Authenticated Encryption with Nonce Misuse and Physical Leakages: Definitions, Separation Results and Leveled Constructions*

Chun Guo*, Olivier Pereira*, Thomas Peters*, François-Xavier Standaert*
* ICTEAM/ELEN/Crypto Group, UCLouvain, Louvain-la-Neuve, Belgium

Abstract. We propose definitions and constructions of authenticated encryption (AE) schemes that offer security guarantees even in the presence of nonce misuse and side-channel leakages. This is part of an important ongoing effort to make AE more robust, while preserving appealing efficiency properties. Our definitions consider an adversary enhanced with the leakages of all the computations of an AE scheme, together with the possibility to misuse nonces, be it during all queries (in the spirit of misuse-resistance), or only during training queries (in the spirit of misuse-resilience recently introduced by Ashur et al.). These new definitions offer various insights on the effect of leakages in the security landscape. In particular, we show that, in contrast with the black-box setting, leaking variants of INT-CTXT and IND-CPA security do not imply a leaking variant IND-CCA security, and that leaking variants of INT-PTXT and IND-CCA do not imply a leaking variant of INT-CTXT. They also bring a useful scale to reason about and analyze the implementation properties of emerging modes of operation with different levels of leakage-resistance, such as proposed in the ongoing NIST lightweight cryptography competition. Eventually, we propose first instances of modes of operations that satisfy our most demanding definitions. In order to optimize their efficiency, we aim at modes that support "leveled implementations" such that the encryption and decryption operations require the use of a small constant number of evaluations of an expensive and heavily protected component, while the bulk of the computations can be performed by cheap and weakly protected block cipher implementations.

1 Introduction

Authenticated encryption (AE) has become the de-facto standard primitive for the protection of secure communications, by offering a robust and efficient alternative to the combination of encryption and MACs, a combination that is challenging enough to have been the source of security issues in numerous high-profile systems [APW09, DR11, PA12]. This effort towards robustness has been intensely pursued and, as a result, a number of strengthened requirements for AE schemes have been proposed in the literature.

A first focus has been on reducing functional requirements, in order to protect users from their failure to provide appropriate inputs to the system. The typical requirement of using random IVs has been lowered to the requirement of providing unique nonces. Further efforts have then been made to reduce the impact of a repeated nonce, by requiring that such a repetition only makes it possible to recognize the repetition of a message, which is the strict minimal consequence. These considerations led Rogaway and Shrimpton to define the central notion of misuse-resistant nonce-based AE [RS06], which goes even one step further, by requiring ciphertexts to be indistinguishable of random strings. Satisfying this notion of misuse-resistance is extremely appealing, as it goes as far as possible in protecting users from their own mistakes or from devices offering poor sources of randomness. However, it comes with a significant memory penalty (two successive passes are needed to perform encryption) and, as we will argue, may also be infeasible to achieve in the presence of many natural types of leakages (e.g., based on the power consumption or electronmagnetic radiation

* An extended abstract is published in the proceedings of LATINCRYPT 2019.

of an implementation). More recently, Ashur et al. [ADL17] proposed the relaxed security notion of misuse-resilience, which requires that nonce misuse does not have an impact on the security of messages encrypted with a fresh nonce. This notion can be satisfied by much more efficient schemes, and we will show that it is also compatible with the side-channel attack scenarios mentioned above.

A second line of efforts aims at protecting from weaknesses that implementers could introduce in an AE scheme, by creating observable behaviors that are not part of its specifications. One type of implementation weakness comes from the decryption of invalid ciphertexts [BDPS13, ABL$^+$14, HKR15, MOSW15, ADL17]. While security models usually assume that the decryption of an invalid ciphertext returns an error signal, the reality is often different, and some implementations return different messages depending on the step at which decryption fails, or would even go as far as releasing the partially decrypted message to the adversary, either explicitly, or by treating it as public garbage.

Another source of weakness coming from implementations is the possibility of side-channel attacks [BKP$^+$18, BMOS17, BPPS17]. Here, the attacker does not (only) exploit explicit software messages, but extracts information from side-effects such as the computation time, the power consumption, or the electromagnetic radiation of the device performing cryptographic operations. In this context, the previous focus on decryption failures must be broadened, as side-channel leakages happen at encryption and decryption times, and happen at decryption time whether a ciphertext is valid or not.

What can be achieved in the presence of leakages of course depends on the type, the permanence and the amount of leakages granted to the adversary. As far as the type is concerned, we separate scalar leakages (like timing) that allow so-called univariate attacks and vector leakages (like the aforementioned power consumption or electromagnetic radiation) that allow so-called multivariate attacks. As far as the permanence is concerned, we separate between full leakages (that allow to leak during all interactions with the device) and partial leakages that exclude the leakages of some interactions. This leads us to define a first taxonomy of leakages as Vector & Full (VF), Scalar & Full (SF), Vector and Partial (VP), Scalar & Partial (SP). In the rest of this paper, we are concerned with the strongest category of VF leakages.

As for the amount of leakage, it is in general hard to quantify and highly depends on the implementations and measurement devices that are at hand. In this respect, our starting observation is that the leakage of all secrets makes confidentiality-preserving cryptography impossible, but the full protection against leakages at the implementation level brings us back to a situation in which we put a lot of pressure on implementers, who we must completely trust to limit the leakages. Furthermore, even in that case, this will come at high cost in terms of extra computation time, energy, or circuit area, since strong protections against side-channel attacks (especially in the case of VF leakages) typically increase the "code size $\times$ cycle count" metric by 2 or 3 orders of magnitude compared to a non-protected implementation [BGS15, GR17].

This state-of-the-art motivated the design of authentication, encryption and AE schemes allowing "leveled implementations" (or implementations in the leveled leakage setting). By leveled implementations, we mean that different levels of security are required for different parts of the computations: some computations must be well protected, while a weaker protection would be sufficient for others. As put forward in [PSV15, BKP$^+$18, BPPS17], this setting usually allows lower cost or more efficient implementations with symmetric building blocks.

The design of such schemes being guided by the security definitions to target, it can be viewed as a tradeoff between the pressure on implementers to limit the leakages and the pressure on the

modes to deal with the remaining leakages. Our following contributions therefore aim at defining security targets and modes of operation with an effective balance between leakage reduction at the implementation level and leakage-resistance at the cryptographic mode level, in order to reach AE with high physical security and a minimum implementation cost.

1.1 Contributions

Our main contributions target deterministic (nonce-based) authenticated encryption with associated data (AEAD) [Rog02]. We:

(i) Define confidentiality and integrity notions in the presence of nonce misuse and leakages. Our security notions capture leakages in encryption and decryption and allow the computation of the "challenge ciphertexts" to leak. Several variants are explored and put in relation.

(ii) Identify the strongest form of security that an AEAD can offer to protect messages in the presence of nonce misuse and VF leakage, which we call $\mathsf{AEML\text{-}VF}$ security, as a combination of black-box misuse-resistance, ciphertext integrity with VF leakage and misuse-resistance and CCA security with VF leakage, and misuse-resilience. We also argue why CCA security with VF leakage and misuse-resistance cannot be achieved.

(iii) Propose modes of operation that satisfy our definitions and allow leveled implementations, using a set of weaker assumptions for the integrity part of the definition and stronger ones for its confidentiality part, leading to "gradual security degradations": even if some of the security requirements for confidentiality are not satisfied, integrity may be preserved.

Inspired by the misuse-resistance vs. misuse-resilience terminology, we denote as leakage-resistant the modes that aim to cope with full leakage, and as leakage-resilient the modes that aim to cope with partial leakage.

Security definition. Our definition of $\mathsf{AEML\text{-}VF}$ security (written more simply as AEML security when VF is understood) is a combination of three requirements: (i) The AE scheme must be misuse-resistant (MR) in the black-box setting (without leakages), in the usual sense of Rogaway and Shrimpton [RS06]. (ii) The AE scheme must offer $\mathsf{CIML2}$ security, which is a natural extension of ciphertext integrity and nonce misuse-resistance in the presence of (full) leakages, introduced by Berti et al. [BKP+18, BPPS17]. (iii) The AE scheme must offer $\mathsf{CCAmL2}$ security, which is an extension of CCA security with nonce misuse-resilience in the presence of (full) leakages that we propose here. Misuse-resilience and full leakages are reflected by the small m and large L in these notations.

The first requirement is there to ensure that, for someone who does not have access to leakages, an AEML scheme is also a traditional MR AE scheme.

In the presence of leakages, we unfortunately cannot just easily extend the Rogaway-Shrimpton definition of MR AE in any natural way that would uniformly combine confidentiality and integrity. Indeed, their definition requires that ciphertexts look random as soon as they are produced from a fresh (nonce, message) pair. But defining the leakage function corresponding to the generation of such random-looking ciphertexts is difficult, since the very definition of this function is implementation-dependent. In order to avoid this caveat, we therefore turn back to the original definitional approach for AE security, as a combination of confidentiality and ciphertext integrity, which we gradually extend to the leakage world in the presence of nonce misuse. Such a combination turns out to be especially relevant in the leakage setting where ensuring confidentiality and integrity may benefit from different types of physical assumptions.

The extension of INT-CTXT (the hardness to forge a fresh ciphertext that would pass decryption) to the setting of misuse and leakages has been recently proposed as the CIML2 notion [BPPS17]. (The same notion excluding decryption leakages has been proposed beforehand in [BKP+18] - we denote it as CIML1). This extension can be viewed as natural as it provides the adversary with full nonce misuse capabilities (as in the black box setting) and leakages from all the computations performed in encryption and decryption. Furthermore and as will be detailed next, it can be obtained under quite mild leakage assumptions, making it a particularly desirable property to reach in practice.

It would be tempting to complete this picture with an extension of CPA security to the misuse and leakage setting, e.g., based on the notions defined/used in [SPY13, BKP+18]. However, this leads to guarantees that are weaker than the theoretical limits. While INT-CTXT + IND-CPA implies (the desired) IND-CCA security in the black box setting [BN08], we show that this is not true anymore when leakages enter the picture: the implication towards an extension of CCA security with leakages does not hold, mainly because it does not capture the risks associated to decryption leakages.

With this difficulty in mind, we introduce the notion of CCAmL2 security as an extension of CCA security that also offers nonce misuse-resilience [ADL17] in the presence of full leakages: as long as the nonce used in the test query is fresh, confidentiality must hold. Besides the aforementioned separation, we also show that leaking variants of INT-PTXT and CCAmL2 do not imply CIML2 security: the alternate definition of AE as INT-PTXT and IND-CCA security [KY00] does not suffice in the leaking setting either. By the above, we propose to use CCAmL2 in combination with CIML2 (and MR) to define AEML-VF security.

Finally, the reason of our focus on nonce misuse-resilience for CCAmL2 security, rather than on the stronger requirement of nonce misuse-resistance, is due to the nature of VF leakages. Concretely, if an implementation of an AE scheme processes a message block-by-block, as it is standard, the leakages happening during the processing of the first blocks will only depend on these blocks, and not on all blocks. So, if an adversary asks for an encryption of two messages that have identical first blocks (but differ otherwise), using a single nonce, the leakages of the computation associated to these first blocks will be highly similar, something that can be easily observed and trivially contradicts misuse-resistance. Nonce misuse-resilience is then the natural form of protection against misuse that can be aimed for in the VF setting. Note that this argument may not hold for scalar leakages (i.e., in the SF of SP settings): if there is a single scalar leaked for a full message encryption process, then that scalar may depend on the full message, not just on its first blocks. It also does not hold in the VP setting since, in that case, the security game does not provide the adversary with the challenge leakages that can help her to distinguish.

New modes of operation. Based on the above definitions, we propose new modes of operation for which the driving motivation, and our choice of leakage models, is to push towards the most effective balance between the pressure on implementers and the pressure on designers of modes (i.e., trading more complicated leakage-resistant modes for simpler implementations).

Traditional (non leakage-resistant) encryption modes, when intended to be used in a VF leakage setting, require implementers to offer an implementation that can at least withstand so-called Differential Power Analysis attacks (DPAs). Informally, DPAs are the most commonly exploited side-channel attacks and take advantage of the leakages about a secret (e.g., key) obtained from computations based on different inputs [MOS11]. A DPA reduces the computational secrecy of the

state manipulated by a device at a rate that is exponential in the number of leakages, by combining the information of these different inputs (e.g., plaintexts).

Leakage-resistant modes have the potential to considerably lower the costs of physical protection (in terms of time/energy/code size needed to perform an encryption) by removing, to a large extent, DPAs from the attack surface (mostly via consistently refreshing internal secrets, so that it's impossible to collect multiple traces), leaving the adversary with the more challenging task of exploiting leakage with Simple Power Analysis attacks (SPAs). SPAs are side-channel attacks taking advantage of the leakage of a single input, possibly measured multiple times to reduce the measurement noise, and therefore correspond to a minimum threat that targets the unavoidable manipulation of the messages to encrypt/decrypt. SPA protection is considerably cheaper than DPA protection (see [DEM+17]), and it is expected that the overhead coming from the use of a more demanding encryption mode (e.g., requiring more block cipher calls per message block) can be compensated by the cheaper physical protections against SPA. This concern is reflected in the assumptions that we make about leakages. They differ in strength depending on the security property that we analyze but, based on these assumptions, any mode offering the possibility to mount a traditional DPA, message block by message block, would be considered insecure.

We define two modes of operation. Our first mode, FEMALE (for Feedback-based Encryption with Misuse, Authentication and LEakages), offers AEML-VF security. The second one, AEDT, is CIML2 and CCAmL2 but not MR, a combination which we denote as AEmL-VF (where the lower-case m indicates that black-box misuse-resistance is abandoned – still, misuse-resilience is offered). It is proposed for situations in which the processing of the message in two passes (inherent to misuse-resistance) leads to excessive memory requirements.

FEMALE is a two-pass encryption scheme offering traditional AE security, which is compatible with leveled implementations: independently of the length of the message and associated data, a strongly protected block cipher (BC) must be called only twice – which we model as a "leak-free" block cipher for simplicity. Apart from that, a weakly protected BC must be called $4\ell + 5$ times for a message of ℓ blocks. The MR security of FEMALE holds in the standard model. The CIML2 security holds in the unbounded leakage model [BPPS17], which lets weakly protected components leak their state completely, hence only relying on the two heavily protected blocks. The CCAmL2 security holds based on the assumption that leakages are simulatable [SPY13]. Our reduction shows that any attacker that breaks the CCAmL2 security of FEMALE is either violating the simulatable leakage assumption, or able to break the eavesdropper security of a very simple encryption scheme that can only encrypt one single block and has no leak-free component (so in some sense we *extend the domain* of the single block encryption *in a security-preserving manner* and *enhance functionality*). The single-block security formally reflects the aforementioned "minimum attack surface" that our modes aim to provide. It cannot be derived from the simulatable leakage assumption: the leakage of one single bit of a secret message may not contradict simulatability but break confidentiality [PSV15]. So this approach is as far as one can go based on the current understanding of leakage-resistance and the fact that accurately defining the confidentiality guarantees offered by a single message block remains an open problem. Still, it makes the application of DPA attacks impossible by design (excepted for the calls to a strongly protected block, which we make two times per message to be encrypted, independently of the length of this message). Also, testing the eavesdropper security of a single-block encryption scheme is a much simpler target for a side-channel evaluation laboratory than evaluating the CCAmL2 security of a fully fledged scheme.

AEDT is a simple variant of the EDT scheme of Berti et al. [BPPS17] which was designed to support leveled implementations. It is already known to be CIML2 secure, and we show that AEDT also offers CCAmL2 security (the combination of which gives AEmL-VF). Giving up on black-box MR security leads to efficiency improvements: AEDT requires two calls of leak-free BC, but only $2\ell - 1$ calls of a weakly protected BC, and the evaluation of a hash function on public values. We note that, if the hash function proceeds by blocks (as any function based on the Merkle-Damgård transformation or sponge constructions), then AEDT does not require any latency for producing ciphertexts, and reduces the memory complexity for encryption to constant instead of requiring the storage of the message in full as for traditional MR. Still, and as usual, the full ciphertext is needed before decryption can start. Combining leakage-resistance with modes like in the works of Fleischmann et al. [FFL12] and Hoang et al. [HRRV15] on on-line misuse-resistance, which explore weakened notions of misuse-resistance that are compatible with single-pass encryption (but do not consider the presence of leakages) is an interesting scope for further research.

The properties achieved by these constructions are summarized in Table 1. We do not mention standard modes that are not designed for leakage-resistance since they do not provide any of these properties (e.g., the CIML1 security of the SIV construction [RS06] is broken with a DPA in [BKP$^+$18]).

Table 1. Existing AE designs that support leveled leakage-resistant implementations. LR hash indicates whether the scheme relies a hash functions that's resistant to some leakages. AD indicates whether the scheme supports the processing associated data.

	LR hash	LF calls	AD	CIML	CCAmL2	MR	Ref.
DTE	✓	2	×	1	×	✓	[BKP$^+$18]
DTE2	✓	2	×	2	×	✓	[BPPS17]
EDT	×	2	×	2	✓	×	[BPPS17]
FEMALE	×	2	✓	2	✓	✓	new
AEDT	×	2	✓	2	✓	×	new

Remark on gradual security degradations. As encouraged by our composite definitions, the security proofs of FEMALE and AEDT are obtained under physical assumptions that differ depending on whether we target confidentiality or integrity guarantees, as outlined above, and as will be carefully discussed in Section 5. In this respect, it is important to note that our definitions of AEML and AEmL security allow expressing gradual security degradations in the sense (present in our modes) that imperfectly simulatable leakages or strongly leaking single-block encryptions (which would harm CCAmL2 security) do not affect the CIML2 guarantees. In other words, CIML2 and CCAmL2 should be seen as gradual improvements that modes of operation can bring to better cope with leakages, with a risk of non-negligible adversarial advantages (especially for CCAmL2) that are in the same time inherent to physical security issues, and significantly reduced compared to modes of operations ignoring physical leakages in their design.

1.2 Related works

Recently, Barwell et al. [BMOS17] introduced notions of misuse-resistant and leakage-resilient AE, and proposed modes of operation satisfying their definitions. We will refer to this work as BMOS, by the initial of its authors. The BMOS definition captures leakage-resilient AE security as follows (we just focus on encryption queries for simplicity): they first follow the Rogaway-Shrimpton strategy

by challenging the adversary to distinguish between non-leaking real or random encryption oracles, then augment the power of the adversary by giving him access to a leaking encryption oracle that cannot be queried with inputs identical to those of the non-leaking oracles. As a result, it is impossible to win their game by exploiting leakages that would reveal information about the messages that the AE scheme is supposed to hide: leakages are excluded from the "challenge" queries. In our terminology, BMOS focuses on the VP (Vector & Partial) leakage model, which we reflect with a small l in our notations.

As a result of the weaker VP leakage model, BMOS can realistically require misuse-resistance to hold for confidentiality, and not only misuse-resilience (i.e., CCAMl1, CCAMl2), which is in line with our previous discussions. As such, the BMOS work can be viewed as dual to ours: we consider full leakages (i.e., leakage-resistance) and as a consequence have to exclude full misuse-resistance; they consider partial leakages (i.e., leakage-resilience) which makes misuse-resistance possible. On the positive side, the BMOS authors show that their definition is compatible with strong composition results. Yet, the VP model may be insufficient in many practical cases: according to the BMOS definition, an implementation that leaks plaintexts in full during encryption may be considered as perfectly secure. By contrast, such an implementation would be considered completely insecure in the VF model, which is our focus here.

A secondary difference is on the modes of operation of BMOS, which require "uniform implementations" (or an implementations in the uniform leakage setting). In contrast with the leveled implementations we propose, all the components in their constructions are required to offer the same strong level of protection against side-channel attacks. This is expected to lead to considerably more expensive implementations. For example, the proposed implementation strategy of the BMOS modes of operation processes each message block with a pairing-based leakage-resilient PRF (while we leverage symmetric building blocks).

Note that in general, our approach is motivated by the observation that restricting the misuse capabilities may be easier to control by implementers than excluding some leakages (which usually depends on when the target device is monitored – a hard to control parameter). Our following results therefore focus on this important (and so far unexplored) part of the problem, and analyze how to deal with adversaries having full leakage capabilities.

Our CCAmL2 definition builds on the notion of misuse-resilience introduced by Ashur, Dunkelman and Luykx [ADL17]. The actual definitions and their motivations are quite different, though. Ashur et al. introduce misuse-resilience to offer a finer grained evaluation of several standard AE schemes that are not misuse-resistant. They do not consider side-channel leakages. In contrast, these leakages are the central concern of our security definitions, and they are our motivation for departing from traditional misuse-resistance in the VF leakage model.

Eventually, we mention the line of works about "after-the-fact" leakages which is complementary to ours [HL11] and allows the adversary to obtain leakage information after the challenge ciphertext. While the latter is meaningful in certain scenarios (e.g., in the context of a cold boot attack [HSH+09], the adversary could first see the encrypted disk – hence getting access to the ciphertext – and then try to design a method of measuring the memory for the purpose of decrypting this ciphertext), it still excludes the leakages during the challenge phase, as will be available in the context of a side-channel attack based on power consumption leakages, which is our main concern here.

1.3 Paper structure

After introducing preliminary notions in Sec. 2, we introduce, in Sec. 3, our definitions of CCAmL2, AEML and AEmL security. These definitions are analyzed and compared with other leaking variants of standard authentication, integrity and confidentiality guarantees in Section 4. (These results are extended to the multi-challenge setting in Appendix A). We then discuss our model and assumptions on leakages in Section 5, and conclude by describing and proving the AEML (resp., AEmL) security of FEMALE (resp., AEDT) in Section 6.

2 Preliminaries

Throughout the paper n denotes the security parameter. For any set S, we define $S^* = \cup_{i=1}^{\infty} S^i$ so that $s \in S^*$ if and only if it exists a non negative integer ℓ such that $s \in S^\ell$. This notation is convenient for instance for variable block-length (authenticated) encryption with message space $\mathcal{M}^*$, where messages of $\mathcal{M}$ consist of one block.

2.1 Notations

Adversary. We denote by a $(q_1, \ldots, q_\omega, t)$-bounded adversary a probabilistic algorithm that has access to ω oracles, can make at most q_i queries to its i-th oracle, and can perform computations bounded by running time t. For algorithms that have no oracle to access, we simply call them t-bounded. In this paper, we use subscripts to make a clear distinction between the number of queries to different oracles: the number of queries to the (authenticated) encryption oracle, decryption oracle, and leakage oracle L are denoted by q_e, q_d, and q_l respectively. For example, a (q_e, q_d, q_l, t)-bounded adversary runs in time t, makes q_e and q_d queries to the encryption and decryption oracles of the Authenticated Encryption with Associated Data (AEAD) scheme respectively, and makes q_l additional queries to the leakage oracle L.

Leaking algorithm. For an algorithm Algo, a leaking version is denoted LAlgo. It runs both Algo and a *leakage function* $\mathsf{L}_{\mathsf{algo}}$ which captures the additional information given by an implementation of Algo during its execution. LAlgo returns the outputs of both Algo and $\mathsf{L}_{\mathsf{algo}}$ which all take the same input.

2.2 Definitions of primitives

We first need the following definition of collision-resistant hash function.

Definition 1 (Collision-Resistant Hash Function). *A (t, ε_{cr})-collision resistant hash function $H : \mathcal{S} \times \{0,1\}^* \to \mathcal{B}$ for security parameter n is a function that is such that, for every t-bounded adversary $\mathcal{A}$, the probability that $\mathcal{A}(1^n, s)$ outputs a pair of distinct bit-strings (m_0, m_1) such that $H_s(m_0) = H_s(m_1)$ is bounded by ε_{cr}, where $s \xleftarrow{\$} \mathcal{S}$ is selected uniformly at random.*

We next need the following definition of range-oriented preimage resistance.

Definition 2 (Range-Oriented Preimage-Resist. Hash Function [AS11]). *A (t, ε_{pr})-range-oriented preimage resistant hash function $H : \mathcal{S} \times \{0,1\}^* \to \mathcal{B}$ for security parameter n is a function such that, for every adversary $\mathcal{A}$ running in time t, the probability that $\mathcal{A}(1^n, s, y)$ outputs a bit-string m such that $H_s(m) = y$ is bounded by ε_{pr}, where $s \xleftarrow{\$} \mathcal{S}$, $y \xleftarrow{\$} \mathcal{B}$ are selected uniformly at random.*

In the following, we assume that the key s is not private, and refer to the hash function simply as H for simplicity, the key s being implicit.

We also need the following definitions of pseudorandom permutation.

Definition 3 (Pseudorandom Function). *A function $\mathsf{F} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ is a $(q, t, \varepsilon_{\mathsf{F}})$-pseudorandom function (PRF) for a security parameter n if, for all (q, t)-bounded adversaries $\mathcal{A}$, we have:*

$$\left|\Pr\left[k \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathsf{F}_k}(1^n) \Rightarrow 1\right] - \Pr\left[f \xleftarrow{\$} \mathcal{F} : \mathcal{A}^{f}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon_{\mathsf{F}},$$

where $\mathcal{F}$ denotes the set of all functions from $\mathcal{M}$ to $\mathcal{M}$.

The *strong* pseudorandom permutation notion is as follows.

Definition 4 (Strong Pseudorandom Permutation). *A function $\mathsf{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ is a $(q, t, \varepsilon_{\mathsf{E}})$-strong pseudorandom permutation (SPRP) for a security parameter n if, for all (q, t)-bounded adversaries $\mathcal{A}$, we have:*

$$\left|\Pr\left[k \xleftarrow{\$} \mathcal{K} : \mathcal{A}^{\mathsf{E}_k, \mathsf{E}_k^{-1}}(1^n) \Rightarrow 1\right] - \Pr\left[P \xleftarrow{\$} \mathcal{P} : \mathcal{A}^{P, P^{-1}}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon_{\mathsf{E}},$$

where $\mathcal{P}$ denotes the set of all permutations on $\mathcal{M}$.

We will focus on authenticated encryption with the following formalism.

Definition 5 (Nonce-Based AEAD [Rog02]). *A* nonce-based authenticated encryption scheme with associated data *is a tuple $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ such that, for any security parameter n, and keys in $\mathcal{K}$ generated from $\mathsf{Gen}(1^n)$:*

- $\mathsf{Enc} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{M} \to \mathcal{C}$ *deterministically maps a key selected from $\mathcal{K}$, a nonce value from $\mathcal{N}$, some blocks of associated data selected from $\mathcal{AD}$, and a message from $\mathcal{M}$ to a ciphertext in $\mathcal{C}$.*
- $\mathsf{Dec} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ *deterministically maps a key from $\mathcal{K}$, a nonce from $\mathcal{N}$, some associated data from $\mathcal{AD}$, and a ciphertext from $\mathcal{C}$ to a message in $\mathcal{M}$ or to a special symbol $\perp$ if integrity checking fails.*

The sets $\mathcal{K}, \mathcal{N}, \mathcal{AD}, \mathcal{M}, \mathcal{C}$ are completely specified by n. Given a key $k \leftarrow \mathsf{Gen}(1^n)$, $\mathsf{Enc}_k(N, A, M) := \mathsf{Enc}(k, N, A, M)$ and $\mathsf{Dec}_k(N, A, M) := \mathsf{Dec}(k, N, A, M)$ are deterministic functions whose implementations may be probabilistic.

Since we only focus on nonce-based authenticated encryption with associated data in this paper, we will simply refer to it as *authenticated encryption*.

2.3 Security definitions

We first recall the definition of Misuse-Resistance (MR) formalized in [RS06].

Definition 6 (MR). *A nonce-based authenticated encryption scheme with associated data $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is $(q_e, q_d, t, \varepsilon)$ misuse resistant for a security parameter n if, for all (q_e, q_d, t)-bounded adversaries $\mathcal{A}$,*

$$\left|\Pr\left[k \xleftarrow{\$} \mathsf{Gen}(1^n) : \mathcal{A}^{\mathsf{Enc}_k, \mathsf{Dec}_k}(1^n) \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\$, \perp}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon,$$

where $\$(N, A, M)$ outputs and associates a fresh random ciphertext $C \leftarrow \mathcal{C}$ of appropriate length to fresh inputs, and the associated C otherwise, and $\perp(N, A, C)$ outputs $\perp$ except if C was associated to (N, A, M) for some message M, in which case it returns M.

3 AE with full vectorial leakages

In order to define AEML security, our proposed security notion for authenticated encryption in the full vectorial leakage setting in the presence of nonce misuse, we start by extending the existing black-box security notions to the leakage setting. Surprisingly, the combination of our strongest extensions of confidentiality and integrity is separated from *any other* combinations unlike the situation without misuse and leakages. This motivates our definition to be at least as secure as this strongest combination.

3.1 Variants of black-box notions with misuse and leakages

We first adapt the IND-CPA and the IND-CCA confidentiality notions of nonce-based authenticated encryption in the setting of nonce misuse and leakages. We focus then on the extension of the INT-PTXT and INT-CTXT integrity notions.

Confidentiality Contrary to existing confidentiality notions in a leaking setting (e.g., [BMOS17]), our definition includes leakages during encryption and decryption even on the challenge ciphertext(s). We first focus on security against chosen-ciphertext attacks with misuse-resilience and leakages, denoted CCAmL2. Then, we derive the weaker notion of security against chosen-plaintext attacks with misuse-resilience and leakages, denoted CPAmL2.

Chosen-ciphertext security with misuse and leakages. To capture the CCAmL2 security, as it is motivated in the introduction, we define the game $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ detailed in Figure 1. This game takes as parameters an adversary $\mathcal{A}$, a nonce-based authenticated encryption AEAD and a (possibly probabilistic) leakage function pair $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ resulting from the implementation of the scheme. During $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$, the adversary $\mathcal{A}$ has to guess the bit b of which depend the challenge ciphertext C^b and the encryption leakages $\mathsf{leak}^b_{\mathsf{enc}}$, computed as a leaking encryption of (N_{ch}, A_{ch}, M^b), where the nonce N_{ch}, the associated data A_{ch} and the messages M^0, M^1 are chosen by $\mathcal{A}$ under certain conditions. All along this game $\mathcal{A}$ is also granted unbounded and adaptive access to three types of oracles: LEnc, a leaking encryption oracle; LDec, a leaking decryption oracle; and $\mathsf{L}_{\mathsf{decch}}$, a challenge decryption leakage oracle that provides the leakage of the decryption process, but not the resulting plaintext.

Overall, this definition follows the general pattern of CCA security. In terms of misuse-resilience, it only forbids the adversary to reuse a nonce N_{ch} in its challenge query. This captures real situations where, for instance, a counter providing the nonce has been unintentionally reset or shifted. As long as the counter recovers increments and provides fresh nonce values, the security of the challenges should remain unaltered even if the previous encryptions leaked. In terms of leakage-resistance, both the encryption and the decryption oracles leak (hence the "2" of CCAmL2 for the two leaking oracles), including during the challenge query. We go one step further with the $\mathsf{L}_{\mathsf{decch}}$ oracle, which offers the leakages corresponding to the decryption of the challenge ciphertext (but of course not the corresponding plaintext, as it would offer a trivial win). This addition captures the fact that the adversary may be allowed to observe the decryption of this challenge ciphertext through side-channels, which might be valuable in applications such as secure bootloading or firmware update with a device controlled by $\mathcal{A}$ [OC15] (see [BPPS17] for more discussion). We let the adversary query the $\mathsf{L}_{\mathsf{decch}}$ oracle multiple times, as leakages can be non-deterministic (e.g., contain measurement

$\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ is the output of the following experiment:

Initialization: generates a secret key $k \leftarrow \mathsf{Gen}(1^n)$ and sets $\mathcal{E} \leftarrow \emptyset$

Pre-challenge queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{LEnc}(\cdot,\cdot,\cdot)$ and $\mathsf{LDec}(\cdot,\cdot,\cdot)$

(1) $\mathsf{LEnc}(N,A,M)$ computes $C \leftarrow \mathsf{Enc}_k(N,A,M)$ and $\mathsf{leak_e} \leftarrow \mathsf{L_{enc}}(k,N,A,M)$ updates $\mathcal{E} \leftarrow \mathcal{E} \cup \{N\}$ and finally returns $(C, \mathsf{leak_e})$

(2) $\mathsf{LDec}(N,A,C)$ computes $M \leftarrow \mathsf{Dec}_k(N,A,C)$ and $\mathsf{leak_d} \leftarrow \mathsf{L_{dec}}(k,N,A,C)$ and returns $(M, \mathsf{leak_d})$ – we stress that $M = \perp$ may occur

Challenge query: on a single occasion $\mathcal{A}^{\mathsf{L}}$ submits a tuple $(N_{\mathsf{ch}}, A_{\mathsf{ch}}, M^0, M^1)$
If M^0 and M^1 have different (block) length or $N_{\mathsf{ch}} \in \mathcal{E}$ return $\perp$
Else compute $C^b \leftarrow \mathsf{Enc}_k(N_{\mathsf{ch}}, A_{\mathsf{ch}}, M^b)$ and $\mathsf{leak}^b_{\mathsf{e}} \leftarrow \mathsf{L_{enc}}(k,N_{\mathsf{ch}},A_{\mathsf{ch}},M^b)$
and return $(C^b, \mathsf{leak}^b_{\mathsf{e}})$

Post-challenge queries: $\mathcal{A}^{\mathsf{L}}$ can keep accessing LEnc and LDec with some restrictions but it can also get an unlimited access to $\mathsf{L_{decch}}$

(3) $\mathsf{LEnc}(N,A,M)$ returns $\perp$ if $N = N_{\mathsf{ch}}$ otherwise computes $C \leftarrow \mathsf{Enc}_k(N,A,M)$ and $\mathsf{leak_e} \leftarrow \mathsf{L_{enc}}(k,N,A,M)$ and finally returns $(C, \mathsf{leak_e})$

(4) $\mathsf{LDec}(N,A,C)$ returns $\perp$ if $(N,A,C) = (N_{\mathsf{ch}}, A_{\mathsf{ch}}, C^b)$ otherwise computes $M \leftarrow \mathsf{Dec}_k(N,A,C)$ and $\mathsf{leak_d} \leftarrow \mathsf{L_{dec}}(k,N,A,C)$ and returns $(M, \mathsf{leak_d})$

(5) $\mathsf{L_{decch}}$ outputs the leakage trace $\mathsf{leak}^b_{\mathsf{d}} \leftarrow \mathsf{L_{dec}}(k,N_{\mathsf{ch}},A_{\mathsf{ch}},C^b)$ of the challenge

Finalization: $\mathcal{A}^{\mathsf{L}}$ outputs a guess bit b' which is defined as the output of the game

Fig. 1: The $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ game.

noise), and $\mathcal{A}$ may benefit from the observation of leakages from multiple decryptions of the same plaintext.

Here, we focus on a single challenge definition but we extend it to the multi-challenge setting in Appendix A.1 where we establish their equivalence.

Definition 7 (CCAmL2). *A nonce-based authenticated encryption with associated data* $\mathsf{AEAD} = (\mathsf{Gen},\mathsf{Enc},\mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L_{enc}},\mathsf{L_{dec}})$ *is* $(q_e,q_d,q_c,q_l,t,\varepsilon)$*-*$\mathsf{CCAmL2}$ *secure for a security parameter* n *if, for every* (q_e,q_d,q_c,q_l,t)*-bounded adversary* $\mathcal{A}^{\mathsf{L}}$,[1] *we have:*

$$\left|\Pr\left[\mathsf{PrivK}^{\mathsf{CCAmL2},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{CCAmL2},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon,$$

where the adversary $\mathcal{A}^{\mathsf{L}}$ *makes at most* q_e *leaking encryption queries,* q_d *leaking decryption queries,* q_c *challenge decryption leakage queries and* q_l *leakage evaluation queries on arbitrarily chosen keys.*

We will sometimes refer to $\mathsf{CCAmL2}^*$ as a weakened version of $\mathsf{CCAmL2}$ where we drop the challenge decryption leakage oracle $\mathsf{L_{decch}}$ from the game of Figure 1.

Chosen-plaintext security with misuse and leakages. Derived from $\mathsf{CCAmL2}$, $\mathsf{CPAmL2}$ is defined by a game $\mathsf{PrivK}^{\mathsf{CPAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ which is exactly as $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$ except that we remove $\mathcal{A}$'s access to the leaking decryption oracle LDec in Figure 1 (Items 2,4). Yet, $\mathcal{A}$ is still able to get challenge decryption leakages $\mathsf{leak}^b_{\mathsf{d}}$—this corresponds to settings in which a passive adversary tries to break confidentiality while being able to observe leakages of encryption and decryption operations. In section 4.1, we will also introduce the $\mathsf{CPAmL1}$ security notion, which can be seen as the $\mathsf{CPAmL2}$

[1] The notation of $\mathcal{A}^{\mathsf{L}}$ indicates that the adversary may query L on chosen inputs including *chosen keys* selected and known by $\mathcal{A}$.

$\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ experiment	$\mathsf{PrivK}^{\mathsf{PIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ experiment
Initialization: 1. $k \leftarrow \mathsf{Gen}(1^n)$, $\mathcal{S} \leftarrow \emptyset$ *Finalization*: 1. $(N, A, C) \leftarrow \mathcal{A}^{\mathsf{LEnc}_k, \mathsf{LDec}_k, \mathsf{L}}(1^n)$ 2. If $(N, A, C) \in \mathcal{S}$, return 0 3. If $\mathsf{Dec}_k(N, A, C) = \perp$, return 0 4. Return 1 *Leaking encryption:* $\mathsf{LEnc}_k(N, A, M)$ 1. $C \leftarrow \mathsf{Enc}_k(N, A, M)$ 2. $\mathcal{S} \leftarrow \mathcal{S} \cup \{(N, A, C)\}$ 3. Return $(C, \mathsf{L}_{\mathsf{enc}}(k, N, A, M))$ *Leaking decryption:* $\mathsf{LDec}_k(N, A, C)$ 1. Return $(\mathsf{Dec}_k(N, A, C), \mathsf{L}_{\mathsf{dec}}(k, N, A, C))$	*Initialization*: 1. $k \xleftarrow{\$} \mathcal{K}$, $\mathcal{S} \leftarrow \emptyset$ *Finalization*: 1. $(N, A, C) \leftarrow \mathcal{A}^{\mathsf{LEnc}_k, \mathsf{LDec}_k, \mathsf{L}}(1^n)$ 2. $M \leftarrow \mathsf{Dec}_k(N, A, C)$ 3. If $M = \perp$ or $(A, M) \in \mathcal{S}$, return 0 4. Return 1 *Leaking encryption:* $\mathsf{LEnc}_k(N, A, M)$ 1. $C \leftarrow \mathsf{Enc}_k(N, A, M)$ 2. $\mathcal{S} \leftarrow \mathcal{S} \cup \{(A, M)\}$ 3. Return $(C, \mathsf{L}_{\mathsf{enc}}(k, N, A, M))$ *Leaking decryption:* $\mathsf{LDec}_k(N, A, C)$ 1. Return $(\mathsf{Dec}_k(N, A, C), \mathsf{L}_{\mathsf{dec}}(k, N, A, C))$

Table 2. The CIML2 and PIML2 security games.

variant without $\mathsf{L}_{\mathsf{decch}}$, corresponding to situations in which an adversary cannot observe any decryption operation, which could happen in settings where keys are dependent of the communication direction, and the adversary only has physical access to one end of the communication.

Definition 8 (CPAmL2). *A nonce-based authenticated encryption with associated data* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *is* $(q_e, q_c, q_l, t, \varepsilon)$*-CPAmL2 secure for a security parameter* n *if, for every* (q_e, q_c, q_l, t)*-bounded adversary* $\mathcal{A}$*, we have:*

$$\left|\Pr\left[\mathsf{PrivK}^{\mathsf{CPAmL2},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{CPAmL2},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon,$$

where the adversary $\mathcal{A}^{\mathsf{L}}$ *makes at most* q_e *leaking encryption queries,* q_c *challenge decryption leakage queries and* q_l *leakage evaluation queries on chosen keys.*

Integrity We next adopt the natural and strong extensions of INT-CTXT and INT-PTXT to nonce-misuse resistance and (full) leakages in encryption and decryption. The INT-CTXT extension, called Ciphertext Integrity with Misuse and Leakages, noted CIML2, comes from [BPPS17] and is an earlier proposal CIML1 [BKP+18] extended with with decryption leakage. Based on this definition, we propose here the corresponding extension of INT-PTXT security, which we call PIML2.

Definition 9 (CIML2, PIML2). *An authenticated encryption* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *provides* $(q_e, q_d, q_l, t, \varepsilon)$*-ciphertext (resp. plaintext) integrity with nonce misuse and leakages for security parameter* n *if, for all* (q_e, q_d, q_l, t)*-bounded adversaries* $\mathcal{A}^{\mathsf{L}}$*, we have:*

$$\Pr\left[\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] \leq \varepsilon,$$
$$\textit{(resp.}\ \Pr\left[\mathsf{PrivK}^{\mathsf{PIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] \leq \varepsilon\textit{)},$$

where the security game $\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *(resp.* $\mathsf{PrivK}^{\mathsf{PIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$*) is defined in the left part of Table 2 (resp. right part) when* $\mathcal{A}^{\mathsf{L}}$ *makes at most* q_e *leaking encryption queries,* q_d *leaking decryption queries and* q_l *leakage evaluation queries.*

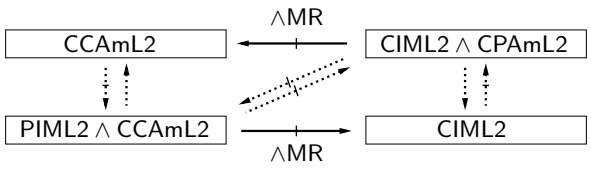

Fig. 2: Relations among notions. Arrows (resp., barred arrows) denote implications (resp., separations). Dotted arrows are trivially implied by other relations.

3.2 Overall requirement on AE

We eventually require that a secure AE intended to support nonce misuse and full leakages satisfies the strongest achievable guarantee presented in the paper: an AEML scheme is expected to offer CCAmL2 and CIML2 security, together with being a MR AEAD scheme without leakages. This definition departs from the traditional ones in the black-box setting, which is based on the combination of CPA and INT-CTXT security [BN08] or CCA and INT-PTXT security [KY00]. We will actually show that there are important separations between these notions: CCAmL2 + PIML2 + MR $\not\Rightarrow$ CIML2, and CPAmL2 + CIML2 + MR $\not\Rightarrow$ CCAmL2. Furthermore, even if CCAmL2 (resp., CIML2) implies both IND-CCA and CPAmL2 (resp., INT-CTXT and PIML2), the combination of CCAmL2 and CIML2 security does not imply MR, which is therefore a separate requirement.

Definition 10 (AEML-VF). *An AE scheme with security against nonce misuse and full vectorial leakages (denoted as* AEML-VF*) is an AE scheme* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with a leakage function pair* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *satisfying the following assertions: (i)* AEAD *is misuse resistant; (ii)* AEAD *is* CIML2 *secure with leakage function* L*; (iii)* AEAD *is* CCAmL2 *secure with leakage function* L*.*

As indicated above, AEML-VF security will typically be abbreviated as AEML in our paper, since the VF attack setting is understood.

3.3 Separation results

We now explain why the strong security notions of MR, CCAmL2 and CIML2 are needed to define AEML, while one could be tempted to make a definition as a combination of weaker notions, which may be easier to prove. Unfortunately, there is no such equivalence and we show that AEML is strictly stronger than any other combinations, assuming that AEML-secure AEAD exists.

We summarize even more relations in Figure 2. In contrast to the black-box setting, these relations show that one cannot choose between different ways to achieve AEML since, for instance, CCAmL2 $\wedge$ PIML2 $\not\Leftrightarrow$ CPAmL2 $\wedge$ CIML2.

MR $\wedge$ CPAmL2 $\wedge$ CIML2 $\not\Rightarrow$ CCAmL2*. It is not surprising that MR does not imply CCAmL2 since the leakage function L is absent from the black-box notion. Contrarily, L appears both in CPAmL2 and CIML2. This claim thus says that leakages in decryption may not alter integrity but may alter confidentiality. To reflect this intrinsic separation of the leakage setting, we show that the implication does not even hold for CCAmL2* where the challenge decryption leakage oracle $\mathsf{L}_{\mathsf{decch}}$ is unavailable. While the latter leakages are motivated in the context of side-channel attacks [BPPS17], is is also quite specific to such attacks. So ignoring it in the separation makes our result stronger and more general.

Theorem 1. *Assuming that there exists an AE scheme which satisfies* MR, CPAmL2, *and* CIML2 *in the unbounded leakage setting, then there exists an AE with the same security properties but which fails to achieve* CCAmL2, *even without challenge decryption leakages (i.e.,* CCAmL2**).*

The proof utilizes information leaked by invalid decryption queries, which was also exploited in the "protocol leakage" setting [BPS15]. The implication does not hold either when starting from the multiple challenge variant of CPAmL2 (discussed in Section A).

Proof. Let $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ be MR, CPAmL2 with respect to L and CIML2 with respect to L^* as such authenticated encryption exists by assumption. Then we build $\mathsf{AEAD}' = (\mathsf{Gen}', \mathsf{Enc}, \mathsf{Dec})$ with leakage $\mathsf{L}' = (\mathsf{L}'_{\mathsf{enc}}, \mathsf{L}'_{\mathsf{dec}})$ such that, for a fixed message $M^\dagger \in \mathcal{M}$:

$\mathsf{Gen}'(1^n)$**:** returns $k \leftarrow \mathsf{Gen}(1^n)$ and $k' \leftarrow \mathsf{Gen}(1^n)$;
$\mathsf{L}'_{\mathsf{enc}}((k,k'),N,A,M)$**:** outputs $(\mathsf{leak}_{\mathsf{e}}, C', \mathsf{leak}_{\mathsf{e}'})$ where $\mathsf{leak}_{\mathsf{e}} = \mathsf{L}_{\mathsf{enc}}(k,N,A,M)$ (comes from the computation of $C \leftarrow \mathsf{Enc}_k(N,A,M)$), the ciphertext $C' = \mathsf{Enc}_{k'}(N,A,M)$ and consequently $\mathsf{leak}_{\mathsf{e}'} = \mathsf{L}_{\mathsf{enc}}(k',N,A,M)$;
$\mathsf{L}'_{\mathsf{dec}}((k,k'),N,A,C)$**:** outputs $\mathsf{leak}_{\mathsf{d}} = \mathsf{L}_{\mathsf{dec}}(k,N,A,C)$ if $M \neq \perp$ (which comes from the computation of $M \leftarrow \mathsf{Enc}_k(N,A,C)$) and outputs $(\mathsf{leak}_{\mathsf{d}}, C^\dagger, \mathsf{leak}^\dagger_{\mathsf{e}'})$ otherwise, where $\mathsf{leak}_{\mathsf{d}} = \mathsf{L}_{\mathsf{dec}}(k,N,A,C)$, $C^\dagger \leftarrow \mathsf{Enc}_{k'}(N,A,M^\dagger)$ and consequently also $\mathsf{leak}^\dagger_{\mathsf{e}'} = \mathsf{L}_{\mathsf{enc}}(k',N,A,M^\dagger)$.

From a black-box standpoint, k' does not even exist so AEAD' is still MR. Therefore, let us focus on the security notions involving leakages.

CPAmL2. In the $\mathsf{PrivK}^{\mathsf{CPAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}(1^n)$ game, the adversary $\mathcal{A}'$ does not have access to $\mathsf{L}'_{\mathsf{dec}}$ except from the challenge decryption leakage through $\mathsf{L}'_{\mathsf{decch}}$. But since the challenge ciphertext is valid, $\mathsf{L}'_{\mathsf{decch}} = \mathsf{L}_{\mathsf{decch}}$ which returns $\mathsf{L}_{\mathsf{dec}}(k, N_{\mathsf{ch}}, A_{\mathsf{ch}}, C^b)$. Consequently, an adversary $\mathcal{A}$ in $\mathsf{PrivK}^{\mathsf{CPAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ can easily simulate the view of $\mathcal{A}'$ simply by picking $k' \leftarrow \mathsf{Gen}(1^n)$ transmitting all the queries to its own oracles and just add the encryption leakage $(C', \mathsf{leak}_{\mathsf{e}'})$ if necessary.

CIML2. In the $\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A}',\mathsf{AEAD}',(\mathsf{L}')^*}(1^n)$ game, the adversary $\mathcal{A}'$ still needs to forge a fresh ciphertext of AEAD with key k while the additional unbounded leakage given by $(\mathsf{L}')^*$ only depends k'. Then, building a reduction to $\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}^*}(1^n)$ is straightforward.

$\neg$CCAmL2*. We build a distinguisher $\mathcal{A}'$ against AEAD'. In the security game $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}(1^n)$, the adversary queries leaking decryption of $(N_{\mathsf{ch}}, A_{\mathsf{ch}}, C)$ for any chosen $N_{\mathsf{ch}}, A_{\mathsf{ch}}$ and C. If the ciphertext is valid, it receives some $M \neq \perp$ and it sets $(M^0, C^0) = (M, C)$. If not, it receives $(\perp, (\mathsf{leak}_{\mathsf{e}}, C^\dagger, \mathsf{leak}^\dagger_{\mathsf{e}'}))$ from $\mathsf{LDec}_{k,k'}(N_{\mathsf{ch}}, A_{\mathsf{ch}}, C)$ and sets $(M^0, C^0) = (M^\dagger, C^\dagger)$. In the challenge phase, $\mathcal{A}'$ sends $(N_{\mathsf{ch}}, A_{\mathsf{ch}}, M^0, M^1)$ for any distinct M^1 than M^0. Since the pair $(N_{\mathsf{ch}}, A_{\mathsf{ch}})$ has never been queried for (leaking) encryption, $\mathcal{A}'$ does not receive $\perp$. In the answer $\mathsf{LEnc}_k(N_{\mathsf{ch}}, A_{\mathsf{ch}}, M^b)$, $\mathcal{A}'$ gets C^b. If C^b equals the known C^0, $\mathcal{A}'$ outputs 0, otherwise it outputs 1. Obviously the distinction holds with probability 1.

Now, it is easy to see that AEAD' with leakage L' fulfills all the desired requirements of the theorem based on the existence of AEAD. □

MR$\wedge$CCAmL2$\wedge$PIML2 $\not\Rightarrow$ CIML2. As for the previous assertion, being MR does not say anything about leakages, so not being CIML2 is obviously compatible. The most interesting part comes from CCAmL2 and PIML2 which include leakages. This claim exploits the fact that leakages on repeated queries may degrade ciphertext integrity but neither confidentiality nor plaintext integrity.

Theorem 2. *Assuming that there exists an AE scheme which satisfies* MR, CCAmL2, *and* PIML2 *in the unbounded leakage model, then there exists an AE which satisfies the same security properties but which fails to achieve* CIML2 *(even if not in the unbounded leakage model).*

The proof proceeds by building a ¬CIML2 scheme AEAD' from an $\mathsf{MR} \wedge \mathsf{CCAmL2} \wedge \mathsf{PIML2}$ scheme AEAD. An interesting feature is that this counterexample AEAD' preserves the tidyness of AEAD. This deviates from Bellare and Namprempre's well-known approach for establishing $\mathsf{INT\text{-}PTXT} \not\Rightarrow \mathsf{INT\text{-}CTXT}$, which did utilize non-tidy counterexamples [BN08]. It is possible in our case due to the presence of leakages.

Proof. Let $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ be MR, CCAmL2 with respect to L and PIML2 with respect to L^* as such authenticated encryption exists by assumption. Then we build $\mathsf{AEAD}' = (\mathsf{Gen}', \mathsf{Enc}, \mathsf{Dec})$ with leakage $\mathsf{L}' = (\mathsf{L}'_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ as follows, where $N^\dagger, A^\dagger, M^\dagger, N^\circ, A^\circ, M^\circ$ below are the outputs of a publicly samplable distribution parametrized by n:

$\mathsf{Gen}'(1^n)$**:** generates $k \leftarrow \mathsf{Gen}(1^n)$. Then, it selects distinct nonces $N^\dagger, N^\circ \in \mathcal{N}$, distinct $A^\dagger, A^\circ \in \mathcal{AD}$ and distinct messages $M^\dagger, M^\circ \in \mathcal{M}$. It computes the ciphertext $C^\dagger \leftarrow \mathsf{Enc}_k(N^\dagger, A^\dagger, M^\dagger)$ and splits it into four random shares: it samples three $|C^\dagger|$-bit strings R_0, R_1, S_0 and sets $S_1 = C^\dagger \oplus R_0 \oplus R_1 \oplus S_0$. It outputs (k, sh) where $sh = (R_0, R_1, S_0, S_1)$.

$\mathsf{L}'_{\mathsf{enc}}((k, sh), N, A, M)$**:** outputs $\mathsf{leak}_{\mathsf{e}} = \mathsf{L}_{\mathsf{enc}}(k, N, A, M)$ (which comes from the computation of the ciphertext $C \leftarrow \mathsf{Enc}_k(N, A, C)$) as well as the additional value B but only in four cases:

- Case 1.1: $(N, A) = (N^\dagger, A^\circ), M \neq M^\circ$: $B = R_0$;
- Case 1.2: $(N, A) = (N^\dagger, A^\circ), M = M^\circ$: $B = R_1$;
- Case 2.1: $(N, A) = (N^\circ, A^\dagger), M \neq M^\dagger$: $B = S_0$;
- Case 2.2: $(N, A) = (N^\circ, A^\dagger), M = M^\dagger$: $B = S_1$.

If we drop the leakage functions AEAD' shows no deviation from AEAD and thus is still MR. It remains to establish the desired leakage-related claims:

CCAmL2. We prove that if there is a CCAmL2 adversary $\mathcal{A}'$ against AEAD', then there is an adversary $\mathcal{A}$ which uses $\mathcal{A}'$ to break the CCAmL2 security of AEAD. In detail, once $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ is setup, $\mathcal{A}(1^n)$ publicly sample $N^\dagger, A^\dagger, M^\dagger, N^\circ, A^\circ, M^\circ$ and sends to $\mathcal{A}'$ whatever specified by AEAD'. Even if the ciphertext $C^\dagger = \mathsf{Enc}_k(N^\dagger, A^\dagger, M^\dagger)$ has never been computed its length is fully determined by n and $(N^\dagger, A^\dagger, M^\dagger)$. Therefore, $\mathcal{A}$ picks random $|C^\dagger|$-bit strings R_1, S_0, S_1. Then it runs $\mathcal{A}'$ and simulates the game $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}$ using its own interaction $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$. For each query from $\mathcal{A}'$, the actions of $\mathcal{A}$ are as follows:

Leaking (non-challenge) encryption queries: On input (N, A, M),

(i) If $N = N^\dagger$ appears for the first time in a leaking encryption query (and so not in the challenge query) $\mathcal{A}$ queries a leaking encryption on $(N^\dagger, A^\dagger, M^\dagger)$ to its own oracle and gets back $C^\dagger$ and then computes $R_0 = C^\dagger \oplus R_1 \oplus S_0 \oplus S_1$;

(ii) $\mathcal{A}$ queries its own leaking encryption oracle on (N, A, M) and gets back some $(C, \mathsf{leak}_{\mathsf{e}})$ or possibly $\perp$ (for forbidden queries);

(iii) $\mathcal{A}$ checks whether it should append an additional leakage B to $(C, \mathsf{leak}_{\mathsf{e}})$ depending on (N, A, M) falls into one of the four cases described above.

$\mathcal{A}$ respectively returns $(C, \mathsf{leak}_{\mathsf{e}})$ or $\perp$ or even $(C, \mathsf{leak}_{\mathsf{e}}, B)$ to $\mathcal{A}'$ according to the above situations.

Leaking decryption queries: on input (N, A, C), $\mathcal{A}$ simply calls its own oracle and reply to $\mathcal{A}'$ with the answer it received.

Leaking challenge query: on input $(N_{\mathsf{ch}}, A_{\mathsf{ch}}, M^0, M^1)$, $\mathcal{A}$ sends it to its leaking challenge oracle and gets the tuple $(C^b, \mathsf{leak}_{\mathsf{e}}^b)$ or possibly $\perp$. If not $\perp$,

(i) if $(N_{\mathsf{ch}}, A_{\mathsf{ch}}) = (N^\dagger, A^\circ)$, $\mathcal{A}$ returns $(C^b, \mathsf{leak}_{\mathsf{e}}^b, R_1)$. In other words, the additional leakage is always R_1 regardless of the messages M^0, M^1;

(ii) if $(N_{\mathsf{ch}}, A_{\mathsf{ch}}) = (N^\circ, A^\dagger)$, $\mathcal{A}$ returns $(C^b, \mathsf{leak}_{\mathsf{e}}^b, S_0)$ regardless of M^0, M^1;

(iii) else, $\mathcal{A}$ just returns $(C^b, \mathsf{leak}_{\mathsf{e}}^b)$.

Challenge decryption leakage: returns the corresponding decryption leakage of the challenge ciphertext.

Eventually, $\mathcal{A}$ outputs the bit returned by $\mathcal{A}'$.

Now we explain why $\mathcal{A}$ properly emulates the $\mathsf{CCAmL2}$ game in front of $\mathcal{A}'$. First, as long as $C^\dagger$ is returned to $\mathcal{A}$ in a leaking encryption queries $\mathcal{A}'$ can not make a leaking challenge query with $N_{ch} = N^\dagger$ and then case (i) in the challenge phase will not occur since $\mathcal{A}$ will receive $\perp$ and will send it to $\mathcal{A}'$ as expected by the game. Therefore, $B = R_0$ or $B = R_1$ cannot be among the encryption leakage of the challenge ciphertext. Second, if N° appears at first in a leaking encryption query, case (ii) of the challenge phase will not occur and $B = S_0$ or $B = S_1$ will never be among the encryption leakage of the challenge ciphertext as in AEAD'. We stress that R_0, R_1, S_0, S_1 might appear several times in next responses to leaking encryption queries but exactly as in the honest run of $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}$. Third, if $N_{ch} = N^\dagger$ appears for the first time in the challenge phase, case (i) of the leaking encryption query phase will not occur and $\mathcal{A}$ will receive $\perp$ and send it to $\mathcal{A}'$ if $\mathcal{A}'$ queries a leaking encryption involving $N^\dagger$ afterwards, which corresponds to the right view of the game. Therefore, R_0 will never be defined and if we also have $A_{ch} = A^\circ$ the additional leakage $B^b = R_1$ will be given in the encryption leakage of the challenge ciphertext. Here it is easy to see, since $\mathcal{A}'$ will receive a single share among those specified in Case 1.1 and Case 1.2 of $\mathsf{L}'_{\mathsf{enc}}$, that this distribution is exactly as the one expected from AEAD'. Four, if $N_{ch} = N^\circ$ appears for the first time in the challenge phase, a similar argument shows that $\mathcal{A}'$ will only get a single share among those specified in Case 1.1 and Case 1.2 of $\mathsf{L}'_{\mathsf{enc}}$ and only if $A_{ch} = A^\dagger$ independently of the choice M^0, M^1. Once again, this is exactly the right distribution where $B^b = S_0$ is random and remains independent of the rest of the game. Five, all the previous arguments show that in any of these situations the potential additional leakage B (B^b included) can be perfectly emulated by $\mathcal{A}$ so that we have

$$\Pr\left[\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] = \Pr\left[\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}(1^n) \Rightarrow 1\right].$$

Finally, in any other cases the above conclusion obviously holds as well.

$\mathsf{PIML2}$. We prove that if there is a $\mathsf{PIML2}$ adversary $\mathcal{A}'$ against AEAD', then there is an adversary $\mathcal{A}$ which uses $\mathcal{A}'$ to break the $\mathsf{PIML2}$ security of AEAD. In detail, once $\mathsf{PrivK}^{\mathsf{PIML2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ is setup, $\mathcal{A}(1^n)$ publicly sample $N^\dagger, A^\dagger, M^\dagger, N^\circ, A^\circ, M^\circ$ and sends to $\mathcal{A}'$ whatever is specified by AEAD'. Since it is assumed that the size of $C^\dagger = \mathsf{Enc}_k(N^\dagger, A^\dagger, M^\dagger)$ is known, $\mathcal{A}$ can pick random $|C^\dagger|$-bit strings R_0, R_1, S_0. Then it runs $\mathcal{A}'$ and simulates $\mathsf{PrivK}^{\mathsf{PIML2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}$ using its own interaction $\mathsf{PrivK}^{\mathsf{PIML2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ such that any query made by $\mathcal{A}'$ is simply relayed by $\mathcal{A}$ to its oracles and answered back with the answer possibly augmented with an additional encryption leakage according to the different four cases defined in $\mathsf{L}'_{\mathsf{enc}}$ except in Case 2.2, namely if $\mathcal{A}'$ requests an encryption of $(N^\circ, A^\dagger, M^\dagger)$. In

the latter case, $\mathcal{A}$ first queries an encryption of $(N^\dagger, A^\dagger, M^\dagger)$ and gets back $C^\dagger$. Then, $\mathcal{A}$ proceeds as in the other cases but adds the leakage $S_1 = C^\dagger \oplus R_0 \oplus R_1 \oplus S_0$.

It should be straightforwards that the simulation is perfect. Furthermore, as long as $\mathcal{A}'$ does not get the share S_1 all the information in its view is exactly the same as the information $\mathcal{A}$ gets in its $\mathsf{PIML2}$ view and therefore any associated-data/plaintext forgery computed by $\mathcal{A}'$ is a valid forgery against AEAD. In order to get some information about $C^\dagger$, $\mathcal{A}'$ must receive the four shares R_0, R_1, S_0, S_1 but it means that it queried $(N^\circ, A^\dagger, M^\dagger)$ but then $C^\dagger = R_0 \oplus R_1 \oplus S_0 \oplus S_1$ is no more considered as a valid forgery since the pair $(A^\dagger, M^\dagger)$ is already involved in a leaking encryption query. As a conclusion $\mathcal{A}$ never makes a leaking encryption query involving (A, M) which has never been among a leaking encryption query made by $\mathcal{A}'$ at the first place.

Note that this reduction works in the unbounded leakage setting as well.

$\neg\mathsf{CIML2}$. We mount a ciphertext forgery attack against AEAD'. Let $\mathcal{A}'$ be an adversary which given $N^\dagger, A^\dagger, M^\dagger, N^\circ, A^\circ, M^\circ$ sequentially queries a leaking encryption on $(N^\dagger, A^\circ, M^\dagger)$, $(N^\dagger, A^\circ, M^\circ)$, $(N^\circ, A^\dagger, M^\circ)$, and $(N^\circ, A^\dagger, M^\dagger)$ which fall into the four cases where each one of the shares R_0, R_1, S_0, S_1 are additionally given in the encryption leakage. Then, $\mathcal{A}'$ output $C^\dagger = R_0 \oplus R_1 \oplus S_0 \oplus S_1$ which is a valid encryption of $(N^\dagger, A^\dagger, M^\dagger)$ which has never been received as an answer to some leaking encryption query. □

As a side note, (i) it can be seen the leakage functions $\mathsf{L}' = (\mathsf{L}'_{\mathsf{enc}}, \mathsf{L}'_{\mathsf{dec}})$ of the counterexample AEAD' preserves the deterministicness of the original one $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$; (ii) we never require that $|\mathcal{M}| > 2$.

4 Completing the definitions' zoo

To give a complete picture of the different security flavors of AE with misuse-resistance or resilience and full vectorial leakages, we list all the security definitions that can be derived from our confidentiality and integrity notions. We then study their relations which may be useful in order to guide future designs with relaxed requirements (e.g., in order to reach better performances). It shows that, apart from the obvious implications between the different flavors of confidentiality (resp., integrity), all the notions are separated from each other. In this section we concentrate on the single challenge notions. The extension to the multi-challenge setting is discussed in Appendix A.

4.1 Security definition list (single challenge setting)

The $\mathsf{CCAmL2}$ security game $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ is defined in Section 3.1, Figure 1. By dropping some accesses to the distinct oracles of this game, we naturally derive other confidentiality notions. For instance $\mathsf{CPAmL2}$ is defined by removing items (2) and (4) from the security game. By doing similar modifications, we can define different integrity notions from the $\mathsf{CIML2}$ security game $\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ defined in Section 2.3, Table 2. We next formalize these variants.

Prefix-suffix definitions In all the notions derived from $\mathsf{CCAmL2}$ and $\mathsf{CIML2}$ we only focus on those capturing full leakages (partial leakage is covered by BMOS). Therefore all the definitions below keep the large L in their notation. This leads us to consider 16 different notions denoted as "pre-suf" with prefix $\mathsf{pre} \in \{\mathsf{CCA}, \mathsf{CPA}, \mathsf{CI}, \mathsf{PI}\}$ and suffix $\mathsf{suf} \in \{\mathsf{ML2}, \mathsf{ML1}, \mathsf{mL2}, \mathsf{mL1}, \mathsf{L2}, \mathsf{L1}\}$: a large "M" corresponds to misuse resistance, a small "m" corresponds to misuse-resilience and no "M/m" means that the security game is nonce-respecting (which only restricts leaking encryption queries).

Zoo of confidentiality notions. For $\mathsf{pre} \in \{\mathsf{CCA}, \mathsf{CPA}\}$ we obtain the following 8 notions, by starting from CCAmL2 and by removing one security layer at a time:

$$\mathsf{CCAmL2} \rightarrow \mathsf{CCAmL1}, \mathsf{CCAL2}, \mathsf{CPAmL2} \rightarrow \mathsf{CPAmL1}, \mathsf{CPAL2}, \mathsf{CCAL1} \rightarrow \mathsf{CPAL1}.$$

Definition 11. *A nonce-based authenticated encryption with associated data* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakages* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *is* $(q_{\mathsf{pre\text{-}suf}}, q_l, t, \varepsilon)$*-pre-suf secure for a security parameter* n *if, for every* $(q_{\mathsf{pre\text{-}suf}}, q_l, t)$*-bounded adversary* $\mathcal{A}^{\mathsf{L}}$*, we have* $\mathsf{pre} \in \{\mathsf{CCA}, \mathsf{CPA}\}$ *and:*

$$\left| \Pr\left[\mathsf{PrivK}^{\mathsf{pre\text{-}suf},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{pre\text{-}suf},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] \right| \leq \varepsilon,$$

where the adversary $\mathcal{A}^{\mathsf{L}}$ *makes at most* $q_{\mathsf{pre\text{-}suf}}$ *queries defined in* $\mathsf{PrivK}^{\mathsf{pre\text{-}suf},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *below, and* q_l *leakage evaluation queries on arbitrarily chosen keys.*

(i) $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CCAmL2}} = (q_e, q_d, q_c)$ with the CCAmL2 game in Figure 1.
(ii) $\mathsf{PrivK}^{\mathsf{CCAmL1},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CCAmL1}} = (q_e, q_d)$ and the CCAmL1 security game "removes 2" from the CCAmL2 game, meaning that $\mathsf{L}_{\mathsf{dec}}$ is removed from all the oracles. In other words items (2),(4) become black-box and (5) disappears.
(iii) $\mathsf{PrivK}^{\mathsf{CCAL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CCAL2}} = (q_e, q_d, q_c)$ and the CCAL2 security game "removes M" from the CCAmL2 game which becomes a nonce-respecting.
(iv) $\mathsf{PrivK}^{\mathsf{CPAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CPAmL2}} = (q_e, q_c)$ and no decryption oracle access is given in Figure 1: items (2) and (4) are removed but not item (5), hence the 2.
(v) $\mathsf{PrivK}^{\mathsf{CPAmL1},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CPAmL1}} = (q_e)$ and the CPAmL1 game only keeps items (1) and (3) from the CCAmL2 game, (like the CPAmL2 game without $\mathsf{L}_{\mathsf{decch}}$).
(vi) $\mathsf{PrivK}^{\mathsf{CPAL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CPAL2}} = (q_e, q_c)$, the CPAL2 game is a nonce-respecting version of the CPAmL2 and $\mathsf{L}_{\mathsf{decch}}$ is still available in item (5).
(vii) $\mathsf{PrivK}^{\mathsf{CCAL1},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CCAL1}} = (q_e, q_d)$ and the CCAL1 is a nonce-respecting version of CCAmL1 (with black-box dec. and nonce-respecting leaking enc.).
(viii) $\mathsf{PrivK}^{\mathsf{CPAL1},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: $q_{\mathsf{CPAL1}} = (q_e)$ with only nonce-respecting leaking encryption.

Zoo of integrity notions. For $\mathsf{pre} \in \{\mathsf{CI}, \mathsf{PI}\}$ we obtain the following 8 notions, by starting from CIML2 and by removing one security layer at a time:

$$\mathsf{CIML2} \rightarrow \mathsf{CIML1}, \mathsf{CIL2}, \mathsf{PIML2} \rightarrow \mathsf{PIML1}, \mathsf{PIL2}, \mathsf{CIL1} \rightarrow \mathsf{PIL1}.$$

Definition 12. *A nonce-based authenticated encryption with associated data* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakages* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *is* $(q_e, d_d, q_l, t, \varepsilon)$*-pre-suf secure for a security parameter* n *if, for every* (q_e, q_d, q_l, t)*-bounded adversary* $\mathcal{A}^{\mathsf{L}}$*, we have* $\mathsf{pre} \in \{\mathsf{CI}, \mathsf{PI}\}$ *and:*

$$\Pr\left[\mathsf{PrivK}^{\mathsf{pre\text{-}suf}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] \leq \varepsilon,$$

where the adversary $\mathcal{A}^{\mathsf{L}}$ *makes at most* q_e *encryption queries and* q_d *decryption queries defined in* $\mathsf{PrivK}^{\mathsf{pre\text{-}suf}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *below, and* q_l *leakage evaluation queries on arbitrarily chosen keys.*

(i) $\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the CIML2 game, see Table 2.
(ii) $\mathsf{PrivK}^{\mathsf{CIML1}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the CIML1 game removes $\mathsf{L}_{\mathsf{dec}}$ (i.e., decryption is black-box).

(iii) $\mathsf{PrivK}^{\mathsf{CIL2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the CIL2 game is a nonce-respecting version of CIML2.
(iv) $\mathsf{PrivK}^{\mathsf{PIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: in the PIML2 game the winning condition changed (Table 2).
(v) $\mathsf{PrivK}^{\mathsf{PIML1}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the PIML1 game removes $\mathsf{L}_{\mathsf{dec}}$ from PIML2.
(vi) $\mathsf{PrivK}^{\mathsf{PIL2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the PIL2 game is a nonce respecting version of PIML2.
(vii) $\mathsf{PrivK}^{\mathsf{CIL1}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the CIL1 game is a nonce-respecting version of CIML2 free of $\mathsf{L}_{\mathsf{dec}}$.
(viii) $\mathsf{PrivK}^{\mathsf{PIL1}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$: the PIL1 game is a nonce-respecting version of PIML2 free of $\mathsf{L}_{\mathsf{dec}}$.

Connection with previous works Among the above sixteen notions, three of them are equivalent to already defined ones: CPAL1 appeared in [PSV15] under the name of LMCPA, CIML1 was introduced in [BKP+18] (under the name CIML) and CIML2 was introduced in [BPPS17].

4.2 Relations within the zoo (single challenge setting)

We picture all the 16 notions with their natural implications in Figure 3.

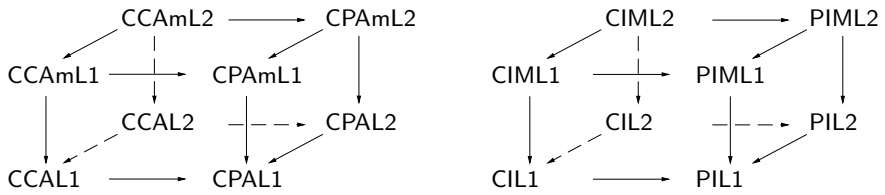

Fig. 3: Single-challenge security notions with various combinations of C/P (Ciphertext/Plaintext), m/M (misuse), 1/2 (# of leaking oracles). Left: confidentiality notions, right: integrity notions. Arrows indicate implications.

Theorem 3 (Long diagonals). *There exist authenticated encryptions schemes showing that:*

$$\mathsf{CCAmL1} \not\leftrightarrows \mathsf{CPAL2} \qquad \mathsf{CCAL2} \not\leftrightarrows \mathsf{CPAmL1} \qquad \mathsf{CPAmL2} \not\leftrightarrows \mathsf{CCAL1}$$
$$\mathsf{CIML1} \not\leftrightarrows \mathsf{PIL2} \qquad \mathsf{CIL2} \not\leftrightarrows \mathsf{PIML1} \qquad \mathsf{PIML2} \not\leftrightarrows \mathsf{CIL1}$$

As a corollary, all the arrows of Figure 3 are strict. The proof only requires to show 4 of the 6 assertions.

Proof. We are to prove 12 non-implications:

(i) $\mathsf{CCAmL1} \not\Rightarrow \mathsf{CPAL2}$,
(ii) $\mathsf{CPAL2} \not\Rightarrow \mathsf{CCAmL1}$,
(iii) $\mathsf{CCAL2} \not\Rightarrow \mathsf{CPAmL1}$,
(iv) $\mathsf{CPAmL1} \not\Rightarrow \mathsf{CCAL2}$,
(v) $\mathsf{CPAmL2} \not\Rightarrow \mathsf{CCAL1}$,
(vi) $\mathsf{CCAL1} \not\Rightarrow \mathsf{CPAmL2}$,
(vii) $\mathsf{CIML1} \not\Rightarrow \mathsf{PIL2}$,
(viii) $\mathsf{PIL2} \not\Rightarrow \mathsf{CIML1}$,
(ix) $\mathsf{CIL2} \not\Rightarrow \mathsf{PIML1}$,
(x) $\mathsf{PIML1} \not\Rightarrow \mathsf{CIL2}$,
(xi) $\mathsf{PIML2} \not\Rightarrow \mathsf{CIL1}$,
(xii) $\mathsf{CIL1} \not\Rightarrow \mathsf{PIML2}$.

We first show that a security notion $\mathsf{X1}$ *without* decryption leakages cannot imply the corresponding notion $\mathsf{X2}$ *with* decryption leakages. This would establish six separations (i), (iv), (vi), (vii), (x), and (xii). For this, assume that AEAD is a $\mathsf{X1}$ secure scheme with master-key K. We define a new scheme AEAD^*, which is the same as AEAD except that its leakages for decryption queries explicitly include the master-key K. In this way, AEAD^* is clearly not $\mathsf{X2}$ secure (as the key is leaked). But it remains $\mathsf{X1}$ secure, since this enhancement of decryption leakage *cannot* be observed in the $\mathsf{X1}$ security game.

We then show that a security notion X *without* supporting nonce-misuse resistance/resilience cannot imply the corresponding notion XM *with* misuse resilience. This would establish four separations (ii), (iii), (viii), and (ix). For this, assume that $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with leakage $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ is a X secure scheme. We define a new scheme $\mathsf{AEAD}^* = (\mathsf{Gen}', \mathsf{Enc}, \mathsf{Dec})$ with leakage $\mathsf{L} = (\mathsf{L}'_{\mathsf{enc}}, \mathsf{L}'_{\mathsf{dec}})$ as follows:

$\mathsf{Gen}'(1^n)$**:** generates two keys $k \leftarrow \mathsf{Gen}(1^n)$ and $k' \leftarrow \mathsf{Gen}(1^n)$, and selects a public pair $(N^\dagger, A^\dagger)$.

$\mathsf{L}'_{\mathsf{enc}}((k, k'), N, A, M)$**:** outputs $\mathsf{leak}_{\mathsf{e}} = \mathsf{L}_{\mathsf{enc}}(k, N, A, M)$ as well as the additional value B but in only two cases:
- if $N = N^\dagger$ and $A = A^\dagger$, $B = k \oplus k'$;
- if $N = N^\dagger$ and $A \neq A^\dagger$, $B = k'$;

Clearly, when multiple encryption queries with the same nonce $N^\dagger$ is made, then both $k \oplus k'$ and k' could be leaked, and the key of the underlying scheme AEAD could be recovered. Therefore, AEAD^* is not misuse-resistant in *any* security setting. This is not the case in the nonce-respecting setting, and it thus remains X secure.

It remains to prove $\mathsf{CPAmL2} \not\Rightarrow \mathsf{CCAL1}$ and $\mathsf{PIML2} \not\Rightarrow \mathsf{CIL1}$. For this we follow the standard idea of showing CPA$\not\Rightarrow$CCA and $\mathsf{INT\text{-}PTXT} \not\Rightarrow \mathsf{INT\text{-}CTXT}$. In detail, consider $\mathsf{CPAmL2} \not\Rightarrow \mathsf{CCAL1}$ first, and assume that $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is $\mathsf{CPAmL2}$ secure. We define a new scheme $\mathsf{AEAD}^* = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}')$ as follows:

$\mathsf{Dec}'_k(N, A, C)$**:** outputs $\mathsf{Dec}_k(N, A, C) \| k$, i.e. the main key k is appended to the decrypted plaintext.

This very artificial scheme "gives up" by appending its key to the decrypted message upon any decryption query. Therefore, it cannot be CCA secure under any reasonable definition. Thus $\mathsf{CPAmL2} \not\Rightarrow \mathsf{CCAL1}$.

For $\mathsf{PIML2} \not\Rightarrow \mathsf{CIL1}$, assume that $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is $\mathsf{PIML2}$ secure. We define a new scheme $\mathsf{AEAD}^* = (\mathsf{Gen}, \mathsf{Enc}', \mathsf{Dec}')$ as follows:

$\mathsf{Enc}'_k(N, A, M)$**:** outputs $\mathsf{Enc}_k(N, A, M) \| 0 \| 0$, i.e., two bits are appended to the ciphertext.
$\mathsf{Dec}'_k(N, A, C)$**:** parses $C = C' \| b \| b'$, and outputs $\mathsf{Dec}_k(N, A, C')$ if and only if $b = b'$.

Then it's clear that AEAD^* is not $\mathsf{CIL1}$ since from any valid ciphertext $(N, A, C \| 0 \| 0)$ obtained before the adversary could use $(N, A, C \| 1 \| 1)$ as a forgery. Yet, it remains $\mathsf{PIML2}$ secure.

Remark #1. By revisiting the proof for $\mathsf{MR} \wedge \mathsf{CCAmL2} \wedge \mathsf{PIML2} \not\Rightarrow \mathsf{CIML2}$ in subsection 3.3, it can be seen that the exhibited $\mathsf{CIML2}$ adversary *only* relies on the leaking encryption. This means that it also breaks the $\mathsf{CIML1}$ security. Therefore, we already know that $\mathsf{MR} \wedge \mathsf{CCAmL2} \wedge \mathsf{PIML2} \not\Rightarrow \mathsf{CIML1}$.

Remark #2. We discuss the relations between these notions and the Eavesdropper Security with Decryption Leakages (EavDL) introduced in [BPPS17] (which is implied by AEML) in Appendix B. We also recall that the treatment of the multi-challenge setting in given in Appendix A.

5 Model and assumptions

Before moving to the description and proof of first modes of operations satisfying the previous security definitions, we now discuss the physical assumptions that will be required for this purpose in more detail.

5.1 Assumptions for CIML2

Starting with the authentication and integrity guarantees, the only assumption required for proving the CIML2 of the following designs is a strongly protected block cipher, that we will formalize as a leak-free SPRP hiding the long-term key, and that will be used a small constant number of times (i.e., 2) per message to encrypt, independently of the length of the message. In our current model, we assume this is perfectly achieved but it is an interesting open problem to investigate how gracefully the security degrades in case of moderate key leakages. For the rest, CIML2 is proven in the very liberal unbounded leakage model introduced in [BKP+18]: all the intermediate values computed by our AE designs (excepted the long term-key) can be leaked in full. Note that the need of a PRP (and not a PRF) is a strict one. As discussed in [BPPS17], this allows avoiding attacks where valid tags are produced thanks to the decryption leakages of invalid ciphertexts.[2]

Concretely, it is expected that the leak-free block is protected with state-of-the-art countermeasures such as masking, which reduces the amount of leakages at a level depending on the number of shares selected by the implementer. For example, [JS17, GPSS18] described how security levels of up to 128 bits can be achieved on ARM Cortex M4/M8 devices, at the cost of significant performance overheads that our limited use of this strongly protected component aim to limit.

5.2 Assumptions for CCAmL2

Confidentiality guarantees are much harder to obtain with leakages. This was already put forward by Micali and Reyzin in their seminal paper on physically observable cryptography [MR04], and is also reflected by the combination of stronger assumptions that we will use to prove the CCAmL2 of our AE designs.

Precisely, we will need a (slightly) stronger version of the leak-free block cipher required for CIML2, together with simulatable leakages for the other block ciphers, and will reduce the security of a full AE implementation to the eavesdropper security of a leaking single-block encryption scheme (which is therefore part of our assumptions).

Strongly protected block cipher Our proofs of CCAmL2 require a strongly protected SPRP that hides its master key, as in our proofs of CIML2. Yet, in this case, we will additionally need that this SPRP hides its output. More precisely, one out of the two strongly protected block ciphers in our following modes of operation will need to enforce this additional feature, which we can explain

[2] We note that such attacks may even prevent to achieve PIL2.

as follows.[3] Given a nonce N, associated data A and message M, the proposals in the next section start by processing a hash of (N, A) which is then sent to the strongly protected SPRP in order to generate a pseudorandom ephemeral key. This ephemeral key must be strongly protected as well, since it will be used to derive all the other keys of the scheme (by contrast, the leakage-restance of these other keys used in the "re-keying" parts of the modes will only rely on the simulatability assumption).

Note that, if the strongly protected block cipher is implemented as a masked design, this stronger requirement is not expected to lead to improved concrete attacks. For example, one could consider a setup where this cipher outputs several key shares, that are then re-combined as part of the key scheduling of the following block cipher in our encryption schemes. From the DPA viewpoint, it is unlikely that XORing these shares provides significantly more information than loading an unprotected key.

Simulatable leakages Besides the previous idealized components that are, by far, the most expensive in terms of side-channel countermeasures, our leakage-resilient modes of operation also require that the bulk of the computation does not leak too much (e.g., does not leak the ephemeral keys in full in just two executions). For this part of the computation, weaker and cheaper countermeasures are expected to be sufficient. Various types of limitations on leakages have been proposed in the literature – see Fuller and Hamlin [FH15] for a review and a comparison (excluding idealized assumptions such as used in [YSPY10]). For instance, leakages may be required to be limited in size at each round of computation, as used by Dziembowski and Pietrzak [DP08] and follow up works [Pie09, DP10, FPS12], or required to be simulatable [SPY13] as we exploit in the next section.

Admittedly, none of these assumptions is fully convincing. Assuming the leakages to be bounded in size is known to be unrealistic (i.e., it is hopeless to bound the output of an oscilloscope), extensions based on preserving the HILL pseudoentropy of the leaking secrets are hard to quantify [SPY$^+$10], while the first instance of simulator proposed in [SPY13] has been falsified in [LMO$^+$14] and it remains an open problem to propose new (implementations of) simulators that withstand the correlation distinguisher put forward by Galea et al.

We selected the simulatability assumption since it provides an elegant way to deal with unrealistic precomputation / future computation attacks (where the leakages in one round provide exploitable information about a later round) [DP08, SPY$^+$10], and it seems the only standard model assumption for arguing security of the single-pass fresh rekeying schemes without public randomness. In a departure, as far as our main goal of minimizing the adversary's surface is concerned, proofs in the bounded leakage model (or even proofs using idealized assumptions such as in [YSPY10]) would provide good hints that a mode of operation is well suited to resist side-channel attacks as well. In general, we leave the security analysis of our constructions under these complementary leakage assumptions as an important scope for further research.

Formally, the leakage-resistance of FEMALE and AEDT relies on the assumption that leakages satisfy (p, q)-recyclable-simulatability defined below, and based on the (p, q)-rsim-game in Table 3. This assumption is an extension of the q-simulatability [SPY13]: (p, q)-recyclable-simulatability is defined as q-simulatability where each of the q leakages can be obtained p times.

[3] For this "output-hidden" ideal component, a PRF would be sufficient.

Game (p,q)-$\mathsf{rsim}(\mathcal{A},\mathsf{E},\mathsf{L},\mathcal{S},b)$.		
The challenger selects two random keys $k, k^* \xleftarrow{\$} \mathcal{K}$. The output of the game is a bit b' computed by $\mathcal{A}^{\mathsf{L}}$ based on th challenger responses to a total of at most qe adversarial queries of the following type, each repeated at most p times:		
Query	Response if $b=0$	Response if $b=1$
$\mathsf{Enc}(x)$	$\mathsf{E}_k(x), \mathsf{L}(k,x)$	$\mathsf{E}_k(x), \mathcal{S}^{\mathsf{L}}(k^*, x, \mathsf{E}_k(x))$
and one query of the following type, repeated at most p times:		
Query	Response if $b=0$	Response if $b=1$
$\mathsf{Gen}(k_{pre}, x)$	$\mathcal{S}^{\mathsf{L}}(k_{pre}, x, k)$	$\mathcal{S}^{\mathsf{L}}(k_{pre}, x, k^*)$

Table 3. The (p,q)-rsim-game.

Definition 13 ((p,q)-recyclable-simulatability of leakages). *Let E be a PRP with L as its leakage function. Then the leakages of E are said to have $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,q)\text{-}rsim})$ (p,q)-recyclable-simulatability, if there exists a $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded simulator $\mathcal{S}^{\mathsf{L}}$ such that, for every (q_l, t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$ (making at most q_l queries to L and running in time t), we have:*

$$\left| \Pr[(p,q)\text{-}\mathsf{sim}(\mathcal{A},\mathsf{E},\mathsf{L},\mathcal{S},1) \Rightarrow 1] - \Pr[(p,q)\text{-}\mathsf{sim}(\mathcal{A},\mathsf{E},\mathsf{L},\mathcal{S},0) \Rightarrow 1] \right| \leq \varepsilon_{(p,q)\text{-}rsim}.$$

The necessity of the recyclable assumption stems from the CCAmL2 game which offers the challenge decryption leakage oracle $\mathsf{L}_{\mathsf{decch}}$ that was not considered in previous works [BKP+18, BPPS17] and (in cooperation with the challenge encryption leakages) may result in the same set of leakages traces being generated more than once. Recyclable simulatability is a stronger assumption than simulatability, since the typical q one considers in leakage-resilient constructions is 2 and the p values allowed by decryption leakages are polynomial.

We note that if only CCAmL1 or CCAmL2* (i.e., the weakened version of CCAmL2 from Section 3.1) is required, in which only challenge encryption leakages are generated (and $\mathsf{L}_{\mathsf{decch}}$ oracle is not available), the original q-simulatability is sufficient (with $q=2$). So overall, what our proofs demonstrate is that security with only challenge encryption leakages for multiple messages and blocks can be reduced to the security of a single block with (noisy, unrepeated) leakages under the assumption that 2 (noisy) leakages can be simulated. By contrast, security with challenge decryption leakages for multiple messages and blocks can only be reduced to the security of a single block with (noise-free, repeated) leakages under the assumption that 2 (noise-free) leakages can be simulated.

Concretely, the recommended designs for (recyclable) simulatable block cipher implementations can be quite different from the strongly protected ones. As already mentioned, leak-free PRPs will typically be implemented with very high-order masking to resist DPA (and thus can be very expensive). By contrast, simulatability is expected to be achievable with weakly protected implementations and essentially requires resistance against SPA with noisy measurements. Similarly, recyclable simulatability requires resistance against SPA with noise-free measurements and could for example be obtained with very low latency hardware (e.g., an unrolled AES implementation in one or two clock cycles [KDH+12], limiting the number of noise-free samples that the adversary can collect).

Reduction to a single message block & gradual security degradations Eventually, our proofs reduce the security of multiple (long) messages to the eavesdropper security of the single-block encryption scheme against CPA, which is not expected to hold with negligible adversarial

advantage. This is because the current state-of-the-art does not provide tools to prevent some minimum leakages on plaintext/ciphertext manipulations that inevitably occur when fully leakage-resilient encryption schemes are considered (i.e., when the leakages of the challenge query is given to the adversary).

Yet, as mentioned in the Introduction, we achieve *security-preserving domain extension* of the single-block encryption (an important direction in theory), and we add mechanisms to render decryption leakage harmless (to resist CCA rather than mere CPA). There is (clearly) a significant theoretical difference between such *security-preserving* modes of operation (as considered next) and the *non-preserving* ones. In practice, the former enforce that DPA attacks are hard(er) to mount, for example thanks to re-keying which is usually instrumental to enable the reductions. This is what we informally denote as minimizing the adversary's surface.

In this respect, it is important to mention that the stronger nature of these confidentiality assumptions is actually the main motivation for the gradual security degradations that our aggregate definition of AEML allows. That is, even if the confidentiality of a message is reduced due to a powerful side-channel attack against the single-block encryption scheme or the (recyclable) simulatability assumption, it will have no impact on other messages.

5.3 Leveled leakage setting

The previous descriptions allow us to clarify the meaning of leveled implementations. In general and informally, leveled implementations mix different types of cipher designs, with different levels of protections against side-channel attacks. In our specific case, leveled implementations mix two types of block cipher designs: a minimally used and strongly protected one that has to resist side-channel adversaries able to collect the leakages corresponding to polynomial number of inputs (i.e., a DPA), and an intensely used and weakly protected one that only has to resist side-channel adversaries able to collect the leakages corresponding to a small constant number of inputs (equivalent to an SPA).

Based on the discussion in this section, a more technical comparison with the work of Barwell et al. [BMOS17] is available in Appendix C.

6 First instantiations: FEMALE and AEDT

We finally present FEMALE and AEDT, two AE schemes making only two calls to a strongly protected block cipher and enabling leveled implementations. FEMALE is AEML. AEDT only offers both CIML2 and CCAmL2 but is more efficient.

6.1 FEMALE design and analysis

FEMALE is named after *Feedback-based Encryption with Misuse, Authentication and LEakage* as it starts processing the message blocks using a (re-keying) ciphertext feedback mode (see the top of Figure 4). The encryption processes the key only twice and the message blocks only once.

Description Given a hash function $\mathsf{H} : \{0,1\}^* \mapsto \mathcal{B}$ and a block cipher E on $\mathcal{M} = \{0,1\} \times \mathcal{B}$ as well as two distinct public constants p_A and p_B of $\mathcal{M}$, the FEMALE encryption algorithm has 3 stages:

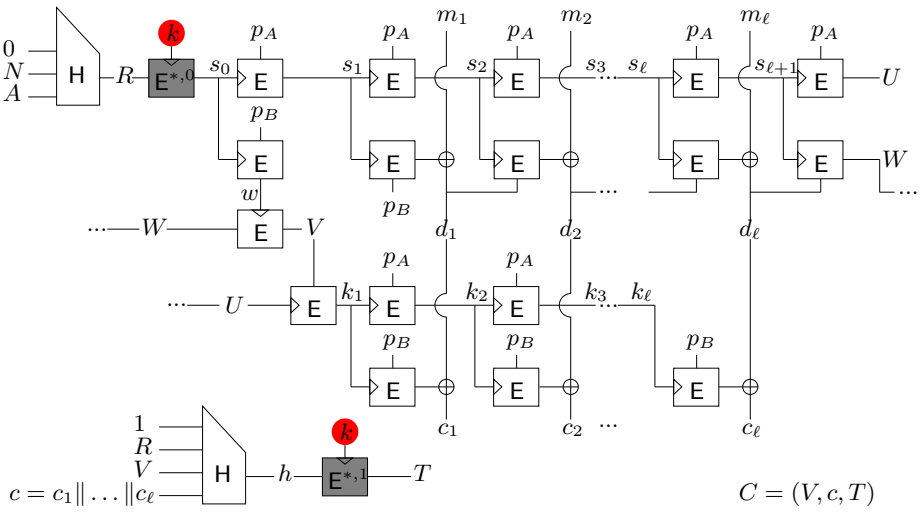

Fig. 4: FEMALE encryption algorithm. "Leak free" block ciphers are in gray with "*". $\mathsf{E}^{*,b} : \mathcal{B} \mapsto \mathcal{M}$ is such that $\mathsf{E}^{*,b}(B) := \mathsf{E}^{*}(b, B)$. Triangles in block ciphers indicate key inputs. If $M \in \mathcal{M}^{*}$ is such that $M = (m_1, \ldots, m_\ell)$, the output ciphertext is $C = (V, c, T) \in \mathcal{M}^{\ell+2}$. (Top) Generates (U, V) and $d = (d_1, \ldots, d_\ell)$, a pre-encryption of $M = (m_1, \ldots, m_\ell)$. (Middle) One-time encryption of d into $c = (c_1, \ldots, c_\ell)$ with one-time key U and pseudorandom IV V. (Bottom) Authentication from tag T.

(i) *Ephemeral key-IV generation:* on input (N, A, M) with $M \in \mathcal{M}^{*}$, derives a pseudorandom ephemeral key U depending only on (N, A) as well as a pseudorandom IV V depending on the whole triple. During this process all the blocks m_i of $M = (m_1, \ldots, m_\ell)$ are "pre-encrypted" as d_i, resulting in $d = (d_1, \ldots, d_\ell)$, where d_i depends on $(N, A, m_1, \ldots, m_i)$;

(ii) *One-time encryption:* on input (V, d) and the ephemeral key U, produces a one-time encryption c of d with initialized vector V;

(iii) *Authentication:* on input (R, V, c), where $R = \mathsf{H}(0\|N\|A)$, computes a pseudorandom tag T.

The ciphertext is given by $C = (V, c, T)$ and it does not include d. To decrypt the ciphertext (N, A, C), FEMALE first checks (iii) before deriving the one-time key U from (N, A) as in step (i) in order to decrypt c into d as the reverse process of step (ii). Eventually, (N, A, d) allows retrieving M at step (i). The full specification of FEMALE is available in Figure 5.

Security Analysis We show that FEMALE satisfies AEML security. In the black-box setting, MR follows from the fact that V is pseudorandom on (N, A, M) even for a variable-length M. This is because W remains secret and already depends on all the (ordered) input. Computing and giving $V = \mathsf{E}_w(W)$ rather than W prevents continuing the computation of the first stage of the encryption algorithm from multiple encryption queries (and captures the message length).

The CIML2 property mainly stems from the last stage of the encryption algorithm and how this stage is verified from the tag in the decryption algorithm. The result follows the technique used in [BPPS17] where tag verifications are made by inverting the SPRP (which is useless to produce valid fresh tags).

Therefore, our main focus is on CCAmL2 security. To make it formal, we define the leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ of FEMALE as:

– $\mathsf{L}_{\mathsf{enc}}$ consists of the leakages that are generated during the encryption:

Description of FEMALE:

Gen(1^n) picks a random key $k \xleftarrow{\$} \{0,1\}^n = \mathcal{K}$.

Enc$_k$(N, A, M) parses $M \in \mathcal{M}^*$ into as many blocks as needed as $M = (m_1, \ldots, m_\ell)$ for some ℓ. Computes $R \leftarrow \mathsf{H}(0\|N\|A)$. Then:

1. *Ephemeral key-IV generation:* (skip step (b) if $\ell = 0$)
   (a) Computes $s_0 \leftarrow \mathsf{E}^*_k(0\|R)$, $w \leftarrow \mathsf{E}_{s_0}(p_B)$, $s_1 \leftarrow \mathsf{E}_{s_0}(p_A)$, and sets $d_0 \leftarrow p_B$;
   (b) Computes $s_{i+1} \leftarrow \mathsf{E}_{s_i}(p_A)$, $y_i \leftarrow \mathsf{E}_{s_i}(d_{i-1})$, $d_i \leftarrow y_i \oplus m_i$, for $i = 1$ to ℓ;
   (c) Computes $U \leftarrow \mathsf{E}_{s_{\ell+1}}(p_A)$, $W \leftarrow \mathsf{E}_{s_{\ell+1}}(d_\ell)$, $V \leftarrow \mathsf{E}_w(W)$.
2. *One-time encryption:* first computes $k_1 \leftarrow \mathsf{E}_U(V)$ and then, for $i = 1$ to $\ell - 1$, computes $k_{i+1} \leftarrow \mathsf{E}_{k_i}(p_A)$, $z_i \leftarrow \mathsf{E}_{k_i}(p_B)$, and $c_i \leftarrow z_i \oplus d_i$.
3. *Authentication:* sets $c = c_1\| \ldots \|c_\ell$, computes $h \leftarrow \mathsf{H}(1\|R\|V\|c)$, and computes $T \leftarrow \mathsf{E}^*_k(1\|h)$.

Eventually, returns the ciphertext $C = (V, c, T)$.

Dec$_k$(N, A, C) parses $C = (V, c, T)$, $c = c_1\| \ldots \|c_\ell$, then proceeds in four phases:

1. *Integrity Checking:* computes $R \leftarrow \mathsf{H}(0\|N\|A)$, $h \leftarrow \mathsf{H}(1\|R\|V\|c)$, and $h^* \leftarrow \mathsf{trunc}((\mathsf{E}^*_k)^{-1}(T))$, where trunc drops the first bit of its input. Then, if $h^* = h$, it enters the next phase, and returns $\perp$ otherwise.
2. *Ephemeral key extraction:* first computes $s_0 \leftarrow \mathsf{E}^*_k(0\|R)$ and $s_{i+1} \leftarrow \mathsf{E}_{s_i}(p_A)$, for $i = 0$ to ℓ, and finally $U \leftarrow \mathsf{E}_{s_{\ell+1}}(p_A)$.
3. *One-time decryption:* first computes $k_1 \leftarrow \mathsf{E}_U(V)$ and then, for $i = 1$ to $\ell - 1$, computes $k_{i+1} \leftarrow \mathsf{E}_{k_i}(p_A)$, $z_i \leftarrow \mathsf{E}_{k_i}(p_B)$, and $d_i \leftarrow z_i \oplus c_i$; Set $d_0 \leftarrow p_B$.
4. *Message recovery:* for $i = 1$ to ℓ, computes $y_i \leftarrow \mathsf{E}_{s_i}(d_{i-1})$ and $m_i \leftarrow y_i \oplus d_i$.

Eventually, returns the message $M = (m_1, \ldots, m_\ell)$.

Fig. 5: The FEMALE AEAD scheme.

- the leakages $\mathsf{L}_\mathsf{E}(s, x)$ generated by all the internal calls to $\mathsf{E}_s(x)$, and
- the leakages $\mathsf{L}_\oplus(a, b)$ generated by all the internal actions $a \oplus b$, and
- all the intermediate values involved in the computations of the hash functions (i.e., hash functions are non-protected, and leak everything).

– L_dec consists of the above that are generated during the decryption.

Our security reduction is made against (i) the simulatability of the leaking blocks, (ii) the security of the encryption of one single block with a fresh key.

Description of LRSE scheme:

RSGen(1^n) picks $k_{ch} \xleftarrow{\$} \{0,1\}^n$, $\mathcal{M}, \mathcal{C} = \{0,1\}^n$ ($p_0, p_1 \in \{0,1\}^n$ can be chosen by the adversary with $p_0 \neq p_1$)

RSEnc$_{k_{ch}}$(m) returns (k_{up}, c), where $c = y_{ch} \oplus m$, $y_{ch} = \mathsf{E}_{k_{ch}}(p_1)$, and $k_{up} = \mathsf{E}_{k_{ch}}(p_0)$. (The term "up" is short for "update".)

RSDec$_{k_{ch}}$(c) proceeds in the natural way.

The leakage $\mathsf{L}_\mathsf{RSE} = (\mathsf{L}_\mathsf{rsenc}, \mathsf{L}_\mathsf{rsdec}, k_{pre})$ resulting from the LRSE implementation is defined as $\mathsf{L}_\mathsf{rsenc}(k_{ch}, m) = (\mathsf{L}_\mathsf{E}(k_{ch}, p_0), \mathsf{L}_\mathsf{E}(k_{ch}, p_1), \mathsf{L}_\oplus(y_{ch}, m), \mathcal{S}^\mathsf{L}(k_{pre}, p_0, k_{ch}))$, $\mathsf{L}_\mathsf{rsdec}(k_{ch}, c) = (\mathsf{L}_\mathsf{E}(k_{ch}, p_0), \mathsf{L}_\mathsf{E}(k_{ch}, p_1), \mathsf{L}_\oplus(y_{ch}, c), \mathcal{S}^\mathsf{L}(k_{pre}, p_0, k_{ch}))$ for a fixed random $k_{pre} \xleftarrow{\$} \{0,1\}^n$. As usual we denote $\mathsf{LRSEnc}_{k_{ch}}(m) = (\mathsf{RSEnc}_{k_{ch}}(m), \mathsf{L}_\mathsf{rsenc}(k_{ch}, m))$.

Fig. 6: Basic unit: the single-block encryption scheme LRSE.

Following Pereira et al.'s approach [PSV15], we consider a Leaking Real Single-block Encryption scheme LRSE defined in Figure 6 as the basic unit of FEMALE. As motivated in the previous section, it is introduced to determine the CCAmL2 security bound. Since for each generated key k_{ch} LRSE is used to encrypt a single message m composed of a single block, we assume that given a security parameter n, LRSE is $(p, q_l, t, \varepsilon_{s\text{-}block})$ secure in the sense that for any (q_l, t)-bounded eavesdropper

adversary $\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}$ choosing $m^0, m^1 \in \mathcal{M}$, we have:

$$\left| \Pr[\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}(\mathsf{LRSEnc}^{+}_{k_{ch}}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}(\mathsf{LRSEnc}^{+}_{k_{ch}}(m^1)) \Rightarrow 1] \right| \leq \varepsilon_{s\text{-}block}, \tag{1}$$

where $\mathsf{LRSEnc}^{+}_{k_{ch}}(m^b) = (\mathsf{LRSEnc}_{k_{ch}}(m^b), [\mathsf{L}_{\mathsf{rsdec}}(k_{ch}, c^b)]^{p-1}, k_{pre})$ for $(c^b, k_{up}) = \mathsf{RSEnc}_{k_{ch}}(m^b)$—here the superscript $p-1$ indicates the adversary could measure the trace $p-1$ times (this typically *doesn't* give identical traces). The reason why the adversary also gets the auxiliary outputs k_{pre} and k_{up} is for composability purpose (which appear in the proof).

With the above, our CCAmL2 result is informally given below. The precise bounds (for both theorems below) can be found in Appendix D.

Theorem 4 (informal). *Let $\mathsf{H} : \{0,1\}^* \to \mathcal{B}$ be a $(0, t', \varepsilon_{cr})$-collision resistant and $(q_d, t', \varepsilon_{pr})$-range-oriented preimage resistant hash function. And let $\mathsf{E} : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2q_e+2q_d+2, t', \varepsilon_{\mathsf{E}})$-SPRP with two implementations: a strongly protected implementation E^* is leak free, and a plain implementation E have leakage function L_{E} that is $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable. Then the* FEMALE *implementation with leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ is $(q_e, q_d, p-1, q_l, t, \varepsilon_{\mathsf{CCAmL2}})$* CCAmL2*-secure, with*

$$\varepsilon_{\mathsf{CCAmL2}} \leq O\left(\frac{t^2}{2^n}\right) + O\left(\frac{t(q_e+q_d)}{2^n}\right) + O\left(\frac{\sigma \cdot t + \sigma^2}{2^n}\right) + O\left(\sigma(\varepsilon_{(p,2)\text{-}rsim} + \varepsilon_{s\text{-}block})\right),$$

where σ denotes the total number of blocks in the challenge messages.

The proof is given in Appendix D. The term $O\big(\sigma(\varepsilon_{(p,2)\text{-}rsim} + \varepsilon_{s\text{-}block})\big)$ corresponds to the security degradation in the presence of side-channel leakage & the reduction to a single block encryption. The bound is roughly of birthday type, and it shows FEMALE is secure up to $t = 2^{n/2} \cdot c^{-1}$ computation and encrypting $2^{n/2} \cdot c^{-1}$ message blocks, where $c \geq 1$ depends on the side-channel strength of the concrete (plain) implementation E.

The CIML2 and MR security of FEMALE are finally presented below.

Theorem 5 (informal). *Let $\mathsf{H} : \{0,1\}^* \to \mathcal{B}$ be a $(0, t', \varepsilon_{cr})$-collision resistant and $(q_d+1, t', \varepsilon_{pr})$-range-oriented preimage resistant hash function. And let $\mathsf{E} : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2q_e+2q_d+2, t', \varepsilon_{\mathsf{E}})$-SPRP, with E^* being its strongly protected leak free implementation. Then:*

(1) FEMALE *provides $(q_e, q_d, t, \varepsilon_{\mathsf{CIML2}})$-ciphertext integrity with coin misuse and unbounded leakage on encryption and decryption as long as $t \leq t' - O(q_e + q_d)$, where $\varepsilon_{\mathsf{CIML2}} \leq O\big(\frac{t^2}{2^n} + \frac{t(q_e+q_d)}{2^n}\big)$;*

(2) FEMALE *scheme is $(q_e, q_d, t, \varepsilon_{mr})$-MR as long as $t \leq t' - O(q_e+q_d)$, where $\varepsilon_{\mathsf{MR}} \leq O\big(\frac{t^2}{2^n} + \frac{\sigma \cdot t}{2^n} + \frac{\sigma^2}{2^n}\big)$, σ denotes the total number of blocks in the encryption queries.*

The proofs are in Appendices D.3 and D.4. Both bounds are roughly birthday.

6.2 AEDT design and analysis

AEDT is a simplified version of FEMALE achieving only AEmL security, that is, dropping the black-box MR requirement. The AEDT encryption algorithm has only two stages: given (N, A, M) with an ℓ-block message M, it computes the hash $R = \mathsf{H}(0\|N\|A)$, and then just uses $\mathsf{E}^*_k(0\|R)$ as the initial key to "start" a one-time encryption of M (in the nonce-respecting case this process will indeed occur once). Then, AEDT applies the same authentication principle as FEMALE. The encryption algorithm is depicted in Figure 7.

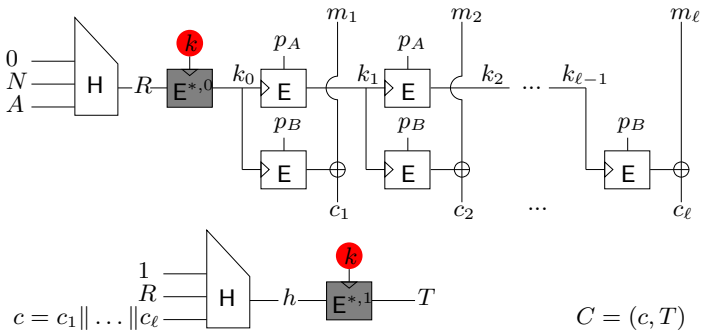

Fig. 7: (Top) Initialization and encryption parts of AEDT, with notations to be used in its specification and security analysis. The meaning of $\mathsf{E}^{*,b}$ is similar to Figure 4. (Bottom) Authentication part of AEDT, with notations consistent with the top half. The final output is $C = (c, T)$.

AEDT uses 2 leak-free blocks, just as FEMALE, but is twice more efficient in terms of number of weakly protected blocks to evaluate. Furthermore, its encryption can be performed "on-the-fly" by pushing each ciphertext block c_i directly into the hash function (if it proceeds block by block).

The CIML2 security of AEDT follows from the one of the EDT mode in [BPPS17]. Its CCAmL2 security is is analyzed in a similar way as FEMALE and is deferred to Appendix E, resulting in similar bounds.

Acknowledgments. Thomas Peters and François-Xavier Standaert are post-doctoral researcher and senior associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S). This work has been funded in part by the ERC consolidator grant SWORD (724725), and also by the EU and Walloon Region through the FEDER project USERMedia (501907-379156).

References

[ABL+14] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. How to securely release unverified plaintext in authenticated encryption. In *ASIACRYPT (1)*, volume 8873 of *LNCS*, pages 105–125. Springer, 2014.

[ADL17] Tomer Ashur, Orr Dunkelman, and Atul Luykx. Boosting authenticated encryption robustness with minimal modifications. In *CRYPTO (3)*, volume 10403 of *LNCS*, pages 3–33. Springer, 2017.

[APW09] Martin R. Albrecht, Kenneth G. Paterson, and Gaven J. Watson. Plaintext recovery attacks against SSH. In *IEEE Symposium on Security and Privacy*, pages 16–26. IEEE Computer Society, 2009.

[AS11] Elena Andreeva and Martijn Stam. The symbiosis between collision and preimage resistance. In *IMA Int. Conf.*, volume 7089 of *LNCS*, pages 152–171. Springer, 2011.

[BDPS13] Alexandra Boldyreva, Jean Paul Degabriele, Kenneth G. Paterson, and Martijn Stam. On symmetric encryption with distinguishable decryption failures. In *FSE*, volume 8424 of *LNCS*, pages 367–390. Springer, 2013.

[BGS15] Sonia Belaïd, Vincent Grosso, and François-Xavier Standaert. Masking and leakage-resilient primitives: One, the other(s) or both? *Cryptography and Communications*, 7(1):163–184, 2015.

[BKP+18] Francesco Berti, François Koeune, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Ciphertext integrity with misuse and leakage: Definition and efficient constructions with symmetric primitives. In *AsiaCCS*, pages 37–50. ACM, 2018.

[BMOS17] Guy Barwell, Daniel P. Martin, Elisabeth Oswald, and Martijn Stam. Authenticated encryption in the face of protocol and side channel leakage. In *ASIACRYPT (1)*, volume 10624 of *LNCS*, pages 693–723. Springer, 2017.

[BN08] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. *J. Cryptology*, 21(4):469–491, 2008.

[BPPS17] Francesco Berti, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. On leakage-resilient authenticated encryption with decryption leakages. *IACR Trans. Symmetric Cryptol.*, 2017(3):271–293, 2017.

[BPS15] Guy Barwell, Daniel Page, and Martijn Stam. Rogue decryption failures: Reconciling AE robustness notions. In *IMA Int. Conf.*, volume 9496 of *LNCS*, pages 94–111. Springer, 2015.

[DEM+17] Christoph Dobraunig, Maria Eichlseder, Stefan Mangard, Florian Mendel, and Thomas Unterluggauer. ISAP - towards side-channel secure authenticated encryption. *IACR Trans. Symmetric Cryptol.*, 2017(1):80–105, 2017.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008.

[DP10] Yevgeniy Dodis and Krzysztof Pietrzak. Leakage-resilient pseudorandom functions and side-channel attacks on feistel networks. In *CRYPTO*, volume 6223 of *LNCS*, pages 21–40. Springer, 2010.

[DR11] Thai Duong and Juliano Rizzo. Cryptography in the web: The case of cryptographic design flaws in ASP.NET. In *IEEE Symposium on Security and Privacy*, pages 481–489. IEEE Computer Society, 2011.

[FFL12] Ewan Fleischmann, Christian Forler, and Stefan Lucks. Mcoe: A family of almost foolproof on-line authenticated encryption schemes. In *FSE*, volume 7549 of *LNCS*, pages 196–215. Springer, 2012.

[FH15] Benjamin Fuller and Ariel Hamlin. Unifying leakage classes: Simulatable leakage and pseudoentropy. In *ICITS*, volume 9063 of *LNCS*, pages 69–86. Springer, 2015.

[FPS12] Sebastian Faust, Krzysztof Pietrzak, and Joachim Schipper. Practical leakage-resilient symmetric cryptography. In *CHES*, volume 7428 of *LNCS*, pages 213–232. Springer, 2012.

[GPSS18] Benjamin Grégoire, Kostas Papagiannopoulos, Peter Schwabe, and Ko Stoffelen. Vectorizing higher-order masking. In *COSADE*, volume 10815 of *LNCS*, pages 23–43. Springer, 2018.

[GR17] Dahmun Goudarzi and Matthieu Rivain. How fast can higher-order masking be in software? In *EUROCRYPT (1)*, volume 10210 of *LNCS*, pages 567–597, 2017.

[HKR15] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust authenticated-encryption AEZ and the problem that it solves. In *EUROCRYPT (1)*, volume 9056 of *LNCS*, pages 15–44. Springer, 2015.

[HL11] Shai Halevi and Huijia Lin. After-the-fact leakage in public-key encryption. In *TCC*, volume 6597 of *LNCS*, pages 107–124. Springer, 2011.

[HRRV15] Viet Tung Hoang, Reza Reyhanitabar, Phillip Rogaway, and Damian Vizár. Online authenticated-encryption and its nonce-reuse misuse-resistance. In *CRYPTO (1)*, volume 9215 of *LNCS*, pages 493–517. Springer, 2015.

[HSH+09] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: cold-boot attacks on encryption keys. *Commun. ACM*, 52(5):91–98, 2009.

[JS17] Anthony Journault and François-Xavier Standaert. Very high order masking: Efficient implementation and security evaluation. In *CHES*, volume 10529 of *LNCS*, pages 623–643. Springer, 2017.

[KDH+12] Stéphanie Kerckhof, François Durvaux, Cédric Hocquet, David Bol, and François-Xavier Standaert. Towards green cryptography: A comparison of lightweight ciphers from the energy viewpoint. In *CHES*, volume 7428 of *LNCS*, pages 390–407. Springer, 2012.

[KY00] Jonathan Katz and Moti Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. In *FSE*, volume 1978 of *LNCS*, pages 284–299. Springer, 2000.

[LMO+14] Jake Longo, Daniel P. Martin, Elisabeth Oswald, Daniel Page, Martijn Stam, and Michael Tunstall. Simulatable leakage: Analysis, pitfalls, and new constructions. In *ASIACRYPT (1)*, volume 8873 of *LNCS*, pages 223–242. Springer, 2014.

[MOS11] Stefan Mangard, Elisabeth Oswald, and François-Xavier Standaert. One for all - all for one: unifying standard differential power analysis attacks. *IET Information Security*, 5(2):100–110, 2011.

[MOSW15] Daniel P. Martin, Elisabeth Oswald, Martijn Stam, and Marcin Wójcik. A leakage resilient MAC. In *IMA Int. Conf.*, volume 9496 of *LNCS*, pages 295–310. Springer, 2015.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In *TCC*, volume 2951 of *LNCS*, pages 278–296. Springer, 2004.

[OC15] Colin O'Flynn and Zhizhang (David) Chen. Side channel power analysis of an AES-256 bootloader. In *CCECE*, pages 750–755. IEEE, 2015.

[PA12] Kenneth G. Paterson and Nadhem J. AlFardan. Plaintext-recovery attacks against datagram TLS. In *NDSS*. The Internet Society, 2012.

[Pie09] Krzysztof Pietrzak. A leakage-resilient mode of operation. In *EUROCRYPT*, volume 5479 of *LNCS*, pages 462–482. Springer, 2009.

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

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In *ACM Conference on Computer and Communications Security*, pages 98–107. ACM, 2002.

[RS06] Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In *EUROCRYPT*, volume 4004 of *LNCS*, pages 373–390. Springer, 2006.

[SPY+10] François-Xavier Standaert, Olivier Pereira, Yu Yu, Jean-Jacques Quisquater, Moti Yung, and Elisabeth Oswald. Leakage resilient cryptography in practice. In *Towards Hardware-Intrinsic Security*, Information Security and Cryptography, pages 99–134. Springer, 2010.

[SPY13] François-Xavier Standaert, Olivier Pereira, and Yu Yu. Leakage-resilient symmetric cryptography under empirically verifiable assumptions. In *CRYPTO (1)*, volume 8042 of *LNCS*, pages 335–352. Springer, 2013.

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

A Extention to the multi-challenge setting

We next revisit the confidentiality against chosen-ciphertext and chosen-plaintext adversaries with partial misuse and full leakages in the multi-challenge setting. We show that the single-challenge and multi-challenge settings are equivalent for chosen-ciphertext notions, but not for chosen-plaintext notions.

A.1 Multi-challenge chosen-ciphertext security

$\mathsf{PrivK}^{\mathsf{mCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ is the output of the following experiment:

Initialization: generates a secret key $k \leftarrow \mathsf{Gen}(1^n)$ and sets $\mathcal{E}, \mathcal{E}_{ch} \leftarrow \emptyset$.

Leaking encryption queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{LEnc}(\cdot,\cdot,\cdot)$,
$\mathsf{LEnc}(N,A,M)$ outputs $\perp$ if $(N,*,*) \in \mathcal{E}_{ch}$, else computes $C \leftarrow \mathsf{Enc}_k(N,A,M)$ and $\mathsf{leak_e} \leftarrow \mathsf{L_{enc}}(k,N,A,M)$, updates $\mathcal{E} \leftarrow \mathcal{E} \cup \{N\}$ and finally returns $(C, \mathsf{leak_e})$.

Leaking decryption queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{LDec}(\cdot,\cdot,\cdot)$,
$\mathsf{LDec}(N,A,C)$ outputs $\perp$ if $(N,A,C) \in \mathcal{E}_{ch}$, else computes $M \leftarrow \mathsf{Dec}_k(N,A,C)$ and $\mathsf{leak_d} \leftarrow \mathsf{L_{dec}}(k,N,A,C)$ and returns $(M, \mathsf{leak_d})$; (Where $M = \perp$ may occur.)

Challenge queries: on possibly many occasions $\mathcal{A}^{\mathsf{L}}$ submits $(N_{ch}, A_{ch}, M^0, M^1)$,
If M^0 and M^1 have different (block) length or $N_{ch} \in \mathcal{E}$ or $(N_{ch},*,*) \in \mathcal{E}_{ch}$, returns $\perp$; Else computes $C^b \leftarrow \mathsf{Enc}_k(N_{ch},A_{ch},M^b)$ and $\mathsf{leak}^b_{\mathsf{e}} \leftarrow \mathsf{L_{enc}}(k,N_{ch},A_{ch},M^b)$, updates $\mathcal{E}_{ch} \leftarrow \mathcal{E}_{ch} \cup \{(N_{ch},A_{ch},C^b)\}$ and finally returns $(C^b, \mathsf{leak}^b_{\mathsf{e}})$;

Decryption challenge leakage queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{L_{decch}}(\cdot)$,
$\mathsf{L_{decch}}(i)$ takes the i-th challenge ciphertext $(N_{ch},A_{ch},C^b) \in \mathcal{E}_{ch}$ and outputs a leakage trace $\mathsf{leak}^b_{\mathsf{d}} \leftarrow \mathsf{L_{dec}}(k,N_{ch},A_{ch},C^b)$;

Finalization: $\mathcal{A}^{\mathsf{L}}$ outputs a guess bit b' which is defined as the output of the game.

Fig. 8: The $\mathsf{PrivK}^{\mathsf{mCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ game.

A natural extension of the CCAmL2 experiment of Figure 1 to the multi-challenge setting is given by the mCCAmL2 experiment in Figure 8 . It leads to the following definition.

Definition 14 (mCCAmL2). *(For a security parameter n) a nonce-based authenticated encryption with associated data* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L_{enc}}, \mathsf{L_{dec}})$ *is* $(q_e, q_d, q_c, q_m, q_l, t, \varepsilon)$-mCCAmL2 *secure if for every* $(q_e, q_d, q_c, q_m, q_l, t)$*-bounded adversary* $\mathcal{A}^{\mathsf{L}}$*, we have:*

$$\left|\Pr\left[\mathsf{PrivK}^{\mathsf{mCCAmL2},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{mCCAmL2},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon,$$

where the adversary $\mathcal{A}^{\mathsf{L}}$ *makes at most* q_e *leaking encryption queries,* q_d *leaking decryption queries,* q_c *challenge decryption leakage queries,* q_m *leaking challenge queries and* q_l *leakage evaluation queries on arbitrarily chosen keys.*

Equivalent notions If the adversary $\mathcal{A}$ in the multi-challenge experiment above is restricted to $q_m \leq 1$ challenge query then the mCCAmL2 security and the CCAmL2 security collide. Therefore, mCCAmL2 security implies CCAmL2 security. The next theorem states that the both notions are equivalent.

Theorem 6. *The* CCAmL2 *security is equivalent to the* mCCAmL2 *security. Formally, for any* $(q_e, q_d, q_c, q_m, q_l, t)$*-bounded adversary* $\mathcal{A}^{\mathsf{L}}$ *against the* mCCAmL2 *security of* AEAD *for a security parameter* n*, if* AEAD *is* $(q_e + q_m - 1, q_d + q_c, q_c, q_l, t, \varepsilon_{\mathsf{CCAmL2}})$-CCAmL2 *secure for a security parameter* n*, we have:*

$$\left|\Pr\left[\mathsf{PrivK}^{\mathsf{mCCAmL2},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{mCCAmL2},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right]\right| \leq q_m \times \varepsilon_{\mathsf{CCAmL2}}.$$

Proof. Given an adversary $\mathcal{A}^{\mathsf{L}}$ against the mCCAmL2 security of AEAD making q_m leaking challenge queries, we build q_m adversaries $\mathcal{A}'_i$ against the CCAmL2 security of AEAD, for $i = 1$ to q_m. To simulate the mCCAmL2 game in front of $\mathcal{A}^{\mathsf{L}}$, each $\mathcal{A}'_i$ has to emulate the responses to the different types of queries in $\mathsf{PrivK}^{\mathsf{mCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$. Therefore, $\mathcal{A}'_i$ proceeds as follows with $\mathcal{E}'_i = \emptyset$:

Leaking encryption queries: when $\mathcal{A}^{\mathsf{L}}$ queries $\mathsf{LEnc}(N, A, M)$,
If $(N, *, *) \in \mathcal{E}'_i$, $\mathcal{A}'_i$ returns $\perp$, else $\mathcal{A}'_i$ queries $\mathsf{LEnc}(N, A, M)$ and gets back either $\perp$ or $(C, \mathsf{leak_e})$ which it sends to $\mathcal{A}^{\mathsf{L}}$.

Leaking decryption queries: when $\mathcal{A}^{\mathsf{L}}$ queries $\mathsf{LDec}(N, A, C)$,
If $(N, A, C) \in \mathcal{E}'_i$, $\mathcal{A}'_i$ returns $\perp$, else $\mathcal{A}'_i$ queries $\mathsf{LDec}(N, A, C)$ and gets back either $\perp$ or $(M, \mathsf{leak_d})$, where $M = \perp$ may occur, and sends it to $\mathcal{A}^{\mathsf{L}}$.

Challenge queries: for $j = 1$ to q_m, $\mathcal{A}^{\mathsf{L}}$ queries $(N^j_{ch}, A^j_{ch}, M^0_j, M^1_j)$,
If $(N^j_{ch}, *, *) \in \mathcal{E}'_i$, $\mathcal{A}'_i$ returns $\perp$, else
- $j < i$, $\mathcal{A}'_i$ queries its leaking encryption oracle $\mathsf{LEnc}(N_{ch}, A_{ch}, M^1)$ and gets either $\perp$ or $(C^1_j, \mathsf{leak}^1_{\mathsf{e},j})$ which it sends to $\mathcal{A}^{\mathsf{L}}$. $\mathcal{A}'_i$ updates $\mathcal{E}'_i = \mathcal{E}'_i \cup (N^j_{ch}, A^j_{ch}, C^1_j)$;
- $j = i$, $\mathcal{A}'_i$ queries its leaking *challenge* oracle on $(N^i_{ch}, A^i_{ch}, M^0_i, M^1_i)$ and gets either $\perp$ or $(C^b, \mathsf{leak}^b_{\mathsf{e}})$ which it sends to $\mathcal{A}^{\mathsf{L}}$;
- $j > i$, $\mathcal{A}'_i$ queries its leaking encryption oracle $\mathsf{LEnc}(N_{ch}, A_{ch}, M^0)$ and gets either $\perp$ or $(C^0_j, \mathsf{leak}^0_{\mathsf{e},j})$ which it sends to $\mathcal{A}^{\mathsf{L}}$. $\mathcal{A}'_i$ updates $\mathcal{E}'_i = \mathcal{E}'_i \cup (N^j_{ch}, A^j_{ch}, C^0_j)$.

Decryption challenge leakage queries: when $\mathcal{A}^{\mathsf{L}}$ queries $\mathsf{L_{decch}}(j)$, for $j = 1$ to q_m,
- $j \neq i$, if $\mathcal{A}'_i$ sent $\perp$ to the j-th challenge query made by $\mathcal{A}^{\mathsf{L}}$ it sends $\perp$, else it queries its leaking decryption oracle $(N^j_{ch}, A^j_{ch}, C^{[j<i]}_j)$ and gets $\mathsf{leak}^{[j<i]}_{\mathsf{d},j}$ which it sends to $\mathcal{A}^{\mathsf{L}}$;

– $j = i$, $\mathcal{A}'_i$ queries its oracle $\mathsf{L}_{\mathsf{decch}}$ and receives either $\perp$ or $\mathsf{leak}^b_{\mathsf{d}}$ which it sends to $\mathcal{A}^{\mathsf{L}}$.

Finalization: $\mathcal{A}'_i$ outputs whatever is the $\mathcal{A}^{\mathsf{L}}$'s output bit b'.

The list $\mathcal{E}'_i = \{(N^j_{ch}, A^j_{ch}, C^{[j<i]}_j)\}^{q_m}_{j=1,j\neq i}$ maintained by $\mathcal{A}'_i$ serves to identify forbidden query attempts made by $\mathcal{A}^{\mathsf{L}}$ that would not be deemed as such in the $\mathsf{CCAmL2}$ experiment played by $\mathcal{A}'_i$. The conclusion of the proof easy follows from standard hybrid arguments, the assumption made on AEAD and by evaluating the efficiency of the adversaries. □

By removing the decryption challenge leakage oracle $\mathsf{L}_{\mathsf{decch}}$ from the game in Figure 8, we define the security notion of $\mathsf{mCCAmL2}^*$. It is easy to see that $\mathsf{CCAmL2}^*$ security is equivalent to $\mathsf{mCCAmL2}^*$ security as well.

A.2 Multi-challenge chosen-plaintext case

A natural extension of the $\mathsf{CPAmL2}$ experiment to the multi-challenge setting is obtained by removing the leaking decryption oracle of the above $\mathsf{mCCAmL2}$ experiment. The resulting experiment is called the $\mathsf{mCPAmL2}$ experiment, where the 2 highlights the access to the decryption challenge leakage oracle.

Definition 15 ($\mathsf{mCPAmL2}$). *A nonce-based authenticated encryption scheme with associated data* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakage function* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *is* $(q_e, q_c, q_m, q_l, t, \varepsilon)$-$\mathsf{mCPAmL2}$ *secure for a security parameter* n *if, for every* (q_e, q_c, q_m, q_l, t)*-bounded adversary* $\mathcal{A}^{\mathsf{L}}$*, we have:*

$$\left|\Pr\left[\mathsf{PrivK}^{\mathsf{mCPAmL2},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{mCPAmL2},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right]\right| \leq \varepsilon,$$

when the adversary $\mathcal{A}^{\mathsf{L}}$ *makes at most* q_e *leaking encryption queries,* q_c *challenge decryption leakage queries,* q_m *leaking challenge queries and* q_l *leakage evaluation queries on arbitrarily chosen keys.*

Non equivalent notions In general, while $\mathsf{mCPAmL2}$ security obviously implies $\mathsf{CPAmL2}$ security, the converse is false.

Theorem 7. *Assuming there exists a* $\mathsf{CPAmL2}$ *secure AE, then there exists* $\mathsf{CPAmL2}$ *secure authenticated encryption which is not* $\mathsf{mCPAmL2}$ *secure.*

Essentially, in the $\mathsf{mCPAmL2}$ game an adversary can obtain decryption leakages for multiple tuples (N, A, C) via the multiple challenge queries, whereas in the $\mathsf{CPAmL2}$ game she can obtain such leakages for only one (challenge) tuple (N, A, C). This difference establishes the separation. By removing the $\mathsf{L}_{\mathsf{decch}}$ oracle as well, the aforementioned difference disappears, and we find back the equivalence of $\mathsf{CPAmL1} := \mathsf{CPAmL2}^*$, for the single-challenge notion, with $\mathsf{mCPAmL1} := \mathsf{mCPAmL2}^*$, for the multi challenge notion.

Proof. Let $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be a $\mathsf{CPAmL2}$ secure authenticated encryption with leakage function pair $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$. Then, we build the following authenticated encryption $\mathsf{AEAD}' = (\mathsf{Gen}', \mathsf{Enc}, \mathsf{Dec})$ with leakage function pairs $\mathsf{L}' = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}'_{\mathsf{dec}})$, where N_0, N_1 below are the outputs of a publicly samplable distribution parametrized by n:

$\mathsf{Gen}'(1^n)$**:** runs $k \leftarrow \mathsf{Gen}(1^n)$, $k_0 \leftarrow \mathsf{Gen}(1^n)$ and computes $k_1 = k \oplus k_0$. The secret key is defined as (k, k_0, k_1).

$\mathsf{L}'_{\mathsf{dec}}((k, k_0, k_1), N, A, C)$ outputs $\mathsf{L}_{\mathsf{dec}}(k, N, A, C)$ and possibly the following additional decryption leakage:

- If $N = N_0$, gives k_0;
- If $N = N_1$, gives k_1.

Since an adversary against the CPAmL2 security of AEAD′ will never receive leakage traces for more than one ciphertext of the form (N, A, C), it will never get both k_0 and k_1. Since they are random shares of the key k, the CPAmL2 security still holds from AEAD because we can simulated the additional leakage by picking $k' \leftarrow \mathsf{Gen}(1^n)$ and set $k_0 = k'$ or $k_1 = k'$ on-the-fly when needed. However, an mCPAmL2 adversary simply has to make two challenge queries involving N_0 and N_1 and then to call the decryption challenge leakage oracle $\mathsf{L}_{\mathsf{decch}}$ on the corresponding challenge ciphertexts received as answers to get k_0 and k_1, from which the secret key $k = k_0 \oplus k_1$ can be efficiently computed. □

B EavDL and AEML

The notion EavDL was introduced by Berti et al. [BKP+18]. It formalizes message confidentiality in a context where an adversary can observe decryption leakages but not the corresponding messages. This setting is motivated by applications such as secure bootloading and bitstream decryption. In this section, we recall this notion, and make discussion on the links between this notion and the other ones.

B.1 mEavDL and its extension mEavDL2

The original definition given by Berti et al. is of a single-challenge form. To save space, we concentrate on its multi-challenge variant mEavDL, which is based on the experiment in Fig. 9.

$\mathsf{PrivK}^{\mathsf{mEavDL}(2),b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}(1^n)$ is the output of the following experiment.

Initialization: generates a secret key $k \leftarrow \mathsf{Gen}(1^n)$ and sets $\mathcal{E}_{ch} \leftarrow \emptyset$.

Challenge queries: on possibly many occasions $\mathcal{A}^{\mathsf{L}}$ submits $(N_{ch}, A_{ch}, M^0, M^1)$,
If M^0 and M^1 have different (block) length or $N_{ch} \in \mathcal{E}$, returns $\perp$; Else updates $\mathcal{E}_{ch} \leftarrow \mathcal{E}_{ch} \cup \{N_{ch}\}$ and finally

- $\mathsf{PrivK}^{\mathsf{mEavDL},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$: computes and returns $C^b \leftarrow \mathsf{Enc}_k(N_{ch}, A_{ch}, M^b)$;
- $\mathsf{PrivK}^{\mathsf{mEavDL2},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$: computes and returns $(C^b, \mathsf{leak}^b_{\mathsf{e}}) \leftarrow \mathsf{LEnc}_k(N_{ch}, A_{ch}, M^b)$.

Decryption leakage queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{L}_{\mathsf{Dec}}(\cdot, \cdot, \cdot)$,
$\mathsf{L}_{\mathsf{Dec}}(N, A, C)$ outputs $\mathsf{leak}_{\mathsf{d}} \leftarrow \mathsf{L}_{\mathsf{dec}}(k, N, A, C)$.

Finalization: $\mathcal{A}^{\mathsf{L}}$ outputs a guess bit b'. If $b = b'$, return 1, else return 0.

Fig. 9: The $\mathsf{PrivK}^{\mathsf{mEavDL},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$ and $\mathsf{PrivK}^{\mathsf{mEavDL2},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$ games.

Note that the challenge nonce has to be respecting for both mEavDL and mEavDL2. If we restrict the number of challenges to 1 in mEavDL then we recover the original EavDL notion of Berti et al. [BKP+18]

Definition 16 (mEavDL: Eavesdropper Security with Decryption Leakage). *An authenticated encryption* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with decryption leakage function* $\mathsf{L}_{\mathsf{dec}}$ *provides* $(q_m, q_d, t, \varepsilon)$*-indistinguishability of ciphertexts against eavesdropping with differential leakage attacks for a security parameter* n*, or is* $(q_m, q_d, t, \varepsilon)$-mEavDL *secure for short, if for any adversary* $\mathcal{A}$ *that makes at most* q_m *challenge queries,* q_d *queries to* $\mathsf{L}_{\mathsf{dec}}$*, and runs in time* t*,*

$$\Pr\left[\mathsf{PrivK}^{\mathsf{mEavDL},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}(1^n) \Rightarrow 1\right] \leq \frac{1}{2} + \varepsilon$$

for the $\mathsf{PrivK}^{\mathsf{mEavDL},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$ *game defined in Fig. 9.*

Stronger variant: mEavDL2 We define a strengthened variant mEavDL2, which is mEavDL enhanced with *challenge encryption leakages* so that the "2" means that both the leaking oracles in encryption and decryption are available.

Definition 17 (mEavDL2: Eavesdropper Security with Encryption & Decryption Leakage). *An authenticated encryption* $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *provides* $(q_m, q_d, t, \varepsilon)$*-indistinguishability of ciphertexts against eavesdropping with differential leakage attacks in encryption and decryption for a security parameter* n*, or is* $(q_m, q_d, t, \varepsilon)$-mEavDL2 *secure for short, if for any adversary* $\mathcal{A}$ *that makes at most* q_m leaking *challenge queries,* q_d *queries to* $\mathsf{L}_{\mathsf{dec}}$*, and runs in time* t*,*

$$\Pr\left[\mathsf{PrivK}^{\mathsf{mEavDL},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}(1^n) \Rightarrow 1\right] \leq \frac{1}{2} + \varepsilon$$

for the $\mathsf{PrivK}^{\mathsf{mEavDL2},b}_{\mathcal{A}^{\mathsf{L}},\mathsf{AEAD}}$ *game defined in Fig. 9.*

mCCAmL2 ⇒ mEavDL2 Our definition of mCCAmL2 almost explicitly incorporates the elements of mEavDL2, and thus mCCAmL2 ⇒ mEavDL2. This means CCAmL2 captures all pre-existing confidentiality notions.

B.2 mCCAmL2* ⇏ EavDL

In this subsection, we show the necessity of including challenge decryption leakage, by showing that the weakened version mCCAmL2* does *not* imply EavDL (neither EavDL2, of course). The idea is that if the decryption leakages always leak the decrypted plaintext, then it remains possible to retain mCCAmL2* security; yet, this feature immediately ruins out the possibility of EavDL, since in the EavDL security game, the challenge plaintext M^b has to be hidden from $\mathcal{A}$.

More clearly, let $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ be mCCAmL2* secure with respect to L. Then we build $\mathsf{AEAD}' = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with leakage $\mathsf{L}' = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}'_{\mathsf{dec}})$ such that:

$\mathsf{L}'_{\mathsf{dec}}(k, N, A, C)$**:** outputs $\mathsf{leak}_{\mathsf{d}} = \mathsf{L}_{\mathsf{dec}}(k, N, A, C)$ if $\mathsf{Dec}_k(N, A, C) = \perp$ and outputs $(\mathsf{leak}_{\mathsf{d}}, M)$ otherwise, where $\mathsf{leak}_{\mathsf{d}} = \mathsf{L}_{\mathsf{dec}}(k, N, A, C)$, $M = \mathsf{Dec}_k(N, A, C)$.

It's not hard to see AEAD* remains mCCAmL2* secure, since the additional decryption leakage (the correct message) essentially contains no new information. However, during the EavDL security game, the challenge plaintext M^b is directly given by this leakage, allowing to precisely determine the value of b.

B.3 $\mathsf{MR} \wedge \mathsf{mCPAmL2} \wedge \mathsf{CIML2} \wedge \mathsf{mEavDL2} \not\Rightarrow \mathsf{CCAmL2}^*$

In subsection 3.3 we have proved $\mathsf{MR} \wedge \mathsf{CPAmL2} \wedge \mathsf{CIML2} \not\Rightarrow \mathsf{CCAmL2}^*$. In this subsection we prove an even stronger claim of $\mathsf{MR} \wedge \mathsf{mCPAmL2} \wedge \mathsf{CIML2} \wedge \mathsf{mEavDL2} \not\Rightarrow \mathsf{CCAmL2}^*$. The idea stems from the following observations:

- in the MR game, no leakage traces are given;
- in the $\mathsf{mCPAmL2}$ game, decryption leakage is only available for challenge ciphertexts, and thus "nonce respecting";
- in the $\mathsf{CIML2}$ game, additional information unrelated to the key k but only to messages M are irrelevant;
- in the $\mathsf{mEavDL2}$ game, assuming $\mathsf{CIML2}$ holds, the decryption leakage oracle will essentially be "nonce respecting";
- on the other hand, in the $\mathsf{CCAmL2}^*$ game, many valid decryption leakages with respect to *a single* are available through trivial leaking decryption query, i.e. from ciphertext returned by the leaking encryption oracle.

Let $\mathsf{AEAD} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be MR, $\mathsf{mCPAmL2} \wedge \mathsf{mEavDL2}$ with respect to leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ and $\mathsf{CIML2}$ with respect to L^*. Then we build $\mathsf{AEAD}' = (\mathsf{Gen}', \mathsf{Enc}, \mathsf{Dec})$ with leakage function $\mathsf{L}' = (\mathsf{L}'_{\mathsf{enc}}, \mathsf{L}'_{\mathsf{dec}})$ as follows, where $N^\dagger, N^\circ, M^\dagger$ below are the outputs of a publicly samplable distribution parametrized by n:

$\mathsf{Gen}'(1^n)$**:** generates $k \leftarrow \mathsf{Gen}(1^n)$ and samples two random bits t_0, t_1. It returns (k, sh) where $sh = (t_0, t_1)$.

$\mathsf{L}'_{\mathsf{enc}}((k, sh), N, A, M)$**:** outputs $\mathsf{leak}_{\mathsf{e}} = \mathsf{L}_{\mathsf{enc}}(k, N, A, M)$ as well as the additional value B but only in two cases:
- Case 1: $N = N^\dagger$ and $M = M^\dagger$, then $B = t_0 \oplus t_1 \oplus 1$;
- Case 2: $N = N^\dagger$ and $M \neq M^\dagger$, then $B = t_0 \oplus t_1$.

$\mathsf{L}'_{\mathsf{dec}}((k, sh), N, A, C)$**:** outputs $\mathsf{leak}_{\mathsf{d}} = \mathsf{L}_{\mathsf{dec}}(k, N, A, C)$ as well as the additional value B if $\mathsf{Dec}_k(N, A, C) = M \neq \perp$ and:
- Case 1: $N = N^\circ$ and $M = M^\dagger$, then $B = t_0$;
- Case 2: $N = N^\circ$ and $M \neq M^\dagger$, then $B = t_1$.

We establish the desired claims in turn:

MR. Both scheme are the same from a black-box perspective.

$\mathsf{CIML2}$. Given a $\mathsf{CIML2}$ adversary $\mathcal{A}'$ against AEAD', it's not hard to see a $\mathsf{CIML2}$ adversary $\mathcal{A}$ could use the oracles of AEAD and internally sampled bits t_0 and t_1 to perfectly simulate the oracles of AEAD' in front of $\mathcal{A}'$. By this, AEAD' is $\mathsf{CIML2}$ as long as AEAD is $\mathsf{CIML2}$.

$\mathsf{mCPAmL2}$. we show that if there is a $\mathsf{mCPAmL2}$ adversary $\mathcal{A}'$ against AEAD', then there is an adversary $\mathcal{A}$ which uses $\mathcal{A}'$ to break the $\mathsf{mCPAmL2}$ security of AEAD. In detail, once $\mathsf{PrivK}^{\mathsf{CCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n)$ is setup, $\mathcal{A}(1^n)$ publicly samples $N^\dagger, N^\circ, M^\dagger$ and sends to $\mathcal{A}'$ whatever is specified by AEAD'. $\mathcal{A}$ also picks two random bits t_0, t_1. Then it runs $\mathcal{A}'$ and simulates $\mathsf{PrivK}^{\mathsf{mCPAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}$ using its own interaction $\mathsf{PrivK}^{\mathsf{mCPAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$. For each query from $\mathcal{A}'$, the actions of $\mathcal{A}$ are as follows:

Leaking (non-challenge) encryption queries: On input (N, A, M),
(i) $\mathcal{A}$ queries its own leaking encryption oracle on (N, A, M) and gets back some $(C, \mathsf{leak}_{\mathsf{e}})$ or possibly $\perp$;

(ii) If not $\perp$, $\mathcal{A}$ checks whether it should append an additional leakage B to $(C, \mathsf{leak_e})$ in the case (N, A, M) falls into one of the two cases described above.

$\mathcal{A}$ respectively returns $(C, \mathsf{leak_e})$ or $\perp$ or even $(C, \mathsf{leak_e}, B)$ to $\mathcal{A}'$ according to the above situations.

Leaking challenge query: on input $(N_{\mathsf{ch}}, A_{\mathsf{ch}}, M^0, M^1)$, $\mathcal{A}$ sends it to its leaking challenge oracle and gets the tuple $(C^b, \mathsf{leak}_{\mathsf{e}}^b)$ or possibly $\perp$. In the latter case it returns $\perp$ as well, otherwise

(i) if $N_{\mathsf{ch}} = N^\dagger$, regardless of what is encrypted by the challenge oracle, $\mathcal{A}$ uniformly samples a new random bit s^* and returns $(C^b, \mathsf{leak}_{\mathsf{e}}^b, s^*)$;

(ii) else, $\mathcal{A}$ just returns $(C^b, \mathsf{leak}_{\mathsf{e}}^b)$.

Challenge decryption leakage: on input i, $\mathcal{A}$ sends i to its challenge decryption leakage oracle and gets $\mathsf{leak}_{\mathsf{d}}^b$. Then,

(i) if the i-th challenge ciphertext contains $N_{\mathsf{ch}} = N^\circ$, $\mathcal{A}$ returns $(\mathsf{leak}_{\mathsf{d}}^b, t_0)$;

(ii) else, $\mathcal{A}$ just returns $\mathsf{leak}_{\mathsf{d}}^b$.

Eventually, $\mathcal{A}$ outputs the bit returned by $\mathcal{A}'$.

Now we explain why $\mathcal{A}$ properly emulates the mCPAmL2 game in front of $\mathcal{A}'$. As long as $N^\dagger$ is never involved in a challenge query, the additional decryption leakage associated to the nonce N° is independent of the involved message. Therefore, the simulation is of no deviation.

We next concentrate on the case $N^\dagger$ is first involved in a challenge query. Since $N^\dagger$ can only appear once there it means that $\mathcal{A}$ will never sees either $t_0 \oplus t_1 \oplus 1$ or $t_0 \oplus t_1$. By the definition of mCPAmL2 we know there is at most one challenge query under the nonce N°. This means that even $\mathcal{A}'$ would have been received either t_0 or t_1 in the real game both equally remains uniform. So when $\mathcal{A}$ always gives t_0 it makes no difference in the view of $\mathcal{A}'$. Therefore, the distributions of the transcript of queries and answers obtained in the game $\mathsf{PrivK}^{\mathsf{mCPAmL2},b}_{\mathcal{A},\mathsf{AEAD}',\mathsf{L}}(1^n)$ and the game simulated by $\mathcal{A}$ are the same, and thus

$$\Pr\left[\mathsf{PrivK}^{\mathsf{mCPAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1\right] = \Pr\left[\mathsf{PrivK}^{\mathsf{mCPAmL2},b}_{\mathcal{A}',\mathsf{AEAD}',\mathsf{L}'}(1^n) \Rightarrow 1\right].$$

mEavDL2. The proof just follows the same line as the proof for mCPAmL2, except that the adversary $\mathcal{A}$ aborts if the internally ran $\mathcal{A}'$ manages to call the decryption leakage oracle on a valid ciphertext which is not a challenge ciphertext. Since the probability to abort is bounded by the probability to create a forgery, the abort probability is at most $\Pr[\mathsf{PrivK}^{\mathsf{CIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}(1^n) \Rightarrow 1]$, which is small since AEAD is CIML2 secure. Now, assuming that abort does not occur we are in the same situation than in the mCPAL1 argument above.

$\neg$CCAmL2*. Let $\mathcal{A}'$ be an adversary which given $N^\dagger, N^\circ, M^\dagger$ sequentially makes the following queries for $M \neq M^\dagger$:

(i) $(N^\circ, A, M^\dagger)$ to the leaking encryption oracle and get $C^\dagger$;
(ii) (N°, A, M) to the leaking encryption oracle and get C;
(iii) $(N^\circ, A, C^\dagger)$ to the leaking decryption oracle and get t_0;
(iv) (N°, A, C) to the leaking decryption oracle and get t_1.

$\mathcal{A}'$ finally submits $(N^\dagger, A, M^\dagger, M)$ and obtains ciphertext C^b and (the additional) leakage bit B. Now whether $B = t_0 \oplus t_1$ or not allows distinguishing.

This concludes our proof for MR $\wedge$ mCPAmL2 $\wedge$ CIML2 $\wedge$ mEavDL2 $\not\Rightarrow$ CCAmL2*. $\square$

C Relation with BMOS

Despite it is hard to prove that our strongly protected SPRP and our leakage assumptions are necessary (which is another important and challenging open problem), we observe that the BMOS work requires very similar ones.

Strongly protected block. The BMOS [BMOS17] security proofs are based on very similar assumptions, and require an ideal component that essentially solves the same challenges as ours which can be viewed as a counterpart to our leak-free SPRP, as we explain next.

Given (N, A, M), the BMOS construction combines an IV-based encryption scheme with a pseudorandom Hash-then-MAC. To verify a tag T, the proposed implementation described in the appendices of the long version of [BMOS17] recomputes fresh shares, say T_1 and T_2, of the right tag $T_1 \cdot T_2$ by applying a PRF on its hashed value, leading to the following observations.

First, the key of this PRF has to be "well protected". The BMOS authors propose an instantiation based on pairings which is proven in the generic group model for this purpose. Critically, this instantiation shares the long-term key in two pieces, which are rerandomized after every call. This creates two security demands: (i) the implementation should be protected well enough so that the two fresh shares do not gradually leak the single long-term secret key of the PRF, and (ii) randomness is needed in order to refresh these shares.

As a result, both BMOS' and our following constructions require the same type of ideal component. Yet, it is worth observing that our model and constructions allow a gradual security degradation. That is, the stronger requirement to perfectly hide one of our leak-free PRF's output only impacts the CCAmL2 security of our schemes. By contrast, it is already required for integrity guarantees in BMOS.

Bounded leakage. For the rest, the BMOS proofs require that each execution of their well protected PRF only leaks a bounded amount of bits per call (i.e., bounded leakages). As already mentioned, this requirement is conceptually similar to our requirement of simulatable leakages (i.e., it aims at avoiding that ephemeral secrets are leaked in full) and analyzing our following constructions under this complementary leakage assumption is an interesting open problem.[4]

D FEMALE Security Proofs

We first analyze the CCA security of FEMALE in sections D.1 and D.2, and then prove CIML2 and MR in sections D.3 and D.4 resp.

D.1 Preparations for CCAmL2 Proof

We are also interested in the mCCAmL2 security, namely our strongest confidentiality notion in the multi-challenge setting introduced in Appendix A.1. An approach is to first prove the CCAmL2 bound claimed in Theorem 4 and then use the generic result of Theorem 6 to derive the mCCAmL2 bound. However, we prefer an "inverse" direction: we directly prove a mCCAmL2 bound, and then obtain the CCAmL2 bound by setting $q_m = 1$. This approach could produce a slightly better bound without the factor q_m in some terms. Formally, sections D.1 and D.2 devote to prove the following theorem.

[4] In the case of BMOS, the systematic exploitation of shared computations (based on pairings) also prevents the issue of precomputation / future computation attacks that is typical of symmetric constructions (which is an additional motivation for using the simulatability assumption in our analyzes).

Theorem 8. *Let $H : \{0,1\}^* \to \mathcal{B}$ be a $(0, t', \varepsilon_{cr})$-collision resistant and $(q_d, t', \varepsilon_{pr})$-range-oriented preimage resistant hash function. And let $E : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2q_e + 2q_d + 2q_m, t', \varepsilon_E)$-SPRP with two implementations: a strongly protected implementation E^* is leak free, and a plain implementation E have leakage function L_E that is $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable. Then the* FEMALE *implementation with leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ defined before is $(q_e, q_d, p-1, q_m, q_l, t, \varepsilon_{\mathsf{mCCAmL2}})$* mCCAmL2*-secure, where*

$$\varepsilon_{\mathsf{mCCAmL2}} \leq 2\varepsilon_E + \varepsilon_{cr} + \varepsilon_{pr} + \textstyle\sum_{i=1}^{q_m} \varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i),$$

ℓ_i is the number of blocks in the i-th challenge message, and $\varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i)$ is the upper bound on the eavesdropper advantage of (q_l, t')-bounded adversaries against FEMALE *on messages with ℓ_i blocks. Concretely, we have:*

$$\varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i) \leq (6\ell_i + 8)(\varepsilon_E + \varepsilon_{(p,2)\text{-}rsim}) + \ell_i \cdot \varepsilon_{s\text{-}block} + \frac{6\ell_i + 4}{2^n}.$$

Here $t' = t + (q_e + q_d + q_m)(t_{\$} + t_{1-pass})$, t_{1-pass} is the maximum running time of FEMALE *upon a single (encryption or decryption) query, and $t_{\$}$ is the time needed for randomly sampling a value from $\mathcal{M}$.*

As mentioned, setting $q_m = 1$ yields the exact CCAmL2 bound which was interpreted as

$$O\left(\frac{t^2}{2^n}\right) + O\left(\frac{t(q_e + q_d)}{2^n}\right) + O\left(\frac{\sigma \cdot t + \sigma^2}{2^n}\right) + O\left(\sigma(\varepsilon_{(p,2)\text{-}rsim} + \varepsilon_{s\text{-}block})\right)$$

in Theorem 4. To further clarify, the first two terms correspond to ε_{cr} and ε_{pr}. The term $\frac{\sigma \cdot t + \sigma^2}{2^n}$ is different from the classical non-rekeying modes (which typically only contains the term $\frac{t}{2^n}$), and it stems from $O(\sigma \cdot \varepsilon_{\mathsf{E}})$ in $\sum_{i=1}^{q_m} \varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i)$: since $t' = t + O(\sigma)$, we have $\varepsilon_{\mathsf{E}} = \frac{t + O(\sigma)}{2^n}$, and thus $O(\sigma \cdot \varepsilon_{\mathsf{E}}) = O(\frac{\sigma \cdot t + \sigma^2}{2^n})$. As mentioned, the term $O\big(\sigma(\varepsilon_{(p,2)\text{-}rsim} + \varepsilon_{s\text{-}block})\big)$ corresponds to the security degradation due to side-channel & the reduction to a single block encryption.

It will be apparent in the proof that FEMALE actually satisfies an even stronger notion of mCCAmL2. Since the encryption starts with $R \leftarrow \mathsf{H}(0\|N\|A)$, only the pairs (N, A)'s must be fresh in the challenge phase to derive the security instead of each of these nonces N's.

The analysis proceeds in five steps, each corresponding to a subsubsection. As the first step, we prove a useful "indistinguishablity-like" lemma for our leaking setting. Then is a preparation: we define a model named *Leaking, Idealized, Single-block Encryption scheme* LISE *with encryption and decryption leakages*. This is actually the idealized version of the LRSE scheme defined in Fig. 6, and could be proved indistinguishable from LRSE (Lemma 2). It will be used in the 4th step (see below), constituting a bridge in the reduction.

Third, (informally speaking) we prove indistinguishability result for the two systems $(\mathsf{FEMALE}(M), \mathsf{L}_{\mathsf{FEMALE}(M)})$ and $(\$, \mathcal{S}_{\mathsf{FEMALE}}(M))$, for a single message M (Lemma 3). In other words, $(\mathsf{FEMALE}(M), \mathsf{L}_{\mathsf{FEMALE}(M)})$, the process of using FEMALE to encrypt a single message, is indistinguishable from an idealized process that produces random outputs \$ and simulated leakages $\mathcal{S}_{\mathsf{FEMALE}}(M)$. This shows the design of FEMALE is good in the sense that it achieves nice confusion and diffusion (so that it produces somewhat pseudorandom outputs).

Yet, the conclusion of step 3 says nothing about the message confidentiality—or eavesdropper security—of $\mathsf{FEMALE}(M)$, since the leakages $\mathsf{L}_{\mathsf{FEMALE}(M)}$ or its indistinguishable counterpart $\mathcal{S}_{\mathsf{FEMALE}}(M)$ may leak M completely. To remedy this, we focus on the idealized process

($\$, \mathcal{S}_{\mathsf{FEMALE}}(M)$), and show how to relate its eavesdropper security to the eavesdropper security of applying LISE—the single-block encryption scheme—to independently encrypt $|M|$ blocks. Since we've established the indistinguishability of LISE and LRSE, the eavesdropper security of $(\mathsf{FEMALE}(M), \mathsf{L}_{\mathsf{FEMALE}(M)})$ can be established via the following chain:

$$\begin{aligned}
&\text{eavesdropper security of } \mathsf{LRSE} \text{ (our assumption, see (1))}\\
\Rightarrow&\text{eavesdropper security of } \mathsf{LISE} \text{ (using indistinguishability of } \mathsf{LRSE} \text{ and } \mathsf{LISE})\\
\Rightarrow&\text{eavesdropper security of the idealized process } (\$, \mathcal{S}_{\mathsf{FEMALE}}(M))\\
\Rightarrow&\text{eavesdropper security of } (\mathsf{FEMALE}(M), \mathsf{L}_{\mathsf{FEMALE}(M)}).
\end{aligned}$$

Eventually, based on the eavesdropper security of $(\mathsf{FEMALE}(M), \mathsf{L}_{\mathsf{FEMALE}(M)})$, we establish the $\mathsf{mCCAmL2}$ security (this step is in subsection D.1). Roughly, the proof relies on the following features of FEMALE:

(i) Every invalid decryption query only leaks a pseudorandom value, i.e. $(\mathsf{E}^*_k)^{-1}(T)$ for some T. So the encryption can be seen as independent from these values;
(ii) For each challenge encryption query, since the nonce is used only once during the experiment, the process starts from a ephemeral key s_0 that is different from any other ephemeral key of the other encryption queries. By this, encryption of this challenge is quite independent from the other encryption queries, and we can view the entire experiment as an eavesdropper adversary against $(\mathsf{FEMALE}(M), \mathsf{L}_{\mathsf{FEMALE}(M)})$ with a lot of offline computations (i.e. all the other encryptions are turned into offline computations).

Indistinguishability of Real-Leaking and Ideal-Simulating Worlds Standaert et al. proved that based on the pseudorandom security of E and the simulatability of the leakage, (roughly) the "real-leaking world" $(\mathsf{E}_k(p), \mathsf{L}(k,p))$ is indistinguishable from the "ideal-simulating" world $(\$, \mathcal{S}^{\mathsf{L}}(k,p,\$))$ [SPY13]. A similar intermediate result could be obtained in our R-simulatability framework. For convenience of applying later in our analysis, we focus on the case $q = 2$. Moreover, we write $[\mathsf{leak}_1, \ldots, \mathsf{leak}_\ell]^p$ for the vector of ℓp leakages, which consist of ℓ (probably distinct) leakages, and each is obtained p times. We stress that trying to obtain the same leakage for p times would *not* result in completely identical traces: each time $\mathsf{L}(x)$ is queried for some input x, the trace would be mixed with random noise, and would probably deviate from the traces generated by previous queries to $\mathsf{L}(x)$.

Lemma 1. *Let $\mathsf{E} : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2, t, \varepsilon_{\mathsf{E}})$-PRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every $k_{pre}, p_0, p_1, z \in \mathcal{M}$ and every $(q_l - q^*, t - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$, the following holds:*

$$\begin{aligned}
&\big|\Pr[k_{ch} \xleftarrow{\$} \mathcal{M} : \mathcal{D}^{\mathsf{L}}(\mathsf{E}_{k_{ch}}(p_0), \mathsf{E}_{k_{ch}}(p_1), [\mathsf{L}_{\mathsf{E}}(k_{ch}, p_0), \mathsf{L}_{\mathsf{E}}(k_{ch}, p_1), \mathcal{S}^{\mathsf{L}}(k_{pre}, z, k_{ch})]^p) \Rightarrow 1]\\
&\quad - \Pr[k_{ch}, c_A, c_B \xleftarrow{\$} \mathcal{M}, c_A \neq c_B \text{ iff. } p_0 \neq p_1 :\\
&\qquad \mathcal{D}^{\mathsf{L}}(c_A, c_B, [\mathcal{S}^{\mathsf{L}}(k_{ch}, p_0, c_A), \mathcal{S}^{\mathsf{L}}(k_{ch}, p_0, c_B), \mathcal{S}^{\mathsf{L}}(k_{pre}, z, k_{ch})]^p) \Rightarrow 1]\big| \leq \varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}.
\end{aligned}$$

Here $q^* = 3p \cdot q_{\mathcal{S}}$, while $t^* = Max\{t_r, t_{sim}\}$, in which t_r is equal to $3p \cdot t_{\mathcal{S}}$ augmented with the time needed to make 2 oracle queries to the PRP challenger and select a uniformly random key in $\mathcal{M}$, and t_{sim} is the time needed to relay the content of $2p$ Enc and p Gen queries from and to a $(p,2)$-rsim challenger.

Proof. The proof consists of two simple transitions: we first replace the real leakages by with simulated ones, relying on the recyclable-simulatability assumption, then replace $\mathsf{E}_{k_{ch}}(p_0)$ and $\mathsf{E}_{k_{ch}}(p_1)$ by two distinct random values to obtain the target inputs, relying on the assumption that E is a PRP.

It's not hard to see the claim remains valid if E is a PRF rather than PRP. To save space we concentrate on the latter case (that will be used in our proof).

Single-Block One-Time Encryption Scheme This is actually an extension of the analogue introduced in [PSV15]. In detail, the use of E for computing k_{up} and y_{ch} is replaced by sampling random values. And we adapt the corresponding leakage traces using $\mathcal{S}^{\mathsf{L}}$. The resulted algorithm is defined in Fig. 10.

Description of LISE: (tool for the proof)
ISGen(1^n) picks $k_{ch} \xleftarrow{\$} \mathcal{M}$, $\mathcal{M}, \mathcal{C} = \mathcal{M}$ ($p_0, p_1 \in \mathcal{M}$ can be chosen)
ISEnc$_{k_{ch}}(m)$ returns (k_{up}, c), where $c = y_{ch} \oplus m$, and $k_{up}, y_{ch} \xleftarrow{\$} \mathcal{M}$, $k_{up} \neq y_{ch}$ as long as $p_1 \neq p_0$ (and $k_{up} = y_{ch}$ otherwise).
ISDec$_{k_{ch}}(c)$ proceeds in the natural way.

The leakage $\mathsf{L}_{\mathsf{ISE}} = (\mathsf{L}_{\mathsf{isenc}}, \mathsf{L}_{\mathsf{isdec}}, k_{pre})$ resulting from the LISE implementation is defined as $\mathsf{L}_{\mathsf{isenc}}(k_{ch}, m) = (\mathcal{S}^{\mathsf{L}}(k_{ch}, p_1, k_{up}), \mathcal{S}^{\mathsf{L}}(k_{ch}, p_0, y_{ch}), \mathsf{L}_{\oplus}(y_{ch}, m), \mathcal{S}^{\mathsf{L}}(k_{pre}, p_1, k_{ch}))$, $\mathsf{L}_{\mathsf{isdec}}(k_{ch}, c) = (\mathcal{S}^{\mathsf{L}}(k_{ch}, p_1, k_{up}), \mathcal{S}^{\mathsf{L}}(k_{ch}, p_0, y_{ch}), \mathsf{L}_{\oplus}(y_{ch}, c), \mathcal{S}^{\mathsf{L}}(k_{pre}, p_1, k_{ch}))$ for a fixed random $k_{pre} \xleftarrow{\$} \mathcal{M}$.

Fig. 10: The ideal single-block encryption scheme ISEnc.

We also define $\mathsf{LISEnc}^{+}_{k_{ch}}(m) = (\mathsf{LISEnc}_{k_{ch}}(m), [\mathsf{L}_{\mathsf{isdec}}(k_{ch}, c)]^{p-1}, k_{pre})$ for $(c, k_{up}) = \mathsf{ISEnc}_{k_{ch}}(m)$. Similarly to Pereira et al. [PSV15], our ISEnc scheme is indistinguishable from its real version RSEnc.

Lemma 2. *Let $E : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2, t, \varepsilon_E)$-PRP, whose implementation has a leakage function L_E having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p, 2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every $p_0, p_1 \in \mathcal{M}$, $p_0 \neq p_1$, and every $(q_l - q^*, t - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$, the following holds:*

$$|\Pr[\mathcal{D}^{\mathsf{L}_{\mathsf{RSE}}}(m, \mathsf{LRSEnc}^{+}_{k_{ch}}(m)) \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{L}_{\mathsf{ISE}}}(m, \mathsf{LISEnc}^{+}_{k_{ch}}(m)) \Rightarrow 1]| \leq \varepsilon_E + \varepsilon_{(p,2)\text{-}rsim}.$$

Here $q^* = 3p \cdot q_{\mathcal{S}} + p$, while $t^* = Max\{t_r, t_{sim}\}$, in which t_r is equal to $3p \cdot t_{\mathcal{S}} + 2t_{\oplus}$ augmented with the time needed to make 2 oracle queries to the PRP challenger and select a uniformly random key in $\mathcal{M}$, $t_{\oplus}$ is the time needed to evaluate the $\oplus$ action on an n-bit input, and t_{sim} is the time needed to relay the content of four Enc and two Gen queries from and to a $(p, 2)$-rsim challenger.

Proof. The proof just follows the same line as Lemma 1. Note that to generate the leakage $\mathsf{L}_{\oplus}(y_{ch}, c)$ $p - 1$ times, one does not need to evaluate $\oplus$ for that many times; instead, one just needs to make $p - 1$ queries L.

It's not hard to see Lemma 2 actually holds even if $p_0 = p_1$. However, as we remarked before, when the input p_1 equals p_0, neither RSEnc not ISEnc ensures eavesdropper security. To highlight this issue and avoid confusion, we put this restriction in Lemma 2.

On the other hand, for the scheme ISEnc we do not enforce the constraint $p_1 \neq p_0$, since the case of $p_1 = p_0$ would be used in some of our arguments below (of course, these arguments do not rely on the eavesdropper security of ISEnc in the case of $p_1 = p_0$).

LRFSM and LIFSM: FEMALE on a Single Message, and its Ideal Version We first formally describe two algorithms (L)RFSM and (L)IFSM, which denote the processes of using (Leaking) Real/Idealized FEMALE to encrypt a Single Message respectively (without generating the tag). The algorithm LRFSM is described in Fig. 11, while LIFSM is in Fig. 12.

Description of RFSM:

- Gen picks $s_0 \xleftarrow{\$} \mathcal{M}$
- $\mathsf{RFSM}_{k_0}(m_1, \ldots, m_\ell)$ proceeds in four steps:
  - (i) Initializes an empty list leak for the leakage;
  - (ii) Computes $s_1 \leftarrow \mathsf{E}_{s_0}(p_A)$ and $w \leftarrow \mathsf{E}_{s_0}(p_B)$, and adds $[\mathsf{L}_\mathsf{E}(s_0, p_A)]^p$ and $\mathsf{L}_\mathsf{E}(s_0, p_B)$ to the list leak. Here the two constants p_A and p_B are those used in FEMALE.
  - (iii) For $i = 1, \ldots, \ell$, computes $s_{i+1} \leftarrow \mathsf{E}_{s_i}(p_A)$, $y_i \leftarrow \mathsf{E}_{s_i}(d_{i-1})$, and $d_i \leftarrow y_i \oplus m_i$, and adds $\mathsf{L}_\mathsf{E}(s_i, p_A)$, $\mathsf{L}_\mathsf{E}(s_i, d_{i-1})$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathsf{L}_\mathsf{E}(s_i, p_A), \mathsf{L}_\mathsf{E}(s_i, d_{i-1}), \mathsf{L}_\oplus(y_i, d_i)]^{p-1}$ to the list leak;
  - (iv) Computes $U \leftarrow \mathsf{E}_{s_{\ell+1}}(p_A)$, $W \leftarrow \mathsf{E}_{s_{\ell+1}}(d_\ell)$, $V \leftarrow \mathsf{E}_w(W)$, and $k_1 \leftarrow \mathsf{E}_U(V)$; and adds $\mathsf{L}_\mathsf{E}(s_{\ell+1}, p_A)$, $\mathsf{L}_\mathsf{E}(s_{\ell+1}, d_\ell)$, $\mathsf{L}_\mathsf{E}(w, W)$, $\mathsf{L}_\mathsf{E}(U, V)$, and $[\mathsf{L}_\mathsf{E}(s_{\ell+1}, p_A), \mathsf{L}_\mathsf{E}(U, V)]^{p-1}$ to the list leak;
  - (v) for $i = 1, \ldots, \ell$, computes $k_{i+1} \leftarrow \mathsf{E}_{k_i}(p_A)$, $z_i \leftarrow \mathsf{E}_{k_i}(p_B)$, and $c_i \leftarrow z_i \oplus d_i$, and adds $\mathsf{L}_\mathsf{E}(k_i, p_A)$, $\mathsf{L}_\mathsf{E}(k_i, p_B)$, $\mathsf{L}_\oplus(z_i, d_i)$, and $[\mathsf{L}_\mathsf{E}(k_i, p_A), \mathsf{L}_\mathsf{E}(k_i, p_B), \mathsf{L}_\oplus(z_i, c_i)]^{p-1}$ to the list leak.

  $\mathsf{RFSM}k_0(m_1, \ldots, m_\ell)$ eventually returns (V, c), where $c = (c_1, \ldots, c_\ell)$.

We define $\mathsf{LRFSM}_{k_0}(m) = (\mathsf{RFSM}_{k_0}(m), \mathsf{leak})$, where leak is the list of traces standing at the end of the computation.

Fig. 11: The RFSM scheme and the involved leakages.

Description of IFSM:

- $\mathsf{IFSM}_{k_0}(m_1, \ldots, m_\ell)$ proceeds in four steps:
  - (i) Initializes an empty list leak for the leakage;
  - (ii) Samples $s_1 \xleftarrow{\$} \mathcal{M}$, and adds $[\mathcal{S}^\mathsf{L}(s_0, p_A, s_1)]^p$ to the list leak;
  - (iii) For $i = 1, \ldots, \ell$, samples $s_{i+1} \xleftarrow{\$} \mathcal{M}$ and $y_i \xleftarrow{\$} \mathcal{M}$ such that $s_{i+1} \neq y_i$ as long as $d_{i-1} \neq p_A$ ($s_{i+1} = y_i$ otherwise), sets $d_i \leftarrow y_i \oplus m_i$, and adds $\mathcal{S}^\mathsf{L}(s_i, p_A, s_{i+1})$, $\mathcal{S}^\mathsf{L}(s_i, d_{i-1}, y_i)$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathcal{S}^\mathsf{L}(s_i, p_A, s_{i+1}), \mathcal{S}^\mathsf{L}(s_i, d_{i-1}, y_i), \mathsf{L}_\oplus(y_i, d_i)]^{p-1}$ to the list leak;
  - (iv) Samples $U \xleftarrow{\$} \mathcal{M}$, $W \xleftarrow{\$} \mathcal{M}$, $U \neq W$ iff. $d_\ell \neq p_A$; $w \xleftarrow{\$} \mathcal{M}$, $w \neq s_1$; $V \xleftarrow{\$} \mathcal{M}$, and $k_1 \xleftarrow{\$} \mathcal{M}$; and adds $\mathcal{S}^\mathsf{L}(s_{\ell+1}, p_A, U)$, $\mathcal{S}^\mathsf{L}(s_{\ell+1}, d_\ell, W)$, $\mathcal{S}^\mathsf{L}(s_0, p_B, w)$, $\mathcal{S}^\mathsf{L}(w, W, V)$, $\mathcal{S}^\mathsf{L}(U, V, k_1)$, $[\mathcal{S}^\mathsf{L}(s_{\ell+1}, p_A, U), \mathcal{S}^\mathsf{L}(U, V, k_1)]^{p-1}$ to the list leak;
  - (v) For $i = 1, \ldots, \ell$, samples $k_{i+1} \xleftarrow{\$} \mathcal{M}$, $z_i \xleftarrow{\$} \mathcal{M}$, and $c_i \leftarrow z_i \oplus d_i$, and adds $\mathcal{S}^\mathsf{L}(k_i, p_A, k_{i+1})$, $\mathcal{S}^\mathsf{L}(k_i, p_B, z_i)$, $\mathsf{L}_\oplus(z_i, d_i)$, and $[\mathcal{S}^\mathsf{L}(k_i, p_A, k_{i+1}), \mathcal{S}^\mathsf{L}(k_i, p_B, z_i), \mathsf{L}_\oplus(z_i, c_i)]^{p-1}$ to the list leak.

  $\mathsf{IFSM}_{k_0}(m_1, \ldots, m_\ell)$ eventually returns (V, c), where $c = (c_1, \ldots, c_\ell)$.

We define $\mathsf{LIFSM}_{k_0}(m) = (\mathsf{IFSM}_{k_0}(m), \mathsf{leak})$ for the list leak standing at the end of the computation.

Fig. 12: The IFSM scheme and the involved leakages.

We then prove that LRFSM and LIFSM are indistinguishable, by relying on the $(p, 2)$-recyclable-simulatability assumption and the cryptographic strength of E.

Lemma 3. *Let $E : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2, t, \varepsilon_E)$-PRP, whose implementation has a leakage function L_E having $(q_\mathcal{S}, t_\mathcal{S}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p, 2)$-R-simulatable leakages, and let $\mathcal{S}^\mathsf{L}$ be an appropriate $(q_\mathcal{S}, t_\mathcal{S})$-bounded leakage simulator. Then, for every ℓ-block message m, every $p_A \neq p_B$, and every $(q_l - p \cdot q_r - q^*, t - p \cdot t_r - t^*)$-bounded distinguisher $\mathcal{D}^\mathsf{L}$ (that makes at most $q_l - p \cdot q_r - q^*$ queries to L and*

runs in time $t - p \cdot t_r - t^*$), the following holds:

$$\begin{aligned}&\left|\Pr[\mathcal{D}^{\mathsf{L}}(m, \mathsf{LRFSM}_{s_0}(m)) \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{L}}(m, \mathsf{LIFSM}_{s_0}(m)) \Rightarrow 1]\right| \\ \leq & 2(\ell+2)(\varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}) + \frac{2\ell+2}{2^n}.\end{aligned}$$

Here $q_r = (4\ell+5)(q_{\mathcal{S}}+1)+2\ell$, q^* and t^* are as defined in Lemma 1, and $t_r = (4\ell+5)(t_{\mathsf{E}} + t_{\$} + t_{\mathcal{S}}) + 2\ell \cdot t_{\oplus}$, where t_{E} is the time needed for evaluating E once, $t_{\$}$ is the time needed for randomly sampling a value from $\mathcal{M}$, and $t_{\oplus}$ is the time needed for evaluating $\oplus$ once.

Proof. We define $\mathsf{G}_{0,1}$ as the security game in which $\mathcal{D}^{\mathsf{L}}$ receives $\mathsf{LRFSM}_{s_0}(m)$ as the input, and G^*_ℓ as the game in which $\mathcal{D}^{\mathsf{L}}$ receives $\mathsf{LIFSM}_{s_0}(m)$ as the input.

We show that $\mathsf{G}_{0,1}$ could be transited to G^*_ℓ via a sequence of games

$$\mathsf{G}_0, \mathsf{G}_1, \ldots, \mathsf{G}_\ell, \mathsf{G}_{\ell+1}, \mathsf{G}_{\ell+2}, \mathsf{G}_{\ell+3} = \mathsf{G}^*_0, \mathsf{G}^*_1, \ldots, \mathsf{G}^*_{\ell-1}.$$

The first half sequence $\mathsf{G}_0, \ldots, \mathsf{G}_\ell, \mathsf{G}_{\ell+1}, \mathsf{G}_{\ell+2}$, and $\mathsf{G}_{\ell+3}$ "idealizes" the *Ephemeral key-IV generation* phase of the encryption. In detail, we first consider $\mathsf{G}_{0,1}$, and replace the two intermediate values $\mathsf{E}_{s_0}(p_A)$ and $\mathsf{E}_{s_0}(p_B)$ by two distinct random values s_1 and w. We also replace the leakages $[\mathsf{L}_{\mathsf{E}}(s_0, p_A)]^p$ and $\mathsf{L}_{\mathsf{E}}(s_0, p_B)$ with $[\mathcal{S}^{\mathsf{L}}(s_0, p_A, s_1)]^p$ and $\mathcal{S}^{\mathsf{L}}(s_0, p_B, w)$. This yields the game G_0.

We next derive an upper bound for $|\Pr[(\mathcal{D}^{\mathsf{L}})^{\mathsf{G}_0} \Rightarrow 1] - \Pr[(\mathcal{D}^{\mathsf{L}})^{\mathsf{G}_{0,1}} \Rightarrow 1]|$. For this, we assume a $(q_l - p \cdot q_r - q^*, t - p \cdot t_r - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$ against G_0 and $\mathsf{G}_{0,1}$, and we build a distinguisher $\mathcal{D}^{\mathsf{L}'}$ against the real-leaking-world and the ideal-simulation-world. Concretely, $\mathcal{D}^{\mathsf{L}'}$ uses $p_0 = p_A$, $p_1 = p_B$, and $z = p_A$ to get his challenge tuple, and we assume that $\mathcal{D}^{\mathsf{L}'}$ receives

$$(c_A, c_B, [\mathsf{leak}_1, \mathsf{leak}_2, \mathcal{S}^{\mathsf{L}}(\cdot, p_A, k_{ch})]^p)$$

as inputs, with $c_A \neq c_B$. $\mathcal{D}^{\mathsf{L}'}$ proceeds in two steps:

(1) $\mathcal{D}^{\mathsf{L}'}$ first uses $[\mathsf{leak}_1]^p$ and leak_2 as the leakages of the first iteration;
(2) $\mathcal{D}^{\mathsf{L}'}$ then sets $s_1 \leftarrow c_A$ and $w \leftarrow c_B$, and emulates all the remaining actions of LRFSM encryption. Eventually, it serves the obtained ciphertext $V\|c_1\|\ldots\|c_\ell$ as well as the leakage traces to $\mathcal{D}^{\mathsf{L}}$, and outputs whatever $\mathcal{D}^{\mathsf{L}}$ outputs.

It can be seen depending on whether the inputs to $\mathcal{D}^{\mathsf{L}'}$ follow real-leaking or ideal-simulating distribution, $\mathcal{D}^{\mathsf{L}}$ is eventually interacting with $\mathsf{G}_{0,1}$ or G_0. We show that to perform the additional operations, $\mathcal{D}^{\mathsf{L}'}$ makes at most $p \cdot s_r$ additional queries to L and spend $p \cdot t_r$ additional time. To this end, we note that the encryption process of LRFSM involves $4\ell+5$ calls to E and 2ℓ xor operations. Moreover,

- in the real world, each call to E costs 1 query to L and t_{E} running time;
- in the ideal world, each "idealized" call to E is translated into sampling a random value and making a call to $\mathcal{S}$, which costs $q_{\mathcal{S}}$ queries to L and $(t_{\$} + t_{\mathcal{S}})$ running time;
- each xor operation costs 1 query to L and $t_{\oplus}$ running time.

Therefore, to emulate the "hybrid" encryption process once, $\mathcal{D}^{\mathsf{L}'}$ needs at most $(4\ell+5)(q_{\mathcal{S}}+1)+2\ell = q_r$ queries to L and $(4\ell+5)(t_{\mathsf{E}}+t_{\$}+t_{\mathcal{S}})+2\ell \cdot t_{\oplus} = t_r$ running time. To obtain the required decryption leakage traces, $\mathcal{D}^{\mathsf{L}'}$ has to additionally perform the "hybrid" *decryption* process for $p-1$ times, which contributes to $(p-1)q_r$ more queries and $(p-1)t_r$ more time. Therefore, as claimed, $\mathcal{D}^{\mathsf{L}'}$

makes at most $p \cdot q_r$ additional queries to L and spends $p \cdot t_r$ additional time for the additional operations. By the above and Lemma 1 we have

$$|\Pr[(\mathcal{D}^{\mathsf{L}})^{\mathsf{G}_0} \Rightarrow 1] - \Pr[(\mathcal{D}^{\mathsf{L}})^{\mathsf{G}_{0,1}} \Rightarrow 1]| \leq \varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}.$$

Then, for $1 \leq i \leq \ell$, to obtain G_i, we modify the game G_{i-1} by replacing the two intermediate values $\mathsf{E}_{s_i}(p_A)$ and $\mathsf{E}_{s_i}(d_{i-1})$ with two values s_{i+1} and y_i, such that s_{i+1} is uniform, and y_i is uniform and $y_i \neq s_{i+1}$ when $d_{i-1} \neq p_A$, and $y_i = s_{i+1}$ otherwise; and further replacing the leakages $[\mathsf{L}_{\mathsf{E}}(s_i, p_A), \mathsf{L}_{\mathsf{E}}(s_i, d_{i-1})]^p$, $\mathsf{L}_{\oplus}(\mathsf{E}_{s_i}(d_{i-1}), m_i)$, and $[\mathsf{L}_{\oplus}(\mathsf{E}_{s_i}(d_{i-1}), d_i)]^{p-1}$ with $[\mathcal{S}^{\mathsf{L}}(s_i, p_A, s_{i+1}), \mathcal{S}^{\mathsf{L}}(s_i, d_{i-1}, y_i)]^p$, $\mathsf{L}_{\oplus}(y_i, m_i)$, and $[\mathsf{L}_{\oplus}(y_i, d_i)]^{p-1}$. To show the indistinguishability of G_i and G_{i-1}, we build $\mathcal{D}^{\mathsf{L}'}$ which proceeds in five steps (to ease description, we define $d_{-1} = p_B$ and $y_0 = w$):

(1) $\mathcal{D}^{\mathsf{L}'}$ first uniformly samples s_0;
(2) For $j = 0, \ldots, i-2$, $\mathcal{D}^{\mathsf{L}'}$ uniformly samples random values s_{j+1}, y_j such that $s_{j+1} \neq y_j$ iff. $d_{j-1} \neq p_A$, simulates the traces $[\mathcal{S}^{\mathsf{L}}(s_j, p_A, s_j), \mathcal{S}^{\mathsf{L}}(s_j, d_{j-1}, y_j)]^p$,[5] computes $d_j \leftarrow y_j \oplus m_j^0$ and $m_j^0 \leftarrow y_j \oplus d_j$ and obtains the traces $\mathsf{L}_{\oplus}(y_j, m_j^0)$ and $[\mathsf{L}_{\oplus}(y_j, d_j)]^{p-1}$;[6]
(3) Then, if $d_{i-2} \neq p_A$, it uniformly samples y_{i-1}, computes $d_{i-1} \leftarrow y_{i-1} \oplus m_{i-1}$, $m_{i-1} \leftarrow y_{i-1} \oplus d_{i-1}$;
(4) Uses $p_0 = p_A$, $p_1 = p_B$, and $z = p_A$ to get the challenge tuple

$$(c_A, c_B, [\mathsf{leak}_1, \mathsf{leak}_2, \mathcal{S}^{\mathsf{L}}(\cdot, p_A, k_{ch})]^p)$$

with $c_A \neq c_B$. Then, it sets $s_{i+1} \leftarrow c_A$ and $y_i \leftarrow c_B$, computes $d_i \leftarrow y_i \oplus m_i$, and simulates $[\mathcal{S}^{\mathsf{L}}(s_{i-1}, p_A, k_{ch}), \mathcal{S}^{\mathsf{L}}(s_{i-1}, d_{i-2}, y_{i-1})]^p$, $\mathsf{L}_{\oplus}(y_{i-1}, m_{i-1})$, $[\mathsf{L}_{\oplus}(y_{i-1}, d_{i-1})]^{p-1}$ as the traces of the i-th iteration,[7] while $[\mathsf{leak}_1, \mathsf{leak}_2]^p$, $\mathsf{L}_{\oplus}(y_i, m_i)$, $[\mathsf{L}_{\oplus}(y_i, d_i)]^{p-1}$ as the leakage of the $i+1$ th iteration in the first pass;
(5) Takes s_{i+1} and d_i as the starting points and emulates the remaining part of the execution of LRFSM encryption. Eventually, $\mathcal{D}^{\mathsf{L}'}$ serves the obtained ciphertext $V \| c_1 \| \ldots \| c_\ell$ as well as the leakage traces to $\mathcal{D}^{\mathsf{L}}$, and outputs whatever $\mathcal{D}^{\mathsf{L}}$ outputs.

It can be seen that depending on the input tuple received by $\mathcal{D}^{\mathsf{L}'}$ is real-leaking or ideal-simulation, $\mathcal{D}^{\mathsf{L}}$ is interacting with G_{i-1} or G_i, unless:

– $d_{i-2} \neq p_A$, yet $y_{i-1} = k_{ch}$, or
– $d_{i-2} = p_A$.

Therefore, denoting this event by Bad_i, we have

$$\Pr[(\mathcal{D}^{\mathsf{L}})^{\mathsf{G}_i} \Rightarrow 1] - \Pr[(\mathcal{D}^{\mathsf{L}})^{\mathsf{G}_{i-1}} \Rightarrow 1] \leq \Pr[\mathsf{Bad}_i] + \varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}.$$

It can be seen that:

– for $i = 3, \ldots, \ell$, $\Pr[d_{i-2} = p_A] = \frac{1}{2^n}$, and $\Pr[y_{i-1} = k_{ch}] = \frac{1}{2^n}$. Therefore, $\Pr[\mathsf{Bad}_i] = \frac{2}{2^n}$; and
– for $i = 1, 2$, $d_{i-2} = p_B \neq p_A$, while $\Pr[y_{i-1} = k_{ch}] = \frac{1}{2^n}$. So $\Pr[\mathsf{Bad}_i] = \frac{1}{2^n}$.

[5] When $j = 0$ the second part of the leakages is $\mathcal{S}^{\mathsf{L}}(s_0, p_B, w)$ without repeating.
[6] These xor actions are omitted when $j = 0$.
[7] When $i = 1$, these xor operations are omitted. More clearly, $\mathcal{D}^{\mathsf{L}'}$ uniformly samples w, and uses $[\mathcal{S}^{\mathsf{L}}(s_0, p_A, k_{ch})]^p$ and $\mathcal{S}^{\mathsf{L}}(s_0, p_B, w)$ as the traces of the 1st iteration.

The hop from G_ℓ to $\mathsf{G}_{\ell+1}$ is slightly different, as no message block would be xored with $\mathsf{E}_{s_{\ell+1}}(d_\ell)$. In detail, we replace the two intermediate values $\mathsf{E}_{s_{\ell+1}}(p_A)$ and $\mathsf{E}_{s_{\ell+1}}(d_\ell)$ by two random values U and W such that $W = U$ iff. $d_\ell = p_A$, and replace the leakage $[\mathsf{L}_\mathsf{E}(s_{\ell+1}, p_A)]^p$ and $\mathsf{L}_\mathsf{E}(s_{\ell+1}, d_\ell)$ by $[\mathcal{S}^\mathsf{L}(s_{\ell+1}, p_A, U)]^p$ and $\mathcal{S}^\mathsf{L}(s_{\ell+1}, d_\ell, W)$. By an analysis similar to the argument above, $\mathcal{D}^{\mathsf{L}'}$ could consistently simulate $\mathsf{G}_{i-1}/\mathsf{G}_i$ against $\mathcal{D}^\mathsf{L}$ unless $d_{\ell-1} = p_A$ or $y_\ell = k_{ch}$ (denoted $\mathsf{Bad}_{\ell+1}$). Therefore,

$$\begin{aligned}&|\Pr[(\mathcal{D}^\mathsf{L})^{\mathsf{G}_{\ell+1}} \Rightarrow 1] - \Pr[(\mathcal{D}^\mathsf{L})^{\mathsf{G}_\ell} \Rightarrow 1]| \le \Pr[\mathsf{Bad}_{\ell+1}] + \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}\\ \le&\frac{2}{2^n} + \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}.\end{aligned}$$

We then replace the intermediate value $\mathsf{E}_w(W)$ by a random V, and replace the leakage $\mathsf{L}_\mathsf{E}(w, W)$ by $\mathcal{S}^\mathsf{L}(w, W, V)$. This yields G_{l+2}. In a similar vein to the above, we have

$$\begin{aligned}&|\Pr[\mathcal{D}^{\mathsf{G}_{\ell+2}} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_{\ell+1}} \Rightarrow 1]| \le \Pr[d_{-1} = p_A \vee w = k_{ch}] + \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}\\ \le&\frac{1}{2^n} + \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}.\end{aligned}$$

We then replace the intermediate value $\mathsf{E}_U(V)$ by a random k_1, and replace the leakage $[\mathsf{L}_\mathsf{E}(U, V)]^p$ by $[\mathcal{S}^\mathsf{L}(U, V, k_1)]^p$. This yields $\mathsf{G}_{l+3} = \mathsf{G}_0^*$. We have

$$\begin{aligned}&|\Pr[\mathcal{D}^{\mathsf{G}_{\ell+3}} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_{\ell+2}} \Rightarrow 1]| \le \Pr[d_\ell = p_A] + \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}\\ \le&\frac{1}{2^n} + \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}.\end{aligned}$$

Therefore,

$$|\Pr[(\mathcal{D}^\mathsf{L})^{\mathsf{G}_{\ell+3}} \Rightarrow 1] - \Pr[(\mathcal{D}^\mathsf{L})^{\mathsf{G}_{0,1}} \Rightarrow 1]| \le \frac{2\ell+2}{2^n} + (\ell+4)(\varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}).$$

Then, for j from 1 to ℓ, we modify the game G_{j-1}^* to obtain G_j^*. The modifications are similar to those made for G_{j-1}: we replace the two intermediate values $\mathsf{E}_{k_j}(p_A)$ and $\mathsf{E}_{k_j}(p_B)$ by two distinct random values k_{j+1} and z_j (note that we always have $p_B \neq p_A$), and replace the leakage $[\mathsf{L}_\mathsf{E}(k_j, p_A), \mathsf{L}_\mathsf{E}(k_j, p_B)]^p$, $\mathsf{L}_\oplus(\mathsf{E}_{k_j}(p_B), d_j)$, and $[\mathsf{L}_\oplus(\mathsf{E}_{k_j}(p_B), c_j)]^{p-1}$ by $[\mathcal{S}^\mathsf{L}(k_j, p_A, k_{j+1}), \mathcal{S}^\mathsf{L}(k_j, p_B, z_j)]^p$, $\mathsf{L}_\oplus(z_j, d_j)$, and $[\mathsf{L}_\oplus(z_j, c_j)]^{p-1}$. We again have $|\Pr[\mathcal{D}^{\mathsf{G}_j^*} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_{j-1}^*} \Rightarrow 1]| \le \varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}$. By these, we eventually obtain the ideal game G_l^*.

Gathering all the above, we have

$$\begin{aligned}&|\Pr[\mathcal{D}^{\mathsf{G}_\ell^*} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_{0,1}} \Rightarrow 1]|\\ =&(|\Pr[\mathcal{D}^{\mathsf{G}_\ell^*} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_0^*} \Rightarrow 1]|) + (|\Pr[\mathcal{D}^{\mathsf{G}_{\ell+3}} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_{0,1}} \Rightarrow 1]|)\\ \le&\ell(\varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}) + \frac{2\ell+2}{2^n} + (\ell+4)(\varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim})\\ \le&2(\ell+2)(\varepsilon_\mathsf{E} + \varepsilon_{(p,2)\text{-}rsim}) + \frac{2\ell+2}{2^n}\end{aligned}$$

as claimed.

From 1-Block to ℓ-Block Security We now evaluate the (eavesdropper) security of an ℓ-block encryption with LIFSM by comparison with the security of ℓ encryptions with LISEnc performed with independent keys, block by block.

Lemma 4. *For every pair of ℓ-block messages m^0 and m^1 and (q_l, t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$, there exists a $(q_l + p \cdot q_r, t + p \cdot t_r)$-bounded adversary $\mathcal{A}^{\mathsf{L}'}$ such that*

$$\begin{aligned}&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m^1)) \Rightarrow 1]| \\ \leq &\frac{2\ell}{2^n} + \sum_{i=1}^{\ell} |\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{s_{i-1}}(m_i^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{s_{i-1}}(m_i^1)) \Rightarrow 1]|,\end{aligned}$$

where $s_0, \ldots, s_{\ell-1}$ are chosen uniformly at random, the p_1 value of $\mathsf{LISEnc}^+_{s_{i-1}}(m_i^b)$ is d_{i-1}, $d_0 = p_1$, $d_1, \ldots, d_{\ell-1}$ are chosen uniformly at random and $d_1, \ldots, d_{\ell-1} \neq p_0$, and m_i^0 and m_i^1 are the i-th block of m^0 and m^1 respectively. Here $q_r = (4\ell+5)q_{\mathcal{S}} + 2\ell$ and $t_r = (4\ell+5)(t_{\$} + t_{\mathcal{S}}) + 2\ell \cdot t_{\oplus}$, where t_{E}, $t_{\$}$, and $t_{\oplus}$ are as assumed in Lemma 3.

Proof. We start by building a sequence of $\ell + 1$ messages $m_{h,0}, \ldots, m_{h,\ell}$ starting from the "left" message m^0 and modifying its blocks one by one until obtaining the "right" message m^1. That is, $m_{h,i} := m_1^0 \| \ldots \| m_{\ell-i}^0 \| m_{\ell-i+1}^1 \| \ldots \| m_{\ell}^1$.

We proceed to argue there exists a $(q_l + p \cdot q_r, t + p \cdot t_r)$-bounded adversary $\mathcal{A}'$ such that

$$\begin{aligned}&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m_{h,i-1})) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m_{h,i})) \Rightarrow 1]| \\ \leq &\frac{2}{2^n} + |\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{s_{\ell-i}}(m^0_{\ell-i+1})) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{s_{\ell-i}}(m^1_{\ell-i+1})) \Rightarrow 1]|.\end{aligned}$$

This along with a simple summation would imply the main claim.

The arguments on $m_{h,i-1}$ and $m_{h,i}$ for all i are similar in general. To make it clearer, we take the first two messages, i.e.,

$$m_{h,0} = m_1^0 \| \ldots \| m_{\ell-1}^0 \| m_{\ell}^0, \text{ and } m_{h,1} = m_1^0 \| \ldots \| m_{\ell-1}^0 \| m_{\ell}^1,$$

as example. For this, assuming a (q_l, t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$ against $\mathsf{LIFSM}_{s_0}(m_{h,0})$ and $\mathsf{LIFSM}_{s_0}(m_{h,1})$, we build a $(q_l + p \cdot q_r, t + p \cdot t_r)$-bounded adversary $\mathcal{A}^{\mathsf{L}'}$ against LISEnc. In detail, $\mathcal{A}^{\mathsf{L}'}$ proceeds in six steps:

(1) $\mathcal{A}^{\mathsf{L}'}$ uniformly samples s_0, s_1, w, such that $s_1 \neq w$, and obtains the simulated leakage traces $[\mathcal{S}^{\mathsf{L}}(s_0, p_A, s_1)]^p$ and $\mathcal{S}^{\mathsf{L}}(s_0, p_B, w)$;

(2) for $j = 1, \ldots, \ell - 3$, $\mathcal{A}^{\mathsf{L}'}$ uniformly samples random values s_{j+1}, y_j such that $s_{j+1} \neq y_j$ iff. $d_{j-1} \neq p_A$, obtains traces $[\mathcal{S}^{\mathsf{L}}(s_j, p_A, s_j), \mathcal{S}^{\mathsf{L}}(s_j, d_{j-1}, y_j)]^p$, computes $d_j \leftarrow y_j \oplus m_j^0$ and obtains the traces $\mathsf{L}_{\oplus}(y_j, m_j^0)$ and $[\mathsf{L}_{\oplus}(y_j, d_j)]^{p-1}$;

(3) $\mathcal{A}^{\mathsf{L}'}$ samples $d_{\ell-1} \neq p_A$, and computes $y_{\ell-1} \leftarrow d_{\ell-1} \oplus m_{\ell-1}^0$. $\mathcal{A}^{\mathsf{L}'}$ then sets the p_0 value of its eavesdropper security challenger of LISEnc to p_A while p_1 to $d_{\ell-1}$, and submits m_{ℓ}^0, and m_{ℓ}^1 to the challenger to obtain the tuple

$$((k_{up}, c_{ch}^b), ([\mathsf{leak}_1, \mathsf{leak}_2]^p, \mathsf{L}_{\oplus}(y_{ch}, m_{ch}^b), [\mathsf{L}_{\oplus}(y_{ch}, c_{ch}^b)]^{p-1}, [\mathcal{S}^{\mathsf{L}}(k_{pre}, p_A, k_{ch})]^p, k_{pre})$$

as outputs (which is produced by either $\mathsf{LISEnc}^+_{k_{ch}}(m_{\ell}^0)$ or $\mathsf{LISEnc}^+_{k_{ch}}(m_{\ell}^1)$).

(4) if $d_{\ell-3} = p_A$ then $\mathcal{A}^{\mathsf{L}'}$ sets $y_{\ell-2} \leftarrow k_{pre}$, otherwise samples $y_{\ell-2}$ such that $y_{\ell-2} \neq k_{pre}$. It then computes $d_{\ell-2} \leftarrow y_{\ell-2} \oplus m_{\ell-2}^0$. At this stage, $\mathcal{A}^{\mathsf{L}'}$ aborts, if either of the following two conditions is fulfilled:

– $d_{\ell-2} = p_A$, yet $y_{\ell-1} \neq k_{ch}$;
– $d_{\ell-2} \neq p_A$, yet $y_{\ell-1} = k_{ch}$.

Otherwise, $\mathcal{A}^{\mathsf{L}'}$ uses the leakage traces $[\mathcal{S}^{\mathsf{L}}(s_{\ell-2}, p_A, k_{pre}), \mathcal{S}^{\mathsf{L}}(s_{\ell-2}, d_{\ell-3}, y_{\ell-2})]^p$, $\mathsf{L}_{\oplus}(y_{\ell-2}, m^0_{\ell-2})$, $[\mathsf{L}_{\oplus}(y_{\ell-2}, d_{\ell-2})]^p$ as the leakages in $\ell-1$ th iteration, and those $[\mathcal{S}^{\mathsf{L}}(k_{pre}, p_A, k_{ch}), \mathcal{S}^{\mathsf{L}}(k_{pre}, d_{\ell-2}, y_{\ell-1})]^p$, $\mathsf{L}_{\oplus}(y_{\ell-1}, m^0_{\ell-1})$, $[\mathsf{L}_{\oplus}(y_{\ell-1}, d$ as leakages in the ℓ th iteration in the first pass;

(5) sets $s_{\ell+1} \leftarrow k_{up}$ and $d_\ell \leftarrow c^b_{ch}$, and uses $[\mathsf{leak}_1, \mathsf{leak}_2]^p, \mathsf{L}_{\oplus}(y_{ch}, m^b_{ch})$, $[\mathsf{L}_{\oplus}(y_{ch}, c^b_{ch})]^{p-1}$ as the corresponding leakage traces (now the message absorbing phase of the first pass has been completed);

(6) Then, $\mathcal{A}^{\mathsf{L}'}$ takes $s_{\ell+1}$ and d_ℓ as the starting points, and emulates the remaining actions of LIFSM encrypting the message $m^0_1 \| \ldots \| m^0_{\ell-1} \| m^b_{ch}$ to obtain $c^0_1 \| \ldots \| c^0_{\ell-1} \| c^b_\ell$. Note that this requires $\mathcal{A}^{\mathsf{L}'}$ to uniformly sample z_ℓ and further compute $c^b_\ell \leftarrow z_\ell \oplus d_\ell = z_\ell \oplus c^b_{ch}$ and generates the traces $\mathsf{L}_{\oplus}(z_\ell, c^b_{ch})$ and $[\mathsf{L}_{\oplus}(z_\ell, c^b_\ell)]^{p-1}$ at the end of the second pass. Eventually, $\mathcal{A}^{\mathsf{L}'}$ serves the ciphertext $V \| c^0_1 \| \ldots \| c^0_{\ell-1} \| c^b_\ell$ as well as all the generated simulated leakages to $\mathcal{A}^{\mathsf{L}}$, and outputs whatever $\mathcal{A}^{\mathsf{L}}$ outputs.

It can be seen that as long as $\mathcal{A}^{\mathsf{L}'}$ does not abort (the probability of which is at most $2/2^n$ since both $d_{\ell-2}$ and $y_{\ell-1}$ are uniform), depending on whether the input tuple received by $\mathcal{A}^{\mathsf{L}'}$ captures LISEnc encrypting m^0_ℓ or m^1_ℓ, the inputs to $\mathcal{A}^{\mathsf{L}}$ capture LIFSM encrypting $m^0_1 \| \ldots \| m^0_{\ell-1} \| m^0_\ell$ or $m^0_1 \| \ldots \| m^0_{\ell-1} \| m^1_\ell$. Moreover, $\mathcal{A}^{\mathsf{L}'}$ is indeed $(q_l + p \cdot q_r, t + p \cdot t_r)$-bounded if $\mathcal{A}^{\mathsf{L}}$ is (q_l, t)-bounded. These complete the proof.

Gathering Lemmas 2, 3, and 4, we obtain Lemma 5 which shows the eavesdropper security bound of LRFSM (which, in fact, is also the eavesdropper bound of FEMALE).

Lemma 5. *Let $E : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2, t, \varepsilon_E)$-PRP, whose implementation has a leakage function L_E having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p, 2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every pair of ℓ-block messages m^0 and m^1 and $(q_l - p \cdot q_r - q^*, t - p \cdot t_r - t^*)$-bounded adversary $\mathcal{A}^{\mathsf{L}}$, it holds*

$$\begin{aligned}
&\left| \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRFSM}_{s_0}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRFSM}_{s_0}(m^1)) \Rightarrow 1] \right| \\
\leq &(6\ell + 8)(\varepsilon_E + \varepsilon_{(p,2)\text{-}rsim}) + \ell \cdot \varepsilon_{s\text{-}block} + \frac{6\ell + 4}{2^n},
\end{aligned}$$

where q_r, t_r are as defined in Lemma 3, and q^*, t^* are as defined in Lemma 2.

Proof.

$$\begin{aligned}
&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRFSM}_{s_0}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRFSM}_{s_0}(m^1)) \Rightarrow 1]| \\
\leq &\underbrace{|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m^1)) \Rightarrow 1]|}_{A} \\
&+ \underbrace{\sum_{b=0,1} |\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRFSM}_{s_0}(m^b)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIFSM}_{s_0}(m^b)) \Rightarrow 1]|}_{\leq 2\left((2\ell+4)(\varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}) + \frac{2\ell+2}{2^n}\right) \text{ (by Lemma 3)}}.
\end{aligned}$$

For the involved term A, by Lemma 4 there exists a $(q_l - q^*, t - t^*)$-bounded adversary $\mathcal{A}^{\mathsf{L}'}$ that satisfies

$$\begin{aligned} A &\le \frac{2\ell}{2^n} + \sum_{i=1}^{\ell} |\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{s_{i-1}}(m_i^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{s_{i-1}}(m_i^1)) \Rightarrow 1]| \\ &\le \underbrace{\sum_{i=1}^{\ell} |\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LRSEnc}^+_{s_{i-1}}(m_i^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LRSEnc}^+_{s_{i-1}}(m_i^1)) \Rightarrow 1]|}_{\le \ell \cdot \varepsilon_{s\text{-}block} \text{ (by (1))}} \\ &\quad + \frac{2\ell}{2^n} + 2\ell(\varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}) \text{ (by Lemma 2).} \end{aligned}$$

The claim thus follows.

Theorem 4, the CCAmL2 security of FEMALE, could then be derived from Lemma 5. The argument is in the next subsection.

D.2 Proof of Theorem 8

We define *trivial decryption queries*: if the adversary queries $\mathsf{Enc}_k(N, A, M) \to C$ and makes the decryption query $\mathsf{Dec}_k(N, A, C)$ later on, the latter is called *trivial*. Trivial decryption queries are usually deemed useless in black-box models. However, with leakage, such queries may give new information to the adversary. We will have explicit arguments for handling such queries.

Then we step into the proof. We start by defining G_0 as the game $\mathsf{PrivK}^{\mathsf{CCAmL2},0}_{\mathcal{A}^{\mathsf{L}},\mathsf{FEMALE}}$, and G^*_0 as the game $\mathsf{PrivK}^{\mathsf{CCAmL2},1}_{\mathcal{A}^{\mathsf{L}},\mathsf{FEMALE}}$. We show that they are indistinguishable from two games G_1 and G^*_1 respectively, the gap between which is the eavesdropper security bound of LRFSM plus an additional loss.

To this end, we first introduce the game G_1, which is obtained from G_0 by replacing all the occurrences of E^*_k and its inverse function by a random permutation π and its inverse. To upper bound $|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_0 \Rightarrow 1]|$, we build a $(2q_e + 2q_d + 2q_m, t_{\mathcal{D}})$-bounded distinguisher $\mathcal{D}_{SPRP}$ against the SPRP security of E^* (more precisely, E, since the crypto strength of E^* stems from E). The challenger $\mathcal{D}_{SPRP}$ picks all the necessary constants itself, and emulates the encryption and decryption oracles to interact with $\mathcal{A}^{\mathsf{L}}$ as follows. On each query made by $\mathcal{A}^{\mathsf{L}}$, $\mathcal{D}_{SPRP}$ emulates all the actions described by FEMALE, except for the computations involving E^*, which is replaced by calls to its own permutation oracle $\mathcal{O}$ (which is either E_k or π). When $\mathcal{A}^{\mathsf{L}}$ outputs its guess bit, $\mathcal{D}_{SPRP}$ returns that bit as its own guess. It's clear that depending on whether $\mathcal{D}_{SPRP}$ is interacting with E_k or π, $\mathcal{A}^{\mathsf{L}}$ is playing G_0 or G_1. Therefore, any difference between $\Pr[\mathsf{G}_1 \Rightarrow 1]$ and $\Pr[\mathsf{G}_0 \Rightarrow 1]$ leads to the same difference in $\mathcal{D}_{SPRP}$ distinguishing E_k from π. During the emulation, the simulated FEMALE scheme receives $q_e + q_m$ encryption queries and q_d decryption queries. Therefore, $\mathcal{D}_{SPRP}$ is a $(2q_e + 2q_d + 2q_m, t')$-bounded adversary against E, making at most $2q_e + 2q_d + 2q_m$ queries to $\mathcal{O}$ and running in time at most $t + (q_e + q_d + q_m)t_{1-pass} \le t'$. Thus $|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_0 \Rightarrow 1]| \le \varepsilon_{\mathsf{E}^*}$ follows from the assumption that E is a $(2q_e + 2q_d + 2q_m, t', \varepsilon_{\mathsf{E}})$-SPRP.

Similarly, we replace E^* by π to turn G^*_0 into G^*_1, which also introduces a gap of ε_{E}.

We then prove

$$\left|\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_1} \Rightarrow 1] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}^*_1} \Rightarrow 1]\right| \le \varepsilon_{cr} + \varepsilon_{pr} + \textstyle\sum_{i=1}^{q_m} \varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i),$$

where ℓ_i is the number of blocks in the ith challenge message, and the bound on $\varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i)$ is as defined in Theorem 5 but where ℓ_i corresponds to the block-length of the i-th challenge messages. This plus the above gap $2\varepsilon_{\mathsf{E}^*}$ yields the claimed bound. To this end, we denote the q_m challenge tuples by

$$(Nc_1, Ac_1, Mc_1^0, Mc_1^1), \ldots, (Nc_{q_m}, Ac_{q_m}, Mc_{q_m}^0, Mc_{q_m}^1).$$

Then, we note that in G_1, the q_m messages being encrypted by the challenge encryption oracle are $Mc_1^0, \ldots, Mc_{q_m}^0$, while those encrypted in G_1^* are $Mc_1^1, \ldots, Mc_{q_m}^1$. We use q_m hops to replace $Mc_1^0, \ldots, Mc_{q_m}^0$ by $Mc_1^1, \ldots, Mc_{q_m}^1$ in turn, to show that G_1 can be transited to G_1^*. For convenience, we define $\mathsf{G}_{2,0} = \mathsf{G}_1$, and define a sequence of games

$$\mathsf{G}_{2,1}, \mathsf{G}_{2,2}, \ldots, \mathsf{G}_{2,q_m},$$

such that in the i-th system $\mathsf{G}_{2,i}$, the first i messages processed by the challenge encryption oracle are $Mc_1^0, \ldots, Mc_i^0$, while the remaining $q_m - i$ messages being processed are $Mc_{i+1}^1, \ldots, Mc_{q_m}^1$. It can be seen actually $\mathsf{G}_{2,q_m} = \mathsf{G}_1^*$.

We then show that for $i = 1, \ldots, q_m$, $\mathsf{G}_{2,i-1}$ and $\mathsf{G}_{2,i}$ are indistinguishable for $\mathcal{A}^{\mathsf{L}}$. For this, from $\mathcal{A}^{\mathsf{L}}$ we build an adversary $\mathcal{A}^{\mathsf{L}'}$, such that if $\mathcal{A}^{\mathsf{L}}$ distinguishes $\mathsf{G}_{2,i-1}$ and $\mathsf{G}_{2,i}$ then $\mathcal{A}^{\mathsf{L}'}$ breaks the eavesdropper security of LRFSM. In detail, $\mathcal{A}^{\mathsf{L}'}$ keeps a pair of tables (Π, Π^{-1}) to simulate the random permutation π (via lazy sampling). And it runs $\mathcal{A}^{\mathsf{L}}$, and reacts as follows:

- Upon an encryption query (N_i, A_i, M_i) from $\mathcal{A}^{\mathsf{L}}$, $\mathcal{A}^{\mathsf{L}'}$ computes $R_i \leftarrow \mathsf{H}(0\|N_i\|A_i)$. Then:
  - if $0\|R_i \notin \Pi$, $\mathcal{A}^{\mathsf{L}'}$ samples $s_0^{(i)} \xleftarrow{\$} \mathcal{M}\backslash\Pi^{-1}$, sets $\Pi(0\|R_i) \leftarrow s_0^{(i)}$ and $\Pi^{-1}(s_0^{(i)}) \leftarrow 0\|R_i$, and then runs $\mathsf{LRFSM}_{s_0^{(i)}}(M_i)$ to get the ciphertext (V_i, c_i) and leakages. $\mathcal{A}^{\mathsf{L}'}$ then computes $h_i \leftarrow \mathsf{H}(1\|R_i\|V\|c_i)$ and $T_i \leftarrow \Pi(1\|h_i)$ (if $1\|h_i \notin \Pi$ then $\mathcal{A}^{\mathsf{L}'}$ defines $\Pi(1\|h_i)$ as a value newly sampled from $\mathcal{M}\backslash\Pi^{-1}$). Finally, $\mathcal{A}^{\mathsf{L}'}$ returns the outputs (V_i, c_i, T_i) and the leakages to $\mathcal{A}^{\mathsf{L}}$;
  - if $0\|R_i \in \Pi$, $\mathcal{A}^{\mathsf{L}'}$ simply runs $\mathsf{LRFSM}_{\Pi(0\|R_i)}(M_i)$, performs the tag generation action $h_i \leftarrow \mathsf{H}(1\|R_i\|V_i\|c_i)$ and $T_i \leftarrow \Pi(1\|h_i)$ on the obtained V_i and c_i, and returns (V_i, c_i, T_i) and the leakages to $\mathcal{A}^{\mathsf{L}}$.
- Upon a trivial decryption query (N_j, A_j, C_j) from $\mathcal{A}^{\mathsf{L}}$ (cf. the beginning of this subsection for the meaning of "trivial"), $\mathcal{A}^{\mathsf{L}'}$ computes $R_j \leftarrow \mathsf{H}(0\|N_j\|A_j)$. Since (N_j, A_j, C_j) is trivial, $0\|R_j \in \Pi$ necessarily holds. Therefore, $\mathcal{A}^{\mathsf{L}}$ parses $C_j = (V_j, c_j, T_j)$, runs $\mathsf{LRFSM.Dec}(\Pi(0\|R_j), c_j)$, and relays the outputs to $\mathcal{A}^{\mathsf{L}}$.
- Upon a non-trivial decryption query (N_j, A_j, C_j) from $\mathcal{A}^{\mathsf{L}}$, $\mathcal{A}^{\mathsf{L}'}$ parses $C_j = (V_j, c_j, T_j)$, and computes $R_j \leftarrow \mathsf{H}(0\|N_j\|A_j)$ and $h_j \leftarrow \mathsf{H}(1\|R_j\|V_j\|c_j)$. Then,
  - if $T_j \notin \Pi^{-1}$, $\mathcal{A}^{\mathsf{L}'}$ samples $Z_j \xleftarrow{\$} \mathcal{M}\backslash\Pi$, and sets $\Pi(Z_j) \leftarrow T_j$ and $\Pi^{-1}(T_j) \leftarrow Z_j$;
  - if $T_j \in \Pi^{-1}$, $\mathcal{A}^{\mathsf{L}'}$ simply sets $Z_j \leftarrow \Pi^{-1}(T_j)$.

  Now $\mathcal{A}^{\mathsf{L}'}$ defines $h_j^* = \mathsf{trunc}(Z_j)$, aborts if $h_j = h_j^*$ (this type of abortion is defined as BadDec), and returns $(\perp, h_j^*)$ to $\mathcal{A}^{\mathsf{L}}$.
- Upon $\mathcal{A}^{\mathsf{L}}$ submitting the j-th challenge tuple $(Nc_j, Ac_j, Mc_j^0, Mc_j^1)$, $\mathcal{A}^{\mathsf{L}'}$ computes $Rc_j \leftarrow \mathsf{H}(0\|Nc_j\|Ac_j)$. Then, if $0\|Rc_j \in \Pi$, $\mathcal{A}^{\mathsf{L}'}$ aborts (this type of abortion is defined as $\mathsf{BadChall}$); otherwise, its action depends on j:
  - When $j < i$, it simply encrypts Mc_j^0 and returns. In detail, $\mathcal{A}^{\mathsf{L}'}$ samples $sc_0^{(j)} \xleftarrow{\$} \mathcal{M}\backslash\Pi^{-1}$, sets $\Pi(0\|Rc_j) \leftarrow sc_0^{(j)}$ and $\Pi^{-1}(sc_0^{(j)}) \leftarrow 0\|Rc_j$, and then runs $\mathsf{LRFSM}_{\Pi(0\|Rc_j)}(Mc_j^0)$, performs the tag generation action $hc_j \leftarrow \mathsf{H}(1\|Rc_j\|Vc_j\|cc_j)$ and $Tc_j \leftarrow \Pi(1\|hc_j)$ on the obtained Vc_j and cc_j, and returns (Vc_j, cc_j, Tc_j) and the leakages to $\mathcal{A}^{\mathsf{L}}$.

- When $j = i$, it relays Mc_j^0 and Mc_j^1 to its eavesdropper security challenger to obtain (Vc_j^b, cc_j^b) and leakages leak_{enc} and $[\mathsf{leak}_{dec}]^{p-1}$, and then computes $Tc_j \leftarrow \Pi(1\|\mathsf{H}(1\|Rc_j\|Vc_j^b\|cc_j^b))$ (possibly defines a new entry in Π) and returns $C_{ch}^b = (Vc_j^b, cc_j^b, Tc_j)$ to $\mathcal{A}^{\mathsf{L}}$. Note that this means the relation $\Pi(0\|Rc_i) = s_0^{ch}$ is implicitly fixed, where s_0^{ch} is the secret key generated inside the eavesdropper security challenger;
- When $j > i$, it simply encrypts Mc_j^1 and returns. The details are similar to the described case $j < i$.

– Upon $\mathcal{A}^{\mathsf{L}}$ making the λ-th query to $\mathsf{L}_{\mathsf{decch}}(j)$ $(1 \le \lambda \le p-1)$,
- When $j \neq i$, $\mathcal{A}^{\mathsf{L}'}$ performs a corresponding decryption process (decrypting (Nc_j, Ac_j, Cc_j^0) when $j < i$, and (Nc_j, Ac_j, Cc_j^1) when $j > i$) and returns the obtained leakages to $\mathcal{A}^{\mathsf{L}}$;
- When $j = i$, $\mathcal{A}^{\mathsf{L}'}$ simply returns the λ-th trace in the vector $[\mathsf{leak}_{dec}]^{p-1}$ as the answer.

Moreover, whenever new entries are added to Π, $\mathcal{A}^{\mathsf{L}'}$ aborts if $0\|Rc_j \in \Pi$ (so that the implicit relation $\Pi(0\|Rc_i) = s_0^{ch}$ never causes inconsistency).

It can be seen that the whole process is the same as either $\mathsf{G}_{2,i-1}$ or $\mathsf{G}_{2,i}$ depending on whether $b = 0$ or 1, given that:

(i) BadDec never occurs, and
(ii) $\mathsf{BadChall}$ never occurs.

On the other hand, besides running $\mathcal{A}^{\mathsf{L}}$, $\mathcal{A}^{\mathsf{L}'}$ samples at most $2(q_e + q_d + q_m)$ random values (to emulate π) and internally processes $q_e + q_d + q_m$ queries. Therefore, the running time of $\mathcal{A}^{\mathsf{L}'}$ is at most $t_{\mathcal{A}^{\mathsf{L}'}} \le t + (q_e + q_d + q_m)(2t_{\$} + t_{1-pass})$ (while $\mathcal{A}^{\mathsf{L}'}$ makes q_l queries to L, which is the same as $\mathcal{A}^{\mathsf{L}}$).

We need to bound the probabilities of the defined events to complete the proof. They are as follows.

Lemma 6. $\Pr[\mathit{BadDec} \vee \mathit{BadChall}] \le \varepsilon_{cr} + \varepsilon_{pr}$.

Proof. To show this, we introduce two other events as follows:

– CollH: during the interaction between $\mathcal{A}^{\mathsf{L}'}$ and the eavesdropper security challenger, there appears two calls to $\mathsf{H}(x)$ and $\mathsf{H}(x')$ such that $x \neq x'$ while $\mathsf{H}(x) = \mathsf{H}(x')$;
– $\mathsf{PreimgH}$: during the interaction between $\mathcal{A}^{\mathsf{L}'}$ and the eavesdropper security challenger, there appears a decryption query $(N_j, A_j, (V_j, c_j, T_j))$ such that $T_j \notin \Pi^{-1}$ before this query is made, yet after $\Pi^{-1}(T_j)$ is defined, it holds $\mathsf{H}(x) = \mathsf{trunc}(\Pi^{-1}(T_j))$ for any hash-call $\mathsf{H}(x)$ occurring during the game.

If CollH happens, then from $\mathcal{A}^{\mathsf{L}}$ we are able to build a collision adversary $\mathcal{A}_{cr}$ against H: $\mathcal{A}_{cr}$ just simulates the eavesdropper security challenger and interacts with $\mathcal{A}^{\mathsf{L}'}$, and waits for CollH to occur. It's clear that the running time of $\mathcal{A}_{cr}$ is no more than $t_{\mathcal{A}^{\mathsf{L}'}}$ plus the time needed to emulate the eavesdropper challenger, which turns out $t + (q_e + q_d + q_m)(2t_{\$} + t_{1-pass}) = t'$ in total. Therefore, by the assumption on the collision resistance of H, we obtain $\Pr[\mathsf{CollH}] \le \varepsilon_{cr}$.

On the other hand, if $\mathsf{PreimgH}$ happens, then from $\mathcal{A}^{\mathsf{L}'}$ we are able to build a preimage adversary against H. To this end, note that by the definition of $\mathsf{PreimgH}$, if it happens with respect to a decryption query $(N_j, A_j, (V_j, c_j, T_j))$, then $T_j \notin \Pi^{-1}$ before this query is made. According to the description of $\mathcal{A}^{\mathsf{L}'}$, $\Pi^{-1}(T_j)$ will be defined to a randomly sampled value, and thus

$\mathsf{H}(1\|R_j\|V_j\|c_j) = \mathsf{trunc}(\Pi^{-1}(T_j))$ exactly fits into the definition of range-oriented preimage resistance. Therefore, we could use an adversary $\mathcal{A}_{pr}$ to emulate the interaction between $\mathcal{A}^{\mathsf{L}'}$ and the eavesdropper security challenger, and if $\mathsf{PreimgH}$ happens then $\mathcal{A}_{pr}$ turns out a preimage adversary against the hash function H. Since $\mathcal{A}_{pr}$'s running time is also at most t', and there are at most q_d such decryption queries, we have $\Pr[\mathsf{PreimgH}] \leq \varepsilon_{pr}$ by the assumption that H is $(q_d, t', \varepsilon_{pr})$-range-oriented preimage resistance. This further yields $\Pr[\mathsf{CollH} \vee \mathsf{PreimgH}] \leq \varepsilon_{cr} + \varepsilon_{pr}$.

We now prove $\Pr[\mathsf{BadDec} \vee \mathsf{BadChall} \mid \neg(\mathsf{CollH} \vee \mathsf{PreH})] = 0$ to complete the proof. Assume that BadDec occurs, and let $(N_i, A_i, (V_i, c_i, T_i))$ be the decryption query with the smallest index that passes the integrity checking. We distinguish several cases:

(i) Case 1: upon this decryption query, it holds $T_i \notin \Pi^{-1}$. Then $\Pi^{-1}(T_i)$ will be defined random and $\mathsf{H}(1\|R_i\|V_i\|c_i) = \mathsf{trunc}(\Pi^{-1}(T_i))$ implies the occurrence of $\mathsf{PreimgH}$.

(ii) Case 2: upon this decryption query it holds $T_i \in \Pi^{-1}$, and $\Pi^{-1}(T_i)$ was defined due to an earlier backward call to π^{-1}. Then it's the same as Case 1.

(iii) Case 3: contrary to Case 1 & 2, and $\Pi^{-1}(T_i) = 0\|Z_i$ for some $Z_i \in \mathcal{B}$. Then the entry $\Pi(0\|Z_i)$ was defined due to the "first" leak free call $\mathsf{E}^*_k(0\|\cdot)$. Then $\mathsf{H}(1\|R_i\|V_i\|c_i) = \mathsf{trunc}(\Pi^{-1}(T_i))$ means during the game there exists N', A' such that $\mathsf{H}(0\|N'\|A') = Z_i = \mathsf{H}(1\|R_i\|V_i\|c_i)$, indicating CollH.

(iv) Case 4: contrary to Case 1 & 2, and $\Pi^{-1}(T_i) = 1\|Z_i$ for some $Z_i \in \mathcal{B}$. Then the entry $\Pi(1\|Z_i)$ was defined in an earlier encryption query, due to the "second" leak free call $\mathsf{E}^*_k(1\|\cdot)$. Assume that this encryption query was (N_j, A_j, M_j), and the corresponding response was $C_j = (V_j, c_j, T_j)$. Clearly it has to be $T_j = T_i$. Since we are considering non-trivial decryption queries, this means $(N_j, A_j, V_j, c_j) \neq (N_i, A_i, V_i, c_i)$. Now, if $V_j\|c_j \neq V_i\|c_i$ then it holds $R_j\|V_j\|c_j \neq R_i\|V_i\|c_i$; otherwise, either $N_j \neq N_i$ or $A_j \neq A_i$ would imply $R_j \neq R_i$ by $\neg\mathsf{CollH}$, which by $\neg\mathsf{CollH}$ further implies $Z_i = \mathsf{H}(1\|R_j\|V_j\|c_j) \neq \mathsf{H}(1\|R_i\|V_i\|c_i)$.

Therefore, $\Pr[\mathsf{BadDec} \mid \neg(\mathsf{CollH} \vee \mathsf{PreH})] = 0$.

Conditioned on that BadDec does not occur, entries of the form $\Pi(0\|R)$ may be defined due to two events:

- First, the "first" leak free call $\mathsf{E}^*(0\|R)$ that occurs during an encryption query (not possible in decryption queries by $\neg\mathsf{BadDec}$);
- Second, the "second" backward leak free call $(\mathsf{E}^*)^{-1}$ during a decryption query.

If $\mathsf{BadChall}$ due to the first type of entries, then there necessarily exists an encryption query (N, A, M) such that $\mathsf{H}(0\|N\|A) = \mathsf{H}(0\|Nc_i\|Ac_i)$. According to the requirements of the $\mathsf{CCAmL2}$ security game, it has to be $(N, A) \neq (Nc_i, Ac_i)$; therefore, $\mathsf{H}(0\|N\|A) = \mathsf{H}(0\|Nc_i\|Ac_i)$ would contradict $\neg\mathsf{CollH}$. If $\mathsf{BadChall}$ due to the second type of entries, then it means $\mathsf{H}(0\|Nc_i\|Ac_i) = \mathsf{trunc}(\Pi^{-1}(T))$ for T specified in some decryption query. By the assumption on this type of entries, $\mathsf{trunc}(\Pi^{-1}(T))$ is defined to random, and thus this case contradicts $\neg\mathsf{PreimgH}$. Thus the claim.

By all the above, define

$$\mathsf{Bad} = \mathsf{BadDec} \vee \mathsf{BadChall},$$

then we have

$$\begin{aligned}
&\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,i}} \Rightarrow 1] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,i-1}} \Rightarrow 1] \\
\leq &\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,i}} \Rightarrow 1 \wedge \mathsf{Bad}] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,i-1}} \Rightarrow 1 \wedge \mathsf{Bad}] + \varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i).
\end{aligned}$$

This means

$$\begin{aligned}|\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_1^*} \Rightarrow 1] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_1} \Rightarrow 1]| &\leq \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,q_m}} \Rightarrow 1] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,0}} \Rightarrow 1] \\ &\leq \sum_{i=1}^{q_m} \Big(\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,i}} \Rightarrow 1] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_{2,i-1}} \Rightarrow 1] \Big) \\ &\leq \varepsilon_{cr} + \varepsilon_{pr} + \textstyle\sum_{i=1}^{q_m} \varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i).\end{aligned}$$

These complete the proof.

Influence of Empty Message. It's meaningless to use empty message for (m)CCAmL2 challenge message, as otherwise it's not possible to pick two distinct challenges. Therefore, in the (m)CCAmL2 game, they can only appear in decryptions and non-challenge encryptions. Our proof has covered both cases (without specific treatment): for such decryptions, it can be seen the arguments around the bad events (in particular, BadDec) remains valid; for such non-challenge encryptions, the adversary $\mathcal{A}^{\mathsf{L}'}$ remains able to simulate the corresponding actions.

D.3 Proof for CIML2

Since the main body only contains an informal theorem we give the formal one as follows.

Theorem 9. *Let $H : \{0,1\}^* \to \mathcal{B}$ be a $(0, t', \varepsilon_{cr})$-collision resistant and $(q_d + 1, t', \varepsilon_{pr})$-range-oriented preimage resistant hash function. And let $E : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2q_e + 2q_d + 2, t', \varepsilon_{E^*})$-SPRP with E^* being its strongly protected leak free implementation. Then* FEMALE *provides $(q_e, q_d, t, \varepsilon_{\mathsf{CIML2}})$-ciphertext integrity with coin misuse and unbounded leakage on encryption and decryption as long as $t \leq t' - (q_e + q_d + 1)t_{1-pass}$, where t_{1-pass} is the maximum running time of* FEMALE *upon a single (encryption or decryption) query, and*

$$\varepsilon_{\mathsf{CIML2}} \leq \varepsilon_{E^*} + \varepsilon_{cr} + \varepsilon_{pr}.$$

The claim can be established by some intermediate results obtained during the proof of Theorem 4. To see this, let's revisit these results:

- First, define G_0 as the real CIML2 security game between the adversary and FEMALE. Then replacing E_k^* by a random permutation π, we obtain a game G_1, with a gap of ε_{E};
- Second, as long as neither of the two bad events CollH and PreimgH happens during the execution of G_1, every non-trivial decryption query would yield $\perp$ as the answer. And we have $\Pr[\mathsf{CollH} \vee \mathsf{PreimgH}] \leq \varepsilon_{cr} + \varepsilon_{pr}$.

Note that the above steps do not rely on additional leakage assumptions. Therefore, the claim holds in the unbounded leakage model. And as remarked in the previous subsection, our proof has covered empty messages.

D.4 Proof for MR

Also we give the formal theorem as follows.

Theorem 10. *Let* $H : \{0,1\}^* \to \mathcal{B}$ *be a* $(0, t', \varepsilon_{cr})$*-collision resistant and* $(q_d + 1, t', \varepsilon_{pr})$*-range-oriented preimage resistant hash function. Let* $E : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ *be a* $(2q_e + 2q_d, t', \varepsilon_{E^*})$*-SPRP. Then the* FEMALE *scheme is* $(q_e, q_d, t, \varepsilon_{\mathsf{MR}})$*-MR as long as* $t \leq t' - (q_e + q_d)(t_{1-pass} + (4\ell + 4)t_{\$})$*, where* t_{1-pass} *is the maximum running time of* FEMALE *upon a single (encryption or decryption) query,* $t_{\$}$ *is the time needed for randomly sampling a value from* $\mathcal{M}$*, and:*

$$\varepsilon_{\mathsf{MR}} \leq \varepsilon_E + \varepsilon_{cr} + \varepsilon_{pr} + (2\ell + 4)q_e\varepsilon_E + \frac{2(\ell+1)q_e + (q_e + q_d)^2 + q_e^2}{2^{n+1}}.$$

Let $\mathcal{A}$ be a $(q_e, q_d, t, \varepsilon_{\mathsf{MR}})$-bounded adversary against the misuse resistance of FEMALE, and assume $\mathcal{A}$ never makes "redundant" queries, i.e.

- $\mathcal{A}$ never repeats queries, and
- $\mathcal{A}$ never decrypts the ciphertext produced by previous encryption queries.

Furthermore, we write the encryption queries as

$$(N_1, A_1, M_1), (N_2, A_2, M_2), \ldots, (N_{q_e}, A_{q_e}, M_{q_e});$$

and we assume the length of M_i is $\ell_i \leq \ell$. We stress that $N_1 \| A_1, \ldots, N_{q_e} \| A_{q_e}$ are *not* necessarily distinct since we are in misuse setting.

We will use a sequence of hybrid games, beginning with the real game, named G_0, where $\mathcal{A}$ interacts with Enc_k and Dec_k, and ending with random-and-invalid game, named G_{q_e+1}, where $\mathcal{A}$ interacts with \$ and $\perp$. Then, using the adversary $\mathcal{A}$ we show that any transition can be reduced to an efficient adversary against one of the assumptions. For convenience, we name E_i the event that $\mathcal{A}$ outputs 1 at the end of G_i.

We start from $\mathsf{G}_0 = \mathcal{A}^{\mathsf{Enc}_k, \mathsf{Dec}_k}$. The first two steps are similar to those appeared in the proof of Theorem 4:

(i) we first replace all the occurrences of E_k^* by a random permutation π to obtain the first intermediate game $\mathsf{G}_{0,1}$. And we introduce two abort conditions CollH and PreimgH into $\mathsf{G}_{0,1}$: the former is fulfilled if there exists two calls to $\mathsf{H}(x)$ and $\mathsf{H}(x')$ such that $x \neq x'$ yet $\mathsf{H}(x) = \mathsf{H}(x')$, while the latter is fulfilled if there exists a hash-call $\mathsf{H}(x)$ and an fresh inverse call $\pi^{-1}(T)$ such that $\mathsf{H}(x) = \mathsf{trunc}(\pi^{-1}(T))$. It can be seen $\Pr[\mathsf{G}_{0,1} \text{ aborts}] \leq \varepsilon_{cr} + \varepsilon_{pr}$, otherwise a corresponding adversary running in time at most $t' \leq t + (q_e + q_d)t_{1-pass}$ can be built against H;

(ii) It holds $|\Pr[E_0] - \Pr[E_{0,1} \mid \mathsf{G}_{0,1} \text{ does not abort}]| \leq \varepsilon_{\mathsf{E}}$, otherwise from $\mathcal{A}$ we can build an SPRP adversary $\mathcal{D}_{SPRP}$ against E_k, and $\mathcal{D}_{SPRP}$ makes at most $2(q_e + q_d)$ queries and runs in time at most t'.

We next define $\ell_0 = 1$, and consider the second intermediate game G^*_{0,ℓ_0}, which is the same as $\mathsf{G}_{0,1}$ except that all decryption queries are simply answered by $\perp$. In a similar vein to the proof of Theorem 4, we know that if neither of the two games abort, then they would have no difference. Therefore, $\Pr[E_{0,1} \mid \mathsf{G}_{0,1} \text{ does not abort}] = \Pr[E^*_{0,\ell_0} \mid \mathsf{G}^*_{0,\ell_0} \text{ does not abort}]$.

We then consider the encryption queries in turn, and use a sequence of games

$$\mathsf{G}_{1,-1} = \mathsf{G}^*_{0,\ell_0}, \mathsf{G}_{1,1}, \ldots, \mathsf{G}_{1,\ell_1}, \mathsf{G}_{1,\ell_1+1}, \mathsf{G}_{1,\ell_1+2} = \mathsf{G}^*_{1,0}, \mathsf{G}^*_{1,1}, \ldots, \mathsf{G}^*_{1,\ell_1}, \mathsf{G}_{2,-1},$$
$$\ldots$$
$$\mathsf{G}_{q_e,-1} = \mathsf{G}^*_{q_e-1,\ell_{q_e-1}}, \ldots, \mathsf{G}_{q_e,\ell_{q_e}}, \mathsf{G}_{q_e,\ell_{q_e}+1}, \mathsf{G}_{q_e,\ell_{q_e}+2} = \mathsf{G}^*_{q_e,0}, \mathsf{G}^*_{q_e,1}, \ldots, \mathsf{G}^*_{q_e,\ell_{q_e}},$$

to show that the ciphertexts are indistinguishable from uniform. In detail, for $i = 1, \ldots, q_e$, we start from $\mathsf{G}^*_{i-1,\ell_{i-1}} = \mathsf{G}_{i,-1}$, and transit to G^*_{i,ℓ_i}. In this game, the tuple (N_i, A_i) would give rise to $R_i = \mathsf{H}(0\|N_i\|A_i)$ and $s_0^{(i)} = \pi(0\|R_i)$. We distinguish two cases:

Non-reuse Case: (N_i, A_i) *never appeared before.* Then conditioned on $\neg\mathsf{Coll}\mathsf{H}$, R_i never appeared before, and thus $s_0^{(i)}$ must be distinct from all the already appeared intermediate values $s_0^{(1)}, \ldots, s_0^{(i-1)}$ since π is a permutation. In the first phase of the transition, for $j = 0, \ldots, \ell_i + 1$, we consider the game $\mathsf{G}_{i,j-1}$, and replace all the subsequently appearing calls of the form $\mathsf{E}_{s_j^{(i)}}(x)$ by calls to a new random permutation $P_{i,j}(x)$. This yields $\mathsf{G}_{i,j}$. It holds $|\Pr[E_{i,j}] - \Pr[E_{i,j-1}]| \leq \varepsilon_{\mathsf{E}}$, otherwise a PRP adversary $\mathcal{B}_{\mathsf{E}}$ against E could be built by emulating the common part of $\mathsf{G}_{i,0}$ and $\mathsf{G}_{i,-1}$. In detail, $\mathcal{B}_{\mathsf{E}}$ interacts with $\mathcal{A}$, and simulates the previously appeared random permutations $P_{1,0}, P_{1,1}, \ldots, P_{i,0}, \ldots, P_{i,j-1}$ by lazy sampling, till the intermediate values $s_j^{(i)}$ and $d_{j-1}^{(i)}$ being computed.[8] Then, $\mathcal{B}_{\mathsf{E}}$ supplies p_A and $d_{j-1}^{(i)}$ to its challenge oracle $\mathcal{O}$ (which is either E_k for a secret key k or the random permutation $P_{i,j}$), and uses the obtained $s_{j+1}^{(i)} \leftarrow \mathcal{O}(p_A)$ to continue emulating the process of real FEMALE encryption. Moreover, to ensure consistency, for queries received in future with (N_i, A_i) as the involved nonce, $\mathcal{B}_{\mathsf{E}}$ uses $P_{i,0}, \ldots, P_{i,j-1}, \mathcal{O}$ for the first $j+1$ calls to E. To emulate these, $\mathcal{B}_{\mathsf{E}}$ makes at most $2q_e$ queries to $\mathcal{O}$, and the running time of $\mathcal{B}_{\mathsf{E}}$ is clearly at most t'. Thus $\mathcal{B}_{\mathsf{E}}$'s advantage does not exceed ε_{E}.

Then, consider the game G_{i,ℓ_i+1}, and let $w^{(i)} = P_{i,0}(p_B)$. We replace all the subsequently appearing calls of the form $\mathsf{E}_{w^{(i)}}(x)$ by calls to a new random permutation $P_{i,\ell_i+2}(x)$, to obtain the game G_{i,ℓ_i+2}. Similarly to the above, $|\Pr[E_{i,\ell+2}] - \Pr[E_{i,\ell+1}]| \leq \varepsilon_{\mathsf{E}}$. We further replace all the subsequently appearing calls of the form $\mathsf{E}_{U^{(i)}}(x)$ by calls to a new random permutation $P_{i,\ell_i+3}(x)$, to obtain the game $\mathsf{G}_{i,\ell_i+3} = \mathsf{G}^*_{i,0}$, with $|\Pr[E_{i,\ell+3}] - \Pr[E_{i,\ell+2}]| \leq \varepsilon_{\mathsf{E}}$. By this, the newly derived initial key $k_1^{(i)} = P_{i,\ell_i+3}(V^{(i)})$ for the one-time encryption is uniform.

Then in the second phase, for $j = 1, \ldots, \ell_i$, we replace all the subsequently appearing calls to $\mathsf{E}_{k_j^{(i)}}$ by corresponding calls to a new random permutation $P'_{i,j}$, and this yields $\mathsf{G}^*_{i,j}$. The gap between each pair of games $(\mathsf{G}^*_{i,j-1}, \mathsf{G}^*_{i,j})$ is at most ε_{E}. We also have $|\Pr[E^*_{i,j}] - \Pr[E^*_{i,j-1}]| \leq \varepsilon_{\mathsf{E}}$.

It can be seen that in the last game G^*_{i,ℓ_i}, the intermediate value $V^{(i)}$ as well as the ciphertext blocks $c_1^{(i)}, \ldots, c_\ell^{(i)}$ resulted from the encryption query (N_i, A_i, M_i) are random. The next paragraph would discuss the nonce-reuse case.

Reuse Case: (N_i, A_i) *has appeared before.* Assume that α is the smallest index such that $(N_\alpha, A_\alpha) = (N_i, A_i)$. We further distinguish two cases:

- Case 1: the length ℓ_i does not exceed the maximum length of the messages already processed with the pair (N_i, A_i). Then it can be seen that in the game $\mathsf{G}_{i,-1}$, when processing the encryption query (N_i, A_i, M_i), all the calls to E during the first pass have been replaced by calls to random permutations $P_{\alpha,0}, \ldots, P_{\alpha,\ell_i+3}$. Therefore, we simply let $\mathsf{G}^*_{i,0} = \mathsf{G}_{i,-1}$.
- Case 2: M_i is the longest message among the messages encrypted with the pair (N_i, A_i). Assume that the calls $\mathsf{E}_{s_0^{(\alpha)}}, \ldots, \mathsf{E}_{s_\beta^{(\alpha)}}$ have been replaced by random permutations $P_{\alpha,0}, \ldots, P_{\alpha,\beta}$ (note

[8] Similarly to the proof of Lemma 3, when $j = 0$ we take $d_{-1}^{(i)} = p_B$ and $w^{(i)}$ as the corresponding output to ease the language.

that $s_{\beta}^{(\alpha)} = U^{(j)}$ for some $j < i$), and let $s_{\beta+1}^{(i)} = P_{\alpha,\beta}(p_A)$. Define $\mathsf{G}_{i,\beta} = \mathsf{G}_{i,\beta}$. Then, for $j = \beta+1, \ldots, \ell_i+2$, we replace all the subsequently appearing calls of the form $\mathsf{E}_{s_j^{(i)}}(x)$ (taking $s_{\ell_i+2}^{(i)} = U^{(i)}$) by calls to a new random permutation $P_{i,j}(x)$. This yields $\mathsf{G}_{i,j+1}$. For convenience we also write $P_{\alpha,j}$ for the permutation $P_{i,j}$. These steps eventually yield the game $\mathsf{G}_{i,\ell_i+3} = \mathsf{G}_{i,0}^*$.

Unlike the non-reuse case, we introduce three abort conditions into the game $\mathsf{G}_{i,0}^*$. In detail, $\mathsf{G}_{i,0}^*$ aborts, if at least one of the following occurs during its execution:

- the event CollD_i, which happens if there exists $j < i$ such that $N_j \| A_j = N_i \| A_i$ and $\ell_i = \ell_j$ for the j-th encryption query (N_j, A_j, M_j), and there exists an index γ such that $m_1^{(i)} \| \ldots \| m_\gamma^{(i)} \neq m_1^{(j)} \| \ldots \| m_\gamma^{(j)}$, yet $d_\gamma^{(i)} = d_\gamma^{(j)}$;
- the event CollW_i, which happens if there exists $j < i$ such that $N_j \| A_j = N_i \| A_i$ for (N_j, A_j, M_j), and $W^{(i)} = W^{(j)}$.

To analyze the events, we consider every index $j < i$. If $N_j \| A_j \neq N_i \| A_i$, then neither CollD_i nor CollW_i could happen with respect to j. If $N_j \| A_j = N_i \| A_i$ and $\ell_i \neq \ell_j$, then CollD_i could happen. In this case, assume that α is the smallest index such that $(N_\alpha, A_\alpha) = (N_i, A_i)$. Then by our convention, we have

$$W^{(i)} = P_{\alpha,\ell_i+1}(d_{\ell_i}^{(i)}), \text{ and } W^{(j)} = P_{\alpha,\ell_j+1}(d_{\ell_j}^{(j)}),$$

with P_{α,ℓ_i+1} and P_{α,ℓ_j+1} being two independent random permutations. Moreover, if abortion does not happen, then conditioned on the transcripts of queries and answers obtained so far, the values $P_{\alpha,\ell_i+1}(d_{\ell_i}^{(i)})$ and $P_{\alpha,\ell_j+1}(d_{\ell_j}^{(j)})$ remain uniform, since

- the ciphertexts in the transcript are clearly independent of these two values, and
- the values $V^{(1)}, \ldots, V^{(i-1)}$ in the transcripts are outputs of the permutations $P_{1,\ell_1+2}, \ldots, P_{i-1,\ell_{i-1}+2}$, thus also independent of these two values.

Therefore, it holds

$$\Pr[W^{(i)} = W^{(j)}] \leq \frac{1}{2^n}.$$

When $N_j \| A_j = N_i \| A_i$ and $\ell_i = \ell_j$, then both CollD_i and CollW_i could happen. In this case, it necessarily be $M_i \neq M_j$ since $\mathcal{A}$ does not repeat queries. Wlog assume that γ is the smallest index such that $m_\gamma^{(j)} \neq m_\gamma^{(i)}$. Then it can be seen $d_{\gamma-1}^{(j)} = d_{\gamma-1}^{(i)}$. Moreover, $N_j \| A_j = N_i \| A_i$ implies $s_0^{(i)} = s_0^{(j)}, \ldots, s_\gamma(i) = s_\gamma^{(j)}$, and thus the first γ random permutations used for processing the two queries are the same ones. Assume that α is the smallest index such that $(N_\alpha, A_\alpha) = (N_i, A_i)$. Then by our convention, these γ random permutations are $P_{\alpha,0}, \ldots, P_{\alpha,\gamma}$. Then it can be seen $d_\gamma^{(j)} \neq d_\gamma^{(i)}$, since $m_\gamma^{(j)} \neq m_\gamma^{(i)}$ implies

$$P_{\alpha,\gamma}(d_{\gamma-1}^{(j)}) \oplus m_\gamma^{(j)} \neq P_{\alpha,\gamma}(d_{\gamma-1}^{(i)}) \oplus m_\gamma^{(i)}.$$

As discussed, conditioned on the transcripts of queries and answers obtained so far, the values $P_{\alpha,\gamma+1}(d_\gamma^{(j)})$ and $P_{\alpha,\gamma+1}(d_\gamma^{(i)})$ remain uniform. By these,

$$\Pr[d_{\gamma+1}^{(j)} = d_{\gamma+1}^{(i)}] = \Pr[P_{\alpha,\gamma+1}(d_\gamma^{(j)}) \oplus m_{\gamma+1}^{(j)} = P_{\alpha,\gamma+1}(d_\gamma^{(i)}) \oplus m_{\gamma+1}^{(i)}] \leq \frac{1}{2^n}.$$

Following the same line, we obtain $\Pr[d_{\gamma+2}^{(j)} = d_{\gamma+2}^{(i)}] \leq \frac{1}{2^n}$, …, $\Pr[d_{\ell_j}^{(j)} = d_{\ell_i}^{(i)}] \leq \frac{1}{2^n}$, and $\Pr[W^{(j)} = W^{(i)}] \leq \frac{1}{2^n}$. Therefore, for such an index j, the probability that either CollD_i or CollW_i happens is

at most $\frac{\ell_i+1}{2^n} \le \frac{\ell+1}{2^n}$.

By the above, in any case, the following holds:

$$\Pr[\mathsf{CollD}_i \vee \mathsf{CollW}_i \mid \neg\mathsf{CollH}] \le \frac{\ell+1}{2^n}.$$

If $\mathsf{G}^*_{i,0}$ does not abort, then the produced intermediate value $W^{(i)}$ is distinct from all the previous value $W^{(j)}$ for $N_j\|A_j = N_i\|A_i$. This means the further generated $V^{(i)}$ are also distinct from these $V^{(j)}$, and further $k_1^{(i)} \neq k_1^{(j)}$. On the other hand, for the previous queries with $N_j\|A_j \neq N_i\|A_i$, conditioned on $\neg\mathsf{CollH}$ we have $R_j \neq R_i$, and thus the subsequent processes utilize completed independent random permutations. This means for such index j the produced one-time encryption initial key $k_1^{(j)}$ is independent from $k_1^{(i)}$. In all, the produced initial key $k_1^{(i)}$ is a new random value independent from all the appeared keys $k_1^{(1)}, \ldots, k_1^{(i-1)}$. Therefore, we start from $\mathsf{G}^*_{i,0}$ and make ℓ_i hops that are similar to those described in the non-reuse case. These hops yield G^*_{i,ℓ_i} at the end, with $|\Pr[E^*_{i,\ell_i}] - \Pr[E^*_{i,0}]| \le \ell\varepsilon_{\mathsf{E}^*} + \frac{\ell+1}{2^n}$.

Finally, in the game $\mathsf{G}^*_{q_e,\ell_{q_e}}$, it can be seen that every intermediate value is derived via a call to a corresponding random permutation. And it can be seen

$$\Pr[\mathsf{G}^*_{q_e,\ell_{q_e}} \text{ aborts}] \le \sum_{i=1}^{q_e} \Pr[\mathsf{CollD}_i \vee \mathsf{CollW}_i \vee \mathsf{CollH} \vee \mathsf{PreimgH}] \le \frac{(\ell+1)q_e}{2^n} + \varepsilon_{cr} + \varepsilon_{pr},$$

and

$$\Pr[E^*_{q_e,\ell_{q_e}} \mid \mathsf{G}^*_{q_e,\ell_{q_e}} \text{ does not abort}] - \Pr[E^*_{0,\ell_0}] \le (2\ell+4)q_e\varepsilon_{\mathsf{E}}.$$

The latter follows from the fact that during the transitions, the number of keys used for calling E is at most $(2\ell+4)q_e$.

We then consider $\Pr[E_{q_e+1}] = \Pr[\mathcal{A}^{\$,\perp} \Rightarrow 1]$, and derive an upper bound for $\Pr[E_{q_e+1}] - \Pr[E^*_{q_e,\ell_{q_e}} \mid \mathsf{G}^*_{q_e,\ell_{q_e}} \text{ does not abort}]$. First, conditioned on that CollD_i never happened for any i, for any two intermediate values $W^{(i)}$ and $W^{(j)}$ we have:

- if their nonce values are the same, i.e. $N_i\|A_i = N_j\|A_j$, then $W^{(i)} \neq W^{(j)}$. Therefore, the two further derived values $V^{(i)} = P_{i,\ell_i+2}(W^{(i)})$ and $V^{(j)} = P_{i,\ell_j+2}(W^{(j)})$ are two random and independent values;
- if $N_i\|A_i \neq N_j\|A_j$, then $W^{(i)}$ and $W^{(j)}$ are random and independent, since they are the outputs of two independent random permutations. This also means the further derived $V^{(i)}$ and $V^{(j)}$ are random and independent.

Thus the q_e outputs $V^{(1)}, \ldots, V^{(q_e)}$ are random and independent. Yet, some of them may be the outputs of the same random permutation (in the extreme case, i.e. all the nonce values are identical, all of them are the outputs of the same random permutation), and will thus be distinct. By this, the statistical distance between $V^{(1)}, \ldots, V^{(q_e)}$ and uniform is at most $q_e^2/2^{n+1}$.

Following the same line, the further derived q_e initial keys $k_1^{(1)}, \ldots, k_1^{(q_e)}$ for the one-time encryption are also uniform and independent. This indicates the $\sum_{i=1}^{q_e} \ell_i$ blocks in the subsequent q_e key streams are given by $\sum_{i=1}^{q_e} \ell_i$ independent random permutations, i.e. $P'_{1,1}, \ldots, P'_{1,\ell_i}$ for $i = 1, \ldots, q_e$,

and thus uniform and independent. Thus for the q_e ciphertexts $(c_1^{(1)}, \ldots, c_\ell^{(1)}), \ldots, (c_1^{(q_e)}, \ldots, c_\ell^{(q_e)})$ the distribution is also uniform.

Finally, we argue that during each encryption query (N_i, A_i, M_i), when the computation proceeds to the tag generation phase $h_i \leftarrow \mathsf{H}(1\|R_i\|V_i\|c_i)$ and $T_i \leftarrow \pi(1\|h_i)$, the random permutation query $\pi(1\|h_i)$ is fresh, i.e. it never happened before, and no previous inverse query resulted in h_i (in this case, the subsequently generated tag T_i is not random either). To this end, we exclude two possibilities:

- If the entry $\pi(1\|h_i)$ is non-fresh due to a previous inverse query $\pi^{-1}(T)$, then $h_i = \mathsf{trunc}(\pi^{-1}(T))$ contradicts $\neg\mathsf{PreimgH}$;
- Otherwise, the query $\Pi(1\|Z_i)$ was necessarily made in an earlier encryption query, due to its "second" leak free call $\mathsf{E}^*(1\|\cdot)$. Assume that this encryption query is (N_j, A_j, M_j). This means $\mathsf{H}(1\|R_i\|V_i\|c_i) = \mathsf{H}(1\|R_j\|V_j\|c_j)$. If $R_i \neq R_j$, or $c_i \neq c_j$, then this clearly contradicts $\neg\mathsf{CollH}$. Otherwise, if $N_i\|A_i \neq N_j\|A_j$ then $\mathsf{H}(0\|N_i\|A_i) = R_i = R_j = \mathsf{H}(0\|N_j\|A_j)$ also contradicts $\neg\mathsf{CollH}$. If $N_i\|A_i = N_j\|A_j$ and $c_i \neq c_j$ then $(N_i\|A_i\|M_i) = (N_j\|A_j\|M_j)$, and it contradicts the assumption that $\mathcal{A}$ never repeats queries;
- there exists an earlier decryption query $(N_j, A_j, (c_j, T_j))$ such that $h_i = \mathsf{trunc}(\pi^{-1}(T_j))$. However, this would contradict $\neg\mathsf{PreimgH}$.

By these, the q_e tags $T_1, \ldots, T_{q_e}$ are distinct and random values, and deviate from the at most q_d tags specified in the decryption queries. Therefore, the statistical distance between the distribution of $T_1, \ldots, T_{q_e}$ and uniform is at most $(q_e + q_d)^2/2^{n+1}$.

Gathering the above, we have

$$\Pr[E_{q_e+1}] - \Pr[E^*_{q_e,\ell_{q_e}} \mid \mathsf{G}^*_{q_e,\ell_{q_e}} \text{ does not abort}] \leq \frac{(q_e+q_d)^2 + q_e^2}{2^{n+1}}.$$

To conclude, we summarize the gaps as follows:

(i) $|\Pr[E_0] - \Pr[E_{0,1} \mid \mathsf{G}_{0,1} \text{ does not abort}]| \leq \varepsilon_{\mathsf{E}^*}$;
(ii) $\Pr[E_{0,1} \mid \mathsf{G}_{0,1} \text{ does not abort}] = \Pr[E^*_{0,\ell_0} \mid \mathsf{G}^*_{0,\ell_0} \text{ does not abort}]$;
(iii) $\Pr[E^*_{q_e,\ell_{q_e}} \mid \mathsf{G}^*_{q_e,\ell_{q_e}} \text{ does not abort}] - \Pr[E^*_{0,\ell_0} \mid \mathsf{G}^*_{0,\ell_0} \text{ does not abort}]$ is upper-bounded by $(2\ell + 4)q_e\varepsilon_{\mathsf{E}}$;
(iv) $\Pr[E_{q_e+1}] - \Pr[E^*_{q_e,\ell_{q_e}} \mid \mathsf{G}^*_{q_e,\ell_{q_e}} \text{ does not abort}] \leq \frac{(q_e+q_d)^2+q_e^2}{2^{n+1}}$;
(v) $\Pr[\mathsf{G}^*_{q_e,\ell_{q_e}} \text{ aborts}] \leq \frac{(\ell+1)q_e}{2^n} + \varepsilon_{cr} + \varepsilon_{pr}$.

Thus in total, $|\Pr[E_0] - \Pr[E_{q_e+1}]|$ is bounded by

$$\varepsilon_{\mathsf{E}^*} + \varepsilon_{cr} + \varepsilon_{pr} + (2\ell+4)q_e\varepsilon_{\mathsf{E}} + \frac{2(\ell+1)q_e + (q_e+q_d)^2 + q_e^2}{2^{n+1}}$$

as claimed.

Empty Messages. can be handled by the proof in this subsection without specific treatment. In detail, for the empty messages appearing in decryption queries the case resembles subsection D.2. When empty messages appear in the encryption queries, it can handle as well: for example, in the non-reuse case, with $\ell_i = 0$ the calls to E_{s_0} and E_{s_1} would be rightfully replaced by random permutations.

Description of AEDT:

Gen(1^n) picks a random key $k \xleftarrow{\$} \{0,1\}^n$, $\mathcal{N}, \mathcal{M}, \mathcal{C} = \{0,1\}^n$.

Enc$_k$(N, A, M) parses $M \in \mathcal{M}^*$ into as many blocks as needed as $M = (m_1, \ldots, m_\ell)$ for some ℓ. Computes $R \leftarrow \mathsf{H}(0\|N\|A)$ and proceeds in three steps:

1. *One-time encryption:* First computes $k_0 \leftarrow \mathsf{E}^*_k(0\|R)$, then, for $i = 1$ to $\ell - 1$, computes $k_i \leftarrow \mathsf{E}_{k_{i-1}}(p_A)$, $z_i \leftarrow \mathsf{E}_{k_{i-1}}(p_B)$, and $c_i \leftarrow z_i \oplus m_i$. Eventually, computes $z_\ell \leftarrow \mathsf{E}_{k_{\ell-1}}(p_B)$ and $c_\ell \leftarrow z_\ell \oplus m_\ell$.
2. *Authentication:* sets $c = c_1\| \ldots \|c_\ell$, and computes $T \leftarrow \mathsf{E}^*_k(1\|\mathsf{H}(1\|R\|c))$.

Eventually, returns the ciphertext $C = (c, T)$.

Dec$_k$(N, A, C) parses $C = (c, T)$, $c = c_1\| \ldots \|c_\ell$, then proceeds in four phases:

1. *Integrity Checking:* computes $R = \mathsf{H}(0\|N\|A)$ and $h^* \leftarrow \mathsf{trunc}((\mathsf{E}^*_k)^{-1}(T))$. Then, if $h^* = \mathsf{H}(1\|R\|c)$, it enters the next phase, and returns $\perp$ otherwise.
2. *One-time decryption:* first computes $s_0 \leftarrow \mathsf{E}^*_k(0\|R)$, then, for $i = 1$ to $\ell - 1$, computes $k_i \leftarrow \mathsf{E}_{k_{i-1}}(p_A)$, $z_i \leftarrow \mathsf{E}_{k_{i-1}}(p_B)$, and $m_i \leftarrow z_i \oplus c_i$. Eventually, computes $z_\ell \leftarrow \mathsf{E}_{k_{\ell-1}}(p_B)$ and $m_\ell \leftarrow z_\ell \oplus c_\ell$.

Eventually, returns the message $M = (m_1, \ldots, m_\ell)$.

Fig. 13: The AEDT AEAD scheme.

E CCAmL2 security of AEDT

We didn't formally describe AEDT in the main body: this gap is filled in Fig. 13.

For its (m)CCAmL2 analysis, the leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ of AEDT is defined similarly to the one of FEMALE, i.e., including all the "bounded" leakages generated by the underlying actions.

Theorem 11. *Let $\mathsf{H} : \{0,1\}^* \to \mathcal{B}$ be a $(0, t', \varepsilon_{cr})$-collision resistant and $(q_d, t', \varepsilon_{pr})$-range-oriented preimage resistant hash function. And let $\mathsf{E} : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ be a $(2q_e + 2q_d + 2, t', \varepsilon_{\mathsf{E}})$-SPRP with two implementations: a strongly protected implantation E^* is leak free, and a plain implementation E have leakage function L_{E} that is $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p, 2)$-R-simulatable. Then the AEDT implementation with leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ defined before is $(q_e, q_d, p-1, q_m, q_l, t, \varepsilon_{\mathsf{mCCAmL2}})$ mCCAmL2-secure, where*

$$\varepsilon_{\mathsf{mCCAmL2}} \leq 2\varepsilon_{\mathsf{E}} + \varepsilon_{cr} + \varepsilon_{pr} + \textstyle\sum_{i=1}^{q_m} \varepsilon_{\mathsf{AEDT}\text{-}eav}(\ell_i),$$

and $\varepsilon_{\mathsf{AEDT}\text{-}eav}(\ell_i)$ is the upper bound on the eavesdropper advantage of (q_l, t')-bounded adversaries on AEDT with ℓ_i block messages:

$$\varepsilon_{\mathsf{AEDT}\text{-}eav}(\ell_i) \leq 4\ell_i(\varepsilon_{\mathsf{E}} + \varepsilon_{(2,2)\text{-}rsim}) + \ell_i \cdot \varepsilon_{s\text{-}block},$$

and ℓ_i is the number of blocks in the i-th challenge messages and $\varepsilon_{s\text{-}block}$ is as defined in equation (1). Here $t' = t + (q_e + q_d + q_m)(t_{\$} + t_{1-pass})$, t_{1-pass} is the maximum running time of AEDT upon a single (encryption or decryption) query, and $t_{\$}$ is the time needed for randomly sampling a value from $\mathcal{M}$.

The proof follows the same line as sections D.1 and D.2. We first formally describe the two algorithms RESM and IESM, which denote the processes of using Real/Idealized AEDT to encrypt a Single Message respectively. They are described in Fig. 14 and 15 respectively.

Lemma 7 (Indistinguishability of LRESM and LIESM). *Let $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\mathsf{E}})$-PRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(2,2)\text{-}rsim})$ $(2, 2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then,*

Description of RESM:

- Gen picks $k_0 \xleftarrow{\$} \{0,1\}^n$
- $\mathsf{RESM}_{k_0}(m_1, \ldots, m_\ell)$ proceeds in two steps:
  (1) Initializes an empty list leak for the leakage;
  (2) for $i = 1, \ldots, \ell$, computes $k_i \leftarrow \mathsf{E}_{k_{i-1}}(p_A)$, $z_i \leftarrow \mathsf{E}_{k_{i-1}}(p_B)$, and $c_i \leftarrow z_i \oplus m_i$, and adds $[\mathsf{L}_{\mathsf{E}}(k_{i-1}, p_A), \mathsf{L}_{\mathsf{E}}(k_{i-1}, p_B)]^p$, $\mathsf{L}_{\oplus}(z_i, m_i)$, and $[\mathsf{L}_{\oplus}(z_i, c_i)]^{p-1}$ to the list leak. Here the two constants p_A and p_B in use are the same as those in AEDT.
  $\mathsf{RESM}_{k_0}(m_1, \ldots, m_\ell)$ eventually returns $(c_1, \ldots, c_\ell)$.

We define $\mathsf{LRESM}_{k_0}(m) = (\mathsf{RESM}_{k_0}(m), \mathsf{leak})$ for the list leak standing at the end of the above process.

Fig. 14: The RESM scheme.

Description of IESM:

- $\mathsf{IESM}_{k_0}(m_1, \ldots, m_\ell)$ proceeds in two steps:
  (1) Initializes an empty list leak for the leakage;
  (2) for $i = 1, \ldots, \ell$, samples $k_i \xleftarrow{\$} \{0,1\}^n$ and $z_i \xleftarrow{\$} \{0,1\}^n$ such that $k_i \neq z_i$, sets $c_i \leftarrow z_i \oplus m_i$, and adds $[\mathcal{S}^{\mathsf{L}}(k_{i-1}, p_A, k_i), \mathcal{S}^{\mathsf{L}}(k_{i-1}, p_B, z_i)]^p$, $\mathsf{L}_{\oplus}(z_i, m_i)$, and $[\mathsf{L}_{\oplus}(z_i, c_i)]^{p-1}$ to the list leak. Here the two constants p_A and p_B in use are the same as those in AEDT.
  $\mathsf{IESM}_{k_0}(m_1, \ldots, m_\ell)$ eventually returns $(c_1, \ldots, c_\ell)$.

We define $\mathsf{LIESM}_{k_0}(m) = (\mathsf{IESM}_{k_0}(m), \mathsf{leak})$ for the list leak standing at the end of the above process.

Fig. 15: The IESM scheme.

for every ℓ-block message m, every p_A, p_B, and every $(q_l - 2q_r - q^*, t - 2t_r - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$, the following holds:

$$|\Pr[\mathcal{D}^{\mathsf{L}}(m, \mathsf{LRESM}_{k_0}(m)) \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{L}}(m, \mathsf{LIESM}_{k_0}(m)) \Rightarrow 1]| \leq \ell(\varepsilon_{\mathsf{E}} + \varepsilon_{(2,2)\text{-}rsim}).$$

Here $q_r = \ell(2q_{\mathcal{S}} + 3)$, q^* and t^* are as defined in Lemma 2, and $t_r = 2\ell(t_{\mathcal{S}} + t_{\$} + t_{\mathsf{E}}) + \ell \cdot t_{\oplus}$, where t_{E}, $t_{\$}$, and $t_{\oplus}$ are as assumed in Lemma 3.

Proof. We define G_0 as the security game in which $\mathcal{A}^{\mathsf{L}}$ receives $\mathsf{LRFSM}_{s_0}(m)$ as the input, and G_ℓ as the game in which $\mathcal{A}^{\mathsf{L}}$ receives $\mathsf{LIFSM}_{s_0}(m)$ as the input. We show that G_0 could be transited to G_ℓ via a sequence of intermediate games $\mathsf{G}_1, \mathsf{G}_2, \ldots, \mathsf{G}_{\ell-1}$, which resemble the games $\mathsf{G}_1^*, \ldots, \mathsf{G}_\ell^*$ appeared in the proof of Lemma 3. In detail, for j from 1 to ℓ, we consider the game G_{j-1}: we replace the two intermediate values $\mathsf{E}_{k_{j-1}}(p_A)$ and $\mathsf{E}_{k_{j-1}}(p_B)$ by two distinct random values k_j and z_j, and replace the leakages $[\mathsf{L}_{\mathsf{E}}(k_{j-1}, p_A), \mathsf{L}_{\mathsf{E}}(k_{j-1}, p_B)]^p$, $\mathsf{L}_{\oplus}(\mathsf{E}_{k_{j-1}}(p_B), m_j)$, and $[\mathsf{L}_{\oplus}(\mathsf{E}_{k_{j-1}}(p_B), c_j)]^{p-1}$ by $[\mathcal{S}^{\mathsf{L}}(k_{j-1}, p_A, k_j), \mathcal{S}^{\mathsf{L}}(k_{j-1}, p_B, z_j)]^p$, $\mathsf{L}_{\oplus}(z_j, m_j)$, and $[\mathsf{L}_{\oplus}(z_j, c_j)]^{p-1}$. This yields the game G_j. Clearly,

$$\Pr[\mathcal{D}^{\mathsf{G}_j} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_{j-1}} \Rightarrow 1] \leq \varepsilon_{\mathsf{E}} + \varepsilon_{(2,2)\text{-}rsim}.$$

Therefore, the ℓ transitions eventually yield

$$\left|\Pr[\mathcal{D}^{\mathsf{G}_\ell} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_0} \Rightarrow 1]\right| \leq \ell(\varepsilon_{\mathsf{E}} + \varepsilon_{(2,2)\text{-}rsim})$$

as claimed.

We then show the eavesdropper security of LIESM encrypting an ℓ-block message relies on the security of ISEnc.

Lemma 8. *For every pair of ℓ-block messages m^0 and m^1 and (q_l, t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$, there exists a $(q_l + 2q_r, t + 2t_r)$-bounded adversary $\mathcal{A}^{\mathsf{L}'}$ such that*

$$\begin{aligned}&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIESM}_{k_0}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIESM}_{k_0}(m^1)) \Rightarrow 1]| \\ \leq &\sum_{i=1}^{\ell} |\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^{+}_{k_{i-1}}(m_i^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^{+}_{k_{i-1}}(m_i^1)) \Rightarrow 1]|,\end{aligned}$$

where $k_0, \ldots, k_{\ell-1}$ are chosen uniformly at random, the p_B value used in the scheme $\mathsf{LISEnc}^{+}_{k_{i-1}}(m_i^1)$ is p_B, and m_i^0 and m_i^1 are the i-th block of m^0 and m^1 respectively. Here $q_r = \ell(2q_{\mathcal{S}} + 1)$ and $t_r = \ell(2t_{\mathcal{S}} + 2t_{\$} + t_{\oplus})$, where t_{E}, $t_{\$}$, and $t_{\oplus}$ are as assumed in Lemma 3.

Proof. Following the same line as the proof of Lemma 3, we start by building a sequence of $\ell + 1$ messages $m_{h,0}, \ldots, m_{h,\ell}$ starting from m^0 and modifying its blocks one by one until obtaining m^1. That is, $m_{h,i} := m_1^0 \| \ldots \| m_{\ell-i}^0 \| m_{\ell-i+1}^1 \| \ldots \| m_{\ell}^1$. It can be shown

$$\begin{aligned}&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIESM}_{k_0}(m_{h,i-1})) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LIESM}_{k_0}(m_{h,i})) \Rightarrow 1]| \\ \leq &|\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^{+}_{k_{\ell-i}}(m_{\ell-i+1}^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^{+}_{k_{\ell-i}}(m_{\ell-i+1}^1)) \Rightarrow 1]|.\end{aligned}$$

Taking a summation yields the main claim.

Lemmas 2, 7, and 8 cinch the eavesdropper security on AEDT.

Lemma 9. *Let $\mathsf{E} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\mathsf{E}})$-PRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(2,2)\text{-}rsim})$ $(2,2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every pair of ℓ-block messages m^0 and m^1 and $(q_l - 2q_r - q^*, t - 2t_r - t^*)$-bounded adversary $\mathcal{A}^{\mathsf{L}}$, it holds*

$$\begin{aligned}&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRESM}_{k_0}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{LRESM}_{k_0}(m^1)) \Rightarrow 1]| \\ \leq &4\ell(\varepsilon_{\mathsf{E}} + \varepsilon_{(2,2)\text{-}rsim}) + \ell \cdot \varepsilon_{s\text{-}block},\end{aligned}$$

where q_r, t_r are as defined in Lemma 7, and q^*, t^* are as defined in Lemma 2.

Proof. In a similar vein to Lemma 5.

And then Theorem 11 (the CCAmL2 security of AEDT) could be derived. Briefly speaking, we note that the designs of AEDT and FEMALE share the following in common:

- First, their authentication components are the same;
- Second, their invalid decryption queries only leak some outputs of $(\mathsf{E}_k^*)^{-1}$, which are indistinguishable from meaningless random values.

Therefore, the proof just follows the same line as that of Theorem 4.

On Empty Message. At the end of this section, we again consider the case the scheme is used on an empty message $M = \perp$. In that case, it can be seen AEDT collapses to a hash-then-MAC function, i.e., $C = \mathsf{E}_k^*(1 \| \mathsf{H}(1 \| R))$, where $R = \mathsf{H}(0 \| N \| A)$. Clearly, this provides authentication for A.

On the other hand, although it appears wasting to hash the input twice, it may not be wise to drop the second call for this special case: on one hand, this would require the scheme to incorporate a sub-mechanism to handle this case, resulting in an increased complexity; on the other hand, the existing CIML2 security proof for AEDT may not hold for the "reduced" authentication function.