)}$ is the secret intermediate value calculated from k and T_j (*i.e.*, output of selection function).

The optimal distinguisher in Equation (1) is applicable to masked implementation; however, it does not represent the features and leakages of masking in a specific way. According to [IUH22a, Equation (2)], for d-th order masked implementation, $p_{Z|\boldsymbol{X}}$ can be represented using a convolution of conditional probability distributions of share given its leakage; namely, it holds that

$$p_{Z|\boldsymbol{X}}(z \mid \boldsymbol{X}) = \sum_{z = a_0 \oplus a_1 \oplus \cdots \oplus a_d} \prod_{i=0}^{d} p_{A_i|\boldsymbol{L}_i}(a_i \mid \boldsymbol{L}_i), \tag{2}$$

where $p_{A_i|\boldsymbol{L}_i}$ is the conditional probability distribution of j-th share A_j given its leakage $\boldsymbol{L}_i$. Equation (2) represents d *convolution* of probability distributions[6], which is essential to achieve an exponential security of masking [IUH22a]. As well, in Figure 3, we have Theorem 2 about an optimal distinguisher on masked implementation.

Theorem 2 (Optimal SCA on masked implementation)**.** *Consider the graphical model in Figure 3. A distinguisher defined as*

$$D_{\mathrm{opt}}(T^n, \boldsymbol{X}^n) = \arg\max_{k \in \mathcal{K}} \sum_j \log \sum_{Z_j^{(k)} = \bigoplus_i A_{i,j}} \prod_{i=0}^{d} p_{A_i|\boldsymbol{L}_i}(A_{i,j} \mid \boldsymbol{L}_{i,j})$$

is optimal, where $a_{i,j}$ and $\boldsymbol{L}_{i,j}$ denote the i-th share of j-th execution and its leakage, respectively.

Proof. See Appendix B. □

This optimal distinguisher is invariant under dataset reordering owing to commutativity of addition as same as Lemma 1. Therefore, a non-cryptographic PRG including LFSRs

[5]Many constructions of optimal distinguisher are known [IUH22b]. For example, the template attack [CRR02], which uses $p_{\boldsymbol{X}|T,K}$, is also optimal [HRG14].

[6]For probability distributions on a finite discrete set, a convolution with a uniform distribution results in a uniform distribution. This implies that the optimal distinguisher in Theorem 2 is not successful in key recovery and has an SR of random guess if at least one leakage of share is not observable (namely, $p_{A_i|\boldsymbol{L}_i} = p_{A_i}$ for some i), which corresponds to upto d-probing on d-th order masking (*i.e.*, black-box attack).

would be sufficient to evaluate a concrete security (*e.g.*, SR and key-lifetme) of d-th order masked circuit against best possible gray-box SCA (*i.e.*, $(d+1)$-th order SCA), as far as its outputs are uniformly equidistributed.

5.3 Consideration of algorithmic leakage (AL)

An optimal distinguisher is promising for gray-box attacker. Here, if the mask-bit generator is inappropriate in its algorithmic nature, the attacker can estimate the values of M. Such an AL enables more efficient order-degraded SCAs like biased mask attack and mask guessing attack, while black-box attacker is not allowed to use it. Thus, the gray-box model is also useful for discussing the AL.

In fact, the upcounter mentioned in Section 4.2 is insecure in the gray-box model because of ALs, although the example provides the black-box security. First, the least significant bits (LSBs) of M_i has a very low frequency if using an upcounter. The gray-box attacker trivially knows collisions in LSBs; that is, for any j, it holds $\mathrm{LSB}(M_{2j}) = \mathrm{LSB}(M_{2(j+1)})$ with probability 1. This implies that a subset of M_j's with only even or odd indices (*i.e.*, $\{ M_{2j} \mid 1 \le j \le \lceil n/2 \rceil \}$ or $\{ M_{2j-1} \mid 1 \le j \le \lceil n/2 \rceil \}$) is not uniform, and the d-th order masked implementation is not d-th order secure against SCAs using the corresponding subset. Such an SCA using a subset can also exploit MSBs, as the MSB is not changed during a consecutive $n/2$ M_j's. Second, both updating and output functions (*i.e.*, F and G) have little diffusion effect. Suppose that the eight-bit LSBs of M is used for initial decomposition of first Sbox of first-order masked AES. Even if the initial seed is given at random, an eight-bit guess of the LSBs yields a hypothetical estimation of eight-bit LSBs of M_j for all j. Hence, the AL would enable a mask-guessing first-order SCA on masked AES, regardless of the bit length of seed.

Quantitatively, AL is unexploitable if the computational cost of mask-guessing grows exponentially by increase of number of traces (up to seed bit length). The computational cost of mask guessing of upcounter-based mask-bits generator is constant but does not increase by the number of traces. In contrast, standard PRGs including LFSR (with appropriate parameter) offer the exponential growth of the computational cost in mask guessing.

To formulate and guarantee such an exponential growth, a high-dimensional uniform equidistribution is useful again owing to its definition. High-dimensionally uniformly equidistributed M is guaranteed not to have partial bits with low frequency, namely, the first AL. Moreover, we must require at least n_{dim}-bit observation of outputs to predict the next outputs of an n_{dim}-dimensional uniform equidistribution, even if the PRG is not keyed. Because the PRG output is not observable, we need at least an n_{dim}-bit guess for a hypothetical estimation of (partial bits of) M_j for all j (when $n_{\mathrm{total}} > n_{\mathrm{dim}}$, where n_{total} is the required mask-bit length in total). The second AL gets computationally unavailable if n_{dim} is sufficiently large. Thus, non-cryptographic PRGs including LFSR ($n_{\mathrm{dim}} = n_g$) would be acceptable as mask-bit generator in the gray-box model as far as its output is a uniform equidistribution in fairly high dimensions.

5.4 Sophisticated gray-box attacks

Biased mask attack. Although other types of higher-order SCAs that utilize leakage of mask bits in a different manner have been known, they cannot achieve higher performance than the optimal distinguisher. For example, a biased mask attack [MOP07] selects traces with highly-biased or fixed mask bits, which is estimated by $\boldsymbol{L}_j$ or AL, and discards other traces. Estimation of highly-based mask bits corresponds to usage of $p_{A_i|\boldsymbol{L}_i}$ or AL because this attack estimates the value of A_i. Meanwhile, discarding traces does not exploit their information, which results in an SR degradation. Thus, optimal distinguishers

(*e.g.*, Theorem 2) achieve higher performance than the biased mask attack even for highly biased mask bits.

Mask guessing attack. If the attacker knows M in Figure 3 due to AL or white-box SCA on PRG, the attacker can guess the shares $a_1, a_2, \dots, a_d$ with probability 1 as $p_{A_i|M}$ and estimate k. The probability distribution in optimal distinguisher is expressed as

$$p_{Z|\boldsymbol{X},M}(z^{(k)} \mid \boldsymbol{X}, m) = p_{A_0|\boldsymbol{L}_0}(z^{(k)} \oplus a_1 \oplus \cdots \oplus a_d \mid \boldsymbol{L}_0),$$

which implies that the log-likelihood computation and key estimation can be performed without convolution. Thus, the attacker can perform a first-order SCA using $\boldsymbol{L}_0$ and its SR gets identical to that on non-masked implementation. Consequently, optimal distinguishers achieve the best gray-box SCA with mask-bit exposure.

Fresh randomness leakage. The attacker may utilize a leakage of correlated pseudorandomness for mask refreshing. However, it would not improve the SR. Let us consider a refreshing of $A'_0, \dots, A'_d$ to $A_0, \dots, A_d$ using a correlated pseudorandomness $R_0, \dots, R_d$ as $A_i = A'_i \oplus R_i$. Note that both $A'_0, \dots, A'_d$ and $A_0, \dots, A_d$ represent identical unshared value Z. Let $\boldsymbol{L}'_0, \dots, \boldsymbol{L}'_d$ be their respective leakages. One way to use it would be using the probability distribution before refreshing by convoluting $p_{A_i|\boldsymbol{L}_i}$ and $p_{R_i|\boldsymbol{L}'_i}$ as

$$p_{A'_i|\boldsymbol{X}}(a'_i \mid \boldsymbol{X}) = \sum_{a'_i = a_i \oplus r_i} p_{A_i|\boldsymbol{L}_i}(a_i \mid \boldsymbol{L}_i) p_{R_i|\boldsymbol{L}'_i}(r_i \mid \boldsymbol{L}'_i),$$

to calculate the log-likelihood. However, it uses a convolution as same as Equation (2), which would degrade the SR, whereas mask refreshing (*i.e.*, adding shared zero) has no impact on and is independent of Z and K. Another way would be soft-analytical SCA (SASCA), which exploits all leakages in the masked circuit to recover secret key by belief propagation; however, it would not improve the SCA performance because $(R_i, \boldsymbol{L}'_i)$ is independent of Z and K.

5.5 Experimental validation

We show an experimental gray-box SCA to validate the claim of Theorem 1 through SR as a major SCA metric [SMY09]. We simulate an optimal second-order SCA on first-order masked implementation with a mask-bit generator of cryptographic PRG or upcounter, where attacker utilizes the leakages of shares as probability distribution. We also perform a mask-guessing first-order SCA when the mask-bit generator is upcounter. We targeted software AES implementation and focused on one-byte partial key recovery from leakages of first-round masked Sbox. Shares S_0 and S_1 are given as $S_0 = Z \oplus M$ and $S_1 = M$. Side-channel traces were generated as $\boldsymbol{L}_i = \mathrm{HW}(S_i) + N$, where $\mathrm{HW}(S_i)$ is the HW of S_i and N is a zero-mean Gaussian noise with variance $\sigma^2 = 3$. When evaluating upcounter, M is generated by an upcounter with a secret seed. Such leakage has been confirmed for many practical devices [MOP07]. Note that we here focused on one Sbox. Hence, the issue about *low-frequency bits* mentioned in Remark 4 is irrelevant to this experiment when the number of traces is more than 256 (*i.e.*, $2^{n_g} < n$).

We empirically evaluated the SR by repeating SCA trials 100 times and counting the number of successful key recovery. Figure 4 displays the SCA result, where the horizontal and vertical axes are the number of traces and SR, respectively[7]. A sound masking should decrease the SR [DFS15, IUH22a]. As claimed, we found little difference of SR between

[7]Due to the heavy computational cost, we evaluated mask-guessing SCA on masked impl. w/ upcounter using up to 200 traces. To perform mask-guessing with n traces, we need a computational cost of $O(2^{v(d+1)}n)$, where v is Sbox bit-length ($v = 8$ for AES).

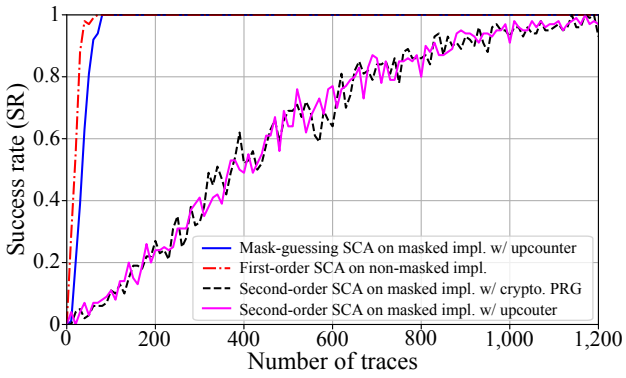


Figure 4: Experimental evaluation result of SCAs.

mask-bit generators of cryptographic PRG and upcounter. In contrast, the mask-guessing first-order SCA offers an SR as high as first-order SCA on non-masked implementation, which means that the security order was degraded to zero due to the AL. Thus, we confirmed the validity of Theorem 1 and the claim on AL.

To validate our claim in the black-box setting, we also performed a computation of empirical mean of traces and performed the first-order TVLA. We found little difference between mask-bit generators of cryptographic PRG and upcounter, and found no first-order leakage in both. Moreover, our claim is valid even in while-box setting, because the white-box attacker exploits physical leakage of mask-bit generator to perform the mask-guessing first-order SCA.

6 Analysis of white-box adversary

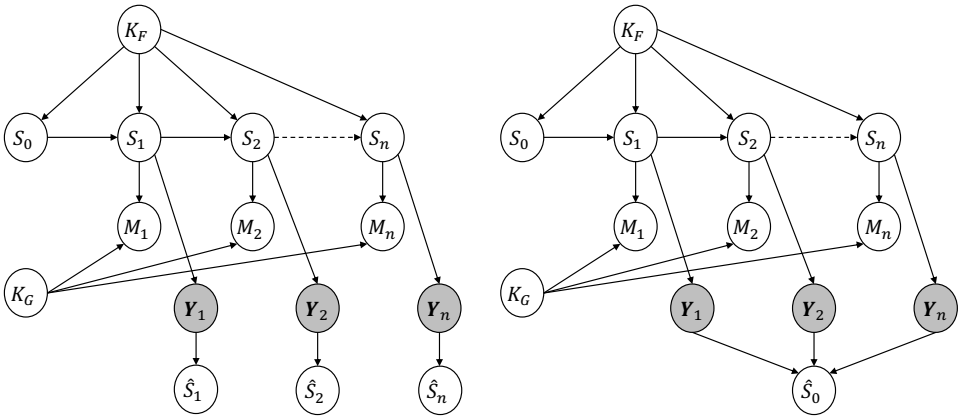
6.1 Notations and graphical model

If the secret seed and key are exposed to the attacker, mask-guessing attack is enabled, meaning that the masked circuit is as insecure as non-masked one as evaluated in Section 5.5. We discuss (in)security of major PRGs against state/key-recovery SCA. See Table 1 for a summary. Figure 5 shows two graphical models representing an SCA on PRG, where estimated keys $\widehat{K}_F$ and $\widehat{K}_G$ from $\boldsymbol{Y}^n$ are omitted. The type depends on the property of F, as described in Section 6.3. Random variables are defined as follows:

- S_0: Initial state of PRG given from seed.
- $S^n = (S_1, \dots, S_n)$: Internal states, where n is the number of traces. $\hat{S}_j$ is its estimation.
- K_F and K_G: Secret keys for updating function F and output function G, respectively (which can be null).
- $\boldsymbol{Y}^n = (\boldsymbol{Y}_1, \boldsymbol{Y}_2, \dots, \boldsymbol{Y}_n)$: Mask-bit generator's leak.

This graphical model means that each state is generated from the previous state (and K_F), outputs are generated from the state (and K_G), and the attacker obtains the physical leakage of each state to estimate, state, seed, and/or keys.

Remark 5 (Soft-analytical mask-guessing attacks.). In this paper, we focus on a key/state-recovery SCA on PRG because the successful recovery immediately reduces masking security to zeroth-order, assuming that the masked implementation has stronger SCA



(a) Type 1 (b) Type 2

Figure 5: Graphical models representing white-box SCA.

resistance than PRG. Meanwhile, a soft-analytical attack, which exploits the PRG leakage to estimate shares by a belief propagation with a probability that is not necessarily 1, would yield key recovery with fewer traces in total than the key/state-recovery SCA and zeroth-order attack if it required fewer traces for full PRG state recovery. This can be considered as a gray- and white-box hybrid attack as the soft-analytical attacker may combine the graphical models in Figures 3 and 5 to perform the belief propagation and optimal SCA. Note that, unless the PRG outputs could be estimated with probability 1, the masked implementation would have $(d+1)$-st order security even against the soft-analytical mask-guessing attacks because the optimal SCA requires d convolution of non-trivial probability distributions [UIM+26]. In this paper, we focus on the zeroth-order attack to highlight the white-box analysis.

6.2 Adaptive attack on masking

In an SCA on masked cryptographic implementation, the attacker is supposed to first acquire n traces, including leakages of both masked circuit and PRG. Because the PRG and masked circuits operate distinctly, the attacker may first perform an SCA to recover the secret key and/or seed of the PRG; thereafter, the attacker can perform a first-order SCA with mask guessing if the attacker has successfully recovered them. In other words, a d-probing attacker can first use all probes on a PRG, then adaptively move the probes to use them on the masked circuit. This is trivially extended to the multiple PRG strategy [CGZ20], meaning that the attacker can adaptively use the probes on each PRG distinctly, as the PRGs operate distinctly. Although such an adaptive attack has been considered impractical for SCAs on masked circuit [SPY+10, FPS12], an adaptive SCA on PRG(s) would be practical if focusing on mask-bit generator.

The feasibility of adaptive SCA would indicate that, *we should evaluate the SCA security of mask-bit generator against practical SCAs, apart from probing-related models and masked circuit.* In fact, we construct an example circuit that is d-probing secure but is vulnerable to an adaptive first-order SCA in Remark 6. Additionally, if we use probing model to evaluate the SCA security of PRGs, they should be d-probing secure as well as masked circuit. However, it may achieve higher security than required because only blind SCA is available owing to unobservability of M, which incurs considerable performance overhead.

6.3 Leakage models

The attacker uses a leakage from PRG to estimate state S, modeled as a conditional probability distribution $p_{S|\boldsymbol{Y}}$, where $\boldsymbol{Y}$ is the leakage of M from PRG. The attacker obtains λ-bit information from a trace, where λ is given as mutual information $I(S;\boldsymbol{Y})$. This indicates that at most $n\lambda$ bits leaks as $I(S^n;\boldsymbol{Y}^n) \leq nI(S;\boldsymbol{Y})$ if the attacker observes n traces. The mutual information represents *leakage amplitude* as in many previous works on LR cryptography [DM19b, DEM+20, BBC+20, DMP22] and theoretical analyses of SR of SCA [DFS15, dCGRP19, IUH22a, MRS23, BCG+23, UIM+26]. Depending on updating function F, we classify PRGs into two types.

Type 1 (F consists in a cryptographic primitive). Cryptographic primitive here can be, for example, BC and pseudorandom permutation (PRP). As the attacker is supposed to know neither secret key and seed, each state $S_1, S_2, \ldots, S_n$ and their leakage $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_n$ seem mutually independent owing to the indistinguishability. In this case, the attacker cannot perform DPAs using $\boldsymbol{Y}^n$, but can only do an SPA to estimate S_j only from $\boldsymbol{Y}_j$ (as $\boldsymbol{Y}_{j'}$ is independent of S_j for $j \neq j'$), as in Figure 5a. If attacker observes n traces, at most $n\lambda$ bits leaks as $I(S^n;\boldsymbol{Y}^n) = nI(S;\boldsymbol{Y})$, but S^n is considered to have $(n \times n_g)$-bit information, which implies the difficulty in state-recovery SCA on this type. Although most symmetric primitive are not proven secure in the presence of leakage (except for LR ones [DP08, PSV15]), some type-1 primitives would be secure in practical settings.

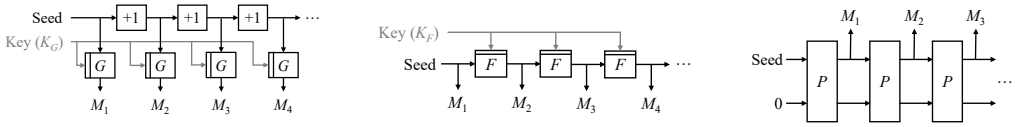
Type 2 (F is not a cryptographic primitive). A major example of this type is a counter-based XOF with $F(S_j) = S_j + 1$. For this type of PRGs, $S_1, S_2, \ldots, S_n$ and $\boldsymbol{Y}_1, \boldsymbol{Y}_2, \ldots, \boldsymbol{Y}_n$ are not independent even for attacker. The attacker may perform key/state-recovery DPAs using n traces. Due to the dependence of S_j's, the attacker may succeed in key recovery if he/she obtains n_f-bit information from leakages as $nI(S;\boldsymbol{Y}) \leq I(S^n;\boldsymbol{Y}^n)$. Thus, the attacker on type-2 may be able to estimate a state S_j from all leakages of $\boldsymbol{Y}^n$. In fact, practical SCAs on some PRGs of this type are known, as discussed in Section 6.4.

6.4 SCA (in)security of major primitives

Plain LFSRs. LFSR is type 2 with $F(S) = \mathrm{double}(S)$ and $G(S) = \mathrm{MSB}(S)$. The LFSR seed can be recovered if the attacker observes $2n_f$-bit output of an n_f-bit LFSR. An SCA attacker obtains $I(S;\boldsymbol{Y})$ bits per trace and at most $n\lambda$-bit information about the state from n traces. Thus, if the attacker observes n traces such that $n\lambda \geq 2n_f$, the attacker may recover the seed or internal state. It would be difficult for LFSR to achieve a provable leakage resilience. Actually, state-recovery SCAs on plain LFSR (Galois and Fibonacci LFSRs) were reported in [JD06, BMV07, SJB20, MMS+23, CMM+24].

Improved LFSRs. In contrast, improved LFSRs (*e.g.*, Mersenne twister and Xorshift) are designed for a high parallelism, which consists of large internal state to realize a long frequency and high dimensionality. State recovery of such LFSRs would be significantly more challenging than plain LFSR, because it requires to observe far more output bits. As the improved LFSRs are implemented with a high parallelism, the leakage would contain less information than non-parallelized implementation due to algorithmic noise; the measurement with a high SNR would be difficult. Thus, although state recovery SCA would be possible *in principle*, improved LFSRs would be more secure than plain LFSRs *in practice*.

Remark 6 (Boolean masked LFSR is broken by first-order adaptive SCA). Because LFSR consists of solely linear operation, d-th order Boolean-masked LFSR is realized by d LFSRs. Here, trace $\boldsymbol{X}$ contains the leakage of each LFSR. In an n-trace SCA, the attacker first



(a) CTR/Hash_DRBG (b) HMAC_DRBG and OFB (c) Sponge

Figure 6: Extendable output functions (XOFs) based on cryptographic primitive.

tries to recover the first LFSR state using the n traces. Then, attacker can also recover the states of second, third, ... , and d-th LFSRs distinctly, as in [CMM+24, Section 4.3]. Thus, d repetition of adaptive first-order SCAs can recover d-probing secure LFSR state.

Stream cipher. It has been highly practical for generating pseudorandomness in terms of performance and implementation cost. For example, Trivium is one of major stream cipher [DC06] included in the ISO/IEC 29192-3 standard [Int23] and the portfolio of eSTREAM project [DCP08]. It has been frequently utilized in masked implementations [CMM+24]. Stream cipher is type 2 as its round function is not PRP, and its leakage resilience is not proven generally. However, stream ciphers are designed in such a way that internal state cannot be recovered from output key stream; the state recovery would be more challenging than that of LFSR even with SCA. Kumar *et al.* reported a state recovery SCA on Trivium in [KDB+22]. However, its feasibility is limited to round-based implementation and very good SNR. Stream ciphers have been frequently implemented in an *unrolled* manner on both hardware and software for low energy [BMA+18, CBT+21][8], and Kobayashi *et al.* claimed that the unrolling would improve the SCA security [KUTH24]. Although stream ciphers have difficulty in achieving a provable leakage resilience, their unrolled implementation would offer both a sufficient performance and practical security.

LR-PRG/stream cipher. Some LR-PRGs/stream ciphers have been developed, which are proven to be secure in the indistinguishability notion under some leakage [DP08, PSV15]. However, their constructions much more complicated than non-LR stream ciphers and incurs considerably high implementation cost. Their security proofs assume that attacker observes its output directly. They would achieve a stronger security goal for mask-bit generation than required.

Type-2 XOFs. A typical PRG as an XOF is counter-based. It uses an upcounter for F and one-way hash H or BC E_{K_G} for output function G; that is, $F(S_i) = S_i + 1$ and $G(S_i) = H(S_i)$ or $E_K(S_i)$, respectively. Figure 6a illustrates its overview, where $G = H$ or E_{K_G}. This construction is the basis of Hash_DRBG and CTR_DRBG, which are PRGs standardized in NIST SP 800-90A [BK12]. Counter-based XOFs do not have leakage resilience as it is type 2. In [DM19a], de Meyer reported a practical blind SCA on CTR_DRBG based on the strategy on the counter-mode AES by Jaffe in [Jaf07], which achieved key and state recoveries within 256 traces and was improved by DL-SCA in [EUH23]. The strategy would be trivially applied to counter-based XOFs with one-way hash including SHA-2 and SHA-3. Thus, counter-based XOF would be inappropriate for mask-bit generation.

Type-1 XOFs. A major way to construct a type-1 XOF from cryptographic hash H or BC E_K is the output-feedback mode (OFB), as illustrated in Figure 6b, where $F = H$ or $F = E_{K_F}$. A NIST-standardized DRBG, HMAC_DRBG, is an instance that uses HMAC

[8] In implementing Trivium on a b-bit microcontroller, b bits are usually processed parallely in a fixslicing manner (up to 64), which corresponds to a b-round unrolling.

as $F(K_F, S_i) = \mathrm{HMAC}_{K_F}(S_i)$. There is no known SCA except for SPA on it if the output is not observable for the attacker, as $S_1, S_2, \ldots, S_n$ are mutually independent, which would imply that it is difficult to exploit their leakage jointly. In [KPP20], Kannwischer *et al.* presented an input-recovery SASCA [VCGS14] on Keccak, which corresponds to state recovery of hash-based OFB; however, its success was limited to small microcontroller, bit-length of observable input (zero in our context), and/or good SNR. Thus, OFB-based XOF may be practically secure in generating mask bits. Note that a key-recovery blind SCA [LDLL14, CR17] would be applicable to E_{K_F} in OFB; but mask guessing attack requires both key and state recovery.

Another major type-1 XOF construction would be sponge using a PRP, such as SHA-3 [BDH+23] and Ascon [DEMS21]. Figure 6c shows a sponge-based XOF, where P is an n_f-bit PRP. The updating and output functions are $F(S_i) = P(S_i)$ and $G(S_i) = \tau_r(S_i)$, respectively, where r is the rate and τ_r is an r-bit truncation. Leakage resilience of sponge-based XOF has been discussed in [DM19b, DEM+20, BBC+20, DMP22]. For example, ISAP [DEM+20], asakey [DMP22], and Ascon [DEMS21] are claimed to be secure against SCAs if PRP is SPA resistant[9]. The aforementioned SASCA in [KPP20] is applicable to this, while its feasibility was limited. Thus, sponge-based XOF would offer high SCA security as well as OFB.

Remark 7 (Reduced-round BC). A reduced-round BC may be used for XOFs in masking, as we do not require cryptographic PRG, which improves the performance. In particular, an XOF based on 4-round AES has been frequently used if a non-cryptographic PRG is sufficient [KS07], for example, in UOV signature [BCD+24].

Pseudorandom function (PRF). One may use a PRF R_K to construct an XOF in Section 6.4 instead of BC or HMAC. The classical GGM would be a typical PRF [GGM86], which was followed by two LR-PRFs, FPS and MSJ [FPS12, MSJ12] (and the leakage resilience of GGM was discussed in [MSJ12]). However, these PRF evaluation requires many calls cryptographic PRG, resulting in a non-negligible latency. They would have difficulty in achieving the practical performance as a mask-bit generator. Another major PRF constructions are sum of BC or PRP [CLM19, HWKS98, ST23] (*e.g.*, $R_{K_1,K_2}(T) = E_{K_1}(T) \oplus E_{K_2}(T)$) and truncation of BC or PRP [BKR98]. Their performance would be comparable to BC or PRP; but, the practical advantage of non-invertibility, which may imply forward-secrecy, is unclear in masking context. Thus, PRF would be less appropriate than BC/PRP.

7 Our recommendations for practitioners

7.1 Black- and gray-box scenarios

Recall that black- and gray-box models correspond to leakage assessment and SCA resistance evaluation of masked circuit by designer/implementer, respectively, apart from the SCA security of PRG. Any PRG can yield valid assessment/evaluation if its output is statistically uniform. Hence, for assessment/evaluation, its availability and implementation costs would be of utmost importance. LFSRs and stream cipher would be useful owing to their high availability and performance (throughput and energy consumption).

[9]Their leakage resilience is not formally proven; but, there are some discussions and positive results [DMP22, BM24, BBC+20]. If we believe their theory, the output is cryptographically secure and the module needs only SPA resistance to offer leakage resilience as an XOF.

7.2 White-box scenario

We list recommendations of mask-bit generator with respect to practical security against white-box SCA and performance.

Improved LFSRs. Some LFSRs like Mersenne twister and Xorshift would have practical security against state-recovery SCAs. Their practical implementation exploits high parallelism and has a large state which renders the state-recovery SCA practically difficult. Yet, they would not have a provable security, implying the feasibility of state recovery if a significant number of traces is observed. This motivates our proposal based on the LR reseeding in Section 7.3

Stream ciphers. Stream cipher, especially its unrolled implementation, would be very practical for mask-bit generation, as recommended by Cassiers *et al.* [CMM+24], A b-round unrolled hardware implementation outputs b pseudorandom bits per clock cycle, and large unrolling degree improves the SCA security [KUTH24]. Banik *et al.* showed that a 288-round unrolling achieves highest energy efficiency [BMA+18]. As well, a software implementation on b-bit microcontroller usually employs a b-round unrolling using b-bit registers.

OFB- and sponge-based XOFs. Some type 1 XOFs (*e.g.*, Ascon) are claimed to achieve an SCA security if the underlying primitive is SPA resistant. Although they would achieve a higher SCA security than improved LFSR and stream cipher, the performance might be worse, as these XOFs require to output random value M_j.

Masked stream ciphers/XOFs. First-order masking (*e.g.*, TI) of some primitives can be implemented with practical performance without fresh randomness (*e.g.*, Ascon [PMSN23], Trivium [SA15], Keccak [Dae17], and AES [Sug18]). Using them, we can realize a first-order secure mask-bit generator, because initial decomposition is not needed in the masking context once it is activated using a seed.

7.3 Proposed LR mask-bit generation method for white-box scenario

From performance perspective, LFSRs (or stream cipher) would be best, whereas type-1 XOFs would offer a leakage resilience. We present an LR mask-bit generation using *reseeding*, which is based on the concept of leveled implementation that combines LR and non-LR primitives to exploit their distinct advantages [BBC+20]. The proposed method comprises a reseeding function and seed expander. The reseeding function, which generates a temporal seed from initial key, should be an LR cryptographic primitive, such as Ascon (a NIST-standardized primitive, claimed to be LR) [DEMS21, TMC+24]. The seed expander is a non-LR high-performance PRG to expand the temporal seed given from LR primitive. The expander is reseeded (*i.e.*, initialized with a new seed) at a frequency to limit the number of traces for SCA, similarly to rekeying [MSGR10, UHIM23]. Our idea is based on the fact that LR primitives may not have high throughput whereas non-LR and non-cryptographic PRG can have. Their combination achieves both leakage resilience and high performance.

Figure 7 displays the overview of proposed method. As a seed expander, we use a high-throughput PRG (*e.g.*, improved LFSR) with a fixed output length, $\alpha : \{0,1\}^{n_f} \to \{0,1\}^{un_g}$ for some $un_g > n_f$, where n_f and n_g are bit lengths of LFSR state and M, respectively, and u is a reseeding interval unit. As a reseeding function, we use an LR primitive $\beta : \mathcal{K} \to \mathcal{K} \times \{0,1\}^{n_f}$, where $\mathcal{K}$ is a key space. Here, β is used to reset α's state (*i.e.*, reseed) as well as to perform rekeying of β. Let $\beta(K) = (\gamma_1(K), \gamma_2(K))$ and $K^j := \gamma_1^j(K)$

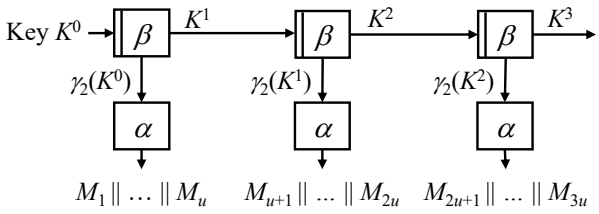


Figure 7: Proposed LR mask-bit generator, where α is non-LR high-throughput seed expander (*e.g.*, LFSR) and β is reseeding function by LR cryptographic primitive like Ascon.

(γ_1^j denotes composition of j γ_1's). Starting with initial uniformly random PRG key of β, denoted by K^0, at each j-th step, we apply β to $\gamma_1(K^{j-1})$ and α to $\gamma_2(K^{j-1})$ and mask bits (values) $(M_{ju} \parallel \cdots \parallel M_{(j+1)u}) = \alpha(\gamma_2(K^j))$. This produces the total output $\alpha(\gamma_2(K^0)) \parallel \alpha(\gamma_2(K^1)) \parallel \cdots \parallel \alpha(\gamma_2(K^{j-1}))$ after the j-th reseeding step. The reseeding interval unit u (*i.e.*, inverse of reseeding frequency) will be determined by considering security-efficiency tradeoff for the given improved LFSR and its implementation security. Namely, given state size of the LFSR n_f and leakage amplitude $\lambda = I(S; \boldsymbol{Y})$, we set u as reseeding interval/frequency such that $n_f \leq u\lambda$, where attacker cannot obtain n_f bits to recover the state for a seed.

This scheme has a similar structure to a forward-secure PRG by Bellare and Yee [BY03], whereas if we use LFSR to boost the throughput, the output is clearly not cryptographically secure. However, thanks to the LR security of β, we can regard the sequence $(M_{ju+1}, M_{ju+2}, \ldots, M_{(j+1)u})$ for each j as an output sequence of α taking an independent and random seed in the presence of leakage. This means that the entire output sequence of M has properties required as mask bits (*i.e.*, high-dimensional uniform equidistribution and a low collision probability) up to the LR security of β.

For practice, a remaining problem is how to determine the reseeding frequency u with a security guarantee ($n_f \leq u\lambda$), as it is generally hard to compute mutual information. We propose the usage of upper-bound evaluation using a DL-based profiling [IUH22a, Section 7]. According to [IUH22a, Lemma 4.4] with $d = 0$, it holds that

$$\lambda = I(S; \boldsymbol{L}) \leq \log \sum_w \mathbb{E}_{\boldsymbol{L}} \left[\left(\hat{p}_{S|\boldsymbol{Y}}(w \mid \boldsymbol{Y}) \right)^2 \right],$$

where $\hat{p}_{S|\boldsymbol{Y}}$ denotes the Walsh–Hadamard transform (WHT) of $p_{S|\boldsymbol{Y}}$. This bound is considerably strong as experimentally evaluated in [IUH22a, Section 7.2]. To use this bound in practice, we need to estimate $p_{S|\boldsymbol{Y}}$. Numerous studies on DL-SCA have exhibited a potential of neural network (NN) to estimate the conditional probability distribution of intermediate value given leakage in SCA [MDP20, IUH22b]. Thus, we first train an NN to estimate $p_{S|\boldsymbol{Y}}$ using the leakages from device focusing on instructions leaking S, and then determine u such that $n_f \leq uq \log \sum_w \mathbb{E}_{\boldsymbol{Y}}[(\hat{p}_{S|\boldsymbol{Y}}(w \mid \boldsymbol{Y}))^2]$ tightly, where q is the number of calls of leaking instruction(s) in α and bit length of S should correspond to the word size of the implementation (*i.e.*, register size of microcontroller or unrolling degree).

We experimentally evaluated the reseeding frequency u of a 128-bit LFSR on eight-bit microcontroller in the condition same as Section 5.5, while α outputs eight bits per call and $q = 1$, and SNR varies. Table 3 displays the experimentally-estimated u, where ground-truth (GT) $I(S; \boldsymbol{Y})$ determines the SNR[10]. In estimating $p_{S|\boldsymbol{Y}}$, we utilized the NN

[10] In generating the simulated side-channel traces, we first determined the GT $I(S; \boldsymbol{Y})$, which is upper-bounded by the SNR of $\boldsymbol{Y} = \mathrm{HW}(S) + N$ according to the Shannon–Hartley theorem. From the SNR, we determined the variance of noise N.

Table 3: Evaluation result of reseeding frequency (Throughput is given in Cycles/byte)

GT $I(S;\boldsymbol{Y})$	0.01	0.02	0.05	0.1	0.2	0.5	1	2
$I(S;\boldsymbol{Y})$ bound	0.013	0.063	0.090	0.186	0.341	0.743	1.254	2.181
Reseeding freq.	9,846	2,327	1,422	688	375	172	102	58
Mask bits/seed	78,768	18,616	11,376	5,504	3,000	1,376	816	464
Ascon calls/Kbyte	0.10	0.43	0.70	1.45	2.67	5.81	9.80	17.2
Throughput	0.22	0.24	0.25	0.29	0.36	0.52	0.74	1.1

Table 4: Values for calculating throughput in Table 3 (cycle/byte is derived from the papers by ourselves)

PRG	Platform	Throughput	Reference
Ascon-p^{12}, $r = 128$	ARM Cortex-M4	53.3 cycles/byte	Adominicăi *et al.* [AMS25]
128-bit LFSR	ARM Cortex-A8	0.213 cycles/byte	Krovetz and Rogaway [KR11]

same as that in [IUH22a]. The result indicates that non-LR LFSR can expands 128-bit seeds significantly. Table 3 also reports the number of Ascon permutation calls per Kbyte (note that the number of LFSR calls per Kbyte are fixed for any GT $I(Z;\boldsymbol{X})$) and ideal throughput of the resulting mask-bit generators. The throughput is calculated using the values in Table 4. LFSR is far faster than Ascon, and the proposed mask-bit generator effectively reduce the Ascon calls by the combination. As a result, the proposed mask-bit generator is at least about 50 times faster than Ascon-only mask-bit generation, while maintaining the leakage resilience. Our method is very effective for smaller $I(S;\boldsymbol{Y})$ (*i.e.*, when noisy). For low-noise regime (*e.g.*, $I(S;\boldsymbol{Y}) \geq 2$), the expansion is not so effective. This is consistent with previous studies showing that any masked implementation cannot be secure if only trivial noise exists in leakage [DFS15, IUH22a, BS21].

8 Conclusion

This study provided a systematic analysis of PRGs for masked cryptographic implementation. We established three adversarial models to analyze the PRG properties required for secure mask-bit generation, and analyzed them to reveal the properties of PRGs required for secure mask-bit generation with an experimental validation. We provided a comprehensive analysis of physical (in)security of major PRGs for practical deployment. Based on our analyses, we showed our recommendations and presented an LR mask-bit generator.

A Proof of Lemma 1

Proof. Before the proof main body, we introduce notations, TVLA, and common SCAs in a formal description.

Notations. Let $X_1, X_2, \ldots, X_n$ be n samples of random variable X. Its empirical mean and unbiased empirical variance are denoted by

$$\hat{\mu}[X] = \frac{1}{n}\sum_j X_j, \quad \hat{\sigma}^2[X] = \frac{1}{n-1}\sum_j (X_j - \hat{\mu}[X])^2,$$

respectively. Let $Y_1, Y_2, \ldots, Y_n$ be a jointly sampled random variables of Y as well. Unbiased empirical covariance between X and Y is given by

$$\hat{\sigma}[X,Y] = \frac{1}{n-1}\sum_j (X_j - \hat{\mu}[X])(Y_j - \hat{\mu}[Y]).$$

Assume that, for $(d-1)$-st order masked software implementation, a trace $\boldsymbol{X}$ consists of d leakages $L_1, L_2, \ldots, L_d \in \mathbb{R}$ corresponding to its share $A_1, A_2, \ldots, A_d$, respectively. In contrast, for $(d-1)$-st order masked hardware implementation, assume that the leakage is univariate and a trace $\boldsymbol{X} = L$ is given as sum of $L_1, L_2, \ldots, L_d$. In the d-th order TVLA/SCA, we use d-th order leakage denoted by $V^{(d)}$. For d-variate SCA (typically on software implementation), the d-th order leakage is given by a product of zero-averaged d sample points as $V^{(d)} = \prod_{i=1}^{d} (L_i - \hat{\mu}[L_i])$. For univariate TVLA/SCA (typically on hardware implementation), the d-th order leakage is given by the d-th power of a zero-averaged sample point as $V^{(d)} = (L - \hat{\mu}[L])^d$.

Test vector leakage assessment (TVLA). A d-th order TVLA assesses the existence of d-th order leakage using a non-parametric t-test [SM15]. The TVLA is designed to reject a null hypothesis that mean of d-th order leakage is not different when the input plaintexts are a fixed value or random. For the d-th order TVLA, the empirical t-value is computed using d-th order leakage in traces as

$$\hat{t} = \frac{\hat{\mu}\left[V_{\mathrm{fix}}^{(d)}\right] - \hat{\mu}\left[V_{\mathrm{rnd}}^{(d)}\right]}{\sqrt{\frac{\hat{\sigma}^2\left[V_{\mathrm{fix}}^{(d)}\right]}{n_{\mathrm{fix}}} + \frac{\hat{\sigma}^2\left[V_{\mathrm{rnd}}^{(d)}\right]}{n_{\mathrm{rnd}}}}},$$

where $V_{\mathrm{fix}}^{(d)}$ and $V_{\mathrm{rnd}}^{(d)}$ are random variables of d-th order leakage with fixed and random plaintexts, respectively, and n_{fix} and n_{rnd} are the respective numbers of traces. If the resultant t-value is higher than threshold, the null hypothesis is rejected and the alternative hypothesis is accepted, implying that there exists a d-th order leakage. Otherwise, the null hypothesis is (negatively) accepted. A double-sided test with a significance level of 10^{-5} have been commonly used, which corresponds to absolute t-value threshold of 4.5.

Correlation power analysis (CPA). Let $\phi : \{0,1\}^{n_s} \to \mathbb{R}$ be a leakage function (*e.g.*, Hamming weight), where n_s denotes bit length of Z. A d-th order CPA estimates the secret key as $\hat{k}$ using Pearson's correlation coefficient $\hat{\rho}$ as

$$\begin{aligned}\hat{k} &= \arg\max_k \left|\hat{\rho}\left[\phi(Z^{(k)}), V^{(d)}\right]\right| \\ &= \arg\max_k \left|\frac{\hat{\sigma}\left[\phi(Z^{(k)}), V^{(d)}\right]}{\hat{\sigma}[\phi(Z^{(k)})] \cdot \hat{\sigma}[V^{(d)}]}\right|.\end{aligned}$$

Profiled deep-learning based SCA (DL-SCA). Let $q_{\hat{\theta}}$ be a trained neural network as a profiled leakage model, which imitates $p_{Z|\boldsymbol{X}}$. A DL-SCA using the neural network estimates the secret key as

$$\hat{k} = \arg\max_k \hat{\mu}\left[\log q_{\hat{\theta}}(Z^{(k)} \mid \boldsymbol{X})\right].$$

Main body of proof. Common SCAs and TVLA compute the results solely using empirical mean, unbiased empirical variance, and unbiased empirical covariance, as described. These statistics are obviously invariant to the order of joint samples in $(T^n, \boldsymbol{X}^n)$ due to commutativity of addition. Thus, the leakage assessment and SCA results are invariant to the trace order. □

B Proof of Theorem 2

Proof. Let $\mathbb{E}$ be the expectation and Pr be the probability measure. Let $\ell(\hat{k}, k) = \mathbb{1}_{\{\hat{k} \neq k\}}$ be a loss function, where $\mathbb{1}$ is the indicator function. Optimal distinguisher is defined as a

distinguisher that minimizes the loss function ℓ in guessing the correct key. Taking the expectation of $\ell(D(T^n, \boldsymbol{X}^n), K)$, we have

$$\begin{aligned}\mathbb{E}\ell(D(T^n, \boldsymbol{X}^n), K) &= \mathbb{E}\mathbb{1}_{\{D(T^n,\boldsymbol{X}^n)\neq K\}} \\ &= \mathbb{E}\mathbb{E}[\mathbb{1}_{\{D(T^n,\boldsymbol{X}^n)\neq K\}} \mid T^n, \boldsymbol{X}^n] \\ &= \mathbb{E}[1 - \Pr(K = D(T^n, \boldsymbol{X}^n) \mid T^n, \boldsymbol{X}^n)].\end{aligned}$$

Therefore,

$$\begin{aligned}D(T^n, \boldsymbol{X}^n) &= \arg\max_k \Pr(K = k \mid T^n, \boldsymbol{X}^n) \\ &= \arg\max_k p_{K|,\boldsymbol{X}^n,T^n}(k \mid T^n, \boldsymbol{X}^n)\end{aligned}$$

is an optimal distinguisher.

Then, we focus on a joint probability distribution $p_{K,\boldsymbol{X}^n,T^n}$ since $\arg\max_k p_{K|\boldsymbol{X}^n,T^n}(k \mid X^n, T^n) = \arg\max_k p_{K,\boldsymbol{X}^n,T^n}(k, \boldsymbol{X}^n, T^n)$. We hereafter omit the subscripts of probability distribution. According to the marginalization in terms of Z^m, we have

$$\begin{aligned}p(k, \boldsymbol{X}^n, T^n) &= \sum_{z^n} p(k, z^n, \boldsymbol{X}^n, T^n) \\ &= \sum_{z^n} p(\boldsymbol{X}^n \mid z^n, T^n, k)p(z^n \mid T^n, k)p(T^n, k).\end{aligned}$$

Here, $p(\boldsymbol{X}^n \mid z^n, T^n, k) = p(\boldsymbol{X}^n \mid z^n)$, which follows from the fact that $\boldsymbol{X}^n$ and (T^n, K) are conditionally independent as Z^n d-separates $\boldsymbol{X}^n$ and (T^n, K) when Z^n is given [Bis06]. As well, $p(T^n, k) = p(T^n)p(k)$, because T^n and K are conditionally independent because Z^n d-separates T^n and K when Z^n is not given. Since $\boldsymbol{X}^n$, Z^n and T^n consists of independent copies of $\boldsymbol{X}$, Z, and T, respectively, we have

$$\begin{aligned}p(k, \boldsymbol{X}^n, T^n) &= \sum_{z^n} p(\boldsymbol{X}^n \mid z^n)p(z^n \mid T^n, k)p(T^n)p(k) \\ &= p(k)\prod_{j=1}^{n}\sum_{z} p(\boldsymbol{X}_j \mid z_j)p(z \mid T_j, k)p(T_j).\end{aligned}$$

According to $Z^{(k)} = g(k, T)$ and Beyes' theorem, we have

$$\begin{aligned}p(k, \boldsymbol{X}^n, T^n) &= p_K(k)\prod_{j=1}^{n} p(\boldsymbol{X}_j \mid Z_j^{(k)})p(T_j) \\ &= p(k)\prod_{j=1}^{n}\frac{p(Z_j^{(k)} \mid \boldsymbol{X}_j)p(\boldsymbol{X}_j)p(T_j)}{p(Z_j^{(k)})}. \qquad (3)\end{aligned}$$

Taking the logarithm of Equation (3), we have

$$\begin{aligned}\log p(k, \boldsymbol{X}^n, T^n) = \log p(k) + \sum_{j=1}^{n}\log p(\boldsymbol{X}_j)p(T_j) \\ + \sum_{j=1}^{n}\log p(Z_j^{(k)} \mid \boldsymbol{X}_j) - \sum_{j=1}^{n}\log p(Z_j^{(k)}). \qquad (4)\end{aligned}$$

Consider the arg max of Equation (4) in terms of k. The first term $\log p_K(k)$ has no impact on the arg max because K is uniformly distributed. The second term $\sum_j \log p(\boldsymbol{X}_j)p(T_j)$ also has no impact because it is independent of k. Thus, assuming that Z is uniformly

distributed, we conclude

$$\begin{aligned} D(T^n, \boldsymbol{X}^n) &= \arg\max_k \log p(k \mid \boldsymbol{X}^n, T^n) \\ &= \arg\max_k \sum_{j=1}^{n} \log p_{Z|\boldsymbol{X}}(Z_j^{(k)} \mid \boldsymbol{X}_j), \end{aligned} \tag{5}$$

is an optimal distinguisher, as required.

The second equality is proven in a similar manner as [IUH22a, Equation (2)]. Marginalizing $A_0, A_1, \ldots, A_d$, we have

$$p(z \mid \boldsymbol{X}) = \sum_{a_0, a_1, \ldots, a_d} p(z, a_0, a_1, \ldots, a_d \mid \boldsymbol{L}_0, \boldsymbol{L}_1, \ldots, \boldsymbol{L}_d).$$

Here, Z is conditionally independent of $\boldsymbol{L}^d$ if A^d is given, which d-separates Z and $\boldsymbol{L}^d$. Because $A_0, A_1, \ldots, A_d$ are mutually independent when Z is not given ($\boldsymbol{L}_0, \boldsymbol{L}_1, \ldots, \boldsymbol{L}_d$ are same) and $z = a_0 \oplus a_1 \oplus \cdots \oplus a_d$, it holds that

$$\begin{aligned} p(z \mid \boldsymbol{X}) &= \sum_{a_0, a_1, \ldots, a_d} p(z \mid a_0, a_1, \ldots, a_d) \prod_i p(a_i \mid \boldsymbol{L}_i) \\ &= \sum_{z = a_0 \oplus a_1 \oplus \cdots \oplus a_d} \prod_{i=0}^{d} p(a_i \mid \boldsymbol{L}_i). \end{aligned} \tag{6}$$

By substituting Equation (6) to Equation (5), we complete the proof. □

Acknowledgment

We are grateful to Akira Ito and Akiko Inoue for fruitful discussion. We sincerely thank the shepherd and all the anonymous reviewers for their feedback. This work has been supported by JSPS Kakenhi Grant No. 25K03117 and 26K21773, JST K Program Grant No. JPMJKP24C1, and JST CRONOS Grant No. JPMJCS25N1, Japan.

References

[ABC+23a] Aikata Aikata, Andrea Basso, Gaetan Cassiers, Ahmet Can Mert, and Sujoy Sinha Roy. Kavach: Lightweight masking techniques for polynomial arithmetic in lattice-based cryptography. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(3):366–390, 2023.

[ABC+23b] Melissa Azouaoui, Olivier Bronchain, Gaëtan Cassiers, Clément Hoffmann, Yulia Kuzovkova, Joost Renes, Tobias Schneider, Markus Schönauer, François-Xavier Standaert, and Christine van Vredendaal. Protecting Dilithium against leakage: Revisited sensitivity analysis and improved implementations. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, (4):58–79, 2023.

[AFL+16] Toshinori Araki, Jun Furukawa, Yehuda Lindell, Ariel Nof, and Kazuma Ohara. High-throughput semi-honest secure three-party computation with an honest majority. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 805–817, 2016.

[AMS25] Alexandre Adomnicăi, Kazuhiko Minematsu, and Junji Shikata. Lightweight yet nonce-misuse secure authenticated encryption for very short inputs. *IEEE Internet of Things Journal*, 12(3):2807–2824, 2025.

[BBC+19] Gilles Barthe, Sonia Balaïd, Gaëtan Cassiers, Pierre-Alan Fouque, Benjamin Grégoire, and François-Xavier Standaert. maskVerif: Automated verification of higher-order masking in presence of physical defaults. In *European Symposium on Research in Computer Security (ESORICS 2019)*, volume 11735 of *Lecture Notes in Computer Science*, pages 300–318. Springer, 2019.

[BBC+20] Davide Bellizia, Olivier Bronchain, Gaëtan Cassiers, Vincent Grosso, Chun Guo, Charles Momin, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Mode-level vs. implementation-level physical security in symmetric cryptography: A practical guide through the leakage-resistance jungle. In *Advances in Cryptology—CRYPTO 2022*, volume 12170 of *Lecture Notes in Computer Science*, pages 369–400, 2020.

[BBD+15] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, and Pierre-Yves Strub. Verified proofs of higher-order masking. In *Advances in Cryptology—EUROCRYPT 2015*, pages 457–485, 2015.

[BBD+16] Gilles Barthe, Sonia Belaïd, François Dupressoir, Pierre-Alain Fouque, Benjamin Grégoire, Pierre-Yves Strub, and Rébecca Zucchini. Strong non-interference and type-directed higher-order masking. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016.

[BCD+24] Ward Beullens, Ming-Shing Chen, Jintai Ding, Boru Gong, Matthias J. Kannwischer, Jacques Patarin, Bo-Yuan Peng, Dieter Schmidt, Cheng-Jhih Shih, Chengdong Tao, and Bo-Yin Yang. Uov: Unbalanced oil and vinegar—algorithm specifications and supporting documentation version 1.0. NIST Post-Quantum Cryptography, Aug 2024.

[BCG+23] Julien Béguinot, Wei Cheng, Sylvain Guilley, Yi Liu, Loïc Masure, Olivier Rioul, and François-Xavier Standeart. Removing the field size loss from Duc et al.'s conjectured bound for masked encodings. In *International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE 2023)*, volume 13979 of *Lecture Notes in Computer Science*, pages 86–104, 2023.

[BCGR24] Julien Béguinot, Wei Cheng, Sylvain Guilley, and Olivier Rioul. Formal security proofs via Doeblin coefficients: Optimal side-channel factorization from noisy leakage to random probing. Advances in Cryptology—CRYPTO 2024, 2024. https://eprint.iacr.org/2024/199.

[BCO04] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 16–29, 2004.

[BDH+23] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Team Keccak, Nov. 2023.

[BFGV12] Josep Balasch, Sebastian Faust, Benedikt Gierlichs, and Ingrid Verbauwhede. Theory and practice of a leakage resilient masking scheme. In *Advances in Cryptology—ASIACRYPT 2012*, pages 758–775, 2012.

[BGG+14] Josep Balasch, Benedikt Gierlichs, Vincent Grosso, Oscar Reparaz, and François-Xavier Standaert. On the cost of lazy engineering for masked software implementations. In *International Conference on Smart Card Research and Advanced Applications (CARDIS 2014)*, volume 8968 of *Lecture Notes in Computer Science*, pages 64–81, 2014.

[BGR+21] Joppe W. Bos, Marc Gourjon, Joost Renes, Tobias Schneider, and Christine van Vredendaal. Masking Kyber: First- and higher-order implementations. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(4):173–214, 2021.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[BK12] Elaine Barker and John Kelsey. NIST special publication 800-90a: Recommendation for random number generation using deterministic random bit generators, Jan. 2012. Computer Security Division Information Technology Laboratory, National Institute of Standards and Technology.

[BKR98] Mihir Bellare, Ted Krovetz, and Phillip Rogaway. Luby-Rackoff backwards: Increasing security by making block ciphers non-invertible. In *Advances in Cryptology—EUROCRYPT'98*, pages 266–280, 1998.

[BM24] Henk Berendsen and Bart Mennink. Tightening leakage resilience of the suffix keyed sponge. *IACR Transactions on Symmetric Cryptology*, 2024(1):459–496, 2024.

[BMA+18] Subhadeep Banik, Vasily Mikhalev, Frederik Armknecht, Takanori Isobe, Willi Meier, Andrey Bogdanov, Yuhei Watanabe, and Francesco Regazzoni. Towards low energy stream ciphers. *IACR Transactions on Symmetric Cryptology*, (2):1–19, 2018.

[BMRT22] Sonia Belaïd, Darius Mercadier, Matthieu Rivain, and Abdul Rahman Taleb. IronMask: Versatile verification of masking security. In *IEEE Symposium on Security and Privacy (SP)*, pages 142–160, 2022.

[BMV07] Sanjay Burman, Debdeep Mukhopadhyay, and Kamakoti Veezhinathan. LFSR based stream ciphers are vulnerable to power attacks. In *Progress in Cryptology—INDOCRYPT 2007*, pages 384–392, 2007.

[BPO+20] Florian Bache, Clara Paglialong, Tobias Oder, Tobias Schneider, and Tim Güneysu. High-speed masking for polynomial comparison in lattice-based KEMs. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(3):483–507, 2020.

[BS21] Olivier Bronchain and François-Xavier Standeart. Breaking masked implementations with many shares on 32-bit software platforms: or when the security order does not matter. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 3:202–234, 2021.

[BY03] Mihir Bellare and Bennet S. Yee. Forward-Security in Private-Key Cryptography. In *CT-RSA*, volume 2612 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2003.

[CBT+21] Andrea Caforio, Subhadeep Banik, Yosuke Todo, Willi Meier, Takanori Isobe, Fukang Liu, and Bin Zhang. Perfect trees: Designing energy-optimal symmetric encryption primitives. *IACR Transactions on Symmetric Cryptology*, (4):36–73, 2021.

[CGC+21] Wei Cheng, Sylvain Guilley, Claude Carlet, Jean-Luc Danger, and Sihem Mesnager. Information leakages in code-based masking: A unified quantification approach. *IACR Trans. Cryptogr. Hardw. Embedded Syst.*, 2021(3):465–495, 2021.

[CGMZ22a] Jean-Sébastien Coron, François Gérard, Simon Montoya, and Rina Zeitoun. High-order polynomial comparison and masking lattice-based encryption. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(1):153–192, 2022.

[CGMZ22b] Jean-Sébastien Coron, François Gérard, Simon Montoya, and Rina Zeitoun. High-order table-based conversion algorithms and masking lattice-based encryption. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022:1–40, 2022.

[CGTZ23a] Jean-Sébastien Coron, François Gérard, Matthias Trannoy, and Rina Zeitoun. High-order masking of NTRU. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(2):180–211, 2023.

[CGTZ23b] Jean-Sébastien Coron, François Gérard, Matthias Trannoy, and Rina Zeitoun. Improved gadgets for the high-order masking of dilithium. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(4):110–145, 2023.

[CGZ20] Jean-Sébastien Coron, Aurélien Greuet, and Rina Zeitoun. Side-channel masking with pseudo-random generator. In *Advances in Cryptology—EUROCRYPT 2020*, pages 342–375, 2020.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In *Advances in Cryptology—CRYPTO' 99*, pages 398–412, 1999.

[CLM19] Yu Long Chen, Eran Lambooij, and Bart Mennink. How to build pseudorandom functions from public random permutations. In *Advances in Cryptology—CRYPTO 2019*, pages 266–293, 2019.

[CMM+24] Gaëtan Cassiers, Loïc Masure, Charles Momin, Thorben Moos, Amir Moradi, and François-Xavier Standaert. Randomness generation for secure hardware masking – unrolled Trivium to the rescue. *IACR Communications in Cryptology*, 1(2), 2024.

[CPRR14] Jean-Sébastien Coron, Emmanuel Prouff, Matthieu Rivain, and Thomas Roche. Higher-order side channel security and mask refreshing. In *International Workshop on Fast Software Encryption (FSE)*, pages 410–424, 2014.

[CR17] Christophe Clavier and Léo Reynaud. Improved blind side-channel analysis by exploitation of joint distributions of leakages. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, pages 24–44, 2017.

[CRR02] Suresh Chari, Josyula R. Rao, and Pankaj Rohatgi. Template attacks. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2002)*, LNCS, pages 13–28, 2002.

[CS20] Gaëtan Cassiers and François-Xavier Standaert. Trivially and efficiently composing masked gadgets with probe isolating non-interference. *IEEE Transactions on Information Forensics and Security*, 15:2542–2555, 2020.

[Dae17] Joan Daemen. Changing of the guards: A simple and efficient method for achieving uniformity in threshold sharing. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 137–153, 2017.

[DC06] Christophe De Canniere. Trivium specifications, 2006. http://www.ecrypt.eu.org/stream/p3ciphers/trivium/trivium_p3.pdf.

[dCBG+17] Thomas de Cnudde, Begül Bilgin, Benedikt Gierlichs, Ventzislav Nikov, Svetla Nikova, and Vincent Rijmen. Does coupling affect the security of masked implementations? In *International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE 2017)*, volume 10348 of *Lecture Notes in Computer Science*, pages 1–18, 2017.

[DCEM18] Thomas De Cnudde, Maik Ender, and Amir Moradi. Hardware masking, revisited. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, (2):123–148, 2018.

[dCGRP19] Eloi de Chérisey, Sylvain Guilley, Olivier Rioul, and Pablo Piantanida. Best information is most successful: Mutual information and success rate in side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2019(2):49–79, 2019.

[DCP08] Christophe De Canniere and Bart Preneel. Trivium. In *New Stream Cipher Designs: The eSTREAM Finalists*, pages 244–266. Springer, 2008.

[DCRB+16] Thomas De Cnudde, Oscar Reparaz, Begül Bilgin, Svetla Nikova, Ventzislav Nikov, and Vincent Rijmen. Masking aes with $d + 1$ shares in hardware. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, pages 194–212, 2016.

[DDF14] Alexandre Duc, Stefan Dziembowski, and Sebastian Faust. Unifying leakage models: From probing attacks to noisy leakage. In *Advances in Cryptology—EUROCRYPT 2014*, pages 423–440, 2014.

[DEM+20] Christoph Dobraunig, Maria Eichlseder, Stefan Mangard, Florian Mendel, Bart Mennik, Robert Primas, and Thomas Unterlaggauer. Isap v2.0. *IACR Transactions on Symmetric Cryptology*, 2020(S1):390–416, 2020.

[DEMS21] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon: Lightweight authentication & hashing, 2021. https://ascon.iaik.tugraz.at/index.html.

[DF12] Stefan Dziembowski and Sebastian Faust. Leakage-resilient circuits without computational assumptions. In *Theory of Cryptography Conference (TCC)*, pages 230–247, 2012.

[DFS15] Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete: Or how to evaluate the security of any leakage device. In *Advances in Cryptology—EUROCRYPT 2015*, volume 9056 of *Lecture Notes in Computer Science*, pages 401–429. Springer, 2015.

[DFS16] Stefan Dziembowski, Sebastian Faust, and Maciej Skórski. Optimal amplification of noisy leakages. In *Theory of Cryptography*, pages 291–318, 2016.

[DHP+22] Jan-Pieter D'Anvers, Daniel Heinz, Peter Pessl, Michiel Van Beirendonck, and Ingrid Verbauwhede. Higher-order masked ciphertext comparison for lattice-based cryptography. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(2):115–139, 2022.

[DM19a] Lauren De Meyer. Recovering the CTR_DRBG state in 256 traces. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(1):37–65, 2019.

[DM19b] Christoph Dobraunig and Bart Mennink. Leakage resilience of the duplex construction. In *Advances in Cryptology—ASIACRYPT 2019*, pages 225–255, 2019.

[DMP22] Christoph Dobraunig, Bart Mennink, and Robert Primas. Leakage and tamper resilient permutation-based cryptography. In *ACM SIGSAC Conference on Computer and Communications Security (CCS 2022)*, pages 859–873, 2022.

[DMRB18] Lauren De Meyer, Oscar Reparaz, and Begül Bilgin. Multiplicative masking for AES in hardware. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(3):431–468, 2018.

[DP08] Stefan Dziembowski and Krzysztof Pietzrak. Leakage-resilient cryptography. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008)*, pages 293–302, 2008.

[DP10] Yavgeniy Dodis and Kryzsztof Pietzrak. Leakage-resilient pseudorandom functions and side-channel attacks on Feistel networks. In *Advances in Cryptology—CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 21–40, 2010.

[dPPRS23] Rafaël del Pino, Thomas Prest, Mélissa Rossi, and Markku-Juhani O. Saarinen. High-order masking of lattice signatures in quasilinear time. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 1168–1185, 2023.

[EUH23] Katsumi Ebina, Rei Ueno, and Naofumi Homma. Side-channel analysis against SecOC-compliant AES-CMAC. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 70(10):3772–3776, 2023.

[FGMDP+18] Sebastian Faust, Vincent Grosso, Santos Merino Del Pozo, Clara Paglialonga, and François-Xavier Standaert. Composable masking schemes in the presence of physical defaults & the robust probing model. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(3):89–120, 2018.

[FPS12] Sebastian Faust, Krzysztof Pietrzak, and Joachim Schipper. Practical leakage-resilient symmetric cryptography. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2012)*, volume 7428 of *Lecture Notes in Computer Science*, pages 213–232. Springer, 2012.

[GD23] John Gaspoz and Siemen Dhooghe. Threshold implementations in software: Micro-architectural leakages in algorithms. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(2):155–179, 2023.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.

[GM11] Louis Goubin and Ange Martinelli. Protecting AES with Shamir's secret sharing scheme. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 79–94, 2011.

[GM17] Hannes Gross and Stefan Mangard. Reconciling $d + 1$ masking in hardware and software. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES 2017)*, volume 10529 of *Lecture Notes in Computer Science*. Springer, 2017.

[GMK16] Hannes Gross, Stefan Mangard, and Thomas Korak. Domain-oriented masking: Compact masked hardware implementations with arbitrary protection order. In *ACM Workshop on Theory of Implementation Security (TIS 2016)*, page 3, 2016.

[GMPO20] Si Gao, Ben Marshall, Dan Page, and Elisabeth Oswald. Share-slicing: Friend or foe? *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(1):152–174, 2020.

[Gou01] Louis Goubin. A sound method for switching between Boolean and arithmetic masking. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 3–15, 2001.

[GSH+26] Anna Guinet, Jan Schoone, Niklas Höher, Dina Hesse, and Tim Güneysu. Lfsrs and boolean masking: An in-depth security analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2026(3):426–464, 2026.

[GT03] Jovan D. Golić and Christophe Tymen. Multiplicative masking and power analysis of AES. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 198–212, 2003.

[HRG14] Annelie Heuser, Olivier Rioul, and Sylvain Guilley. Good is not good enough: Deriving optimal distinguishers from communication theory. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2014)*, pages 55–74, 2014.

[HWKS98] Chris Hall, David Wagner, John Kelsey, and Bruce Schneier. Building PRFs from PRPs. In *Advances in Cryptology—CRYPTO '98*, pages 370–389, 1998.

[IKL+13] Yuval Ishai, Eyal Kushilevitz, Xin Li, Rafail Ostrovsky, Manoj Prabhakaran, Amit Sahai, and David Zuckerman. Robust pseudorandom generators. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 576–588, 2013.

[Int23] International Organization for Standardization. ISO/IEC 29192-3:2012 Information technology—Security techniques—Lightweight cryptography—Part 3: Stream ciphers. https://www.iso.org/standard/56426.html, Nov. 2023.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology—CRYPTO 2003*, pages 463–481, 2003.

[IUH21] Akira Ito, Rei Ueno, and Naofumi Homma. Toward optimal deep-learning based side-channel attacks: Probability concentration inequality loss and its usage. Cryptology ePrint Archive, Report 2021/1216, 2021. https://ia.cr/2021/1216.

[IUH22a] Akira Ito, Rei Ueno, and Naofumi Homma. On the success rate of side-channel attacks on masked implementations: Information-theoretical bounds and their practical usage. In *ACM SIGSAC Conference on Computer and Communications Security (CCS 2022)*, pages 1521–1535, 2022.

[IUH22b] Akira Ito, Rei Ueno, and Naofumi Homma. Perceived information revisited: New metrics to evaluate success rate of side-channel attacks. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 4:228–254, 2022.

[Jaf07] Josh Jaffe. A first-order DPA attack against AES in counter mode with unknown initial counter. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 1–13, 2007.

[JD06] Antoine Joux and Pascal Delaunay. Galois lfsr, embedded devices and side channel weaknesses. In *Progress in Cryptology—INDOCRYPT 2006*, pages 436–451, 2006.

[KDB+22] Satyam Kumar, Vishnu Asutosh Dasu, Anubhab Baksi, Santanu Sarkar, Dirmanto Jap, Jakub Breier, and Shivam Bhasin. Side channel attack on stream ciphers: A three-step approach to state/key recovery. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(2):166–191, 2022.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *Advances in Cryptology—CRYPTO' 99*, volume 1666 of *Lecture Notes in Computer Science*, pages 388–397, 1999.

[KMMS22] David Knichel, Amir Moradi, Nicolai Müller, and Pascal Sasdrich. Automated generation of masked hardware. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 1:589–629, 2022.

[KPP20] Matthias J. Kannwischer, Peter Pessl, and Robert Primas. Single-trace attacks on Keccak. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(3):243–268, 2020.

[KR11] Ted Krovetz and Phillip Rogaway. The software performance of authenticated-encryption modes (full version). In *Fast Software Encryption*, pages 306–327, 2011.

[KS07] Liam Keliher and Jiayuan Sui. Exact maximum expected differential and linear probability for two-round Advanced Encryption Standard. *IET Inf. Secur.*, 1(2):53–57, 2007.

[KSM20] David Knichel, Pascal Sasdrich, and Amir Moradi. SILVER—statistical independence and leakage verification. In *Advances in Cryptology—ASIACRYPT 2020*, volume 12491 of *Lecture Notes in Computer Seience*, pages 787–816, 2020.

[KSM21] David Knichel, Pascal Sasdrich, and Amir Moradi. Generic hardware private circuits: Towards automated generation of composable secure gadgets. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022:323–344, 2021.

[KUTH24] Soichiro Kobayashi, Rei Ueno, Yosuke Todo, and Naofumi Homma. Side-channel linearization attack on unrolled Trivium hardware. *IACR Commun. Cryptol.*, 1(3), 2024.

[LDLL14] Yanis Linge, Cécile Dumas, and Sophie Lambert-Lacroix. Using the joint distributions of a cryptographic function in side channel analysis. In *International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE)*, pages 199–213, 2014.

[Mar03] George Marsaglia. Xorshift RNGs. *Journal of Statistical Software*, 8(14):1–6, 2003.

[MDP20] Loïc Masure, Cécile Dumas, and Emmanuel Prouff. A comprehensive study of deep learning for side-channel analysis. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(1):348–375, 2020.

[MKSM22] Nicolai Müller, David Knichel, Pascal Sasdrich, and Amir Moradi. Transitional leakage in theory and practice: Unveiling security flaws in masked circuits. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2:266–288, 2022.

[MMMS23] Loïc Masure, Pierrick Méaux, Thorben Moos, and François-Xavier Standaert. Effective and efficient masking with low noise using small-Mersenne-prime ciphers. In *Advances in Cryptology—EUROCRYPT 2023*, pages 596–627, 2023.

[MMS+23] Julien Maillard, Awaleh Houssein Meraneh, Modou Sarry, Christophe Clavier, Hélène Le Bouder, and Gaël Thomas. Blind side channel analysis on the Elephant LFSR extended version. In *E-Business and Telecommunications*, pages 20–42, 2023.

[MN98] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30, 1998.

[MOP07] Stefan Mangard, Elisabeth Oswald, and Thomas Popp. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer New York, 2007.

[MPO05] Stefan Mangard, Norbert Pramstaller, and Elisabeth Oswald. Successfully attacking masked AES hardware implementations. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 157–171, 2005.

[MPP16] Houssem Maghrebi, Thibault Portigliatti, and E. Prouff. Breaking cryptographic implementations using deep learning techniques. *Security, Privacy, and Applied Cryptography Engineering (SPACE)*, pages 3–26, 2016.

[MPW21] Ben Marshall, Dan Page, and James Webb. MIRACLE: MIcRo-ArChitectural Leakage Evaluation: A study of micro-architectural power leakage across many devices. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2022(1):175–220, 2021.

[MRS23] Loïc Masure, Olivier Rioul, and François-Xavier Standaert. A nearly tight proof of Duc et al.'s conjectured security bound for masked implementations. In *Smart Card Research and Advanced Applications*, pages 69–81, 2023.

[MSGR10] Marcel Medwed, François-Xavier Standaert, Johann Großschädl, and Francesco Regazzoni. Fresh re-keying: Security against side-channel and fault attacks for low-cost devices. In *Progress in Cryptology—AFRICACRYPT 2010*, volume 6055 of *Lecture Notes in Computer Science*, pages 279–296. Springer, 2010.

[MSJ12] Marcel Medwed, François-Xavier Standaert, and Antoine Joux. Towards super-exponential side-channel security with efficient leakage-resilient PRFs. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2012)*, volume 7428 of *Lecture Notes in Computer Science*, pages 193–212. Springer, 2012.

[NRS11] Svetla Nikova, Vincent Rijmen, and Martin Schläffer. Secure hardware implementation of nonlinear functions in the presence of glitches. *Journal of Cryptology*, 24(2):292–321, 2011.

[OSPG18] Tobias Oder, Tobias Schneider, Thomas Pöppelmann, and Tim Güneysu. Practical CCA2–secure and masked ring-LWE implementation. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):142–174, 2018.

[Pie09] Kryzsztof Pietzrak. A leakage-resilient mode of operation. In *Advances in Cryptology—EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 462–482, 2009.

[PMSN23] Srinidhi Hari Prasad, Florian Mendel, Martin Schläffer, and Rishub Nagpal. Efficient low-latency masking of ascon without fresh randomness. Cryptology ePrint Archive, Paper 2023/1914, 2023.

[PR13] Emmanuel Prouff and Matthieu Rivain. Masking against side-channel attacks: A formal security proof. In *Advances in Cryptology—EUROCRYPT 2013*, pages 142–159, 2013.

[PSV15] Olivier Pereira, François-Xavier Standaert, and Srinivas Vivek. Leakage-resilient authentication and encryption from symmetric cryptographic primitives. In *ACM SIGSAC Conference on Computer and Communications Security (CCS 2015)*, pages 96–108, 2015.

[RBN+15] Oscar Reparaz, Begül Bilgin, Svelta Nikova, Benedikt Gierlichs, and Ingrid Verbauwhede. Consolidating masking schemes. In *Advances in Cryptology—CRYPTO 2015*, pages 764–783, 2015.

[Rep16] Oscar Reparaz. Detecting flawed masking schemes with leakage detection tests. In *Fast Software Encryption (2016)*, volume 9783 of *Lecture Notes in Computer Science*, pages 204–222, 2016.

[RP10] Matthieu Rivain and Emmanuel Prouff. Provably secure higher-order masking of AES. In *International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 413–427, 2010.

[RSVC+11] Mathieu Renauld, François-Xavier Standeart, Nicolas Veyrat-Charvillon, Dina Kamel, and Denis Flandre. A formal study of power variability issues and side-channel attacks for nanoscale devices. In *Advances in Cryptology—EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 109–128, 2011.

[SA15] Dillibabu Shanmugam and Suganya Annadurai. Secure implementation of stream cipher: Trivium. In *SECITC*, pages 253–266, 2015.

[SCS+21] A. Madura Shelton, Łukasz Chmielewski, Niels Samwel, Markus Wagner, Lejla Batina, and Yuval Yarom. Rosita++: Automatic higher-order leakage elimination from cryptographic code. In *ACM SIGSAC Conference on Computer and Communications Security (CCS 2021)*, pages 685–699, 2021.

[SJB20] Siang Meng Sim, Dirmanto Jap, and Shivam Bhasin. DAPA: Differential analysis aided power attack on (non-)linear feedback shift registers. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2021(1):169–191, 2020.

[SM15] Tobias Schneider and Amir Moradi. Leakage assessment methodology: A clear roadmap for side-channel evaluations. In *International Wrokshop on Cryptographic Hardware and Embedded Systems (CHES)*, pages 495–513, 2015.

[SMY09] François-Xavier Standeart, Tal G. Malkin, and Moti Yung. A unified framework for the analysis of side-channel key recovery attacks. In *Advances in Cryptology—EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 443–461, 2009.

[SPY+10] François-Xavier Standaert, Olivier Pereira, Yu Yu, Jean-Jacques Quisquater, Moti Yung, and Elisabeth Oswald. Leakage resilient cryptography in practice. In Ahmad-Reza Sadeghi and David Naccache, editors, *Towards Hardware-Intrinsic Security: Foundations and Practice*, pages 99–134. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[SSB+21] A. Madura Shelton, Niels Samwel, Lejla Batina, Francesco Regazzoni, Markus Wagner, and Yuval Yarom. Rosita: Towards automatic elimination of power-analysis leakage in ciphers. In *Network and Distributed System Security Symposium (NDSS 2021)*, 2021.

[ST23] Ferdinand Sibleyras and Yosuke Todo. Keyed sum of permutations: A simpler RP-based PRF. In *Topics in Cryptology—CT-RSA 2023*, pages 573–593, 2023.

[Sug18] Takeshi Sugawara. 3-share threshold implementation of AES S-box without fresh randomness. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2019(1):123–145, 2018.

[TMC+24] Meltem Sönmez Turan, Kerry A. McKay, Donghoon Chang, Jinkeon Kang, and John Kelsey. Nist special publication 800 NIST SP 800-232 ipd: Ascon-based lightweight cryptography standards for constrained devices: Authenticated encryption, hash, and extendable output functionsg, 2024. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-232.ipd.pdf.

[Uen26] Rei Ueno. Side-channel attack on masked cryptographic implementation with non-uniform random numbers. In *56th IEEE International Symposium on Multiple-Valued Logic, ISMVL 2026, Sendai, Japan, May 19-21, 2026*, pages 222–226. IEEE, 2026.

[UHA17] Rei Ueno, Naofumi Homma, and Takafumi Aoki. Toward more efficient DPA-resistant AES hardware architecture based on threshold implementation. In *International Workshop on Constructive Side-Channel Analysis and Secure Design (COSADE 2017)*, volume 10348 of *Lecture Notes in Computer Science*, pages 50–64, 2017.

[UHIM23] Rei Ueno, Naofumi Homma, Akiko Inoue, and Kazuhiko Minematsu. Fallen sanctuary: A higher-order and leakage-resilient rekeying scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 1(2024):264–308, 2023.

[UHMA17] Rei Ueno, Naofumi Homma, Sumio Morioka, and Takafumi Aoki. Automatic generation of formally-proven tamper-resistant Galois-field multipliers based on generalized masking scheme. In *Design, Automation and Test in Europe Conference and Exhibition (DATE 2017)*, pages 978–983. IEEE, 2017.

[UHMA21] Rei Ueno, Naofumi Homma, Sumio Morioka, and Takafumi Aoki. A systematic design methodology of formally proven side-channel-resistant cryptographic hardware. *IEEE Design & Test*, 38(3):84–92, 2021.

[UIM+26] Rei Ueno, Akiko Inoue, Kazuhiko Minematsu, Akira Ito, and Naofumi Homma. A formal security proof of masking: Reduction from strong noisy leakage to probing model without random probing and application to LR primitive. In *Advances in Cryptology—CRYPTO 2026*, pages 326–358, 2026.

[VCGS14] Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert. Soft analytical side-channel attacks. In *Advances in Cryptology—ASIACRYPT 2014*, pages 282–296, 2014.

[WCG+22] Qianmei Wu, Wei Cheng, Sylvain Guilley, Fan Zhang, and Wei Fu. On efficient and secure code-based masking: A pragmatic evaluation. *IACR Trans. Cryptogr. Hardw. Embedded Syst.*, 2022(3):192–222, 2022.

[WMCS20] Weijia Wang, Pierrick Méaux, Gaëtan Cassiers, and François-Xavier Standaert. Efficient and private computations with code-based masking. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020(2):128–171, 2020.

[Yao82] Andrew C. Yao. Theory and applications of trapdoor functions. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.

[YSPY10] Yu Yu, François-Xavier Standaert, Olivier Pereira, and Moti Yung. Practical leakage-resilient pseudorandom generators. In *ACM SIGSAC Conference on Computer and Communications Security (CCS 2010)*, pages 141–151, 2010.

[ZM24] Jannik Zeitschner and Amir Moradi. PoMMES: Prevention of microarchitectural leakages in masked embedded software. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2024(3):342–376, 2024.

[ZMM23] Jannik Zeitschner, Nicolai Müller, and Amir Moradi. PROLEAD_SW: Probing-based software leakage detection for ARM binaries. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023(3):391–421, 2023.