Towards Lightweight Side-Channel Security and the Leakage-Resilience of the Duplex Sponge

Chun Guo, Olivier Pereira, Thomas Peters, François-Xavier Standaert.

UCL Crypto Group, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

Abstract. Authenticated Encryption (AE) has become the de facto standard for encryption in modern protocols, and the ubiquitous deployment of small connected devices naturally calls for the availability of lightweight AE schemes, as reflected by a new standardisation process initiated by the NIST. Small devices do not only have a limited computational power: they are also targets of choice for side-channel attacks, and the ease of protecting against these attacks is therefore an important aspect of the selection criteria in this standardisation process.
In this paper, we address this challenge by presenting a new AE mode, TETSponge, which carefully combines a tweakable block cipher with strong protections against side-channel attacks that only needs to be called twice per encryption and decryption, and a sponge-style permutation that only needs weak side-channel protections and is used to frugally process the message and associated data.
TETSponge offers many desirable features: *(i)* it supports single-pass encryption and decryption, and is compatible with limited memory requirements, *(ii)* it offers black-box security as an AE mode, with good bounds, including in the multi-user setting, *(iii)* it offers strong resistance against side-channel attacks during both encryption and decryption, and guarantees nonce misuse-resilience. As such, we conclude that TETSponge offers an appealing option for the implementation of lightweight AE in settings where side-channel attacks are an actual concern.
Along our way, we propose the first rigorous methodology that can be used to analyze the leakage-resilience of sponge-based constructions.

Keywords: Authenticated Encryption, Duplex Construction, Leakage-Resilience, Leveled Implementations, Multi-User / Beyond Birthday Security.

1 Introduction

Problem statement. In 2013, the NIST initiated a lightweight cryptography project to understand the need for dedicated lightweight Authenticated Encryption (AE) schemes, which has led to the launching of a standardization process in 2019.[1] In this context, resistance to side-channel attacks is identified as one of the desirable features that is missing from existing solutions. From an application viewpoint, it is easily motivated by the observation that lightweight devices

[1] See https://csrc.nist.gov/projects/lightweight-cryptography.

may be deployed in environments where they can be under physical control of an adversary, yet be responsible of sensitive tasks (e.g., automotive, drone-related). Maybe more worryingly, a lack of embedded security can also be the root of serious distributed attacks starting from seemingly non-critical connected objects (see for example [37]). However, from a cryptographic viewpoint, NIST's lightweight physical security goal challenges the current understanding of side-channel countermeasures, which are known to imply significant overheads. To take one example, the cycle counts of the (optimized) masked implementations of block ciphers by Goudarzi and Rivain presented at Eurocrypt 2017 [23] blows up by factors ranging from tenths to hundreds for number of shares ranging from 2 or 3 to more than 4, compared to a non-protected implementation.

This state of affairs raises the question of the design of AE modes allowing both efficient and lightweight implementations (e.g., supporting streaming) and embedding side-channel resistance features, so that the secure implementation of the mode can circumvent, at least in part, the high costs associated to the uniform protection of its underlying primitives. Besides, and if aiming at standardization, it is also desirable that such modes maintain as much of other standard security features as possible. For example, good multi-user security appears important in front of mass-surveillance and distributed adversaries [5], beyond-birthday security is useful to increase the lifetime of keys [24] (which is relevant in a lightweight scenario), and some kind of resistance (or resilience) against nonce misuse is in general a welcome option [36,2].

State-of-the-art. Taken separately, the design of lightweight symmetric primitives & modes and the design of leakage-resilient primitives & modes have been topics of quite intense research over the last years. For lightweight designs, we refer to the recent survey of Biryukov and Perrin [13], and to the CAESAR competition.[2] For leakage-resilient primitives, we refer to the line of works initiated by Dziembowski and Pietrzak's leakage-resilient stream cipher [19], which has then been the seed for the design of PRGs, PRFs and PRPs [40,20,18], with contrasted practical impact [4]. For leakage-resilient authentication, encryption and AE modes, we refer to the CCS 2015 work of Pereira et al. [34] and follow-ups [7,8,3]. None of these published proposals considers NIST's combined requirements as part of their design goals. We further justify this claim by describing two concrete proposals that get slightly closer to these goals.

First, ISAP was proposed at FSE 2017 by Dobraunig et al. as a potential solution for side-channel secure AE [17]. ISAP makes an important step in putting forward the good properties of sponge-based constructions for side-channel security — a conclusion that was reached independently by the Keccak team in the design of Keyak.[3] In short, the main observation exploited by these designs is that some kind of leakage can be heuristically captured by reducing the capacity of the sponges. Yet, besides the lack of a systematic leakage-resilience analysis which (to the best of our knowledge) is common to all sponge-based construc-

[2] See https://competitions.cr.yp.to/caesar.html.
[3] See https://keccak.team/files/Keyakv2-doc2.2.pdf.

tions, ISAP is a two-pass mode of operation, which is hardly compatible with lightweight encryption that mandates very limited memory requirements.

Second, TEDT is a recent AE mode of operation by Berti et al. [6], aimed at high physical security. TEDT provides strong security guarantees against leakages, that correspond to the top of the hierarchy of confidentiality and integrity definitions established in [25]. Namely, TEDT ensures Ciphertext Integrity with Misuse and Leakage in encryption and decryption (CIML2) in a liberal model where all the intermediate computations are leaked to the adversary but a long-term key. TEDT also ensures security against Chosen Ciphertext Adversaries with misuse-resilience[4] and leakage in encryption and decryption (CCAmL2), in two different leakage models. From an implementation viewpoint, it encourages leveled implementations, where (expensive) protections against side-channel attacks are used in a limited way, while the bulk of the computation can be executed by cheap and weakly protected designs. It is shown in [6] that such implementations can bring significant performance gains compared to implementations where side-channel protections are uniformly used during encryption. Yet, TEDT is also a two-pass design and requires four calls to a Tweakable Block Cipher (TBC) for each message block, which can be expected to be much more expensive than the sponge-based approach adopted in ISAP.

Contribution. Based on this state-of-the-art, we can re-state problem as:

Can we design and analyze a single-pass AE mode based on a sponge construction and providing security against side-channel leakages?

We answer this question positively by proposing a new AEAD mode, denoted as TETSponge, based on a keyed duplex [12]. The abbreviation TETSponge stands for Tweakable (due to the use of TBC), (simultaneously) Encrypt & Tag (the natural feature of Duplex) Sponge. It enhances the duplex with an efficient TBC-based key derivation function (KDF) and tag generation function (TGF), in a appealingly simple design suitable for efficient implementations on various platforms. We show that in addition to black box CCA security, such a single-pass duplex variant already satisfies some nice leakage confidentiality notions, namely CCAmL1 which corresponds to CCA security with misuse-resilience, in the presence of encryption leakages. In addition, the invertibility of the TBC-based TGF endows the strong CIML2 authenticity notion. Furthermore, by using only additional public key material, TETSponge achieves high security bounds in the multi-user setting. As main price to pay, TETSponge does not achieve CCAmL2 security (see appendix A for an attack), which appears to require two passes. To address this limitation and for completeness, we propose and analyze TEDTSponge which is a more direct sponge variant of TEDT that additionally brings CCAmL2 guarantees in two passes. For comparison, we also show why ISAP, the mode that is closest to those introduced in this paper, does not provide either CIML2 or CCAmL2 security.

[4] Following the definition of misuse-resilience in [2]. As discussed in [25], misuse-*resistance* in the sense of [36] is believed to be impossible in many leakage settings.

For brevity, we next refer to TETSponge and TEDTSponge as S1P and S2P, standing for *Sponge with 1/2 Pass(es)*. The guarantees we achieve for these two modes are summarized in Table 1.

Table 1. Summary of our leakage-resilient sponges, using capacity c and an n-bit TBC. Multi-user security (referred to as mu in the acronyms) is ensured up to $\approx 2^{n_p}$ users.

	key size (bits)	stretch	#passes	muCIML2 & muCCAm$	mu LR-CCA
S1P	n priv., n_p pub.	n bits	1	$\min\{2^{c/2}, 2^n/n^2\}$	$2^{n/2}$ muCCAmL1
S2P	n priv., n_p pub.	n bits	2	$\min\{2^{c/2}, 2^n/n^2\}$	$2^{n/2}$ muCCAmL2

As a contribution of independent interest, and in order to investigate the security of our proposals in the presence of leakages, we propose a methodology for the leakage security analysis of keyed duplex/sponge constructions. Since working in the ideal permutation model is somewhat unavoidable with sponges [10], our leakage security analyzes are also made in the ideal permutation model, which is then naturally combined with the oracle-free leakage functions introduced in [40]. As a compensation for this idealized analysis, all results are obtained under the weakest and easiest to validate leakage assumption, namely non-invertibility [21]. Building upon these, we prove security bounds that are expressive and easy-to-understand. We believe that our methodology is general and could be re-used to analyze other keyed duplex/sponge constructions.

2 Preliminaries

Given a bit-string $x \in \{0,1\}^*$, $|x|$ denotes its length. For any value x, we denote by $\mathsf{ls}_a(x)/\mathsf{ms}_a(x)$ the least/most significant a bits of x. $[\mathsf{num}]_a$ is the binary encoding of the integer num using a representation of a bits.

We denote by a $(q_1, \ldots, q_\omega, t)$-bounded adversary a probabilistic algorithm that has access to ω oracles, $O_1, \ldots, O_\omega$, can make at most q_i queries to its i-th oracle O_i, and can perform computation bounded by running time t. Security notions define the oracles $O_1, \ldots, O_\omega$ available to the adversary in a security experiment. In a proof in the *ideal model*, the adversary is also granted access to ideal objects (see later) that we do not always make explicit in the notation.

A leaking version of an algorithm Algo 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 simply returns the outputs of both Algo and $\mathsf{L}_{\mathsf{Algo}}$ which all take the same input.

2.1 Primitives

A random key-less permutation π, as used in sponge designs, refers to a permutation of $\{0,1\}^\ell$ drawn uniformly at random among the set of all permutations of $\{0,1\}^\ell$. The permutation π is then seen as an ideal object.

A Tweakable Block Cipher (TBC) with key space $\{0,1\}^\kappa$, tweak space $\{0,1\}^t$, and domain $\{0,1\}^n$, also denoted (κ, t, n)-TBC, is a mapping $\widetilde{\mathsf{E}} : \{0,1\}^\kappa \times$

$\{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$ such that for any key $K \in \{0,1\}^\kappa$ and any tweak $T \in \{0,1\}^t$, $X \mapsto \widetilde{\mathsf{E}}(K,T,X)$ is a permutation of $\{0,1\}^n$. We only focus on (n,n,n)-TBC in this paper. An ideal TBC $\widetilde{\mathsf{IC}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a TBC sampled uniformly at random from all (n,n,n)-TBCs: the spirit is the same as ideal (block) ciphers. In this case, $\widetilde{\mathsf{IC}}_K^T$ is a random independent permutation of $\{0,1\}^n$ for each $(K,T) \in \{0,1\}^n \times \{0,1\}^n$ even if the key K is *public*.

Definition 1 (Nonce-based AEAD). *A* nonce-based authenticated encryption scheme with associated data *is a tuple* $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$ *such that:*

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

The message size ℓ_m *uniquely determines the ciphertext size* $\ell_c = \ell_m + \mathsf{oh}$*, where the constant* oh *is the stretch.* $\mathcal{C}_{\ell_m}$ *denotes the set of all the ciphertexts encrypting* ℓ_m*-size messages. Given a key* $k \leftarrow \mathcal{K}$*,* $\mathsf{Enc}_k(N,A,M) := \mathsf{Enc}(k,N,A,M)$ *and* $\mathsf{Dec}_k(N,A,C) := \mathsf{Dec}(k,N,A,C)$ *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 often simply refer to it as *authenticated encryption*.

2.2 Security definitions in the multi-user setting

We borrow the security model of [6], which captures attacks in the multi-user setting. First, we recall the notion of *multi-user misuse-resilience*.

Misuse-resilience. Ashur et al. [2] proposed a strong indistinguishability notion for authenticated encryption which divides adversarial encryption queries into *challenge* and *non-challenge* ones, and only requires the adversary to be nonce-respecting among the former type of queries. The nonce-misuse in non-challenge queries should not affect the pseudorandomness of the responses to the challenge queries, i.e., of the challenge ciphertexts. To avoid confusion with (the initials of) *misuse-resistance* [36], we will refer to *misuse-resilience* as $\mathsf{CCAm\$}$ (with a small m), which is a "real-or-random" indistinguishability game between the real world $(\mathsf{Enc}_k, \mathsf{Enc}_k, \mathsf{Dec}_k)$ and the random (or ideal) world $(\mathsf{Enc}_k, \$, \perp)$, hence the \$, where the second oracle is the challenge oracle.

Definition 2 ($\mathsf{muCCAm\$}$ advantage). *Given a nonce-based authenticated encryption* $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$*, the* multi-user chosen ciphertext misuse resilience advantage *of an adversary* $\mathcal{A}$ *against* AEAD *with* u *users is*

$$\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathcal{A},\mathsf{AEAD},u} := \left| \Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K}},\mathsf{Enc}_{\mathbf{K}},\mathsf{Dec}_{\mathbf{K}},\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K}},\$,\perp,\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] \right|,$$

where the probability is taken over the u *user keys* $\mathbf{K} = (K_1, \ldots, K_u)$*, where* $K_i \leftarrow \mathcal{K}$*, over* $\mathcal{A}$*'s random tape and the ideal oracles* π *and* $\widetilde{\mathsf{IC}}$ *and where:*

– $\mathsf{Enc}_{\mathbf{K}}(i, N, A, M)$*: if* $1 \leq i \leq u$*, outputs* $\mathsf{Enc}_{K_i}(N, A, M)$*;*
– $\$(i, N, A, M)$ *outputs and associates a fresh random ciphertext* $C \xleftarrow{\$} \mathcal{C}_{|M|}$ *to fresh input, and the already associated* C *otherwise;*
– $\mathsf{Dec}(i, N, A, C)$ *outputs* $\mathsf{Dec}_{K_i}(N, A, C)$ *if* (i, N, A, C) *is not an oracle answer to an encryption query* (i, N, A, M) *for some* M*, and* $\perp$ *otherwise;*
– $\perp(i, N, A, C)$ *outputs* $\perp$*;*
– *for each user* i*: (i) nonce* N *cannot be used both in query to* $O_1(i, N, *, *)$ *and* $O_2(i, N, *, *)$*; (ii)* $O_2(i, *, *, *)$ *is nonce-respecting; (iii) if* C *is returned by* $O_1(i, N, A, M)$ *or* $O_2(i, N, A, M)$ *query* $O_3(i, N, A, C)$ *is forbidden; (iv) a nonce used twice with* O_1 *cannot be used for an* O_3 *query.*

The *extended* muCCAm$ advantage muCCAm* is defined as the muCCAm$ advantage where the last limitation, restriction *(iv)*, is waived from the definition.

Leakage-resilience. In front of a leakage adversary, separate definitions for integrity and confidentiality potentially offer more gradual degradations. This results from the important general feature of physically observable cryptography that *unpredictability is much easier to ensure than indistinguishability* [32], which naturally splits the level of confidence we might expect to achieve both notions. Here, we will focus on misuse-resilient AEAD with misuse-*resistant* integrity and misuse-*resilient* confidentiality. To formalize the leakage depending on an implementation, AEAD is associated to both an encryption leakage function $\mathsf{L}_{\mathsf{Enc}}$ and a decryption leakage function $\mathsf{L}_{\mathsf{Dec}}$.

Berti et al. defined the leakage integrity notion of *Ciphertext Integrity with Misuse*-resistance *and (encryption & decryption) Leakage* in [7,8], which is denoted CIML2. In some sense, the definition is obtained by enhancing the traditional INT-CTXT security game with encryption and decryption leakages. Below, we recall the multi-user extension of this definition from [6].

Definition 3 (muCIML2 advantage). *Given a nonce-based authenticated encryption* $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L}_{\mathsf{Enc}}, \mathsf{L}_{\mathsf{Dec}})$*, the multi-user ciphertext integrity advantage with misuse-*resistance *and leakage of an adversary* $\mathcal{A}$ *against* AEAD *with* u *users is*

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u} := \left| \Pr\left[\mathcal{A}^{\mathsf{L},\mathsf{LEnc}_{\mathbf{K}},\mathsf{LDec}_{\mathbf{K}},\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{L},\mathsf{LEnc}_{\mathbf{K}},\mathsf{LDec}^{\perp}_{\mathbf{K}},\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] \right| ,$$

where the probability is taken over the u *user keys* $\mathbf{K} = (K_1, \ldots, K_u)$*, with* $K_i \leftarrow \mathcal{K}$*, over* $\mathcal{A}$*'s random tape and the ideal oracles* π *and* $\widetilde{\mathsf{IC}}$ *and where:*

– L *gives access to* $\mathsf{L}_{\mathsf{Enc}}$ *and* $\mathsf{L}_{\mathsf{Dec}}$ *for any* K*,* N*,* A*,* M *or* C *provided by* $\mathcal{A}$*, modeling access to a device under full adversarial control but not containing any user's secret key.*
– $\mathsf{LEnc}_{\mathbf{K}}(i, N, A, M)$*: if* $1 \leq i \leq u$*, outputs the cipher* $\mathsf{Enc}_{K_i}(N, A, M)$ *and the leakage trace* $\mathsf{L}_{\mathsf{Enc}}(K_i, N, A, M)$*;*
– $\mathsf{LDec}_{\mathbf{K}}(i, N, A, C)$*: if* $1 \leq i \leq u$*, outputs* $\big(\mathsf{Dec}_{K_i}(N, A, C), \mathsf{L}_{\mathsf{Dec}}(K_i, N, A, C)\big)$*;*

– $\mathsf{LDec}^{\perp}_{\mathbf{K}}(i, N, A, C)$: *for* $1 \leq i \leq u$, *computes* $\mathsf{leak_d} \leftarrow \mathsf{L_{Dec}}(K_i, N, A, C)$ *and if* C *is an output of some leaking encryption query* (i, N, A, M) *for some* M *outputs* $(M, \mathsf{leak_d})$, *else outputs* $(\perp, \mathsf{leak_d})$.

The muCIML1 variant of this definition is obtained by removing the leakages from the decryption oracles or, formally, by setting $\mathsf{L_{Dec}} = \perp$.

Regarding confidentiality, we rely on the notions of security against *multi-user Chosen-Ciphertext Attacks with misuse-resilience and Leakage*, with leakages happening either during encryption only – muCCAmL1, or during both encryption and decryption – muCCAmL2 [6].

Definition 4 (muCCAmL2 & muCCAmL1 advantages). *Given a nonce-based authenticated encryption* $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$ *with leakage function pair* $\mathsf{L} = (\mathsf{L_{Enc}}, \mathsf{L_{Dec}})$, *the* multi-user chosen-ciphertext advantage with misuse-resilience and leakage *of an adversary* $\mathcal{A}$ *against* AEAD *with* u *users is*

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u} := \left|\Pr\left[\mathsf{PrivK}^{\mathsf{muCCAmL2},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}} \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{muCCAmL2},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}} \Rightarrow 1\right]\right|,$$

$$\mathbf{Adv}^{\mathsf{muCCAmL1}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u} := \left|\Pr\left[\mathsf{PrivK}^{\mathsf{muCCAmL1},0}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}} \Rightarrow 1\right] - \Pr\left[\mathsf{PrivK}^{\mathsf{muCCAmL1},1}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}} \Rightarrow 1\right]\right|,$$

where the security game $\mathsf{PrivK}^{\mathsf{muCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *is defined in Figure 1 and where the security game* $\mathsf{PrivK}^{\mathsf{muCCAmL1},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *removes all the decryption leakage traces* leak_d *and* leak^b_d *from Figure 1.*

$\mathsf{PrivK}^{\mathsf{muCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u}$ is the output of the following experiment:

Initialization: generates u secret keys $K_1, \ldots, K_u \leftarrow \mathcal{K}$ and sets $\mathcal{E}_{ch}, \mathcal{E}_1, \ldots, \mathcal{E}_u \leftarrow \emptyset$.

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

Leaking decryption queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{LDec}(\cdot,\cdot,\cdot,\cdot)$, $\mathsf{LDec}(i, N, A, C)$ outputs $\perp$ if $(i, N, A, C) \in \mathcal{E}_{ch}$, else computes the plaintext $M \leftarrow \mathsf{Dec}_{K_i}(N, A, C)$ and $\mathsf{leak_d} \leftarrow \mathsf{L_{Dec}}(K_i, N, A, C)$ and returns $(M, \mathsf{leak_d})$;

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

Decryption challenge leakage queries: $\mathcal{A}^{\mathsf{L}}$ gets adaptive access to $\mathsf{L_{Decch}}(\cdot,\cdot,\cdot,\cdot)$, $\mathsf{L_{Decch}}(i, N_{ch}, A_{ch}, C^b)$ computes and outputs $\mathsf{leak}^b_{\mathsf{d}} \leftarrow \mathsf{L_{Dec}}(k, N_{ch}, A_{ch}, C^b)$ if $(i, N_{ch}, A_{ch}, C^b) \in \mathcal{E}_{ch}$; Else it outputs $\perp$;

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

Fig. 1: The $\mathsf{PrivK}^{\mathsf{muCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u}$ game.

The muCCAmL1 and muCCAmL2 notions are roughly the traditional CCA notion in the multi-user setting enhanced with encryption and decryption leakages. The presence of $\mathsf{L_{Decch}}$, which provides decryption leakages to challenge

queries, is intended to capture the informativeness of valid decryption leakages, and its motivation stems from the harmfulness of decryption leakages where the plaintext is supposed to remain a secret locked inside the decrypting device, a proprerty that is relevant in some applications such as secure bootloading [33].

We note that CCA security with leakage is not defined in a "real-or-random" form, as it is done in other places. The reason is that such definitions raise challenges when leakages need to happen on the "random" side, which has no physical existence.

In the following discussions, and when the multi-user setting is irrelevant, we will omit the mu from our notations and talk about CIML1, CIML2, CCAmL1 and CCAmL2, which refer to the definitions above when $u = 1$.

3 Single-pass design: TETSponge/S1P

3.1 Background and design considerations

Inner keyed duplex schemes are in general attractive for efficient AE: they can achieve this functionality in a single pass, are highly flexible and ensure nice security bounds in the multi-user setting [12,16]. Most importantly in our context, it is believed that they offer some level of leakage-resilience by design [9,17], even though it has not yet been formalized.

The starting point of our constructions is the (new) observation that the full leakage of internal states in inner keyed duplex may not fully ruin integrity. In fact, once the internal state has been recovered, the construction collapses to a Hash-then-MAC authentication scheme (with the duplex being the keyless hash function and the last permutation call being the fixed input length MAC). Thus, if the last permutation call is carefully protected, the inner keyed duplex could offer CIML1 security (i.e., ciphertext integrity with nonce-misuse and encryption leakages, defined in Section 2.2).

In this respect, the main limitation of this duplex construction in terms of leakage security lies in the verification of the tag. Decryption requires to perform the entire computation to recover the valid tag to compare it with the one included in the ciphertext. In the presence of decryption leakage, this valid tag may leak from a decryption attempt on an invalid ciphertext, leading to a CIML2 attack. For an n-bit tag, an approach to cope with this decryption leakage is to rely on an n-bit invertible primitive like a (tweakable) block cipher [8], so that the verification only compares (and leaks) pre-images of tags without ever computing the valid tag, which then remains unpredictable. Inversion may be challenging though, as the n-bit tag size is typically much smaller than the size of the sponge permutation ℓ and, due to the presence of leakages, it is hard to build such an n-bit invertible primitive from the ℓ bit permutation π. (Feistel-based solution requires super-logarithmic number of rounds [18]).

With these considerations in mind, we leverage a TBC for this part of our modes. That is, we derive a $2n$-bit (hash) digest from the duplex function, and we then use an (n, n, n)-TBC to absorb this digest and generate the (n-bit) tag.

As such, when internal states are leaked, the keyed duplex collapses to a simple Hash-then-TBC scheme, which still naturally achieves CIML2. The motivation for employing $2n$-bit digests is to boost the complexity of hash collision-based forgery attacks to 2^n, which is inherited from [6].

For the rest, the use of nonce-based key-derivation for state-refreshing per message is a common design strategy for SCA secure AEs [7,17,8]. As will be detailed next, as long as the internal state is not fully exposed by SCAs, it enables the inner keyed duplex to maintain some level of privacy and integrity. The use of "public keys" in order to offer multi-user security beyond $n/2$ is also inherited from [6]. In detail, the presence of possibly public additional randomness PK immediately reduces the collision probability between user keys. Moreover, using these public keys as the tweak inputs creates a strong independence between different users and prevents the effectiveness of offline computation brought in by the multiple user keys. Besides, as these public keys do not need any physical protection for confidentiality, they can be freely used in weakly protected parts of our mode of operation. We refer to [6] for more information.

3.2 Specification

The TETSponge/S1P scheme is a one-pass sponge-based mode for AEAD.

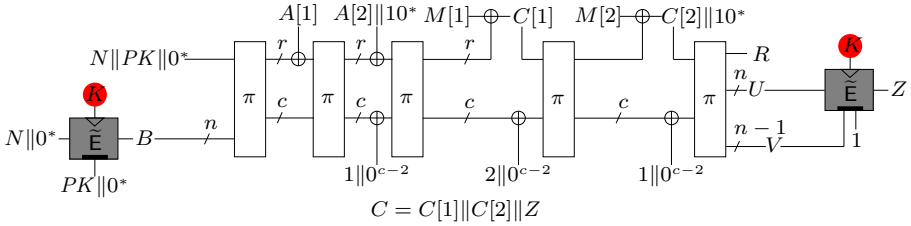

Fig. 2: $\mathsf{S1P}[\pi, \widetilde{\mathsf{E}}]_{K,PK}$ AEAD with $\nu = 2$ r-bit block of associated data and $\ell = 2$ r-bit blocks of message. Dark squares indicate the (n, n, n)-TBC where, the triangle denotes the key input and the small black rectangle denotes the tweak input. The value $1\|0^{c-2}$ is inserted only if $|A[\nu]| < r$, resp. $|M[\ell]| < r$.

Parameters. $\widetilde{\mathsf{E}}$ is an (n, n, n)-TBC and π is an $\ell = r+c$ bit keyless permutation. The key of S1P is $K\|PK$, where $|K| = n$ and $|PK| = n_p$. We stress that only K has to be kept *secret*, but PK can be public. The secret key K is picked *uniformly* at random in $\{0,1\}^n$. The public key PK only needs to be *distinct* for each session. For simplicity, in this paper we focus on uniform $PK \in \{0,1\}^{n_p}$. Let $n_N = |N|$ be the fixed length of the nonces. We require that $n_p \leq r$, $n_N + n_p + n \leq r + c$, and $2n \leq r + c + 1$. Yet, we recommend $n_p \approx n$ and $c \approx 2n$ and we actually choose $n_p = n - 1$ and $c = 2n$ as this leads to a security up to $2^n/n^2$ complexity. There is no recommendation for n_N, but when $n = 128$ one could take $n_N = 96$ which is a standard choice.

The encryption. Upon encrypting (N, A, M), the mode first derives an n-bit initial seed B from N, using a strongly protected TBC-call to $\widetilde{\mathsf{E}}_K^{PK\|0^*}(N\|0^*)$.

The initial seed B is then used as the key of the inner keyed duplex to process A and $M = M[1]\| \ldots \|M[\ell]$ and produce $\mathbf{c} = C[1]\| \ldots \|C[\ell]$. Note that 2 bits are used for domain separation, in order to distinguish M from A and mark if the last blocks of A and M are of full r bits or not.

Let $U\|V$ be the least significant $2n-1$ bits of the final state with $|U| = n$. As discussed, this truncated state is *not* immediately used as the tag. Instead, another strongly protected TBC-call is made, which generates the n-bit tag $Z = \widetilde{\mathsf{E}}_K^{V\|1}(U)$. The final ciphertext is $C[1]\| \ldots \|C[\ell]\|Z$.

The Decryption. Upon decrypting (N, A, C), $C = C[1]\| \ldots \|C[\ell]\|Z$, the mode first derives the initial seed B via $B = \widetilde{\mathsf{E}}_K^{PK\|0^*}(N\|0^*)$, and then runs the inner keyed duplex on A and $C[1]\| \ldots \|C[\ell]$ to derive M and the $2n-1$ bit truncated state $U\|V$. Finally, it makes an inverse TBC call $U^* = (\widetilde{\mathsf{E}}_K^{V\|1})^{-1}(Z)$, and outputs M if and only if there is a match $U = U^*$. In such a way, invalid decryption only leaks meaningless random values U^* (instead of the correct tag) which is instrumental to achieve $\mathsf{CIML2}$.

The encryption and decryption are formally described below. Note that when $\mathbf{c}$ is empty while A passes the integrity checking, $\mathsf{S1P.Dec}$ explicitly returns a special value **true**, so that it can be used for authenticating A.

---

algorithm $\mathsf{S1P}[\pi, \widetilde{\mathsf{E}}].\mathsf{Enc}_{K,PK}(N, A, M)$

1. $\ell \leftarrow \lceil |M|/r \rceil$, $\nu \leftarrow \lceil |A|/r \rceil$
2. parse M as $M[1]\| \ldots \|M[\ell]$, with $|M[1]| = \ldots = |M[\ell-1]| = r$ and $1 \le |M[\ell]| \le r$
3. parse A as $A[1]\| \ldots \|A[\nu]$, with $|A[1]| = \ldots = |A[\nu-1]| = r$ and $1 \le |A[\nu]| \le r$
4. $B \leftarrow \widetilde{\mathsf{E}}_K^{PK\|0^{n-n_P}}(N\|0^{n-n_N})$
5. $ivsize \leftarrow r + c - n$
6. $IV \leftarrow N\|PK\|0^{ivsize-n_N-n_P}$
7. $S_0 \leftarrow IV\|B$, $S_1 \leftarrow \pi(S_0)$
8. **if** $\nu \ge 1$ **then**
9. **for** $i = 1$ **to** $\nu - 1$ **do**
10. $S_i \leftarrow S_i \oplus (A[i]\|0^c)$
11. $S_{i+1} \leftarrow \pi(S_i)$
12. **if** $|A[\nu]| < r$ **then**
13. $A[\nu] \leftarrow A[\nu]\|10^{r-|A[\nu]|-1}$
14. $S_\nu \leftarrow S_\nu \oplus (0^r\|[1]_2\|0^{c-2})$
15. $S_\nu \leftarrow S_\nu \oplus (A[\nu]\|0^c)$
16. $S_{\nu+1} \leftarrow \pi(S_\nu)$
17. **if** $\ell \ge 1$ **then**
18. $S_{\nu+1} \leftarrow S_{\nu+1} \oplus (0^r\|[2]_2\|0^{c-2})$
19. **for** $i = 1$ **to** $\ell - 1$ **do**
20. $j \leftarrow i + \nu$
21. $C[i] \leftarrow \mathsf{ms}_r(S_j) \oplus M[i]$
22. $S_j \leftarrow C[i]\|\mathsf{ls}_c(S_j)$
23. $S_{j+1} \leftarrow \pi(S_j)$
24. $C[\ell] \leftarrow \mathsf{ms}_{|M[\ell]|}(S_{\nu+\ell}) \oplus M[\ell]$
25. **if** $|C[\ell]| < r$ **then**
26. $S_{\nu+\ell} \leftarrow S_{\nu+\ell} \oplus (0^r\|[1]_2\|0^{c-2})$
27. $S_{\nu+\ell} \leftarrow C[\ell]\|10^{r-|C[\ell]|-1}\|\mathsf{ls}_c(S_{\nu+\ell})$
28. **else** $S_{\nu+\ell} \leftarrow C[\ell]\|\mathsf{ls}_c(S_{\nu+\ell})$
29. $S_{\nu+\ell+1} \leftarrow \pi(S_{\nu+\ell})$
30. $U\|V \leftarrow \mathsf{lsb}_{2n-1}(S_{\nu+\ell+1})$
31. $Z \leftarrow \widetilde{\mathsf{E}}_K^{V\|1}(U)$
32. $\mathbf{c} \leftarrow C[1]\| \ldots \|C[\ell]$, $C \leftarrow \mathbf{c}\|Z$
33. **return** C

algorithm $\mathsf{S1P}[\pi, \widetilde{\mathsf{E}}].\mathsf{Dec}_{K,PK}(N, A, C)$

1. $\ell \leftarrow \lceil \frac{|C|-n}{r} \rceil$, $\nu \leftarrow \lceil |A|/r \rceil$
2. parse C as $C[1]\| \ldots \|C[\ell]\|Z$, with $|C[1]| = \ldots = |C[\ell-1]| = r$, $1 \le |C[\ell]| \le r$, and $|Z| = n$
3. parse A as $A[1]\| \ldots \|A[\nu]$, with $|A[1]| = \ldots = |A[\nu-1]| = r$ and $1 \le |A[\nu]| \le r$
4. $B \leftarrow \widetilde{\mathsf{E}}_K^{PK\|0^{n-n_P}}(N\|0^{n-n_N})$
5. $ivsize \leftarrow r + c - n$
6. $IV \leftarrow N\|PK\|0^{ivsize-n_N-n_P}$
7. $S_0 \leftarrow IV\|B$, $S_1 \leftarrow \pi(S_0)$
8. **if** $\nu \ge 1$ **then**
9. **for** $i = 1$ **to** $\nu - 1$ **do**
10. $S_i \leftarrow S_i \oplus (A[i]\|0^c)$
11. $S_{i+1} \leftarrow \pi(S_i)$
12. **if** $|A[\nu]| < r$ **then**

13. $A[\nu] \leftarrow A[\nu] \| 10^{r-|A[\nu]|-1}$
14. $S_\nu \leftarrow S_\nu \oplus (0^r \| [1]_2 \| 0^{c-2})$
15. $S_\nu \leftarrow S_\nu \oplus (A[\nu] \| 0^c)$
16. $S_{\nu+1} \leftarrow \pi(S_\nu)$
17. **if** $\ell \geq 1$ **then**
18. $S_{\nu+1} \leftarrow S_{\nu+1} \oplus (0^r \| [2]_2 \| 0^{c-2})$
19. **for** $i = 1$ **to** $\ell - 1$ **do**
20. $j \leftarrow i + \nu$
21. $M[i] \leftarrow \mathsf{ms}_r(S_j) \oplus C[i]$
22. $S_j \leftarrow C[i] \| \mathsf{ls}_c(S_j)$
23. $S_{j+1} \leftarrow \pi(S_j)$
24. $M[\ell] \leftarrow \mathsf{ms}_{|C[\ell]|}(S_{\nu+\ell}) \oplus C[\ell]$
25. **if** $|C[\ell]| < r$ **then**
26. $S_{\nu+\ell} \leftarrow S_{\nu+\ell} \oplus (0^r \| [1]_2 \| 0^{c-2})$
27. $S_{\nu+\ell} \leftarrow C[\ell] \| 10^{r-|C[\ell]|-1} \| \mathsf{ls}_c(S_{\nu+\ell})$
28. **else** $S_{\nu+\ell} \leftarrow C[\ell] \| \mathsf{ls}_c(S_{\nu+\ell})$
29. $S_{\nu+\ell+1} \leftarrow \pi(S_{\nu+\ell})$
30. $U \| V \leftarrow \mathsf{lsb}_{2n-1}(S_{\nu+\ell+1})$
31. $U^* \leftarrow (\widetilde{\mathsf{E}}_K^{V\|1})^{-1}(Z)$
32. **if** $U \neq U^*$ **then return** $\perp$
33. **else if** $\ell > 0$ **then return** $M[1] \| \dots \| M[\ell]$
34. **else return true**

3.3 Black box security of S1P

We focus on the black-box muCCAm$ security of S1P and defer the leakage security analyzes to Sections 5 and 6. Let $\overrightarrow{q} = (q_m, q_e, q_d, q_{\widetilde{\mathsf{IC}}}, q_\pi)$ and denote by $(\overrightarrow{q}, \sigma)$-*adversaries* the adversaries that make q_m, q_e, q_d, $q_{\widetilde{\mathsf{IC}}}$, and q_π queries to the non-challenge encryption oracle, the challenge encryption oracle, the LDec oracle, $\widetilde{\mathsf{IC}}$, and π, and have at most σ blocks (of r bits) in all their queried plaintexts (both non-challenge and challenge) and ciphertexts, including associated data.

Theorem 1. *Assume* $u \leq 2^{n_p}$, $n_p \leq n$, $n \geq 5$, *and* $q^* = \sigma + q_e + q_d + q_m + q_\pi \leq \min\{2^n/4, 2^{r+c}/2\}$. *Then in the ideal TBC and permutation model, for any* $(\overrightarrow{q}, \sigma)$*-adversary* $\mathcal{A}$ *it holds*

$$\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathcal{A},\mathsf{S1P},u} \leq \frac{5u}{2^{n_p}} + \frac{56(q^*)^2}{2^c} + \frac{23nq^* + 5n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{1}$$

The proof is available in appendix F.[5] Here $q_\pi, q_{\widetilde{\mathsf{IC}}} = t$ represents the time complexity. If $n = 128$, and with our chosen parameters $n_p = n - 1$ and $c = 2n$, the bound simplifies to $\frac{5u}{2^{127}} + \frac{2^{14}t + 2^7\sigma}{2^{128}}$, implying security up to 2^{124} users, 2^{114} computations, and roughly 2^{120} message blocks. For comparison, we informally remark that from the analysis in Appendix F, it can be seen the security bound increases up to $2^n/n$ in the single user setting.

While it is possible to establish bounds beyond $2^{c/2}$ [29,14], we will see that the term $2^{c/2}$ constitutes the CIML2 security limit and cannot be improved. Thus, better muCCAm$ bounds cannot result in a smaller choice of c when CIML2 security is a concern.

3.4 Other remarks

One could instantiate key-derivation and tag-generation with the permutation π to get a purely permutation-based AEAD close to Daemen et al.'s keyed duplex

[5] For simplicity, the proof leverages some technical ingredients introduced in the next sections, so it is better read after the introduction of Appendices C to E.

with built-in multi-user support [16]. We illustrate it in Appendix B, Figure 5. But this variant suffers anew from the re-computation of the valid tag Z to verify the validity of ciphertexts, which might ruin decryption leakage security. So overall, although purely permutation-based designs may be preferable for lightweight cryptography without leakage, we believe their vulnerability to decryption leakages limits their interest (even in lightweight scenarios), in case physical attacks are in the application spectrum envisioned.

We also note that since we expect the sponge to act as a keyless hash when the internal state has been leaked, we cannot follow the more efficient designs approaches that encroach the capacity part of the sponge — including *full-state* duplex [31,16] and *concurrent absorption* [38].

4 Two-pass design: TEDTSponge/S2P

As mentioned in introduction, the main limitation of TETSponge/S1P is that it does not achieve CCAmL2 security (which seems to be the price to pay to obtain a one-pass design). With two passes, an Encrypt-then-MAC composition, the de facto standard choice for leakage-resilient AEs [3,17,8] becomes possible. Concretely, we use the TBC-based key-derivation to start a keyed duplex for encrypting, and then use a hash-then-TBC MAC function to generate the tag from the nonce, the associated data, the ciphertext, and the public key. The keyless hash function is, of course, built upon the sponge function. The ideas of using hash-then-TBC for beyond $n/2$ security and hashing PK for beyond $n/2$ multi-user security are again inherited from [6]. In summary, $\mathsf{S2P}[\pi, \widetilde{\mathsf{E}}]$ can be seen as a (more efficient) sponge-based variant of the TEDT TBC-mode [6], or an ISAP variant using a TBC to improve leakage-resilience (i.e., CIML2 and CCAmL2) security.

Formally, with the same conventions $n_p \leq r$, $n_N + n_p + n \leq r + c$, and $2n \leq r + c + 1$ as in Section 3.2, the mode is given in Figure 3, and described as follows. We recommend the parameters $n_p = n - 1$ and $c = 2n$.

---

algorithm $\mathsf{S2P}[\pi, \widetilde{\mathsf{E}}].\mathsf{Enc}_{K,PK}(N, A, M)$

1. $\ell \leftarrow \lceil |M|/r \rceil$
2. parse M as $M[1]\| \ldots \|M[\ell]$, with $|M[1]| = \ldots = |M[\ell-1]| = r$ and $1 \leq |M[\ell]| \leq r$
3. **if** $\ell > 0$ **then**
4. $\quad B \leftarrow \widetilde{\mathsf{E}}_K^{PK\|0^{n-n_P}}(N\|0^{n-n_N})$
5. $\quad ivsize \leftarrow r + c - n$
6. $\quad IV \leftarrow N\|PK\|0^{ivsize-n_N-n_P}$
7. $\quad S_0 \leftarrow IV\|B$
8. $\quad$ **for** $i = 1$ **to** ℓ **do**
9. $\quad\quad S_i \leftarrow \pi(S_{i-1})$
10. $\quad\quad C[i] \leftarrow \mathsf{ms}_{|M[i]|}(S_i) \oplus M[i]$
11. $\quad\quad S_i \leftarrow C[i]\|\mathsf{ls}_c(S_i)$
12. $\quad \mathbf{c} \leftarrow C[1]\| \ldots \|C[\ell]$
13. **else** $\mathbf{c} \leftarrow \bot$
14. $U\|V \leftarrow \mathsf{H}[\pi](A, \mathbf{c}, N, PK)$
15. $Z \leftarrow \widetilde{\mathsf{E}}_K^{V\|1}(U)$, $C \leftarrow \mathbf{c}\|Z$
16. **return** C

algorithm $\mathsf{S2P}[\pi, \widetilde{\mathsf{E}}].\mathsf{Dec}_{K,PK}(N, A, C)$

1. $\ell \leftarrow \lceil \frac{|C|-n}{r} \rceil$
2. parse C as $\mathbf{c}\|Z$, with $\mathbf{c} = C[1]\| \ldots \|C[\ell]$, $|C[1]| = \ldots = |C[\ell-1]| = r$, and $1 \leq |C[\ell]| \leq r$
3. $U\|V \leftarrow \mathsf{H}[\pi](A, \mathbf{c}, N, PK)$
4. $U^* \leftarrow (\widetilde{\mathsf{E}}_K^{V\|1})^{-1}(Z)$
5. **if** $U \neq U^*$ **then return** $\bot$
6. **else if** $\ell = 0$ **then return true**

7. **else** // $\ell > 0$
8. $B \leftarrow \widetilde{\mathsf{E}}_K^{PK\|0^{n-n_P}}(N\|0^{n-n_N})$
9. $ivsize \leftarrow r + c - n$
10. $IV \leftarrow N\|PK\|0^{ivsize-n_N-n_P}$
11. $S_0 \leftarrow IV\|B$
12. **for** $i = 1$ **to** ℓ **do**
13. $S_i \leftarrow \pi(S_{i-1})$
14. $M[i] \leftarrow \mathsf{ms}_{|C[i]|}(S_i) \oplus C[i]$
15. $S_i \leftarrow C[i]\|\mathsf{ls}_c(S_i)$
16. **return** $M[1]\|\dots\|M[\ell]$

algorithm $\mathsf{H}[\pi](A, \mathbf{c}, N, PK)$

1. $\ell \leftarrow \lceil |M|/r \rceil$, $\nu \leftarrow \lceil |A|/r \rceil$, $\omega \leftarrow \nu + \ell$
2. parse M as $M[1]\|\dots\|M[\ell]$, with $|M[1]| = \dots = |M[\ell-1]| = r$ and $1 \le |M[\ell]| \le r$
3. parse A as $A[1]\|\dots\|A[\nu]$, with $|A[1]| = \dots = |A[\nu-1]| = r$ and $1 \le |A[\nu]| \le r$
4. $S_0 \leftarrow 0^{r+c}$
5. **if** $\nu \ge 1$ **then**
6. **for** $i = 1$ **to** $\nu - 1$ **do**
7. $S_i \leftarrow \pi(A[i]\|\mathsf{ls}_c(S_{i-1}))$
8. **if** $|A[\nu]| < r$ **then**
9. $A[\nu] \leftarrow A[\nu]\|10^{r-|A[\nu]|-1}$
10. $S_{\nu-1} \leftarrow S_{\nu-1} \oplus (0^r\|[1]_2\|0^{c-2})$
11. $S_\nu \leftarrow \pi(A[\nu]\|\mathsf{ls}_c(S_{\nu-1}))$
12. **if** $\ell \ge 1$ **then**
13. $S_\nu \leftarrow S_\nu \oplus (0^r\|[2]_2\|0^{c-2})$
14. **for** $i = 1$ **to** $\ell - 1$ **do**
15. $S_{i+\nu} \leftarrow \pi(C[i]\|\mathsf{ls}_c(S_{i+\nu-1}))$
16. **if** $|C[\ell]| < r$ **then**
17. $S_{\omega-1} \leftarrow S_{\omega-1} \oplus (0^r\|[1]_2\|0^{c-2})$
18. $S_{\omega-1} \leftarrow C[\ell]\|10^{r-|C[\ell]|-1}\|\mathsf{ls}_c(S_{\omega-1})$
19. **else** $S_{\omega-1} \leftarrow C[\ell]\|\mathsf{ls}_c(S_{\omega-1})$
20. $S_\omega \leftarrow \pi(S_{\omega-1})$
21. $S_{\omega+1} \leftarrow \pi\big(N\|0^{r-n_N}\|\mathsf{ls}_c(S_\omega)\big)$
22. $S_{\omega+2} \leftarrow \pi\big(PK\|0^{r-n_P}\|\mathsf{ls}_c(S_{\omega+1})\big)$
23. $U\|V \leftarrow \mathsf{lsb}_{2n-1}(S_{\omega+2})$
24. **return** $U\|V$

Note that it may be tempting to decrease the capacity of the first (keyed) sponge pass in order to improve efficiency. However, such an optimization is unlikely to be easily exploitable in practice. The tricky issue is that this would induce a significant difference between the throughput of the two passes (so that the second pass would remain the bottleneck anyway). As a result, we prefer using the same c twice. a consistent c parameter.

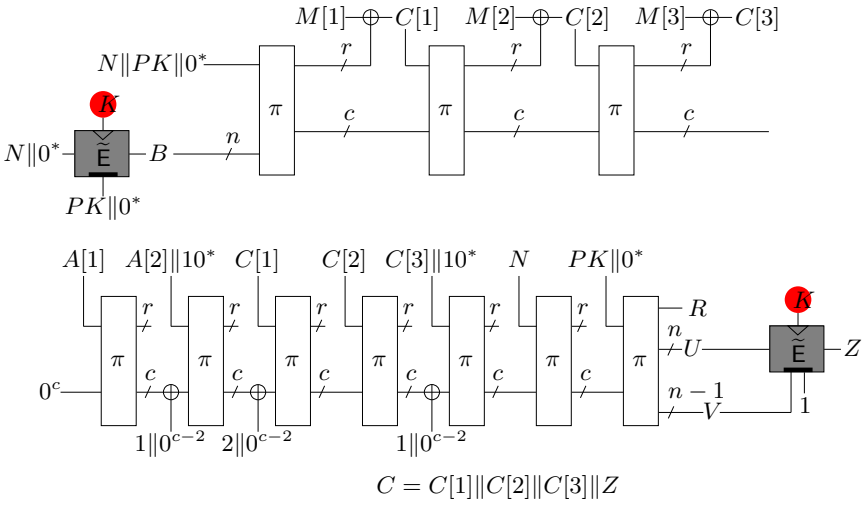

Fig. 3: $\mathsf{S2P}[\pi, \widetilde{\mathsf{E}}]_{K,PK}$ AEAD (with the same notation as before).

As in section 3.3, we present the muCCAm$ security results below (with similar bounds as Theorem 1), and defer leakage analyzes to Sections 5 and 6.

Theorem 2. *Assume* $u \leq 2^{n_p}$, $n_p \leq n$, $n \geq 5$, *and* $q^* = 2\sigma + 2(q_e + q_d + q_m) + q_\pi \leq \min\{2^n/4, 2^{r+c}/2\}$. *Then in the ideal TBC and permutation model, for any* $(\overrightarrow{q}, \sigma)$*-adversary* $\mathcal{A}$, $\overrightarrow{q} = (q_m, q_e, q_d, q_{\widetilde{\mathsf{IC}}}, q_\pi)$, *it holds that:*

$$\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathcal{A},\mathsf{S2P},u} \leq \frac{4u}{2^{n_p}} + \frac{53(q^*)^2}{2^c} + \frac{8nq^* + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}\,. \tag{2}$$

The proof is offered in Appendix G.[6]

5 Leakage integrity analyzes: muCIML2

For integrity, we combine a very conservative model of "unbounded leakage" for the sponge part of our designs, in which the input/output of each call to the π permutation is leaked in full, with strongly protected implementations of the TBC calls, following [7,8]. More precisely, the protected TBC calls are assumed to be secure against key recovery attacks (e.g., DPAs exploiting multiple queries to the TBCs). Informally, these assumptions are instrumental in providing the separation of duties that enables leveled implementations. As a result, our AE modes are closer to *keyless sponge-based hash functions* than the usual *keyed sponges*. Hence, the integrity guarantees rely on the physical protection of the TBC-calls & the ideal permutation assumption (which is a common assumption for analyzing sponges). Formally, we define $\mathsf{L}^* = (\mathsf{L}^*_{\mathsf{Enc}}, \mathsf{L}^*_{\mathsf{Dec}})$, where:

- $\mathsf{L}^*_{\mathsf{Enc}}$ consists of the following information appearing during the encryption:
  - $\{S^{in}, S^{out}\}$ for each internal call to $\pi(S^{in}) \to S^{out}$, and
  - $\{T, X, Y\}$ for each internal call to $\widetilde{\mathsf{E}}^T_K(X) \to Y$ or $(\widetilde{\mathsf{E}}^T_K)^{-1}(Y) \to X$ (i.e., all values are completely leaked except for the key K), and
  - $\{a, b\}$ for each internal XOR action $a \oplus b$.
- $\mathsf{L}^*_{\mathsf{Dec}}$ consists of the above that are generated during the decryption.

We further write $\overrightarrow{q} = (q_e, q_d, q_{\widetilde{\mathsf{IC}}}, q_\pi)$, and denote by $(\overrightarrow{q}, \sigma)$*-adversaries* the adversaries that make q_e, q_d, $q_{\widetilde{\mathsf{IC}}}$, and q_π queries to $\mathsf{LEnc}_{\mathbf{K}}$, $\mathsf{LDec}_{\mathbf{K}}$, $\widetilde{\mathsf{IC}}$, and π, and have at most σ blocks (of r bits) in all their queried plaintext and ciphertext including associated data. In the unbounded leakage model, queries to the training oracle L^* are useless since everything is leaked in full anyway, and we therefore omit them. With these, we first present the results on S1P.

Theorem 3. *Assume* $u \leq 2^{n_p}$, $n_p \leq n$, $n \geq 5$, $q^* = \sigma + q_e + q_d + q_\pi \leq \min\{2^n/4, 2^{r+c}/2\}$, *and leakage* L^* *is "unbounded" as above. Then in the ideal TBC and permutation model, for any* $(\overrightarrow{q}, \sigma)$*-adversary* $\mathcal{A}$ *it holds*

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{S1P},\mathsf{L}^*,u} \leq \frac{3u}{2^{n_p}} + \frac{32(q^*)^2}{2^c} + \frac{7nq^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{3}$$

[6] Again, the proof leverages some technical ingredients introduced in the next sections, so it is better read after the introduction of Appendices C to E.

We then present the results on S2P. The bounds are comparable to Theorem 1.

Theorem 4. *Assume* $u \leq 2^{n_p}$, $n_p \leq n$, $n \geq 5$, $q^* = 2\sigma + 2(q_e + q_d) + q_\pi \leq \min\left\{2^n/4, 2^{r+c}/2\right\}$, *and leakage* L^* *is "unbounded" as above. Then in the ideal TBC and permutation model, for any* $(\overrightarrow{q}, \sigma)$*-adversary* $\mathcal{A}$ *it holds that*

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{S2P},\mathsf{L}^*,u} \leq \frac{2u}{2^{n_p}} + \frac{32(q^*)^2}{2^c} + \frac{3q^* + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}\,. \tag{4}$$

See appendices C and D for their proofs.

6 Leakage privacy analyzes: muCCAmL1 & muCCAmL2

The CCAmL1 and CCAmL2 notions from Definition 4 mainly capture the privacy of *challenge encryption messages*. As mentioned in Section 2.2, this privacy is in general harder to achieve (and analyze) than integrity and, regarding the challenge leakages, some type of "bounded" assumptions are clearly necessary: cleartext messages would be leaked in full otherwise.

This section starts with an introuction to the general leakage model that we will consider in our analyzes (Section 6.1) and the non-invertible leakage assumption that we use as an alternative to the unbounded leakages of the previous integrity analyzes (Section 6.2). We then discuss the (unavoidable) leakage (in)security of a single-round XOR (to which we therefore reduce security) in Section 6.3. Based on these tools, we establish the leakage eavesdropper security of generic keyed duplex in Section 6.4, which allows concluding about the muCCAmL1 and muCCAmL2 security of our new modes in Section 6.5.

To the best of our knowldege, this is the first example of a rigourous model and analysis of a sponge-based mode in the presence of leakages.

6.1 Modeling leakage functions

Most analyzes of sponge-based constructions rely on the ideal (permutation) model.[7] We follow that practice, which has the advantage of offering an easy compatibility with extremely permissive assumptions: we will assume that the leakages resulting from each call to the permuation π is non-invertible, an assumption that is quite minimal and was used before by Yu et al. [40] for modeling leakages around a random oracle. In order to reach a consistent analysis, we stick to the same approach for the $\widetilde{\mathsf{E}}$ block. This approach also comes with the important benefit that it can be easily measured/challenged by cryptanalytic practice (as will be detailed in the next sections), and therefore might lead to a better understanding of how to implement and design modes from a real-world perspective.

Formally, we model leakage traces of $\mathsf{F} \in \{\pi, \widetilde{\mathsf{E}}, \oplus\}$ as the outputs of an *efficient probabilistic* function L_F on the same inputs. Each computation of F comes with leakage traces from L_F. For $\widetilde{\mathsf{E}}$ and π, we further split the leakage traces into an *input* and an *output* part:

[7] In Section 7 we discuss the complications raised by other possibilities.

– For sponge, the leakage trace due to the evaluation of $\pi(S^{in}) \rightarrow S^{out}$ or of $\pi^{-1}(S^{out}) \rightarrow S^{in}$ is written $(\mathsf{L}_{\pi}^{in}(S^{in}), \mathsf{L}_{\pi}^{out}(S^{out}))$;
– For a TBC, the leakage trace due to the evaluation of $\widetilde{\mathsf{E}}_K^T(X) \rightarrow Y$ or of $(\widetilde{\mathsf{E}}_K^T)^{-1}(Y) \rightarrow X$ is written $(\mathsf{L}_{\widetilde{\mathsf{E}}}^{in}(K,T;X), \mathsf{L}_{\widetilde{\mathsf{E}}}^{out}(K,T;Y))$ with semicolon.

This distinction between $\mathsf{L}_{\mathsf{F}}^{in}$ and $\mathsf{L}_{\mathsf{F}}^{out}$ allows to independently quantify the secrecy of the input and the output which better reflects the designers implementation goals for each functions/calls. This also allows interpreting the security bounds based on cryptanalytic experiments (see section 6.2).

We insist again on the probabilistic nature of the leakage functions, even for $\mathsf{L}_{\mathsf{F}}^{in}$ and $\mathsf{L}_{\mathsf{F}}^{out}$ and any F. It corresponds to the practical observation that measuring p times the leakage from the same computation does generally *not* result in completely identical traces, due to thermal noise in devices and to measurement noise for instance. Therefore, we will write $[\mathsf{L}_{\mathsf{F}}]^p$ for a vector of p leakage traces of F.

Oracle-free leakage function. We stress that leakage functions L_{F} must be (probabilistic) functions, and have no access to oracles like $\widetilde{\mathsf{E}}$ or π. This guarantees that L_{F} only leaks information about the computation that is happening in the device, and not about computation that could happen in other calls of $\widetilde{\mathsf{E}}$ or π. Granting access to such oracles to the leakage function would not only be against the locality of leakages (a device cannot leak about values that never appeared in any $\widetilde{\mathsf{E}}$ or π query before), it would also lead to consider artificial attacks. For instance, while it is artificial that L_{π} (say, related to $\pi(S_0) \rightarrow S_1$) outputs information related to another or a "future" call to π (say, $S_2 = \pi(C[1] \| \mathsf{ls}_c(S_1))$), such leakages might prevent the proof to later show a necessary transition related to a call that already leaked before it has been computed (like the unpredictability of the key stream $\mathsf{ms}_r(S_2)$ in the next block) [19,35].

Such artificial "attacks" are made impossible by preventing L_{F} from making any oracle call, an approach that was previously adopted in [40]. Formally, we say that the leakage function associated to the function F is oracle-free, if $\tau(\mathsf{L}_{\mathsf{F}}^{in}) = \tau(\mathsf{L}_{\mathsf{F}}^{out}) = \emptyset$, where $\tau(\mathsf{L}_{\mathsf{F}}^{*})$ is the transcript of queries made by $\mathsf{L}_{\mathsf{F}}^{*}$ to π and $\widetilde{\mathsf{E}}$ when $\mathsf{L}_{\mathsf{F}}^{*}$ is evaluated on its inputs.

6.2 Non-invertible leakage assumption

As explained above, we assume that leakage functions are essentially non-invertible, and on some parts of them in particular.

In the context of a permutation π, and in line with the intuition informally proposed for ISAP [17], we require that, given a sequence of p leakages about an execution of π, in which the adversary may be able to pick r (or c) bits of the input and/or outputs of π, the remaining $c = \ell - r$ bits (or $r = \ell - c$ bits) remain unpredictable, even if $\mathcal{A}$ can make a certain number of guessing attempts.

Internal states' leakages. More formally, we define

$$\mathbf{Adv}^{\mathsf{uplsc}[p]}(\mathcal{A}) := \Pr\big[s_{ch} \xleftarrow{\$} \{0,1\}^c, Guesses \leftarrow \mathcal{A}^{\pi}(\mathsf{leak}) : s_{ch} \in Guesses\big], \quad (5)$$

where $Guesses$ is a finite set of guesses, and $\mathcal{A}$'s input leak is a list of leakages depending on two r-bit values Y^{in} and Y^{pre} specified by $\mathcal{A}$, i.e.,

$$\mathsf{leak} = \Big(\mathsf{ms}_2(s_{ch}), \big[\mathsf{L}_\pi^{out}(Y^{pre}\|s_{ch}), \mathsf{L}_\pi^{in}(Y^{in}\|s_{ch})\big]^p\Big). \tag{6}$$

The presence of the 2 bits leakage $\mathsf{ms}_2(s_{ch})$ is due to assuming that the insertion of the 2-bit domain separation constants completely leaks. We further define

$$\mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, t, q_g) := \max_{(q_\pi,t,q_g)\text{-}\mathcal{A}} \Big\{\mathbf{Adv}^{\mathsf{uplsc}[p]}(\mathcal{A})\Big\}, \tag{7}$$

with the maximal taken over all adversaries that makes q_π queries to π, runs in time t, and makes q_g guesses (i.e., resulting in $|Guesses| = q_g$).

Initial state leakage. In addition to Equation (5), we define

$$\mathbf{Adv}^{\mathsf{uplsn}[p]}(\mathcal{A}) := \Pr\big[B \xleftarrow{\$} \{0,1\}^n, Guesses \leftarrow \mathcal{A}^\pi(\mathsf{leak}) : B \in Guesses\big], \tag{8}$$

where leak depends on $\mathcal{A}$ specified values K, T of n bits and IV of $c+r-n$ bits:

$$\mathsf{leak} = \Big(\big[\mathsf{L}_{\widetilde{\mathsf{E}}}^{out}(K, T; B), \mathsf{L}_\pi^{in}(IV\|B)\big]^p\Big).$$

For simplicity, we also define

$$\mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, t, q_g) := \max_{(q_\pi,t,q_g)\text{-}\mathcal{A}} \Big\{\mathbf{Adv}^{\mathsf{uplsn}[p]}(\mathcal{A})\Big\}. \tag{9}$$

Final state leakage. The last assumption is made on the secrecy of the final state $S_{\nu+\ell+1}$ in $\mathsf{S1P}$ (see line 19 in $\mathsf{S1P}[\pi, \widetilde{\mathsf{E}}].\mathsf{Enc}$):

$$\mathbf{Adv}^{\mathsf{upfi}[p]}(\mathcal{A}) := \Pr\big[S \xleftarrow{\$} \{0,1\}^{r+c}, Guesses \leftarrow \mathcal{A}^\pi(\mathsf{leak}) : S \in Guesses\big],$$

where $R = \mathsf{ms}_{r+c+1-2n}(S)$ and $\mathsf{leak} = (\mathsf{ls}_{2n-1}(S), [\mathsf{L}_\pi^{out}(S)]^p)$. We also define

$$\mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, t, q_g) := \max_{(q_\pi,t,q_g)\text{-}\mathcal{A}} \Big\{\mathbf{Adv}^{\mathsf{upfi}[p]}(\mathcal{A})\Big\}. \tag{10}$$

In the notations, the suffix $[p]$ indicates the number of repeated measurements.

Understanding the up[p] assumptions. The above three assumptions characterize leakage properties of a single call to the sponge permutation π, as summarized in Figure 4 (right). We take Equation (5) as the main example. Concretely, it captures that the value of s_{ch}, which is supposed to be secret in the black-box setting, cannot be recovered from the involved leakages. On the one hand, s_{ch} appears in the leakage of the subsequent permutation-call $\pi(Y^{in}\|s_{ch}) \to \cdot$, which justifies the presence of $\mathsf{L}_\pi^{in}(Y^{in}\|s_{ch})$. On the other hand, s_{ch} itself is the output of the previous permutation-call, which in turn justifies the presence of $\mathsf{L}_\pi^{out}(Y^{pre}\|s_{ch})$. By authorizing p measurements, we capture the repeated measurements that can happen when repeatedly asking for the decryption of a single ciphertext, something that is possible when playing the $\mathsf{muCCAmL2}$

security game. It should be noted that, while this repetition may reduce the measurement noise, it does not contradict the informal separation between SPA and DPA attacks (put forward in [39] and more formally exploited in [7,8,6]). Namely, in practice, it is expected that the (SPA) advantage of Equation (5) is still much smaller than that of a DPA against a block protected with similar countermeasures. The additional leakages $[\mathsf{L}_{\widetilde{\mathsf{E}}}^{out}(K, T_1; B)]^p$ in Equation (8) capture the generation of the initial seed B.

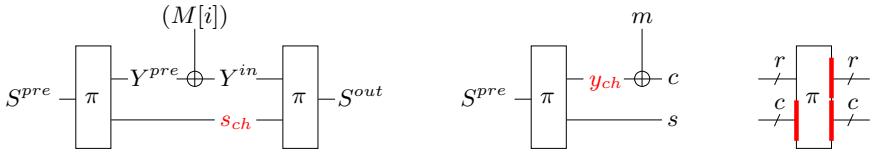

Fig. 4: (Left) The $\mathsf{uplsc}[p]$ assumption where s_{ch} in red is the secret. Other values are defined as in Eq. (5) except S^{pre} and S^{out} which appear in the subsequent security tester. (Middle) The "basic" message processing where y_{ch} in red is the secret. (Right) Summary: all the values in red need to keep some secrecy (i.e., the ls_c bits of the input, and the entire output, as stressed by the red bold lines).

Testers: Measuring advantages in practice. The concrete values of the advantages $\mathbf{Adv}^{\mathsf{uplsc}[p]}$, $\mathbf{Adv}^{\mathsf{upfi}[p]}$ and $\mathbf{Adv}^{\mathsf{uplsn}[p]}$ can be measured for a specific implementation on a specific device by running the following tester against the best known challenging SCA adversary $\mathcal{A}$. This along with Theorems 6 and 7 (later in section 6.5) allows us to determine how many plaintext blocks can be processed before key updating. We also serve Theorem 5 (in Section 6.4) as a general result on keyed duplex constructions, that may be useful for future works. For brevity, we only describe the tester for $\mathbf{Adv}^{\mathsf{uplsc}[p]}$.

1: **Tester for UP** $\mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, t, q_g)$
2: Let the challenging adversary $\mathcal{A}$ serve r bit values Y^{pre} and Y^{in}
3: Pick the secret: $s_{ch} \xleftarrow{\$} \{0,1\}^c$
4: Repeating $S^{pre} \leftarrow \pi^{-1}(Y^{pre} \| s_{ch})$ and $S^{out} \leftarrow \pi^{-1}(Y^{in} \| s_{ch})$ for p times
5: Serve $\mathcal{A}$ with the 2 bits $\mathsf{ms}_2(s_{ch})$ and the leakage traces resulted from step 4. This gives $\mathcal{A}$ the leakages $[\mathsf{L}_\pi^{out}(Y^{pre} \| s_{ch})]^p$ and $[\mathsf{L}_\pi^{in}(Y^{in} \| s_{ch})]^p$.
6: Let the challenging adversary $\mathcal{A}$ output q_g guesses $s_1, \ldots, s_{q_g}$, $\mathcal{A}$ wins as long as $s_{ch} \in \{s_1, \ldots, s_{q_g}\}$

6.3 Capturing the (in)security of the XOR

As a final step towards characterizing the leakage privacy of keyed duplex implementations, we have to measure the leakage resulting from XORing blocks of key stream with message blocks. We stress that, as observed in [34] and follow-up works, the leakage privacy of any such XOR-based design has to rely, in a way or another, on a leakage protection of the XOR. To this end, we follow [34,6]

and define

$$\mathbf{Adv}^{\mathsf{LORL}[p]}(\mathcal{A}) := \Big|\Pr_{\pi}\Big[y_{ch} \xleftarrow{\$} \{0,1\}^r, c^0 \leftarrow y_{ch} \oplus m^0 : \mathcal{A}^{\pi}(c^0, \mathsf{leak}_0) \Rightarrow 1\Big] - \Pr_{\pi}\Big[y_{ch} \xleftarrow{\$} \{0,1\}^r, c^1 \leftarrow y_{ch} \oplus m^1 : \mathcal{A}^{\pi}(c^1, \mathsf{leak}_1) \Rightarrow 1\Big]\Big| \ , \quad (11)$$

where leak_b again depends on a c-bit value s specified by $\mathcal{A}$:

$$\mathsf{leak}_b = \Big(\big[\mathsf{L}_{\pi}^{out}(y_{ch}\|s)\big]^p, \mathsf{L}_{\oplus}(y_{ch}, m^b), \big[\mathsf{L}_{\oplus}(y_{ch}, c^b)\big]^{p-1}\Big). \quad (12)$$

In the abbreviation $\mathsf{LORL}[p]$, the suffix L stands for *leaking*, and the suffix $[p]$ is similar as before. We also define

$$\mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, t) := \max_{\mathcal{A}} \Big\{\mathbf{Adv}^{\mathsf{LORL}[p]}(\mathcal{A})\Big\}. \quad (13)$$

The assumptions with $p > 1$ capture the case of giving (more than one) challenge decryption leakages (from $\mathsf{L}_{\mathsf{Decch}}$). This means the muCCAmL1 security (with no challenge decryption leakage) can be reduced to $\mathbf{Adv}^{\mathsf{uplsc}[1]}, \ldots, \mathbf{Adv}^{\mathsf{LORL}[1]}$.

Understanding $\mathsf{LORL}[p]$ advantage. Equation (11) defines the information an adversary might extract from the "basic" message manipulation made in a keyed duplex, which involves XORing: see Figure 4 (middle). Concretely, the sensitive point is the key stream block y_{ch}. This block is the output of a permutation-call, hence the presence of $\mathsf{L}_{\pi}^{out}(y_{ch}\|s)$. Then, the block y_{ch} is used to mask the message block, and thus $\mathsf{L}_{\oplus}(y_{ch}, m^b)$ comes. As discussed earlier, the vectors of p leakages are relevant when repeated leaking decryption happen, and therefore are used for proving the muCCAmL2 security of modes. We can stick to $p = 1$ when proving muCCAmL1 security.

As discussed in [27,34], if a single XOR of the message leaks a single bit, then no muCCAmL1/muCCAmL2 security would spring up. Thus, it is legitimate to focus on protecting this part of the implementations. Concretely, and while it may not be possible to guarantee that $\mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, t)$ is negligible, we can claim the following: first, this quantity can again be measured by a tester (given in Appendix E.1); second, it allows us to faithfully reduce the muCCAmL1/muCCAmL2 security to very simplified one-time components, possibly with many leaking repetitions but for a single computation from a fixed random string, which further makes it easier to study and to protect as isolated components. Our security proofs then show how the advantage *degrades to the apparent best possible security, at an inevitable rate* during the encryption (or challenge decryption).

6.4 Leakage eavesdropper security of duplex stream ciphers

The argument will proceed in two steps.

The ideal stream cipher with leakages. Consider our target construction, i.e., the duplex stream cipher. Its ideal reference can be obtained by replacing all the internal states in the duplex by uniformly sampled $r+c$ bit values. Below we formally describe these two processes from pseudocode.

Description of $\mathsf{Real}^{\mathsf{t}}[\pi]$:

- Gen picks $B \xleftarrow{\$} \{0,1\}^n$
- $\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M)$ proceeds in five steps:
  (1) Initializes an empty list leak for the leakage;
  (2) Computes $S_0 \leftarrow IV \| B$, $S_1 \leftarrow \pi(S_0)$, and adds the leakages $[\mathsf{L}^{in}_\pi(S_0)]^p$ and $[\mathsf{L}^{out}_\pi(S_1)]^p$ to leak;
  (3) For $i = 1, \ldots, \nu = \lceil |A|/r \rceil$, computes $S_{i+1} \leftarrow \pi((A[i] \| 0^c) \oplus S_i)$, and adds the leakage $[\mathsf{L}^{in}_\pi((A[i] \| 0^c) \oplus S_i), \mathsf{L}^{out}_\pi(S_{i+1}), \mathsf{L}_\oplus(\mathsf{ms}_r(S_i), A[i])]^p$ and $\mathsf{ms}_2(\mathsf{ls}_c(S_i))$ to leak;
  (4) For $i = 1, \ldots, \ell = \lceil |M|/r \rceil$, computes $j \leftarrow i+\nu$, $C[i] \leftarrow \mathsf{ms}_r(S_j) \oplus M[i]$, $S_{j+1} \leftarrow \pi(C[i] \| \mathsf{ls}_c(S_j))$, and adds the leakages $\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), M[i])$, $[\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), C[i])]^{p-1}$, $[\mathsf{L}^{in}_\pi((M[i] \| 0^c) \oplus S_j), \mathsf{L}^{out}_\pi(S_{j+1})]^p$, and $\mathsf{ms}_2(\mathsf{ls}_c(S_i))$ to leak;
  (5) If the tag $t = 1$ then defines $\mathbf{c} = C[1] \| \ldots \| C[\ell] \| \mathsf{ls}_{2n-1}(\mathsf{L}^{out}_\pi(S_{\nu+\ell+1}))$, else defines $\mathbf{c} = C[1] \| \ldots \| C[\ell]$.

  $\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M)$ eventually returns $\mathbf{c}$. For the list leak standing at the end of the above process, we define $\mathsf{LReal}^{\mathsf{t}}_B[\pi](IV, A, M) = (\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M), \mathsf{leak})$.

The process of $\mathsf{Ideal}^{\mathsf{t}}(IV, A, M)$ is the same as $\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M)$, except that the internal states $S_1, \ldots, S_{\nu+\ell+2}$ are all uniformly sampled from $\{0,1\}^{r+c}$. Its complete description is deferred to appendix E.2. We define $\mathsf{LIdeal}^{\mathsf{t}}(IV, A, M) = (\mathsf{Ideal}^{\mathsf{t}}(IV, A, M), \mathsf{leak})$ for the list leak standing at the end of the above process.

Note that the above stream cipher roughly captures the underlying structures of S1P and S2P: for the former, we omit the domain separation constants for simplicity, but since we assumed inserting these constants completely leaks (the 2 bits $\mathsf{ms}_2(\mathsf{ls}_c(S_i))$), our simplification does not affect the result.

We show that the real and ideal stream ciphers (with leakages) are indistinguishable (upon encrypting a single message). This step mainly requires us to bound certain "bad events" in the ideal world, for which the non-invertible leakage assumption is helpful.

Lemma 1. *For every adversary-chosen pair (A, M) with ℓ blocks in total, every T, N, and every (q_π, t)-bounded distinguisher $\mathcal{D}^\pi$, it holds*

$$\begin{aligned}
&\left| \Pr[\mathcal{D}^\pi(\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M)) \Rightarrow 1] - \Pr[\mathcal{D}^\pi(\mathsf{Ideal}^{\mathsf{t}}_B(IV, A, M)) \Rightarrow 1] \right| \\
\leq &\frac{(\ell+2)^2}{2^{c+1}} + \mathbf{Adv}^{\mathsf{upisn}[p]}(q_\pi, O(t + p\ell t_l), q_\pi) + \ell \cdot \mathbf{Adv}^{\mathsf{upisc}[p]}(q_\pi, O(t + p\ell t_l), 2q_\pi) \\
&+ \mathsf{t} \cdot \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, O(t + p\ell t_l), q_\pi), \qquad (14)
\end{aligned}$$

where t_l is the total time needed for evaluating L^{in}_π, L^{out}_π, $\mathsf{L}_\oplus$, and the xor of two r-bit values.

Proof. Wlog we consider the case of $|A| = 0$ and $\ell = \frac{|M|}{r}$ for simplicity.

Preparations. Denote $\mathsf{G}_1(\mathcal{D}, \mathsf{Real}_B^{\mathsf{t}}[\pi], \pi)$ and $\mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}^{\mathsf{t}}, \pi)$ the games capturing the interactions between $\mathcal{D}$ and the real $(\mathsf{Real}_B^{\mathsf{t}}[\pi], \pi)$ and the ideal $(\mathsf{Ideal}^{\mathsf{t}}, \pi)$ resp.; and simplified as G_1 and G_2. We'll prove the indistinguishability of G_1 and G_2. To make it rigorous, we use the H-coefficients technique. A deviation from common applications of this technique is that, instead of considering the mere adversarial transcripts, we focus on an extended notion of transcripts, which summarize the whole *interactions*. This is because during the games in question, there are various other randomness sources such as the random coins of the leakage functions, and these could only be captured by the extended transcripts.[8]

Concretely, note that the real adversarial transcripts could be summarized as two lists τ_{le} and τ_π: the former includes the ciphertext as well as the leakages (its concrete representation won't be needed in this proof), while the latter

$$\tau_\pi = \big((S_1^{in}, S_1^{out}), \ldots, (S_{q_\pi}^{in}, S_{q_\pi}^{out})\big) \tag{15}$$

includes the adversarial permutation queries and responses, and indicates the i-th query is either forward $\pi(S_i^{in}) \to S_i^{out}$ or backward $\pi^{-1}(S_i^{out}) \to S_i^{in}$. Besides, at the end of the interaction, we reveal the involved internal state $\mathbf{S} = (S_0, S_1, \ldots)$ to $\mathcal{D}$, and append this to its transcript. Clearly, this doesn't reduce its advantage.

Moreover, note that the interactions with the (real or ideal) encryption process additionally rely on $\mathbf{r}$, the random coins of the distinguisher $\mathcal{D}$ & the involved leakage functions. Yet, it can be seen that during two games $\mathsf{G}_1(\mathcal{D}, \mathsf{Real}_B^{\mathsf{t}}[\pi_1], \pi_1)$ and $\mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}^{\mathsf{t}}, \pi_2)$, if the following conditions are fulfilled, then the queries and responses of $\mathcal{D}$ are the same, and thus $\mathcal{D}$ outputs the same:

- $\pi_1 \vdash \tau_\pi$, and $\pi_2 \vdash \tau_\pi$;
- the internal state values produced in the two games are the same $\mathbf{S}$;
- the random coins $\mathbf{r}$ used in G_1 and G_2 are the same.

The reason is that, all the internal actions in $\mathsf{G}_1(\mathcal{D}, \mathsf{Real}_B^{\mathsf{t}}[\pi_1], \pi_1)$ and $\mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}^{\mathsf{t}}, \pi_2)$ give rise to the same results. With the above considerations, we summarize all the randomness in what we call *extended transcripts*, i.e., $\tau = (\tau_\pi, \mathbf{S}, \mathbf{r})$. Note that τ_{le} disappears in τ, as it can be recovered from $\mathbf{r}$ and $\mathbf{S}$.

With respect to some fixed distinguisher $\mathcal{D}$, an extended transcript $\tau = (\tau_\pi, \mathbf{S}, \mathbf{r})$ is said *attainable* if there exists randomness $(\mathbf{r}, \pi)$ such that using $\mathbf{r}$, the ideal execution of $\mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}^{\mathsf{t}}, \pi)$ yields $(\tau_\pi, \mathbf{S})$. We denote $\mathcal{T}$ the set of attainable transcripts. In all the following, we denote T_{re}, resp. T_{id}, the probability distribution of the transcript τ induced by the real world, resp. the ideal world (note that these two probability distributions depend on the distinguisher). By extension, we use the same notation to denote a random variable distributed according to each distribution.

Given a set τ_π and a random permutation π, we say that π *extends* τ_π, denoted $\pi \vdash \tau_\pi$, if $\pi(S^{in}) = S^{out}$ for all $(S^{in}, S^{out}) \in \tau_\pi$. It is easy to see that

[8] The idea of focusing on the whole internal randomness rather than the mere adversarial transcripts has appeared in indifferentiability proofs [28].

for any attainable transcript $\tau = (\tau_\pi, \mathbf{S}, \mathbf{r})$, the event $T_{id} = \tau$ happens if and only if $\pi \vdash \tau_\pi$, $\mathbf{S}$ is generated in the execution, and the randomness $\mathbf{r}$ is used (by $\mathcal{D}$ and L), while the event $T_{re} = \tau$ happens if and only if $\pi \vdash \tau_\pi$, $\pi(S_0) = S_1$, $\pi\big(C[1]\|\mathsf{ls}_c(S_i)\big) = S_{i+1}$ for every S_i in $\mathbf{S}$, and the randomness $\mathbf{r}$ is used.

With the above, the H-coefficients main lemma [15] is as follows.

Lemma 2. *Fix a distinguisher $\mathcal{D}$. Let $\mathcal{T} = \mathcal{T}_{good} \cup \mathcal{T}_{bad}$ be a partition of the set of attainable transcripts $\mathcal{T}$. Assume that there exists ε_1 such that for any $\tau \in \mathcal{T}_{good}$, one has*

$$\frac{\Pr[T_{re} = \tau]}{\Pr[T_{id} = \tau]} \geq 1 - \varepsilon_1,$$

and that there exists ε_2 such that $\Pr[T_{id} \in \mathcal{T}_{bad}] \leq \varepsilon_2$. Then $\mathbf{Adv}(\mathcal{D}) \leq \varepsilon_1 + \varepsilon_2$.

Bad Extended Transcripts. An attainable transcript $\tau = (\tau_\pi, \mathbf{S}, \mathbf{r})$ is bad, if either of the following two conditions is fulfilled:

- (B-1) *contradiction*: there exists two distinct indices $i, j \in [0, \ldots, \ell+1]$ such that $\mathsf{ls}_c(S_i) = \mathsf{ls}_c(S_j) \wedge \mathsf{ls}_c(S_{i+1}) \neq \mathsf{ls}_c(S_{j+1})$, or $\mathsf{ls}_c(S_i) \neq \mathsf{ls}_c(S_j) \wedge \mathsf{ls}_c(S_{i+1}) = \mathsf{ls}_c(S_{j+1})$;
- (B-2) *exposure of secret state*: if any of the following is fulfilled:
  - $(S_0, \star) \in \tau_\pi$; or
  - there exits $i \in [1, \ldots, \ell+1]$ (when $\mathsf{t} = 1$) such that $(\star, S_i) \in \tau_\pi$; or
  - there exits $i \in [1, \ldots, \ell]$ such that $(C[i]\|\mathsf{ls}_c(S_i), \star) \in \tau_\pi$.

For clearness, below we mainly focus on the case $\mathsf{t} = 1$, as the possibilities of leakages are more than the case $\mathsf{t} = 0$. The condition (B-1) is easy to bound:

$$\Pr[\text{(B-1)}] \leq \Pr[\exists i, j : S_i = S_j] \leq \frac{(\ell+2)^2}{2^{c+1}}. \tag{16}$$

To bound $\Pr[\text{(B-2)}]$, we need the up leakage assumption. Consider an execution of G_2 with the inputs (IV, M). We divide (C-2) into three subevents:

(1) $\mathsf{BadInner}$: that occurs when there exists an index $i \in [1, \ldots, \ell]$ such that $\big(\star, S_i\big) \in \tau_\pi$ or $\big(C[i]\|\mathsf{ls}_c(S_i), \star\big) \in \tau_\pi$;
(2) $\mathsf{BadInit}$: that occurs when $(S_0, \star) \in \tau_\pi$;
(3) $\mathsf{BadFinal}$: that occurs when $\mathsf{t} = 1$ and $(\star, S_{\ell+1}) \in \tau_\pi$.

Consider $\mathsf{BadInner}$ first. We follow Yu et al.'s argument for an event Query_a with somewhat similar meaning [40, Appendix A]: given an adversary $\mathcal{D}^\pi$, we construct an adversary $\mathcal{A}^\pi$ such that

$$\mathbf{Adv}^{\mathsf{uplsc}[p]}(\mathcal{A}) \leq \Pr_{\mathbf{r},\mathbf{S},\pi}[\mathsf{BadInner} \text{ in } \mathcal{D}^\pi(\mathsf{Ideal}^{\mathsf{t}}(IV, M))]. \tag{17}$$

To this end, $\mathcal{A}^\pi$ runs an instance of $\mathcal{D}$, and keeps τ_π, i.e., the set of $\mathcal{D}$'s queries to π. $\mathcal{A}^\pi$ simulates the following process against $\mathcal{D}$:

(1) $\mathcal{A}^{\pi}$ guesses an index $i \xleftarrow{\$} [1,\ell]$, samples an initial seed $B \xleftarrow{\$} \{0,1\}^n$, sets $S_0 \leftarrow IV \| B$, and initializes a list leak with the leakage traces $[\mathsf{L}^{in}_{\pi}(S_0)]^p$;

(2) For $j = 1, \ldots, i-1$, $\mathcal{A}^{\pi}$ samples the new state $S_j \xleftarrow{\$} \{0,1\}^{r+c}$, computes $C[j] \leftarrow \mathsf{ms}_r(S_j) \oplus M[j]$, and adds the leakages $\left[\mathsf{L}^{out}_{\pi}(S_j), \mathsf{L}^{in}_{\pi}(C[j] \| \mathsf{ls}_c(S_j))\right]^p$, $\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), M[j])$, $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), C[j])]^{p-1}$, and $\mathsf{ms}_2(\mathsf{ls}_c(S_j))$ to leak;

(3) $\mathcal{A}^{\pi}$ samples $Y \xleftarrow{\$} \{0,1\}^r$ and computes $C[i] \leftarrow Y \oplus M[i]$. $\mathcal{A}^{\pi}$ then submits Y and $C[i]$ to its $\mathsf{uplsc}[p]$ challenger and obtains the leakages

$$\mathsf{leak}_{ch} = \left(\mathsf{ms}_2(s_{ch}), \left[\mathsf{L}^{out}_{\pi}(Y \| s_{ch}), \mathsf{L}^{in}_{\pi}(C[i] \| s_{ch})\right]^p\right)$$

for the challenge secret $s_{ch} \in \{0,1\}^c$. $\mathcal{A}^{\pi}$ then adds the leakages $\mathsf{L}_{\oplus}(Y, M[i])$, $[\mathsf{L}_{\oplus}(Y, C[i])]^{p-1}$, and leak_{ch} to leak. This means $Y \| s_{ch}$ is taken as S_i—though $\mathcal{A}^{\pi}$ doesn't know its value.

(4) $\mathcal{A}^{\pi}$ then performs the remaining evaluations: for $j = i+1, \ldots, \ell$, $\mathcal{A}^{\pi}$ samples $S_j \xleftarrow{\$} \{0,1\}^{r+c}$, computes $C[j] \leftarrow \mathsf{ms}_r(S_j) \oplus M[j]$, and adds the leakages $\left[\mathsf{L}^{out}_{\pi}(S_j), \mathsf{L}^{in}_{\pi}(C[j] \| \mathsf{ls}_c(S_j))\right]^p$, $\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), M[j])$, $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), C[j])]^{p-1}$, and $\mathsf{ms}_2(\mathsf{ls}_c(S_j))$ to leak.

(5) Finally, $\mathcal{A}^{\pi}$ samples $S_{\ell+1} \xleftarrow{\$} \{0,1\}^{r+c}$, adds $[\mathsf{L}^{out}_{\pi}(S_{\ell+1})]^p$ to leak, returns $(C[1] \| \ldots \| C[\ell] \| \mathsf{ls}_{2n-1}(S_{\ell+1}), \mathsf{leak})$ to $\mathcal{D}$, and outputs $\{\mathsf{ls}_c(S^{in}), \mathsf{ls}_c(S^{out}) : (S^{in}, S^{out}) \in \tau_{\pi}\}$ as the set $Guesses$.

The strategy of $\mathcal{A}^{\pi}$ is to make a uniform guess on the position of the first inner secret value that appears in τ_{π}, as this value is the "first", its being queried was necessarily due to the corresponding leakages (rather than the compromising of the other inner states). This guess will be correct with probability $1/\ell$. Then, $\mathcal{A}^{\pi}$ simulates $\mathsf{Ideal}^1(IV, M)$ and provides the leakages to $\mathcal{D}$, except for the i index, for which the leakages are replaced by those obtained from an uplsc challenger. Now if the guess on the index i is correct, then all the inputs sent to $\mathcal{D}$ are distributed exactly as those in a normal execution of G_2. Therefore, when $\mathcal{D}$ halts, if $\mathcal{D}$ made a query on s_{ch}, then outputting the aforementioned set $Guesses$ (based on τ_{π}) would break the uplsc game. So we have

$$\Pr[s_{ch} \in Guesses \mid \mathsf{BadInner} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}, \pi)] \geq \frac{1}{\ell}.$$

Now, we observe that

$$\Pr[s_{ch} \in Guesses \mid \mathsf{BadInner} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}, \pi)] \leq \frac{\Pr[s_{ch} \in Guesses]}{\Pr[\mathsf{BadInner} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}, \pi)]}.$$

And it can be seen $\mathcal{A}$ is $(q_{\pi}, t^*, 2q_{\pi})$-bounded, with $t^* = O(t + p\ell t_l)$. By this,

$$\begin{aligned}
\Pr[\mathsf{BadInner} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}, \pi)] &\leq \ell \cdot Pr[s_{ch} \in Guesses] \\
&\leq \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(\mathcal{A}) \quad \text{(Eq. 5)} \\
&\leq \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_{\pi}, t^*, 2q_{\pi}). \quad \text{(Eq. 7)}
\end{aligned}$$

These finish the analysis of BadInner. For the events BadInit and BadFinal, using Eqs. (9) and (10), similar arguments could establish[9]

$$\Pr[\mathsf{BadInit} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}, \pi)] \le \mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, t^*, q_\pi),$$
$$\Pr[\mathsf{BadFinal} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{Ideal}, \pi)] \le \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, t^*, q_\pi).$$

The three terms plus Eq. (16) yield

$$\begin{aligned}\Pr[T_{id} \in \mathcal{T}_{bad}] \le & \frac{(\ell+2)^2}{2^{c+1}} + \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, t^*, 2q_\pi) \\ & + \mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, t^*, q_\pi) + \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, t^*, q_\pi). \end{aligned} \tag{18}$$

Summarizing. When $\mathsf{t} = 1$, for any good transcript $\tau = (\tau_\pi, \mathbf{S}, \mathbf{r})$ we have

$$\begin{aligned}\Pr[T_{id} = \tau] &= \Pr[\pi \vdash \tau_\pi] \cdot \Pr[\mathbf{r}] \cdot \Pr[\mathbf{S}] \\ &= \Pr[\pi \vdash \tau_\pi] \cdot \Pr[\mathbf{r}] \cdot \frac{1}{2^n} \cdot \frac{1}{2^{c(\ell+1)}},\end{aligned}$$

As $\Pr[S_0] = \Pr[B] = \frac{1}{2^n}$, while $\Pr[S_i] = \frac{1}{2^{r+c}}$ for $i = 1, \ldots, \ell+1$. Whereas

$$\Pr[T_{re} = \tau] = \Pr[\pi \vdash \tau_\pi] \cdot \Pr[r] \cdot \frac{1}{2^n} \cdot \Pr[\forall i \in [0, \ell] : \pi(S_i) = S_{i+1} \mid \pi \vdash \tau_\pi].$$

Conditioned on ¬(B-1) and ¬(B-2), it can be seen $\Pr[\forall i \in [0, \ell] : \pi(S_i) = S_{i+1} \mid \pi \vdash \tau_\pi \mid \forall j < i : \pi(S_j) = S_{j+1}] \ge \frac{1}{2^{c(\ell+1)}}$. Therefore,

$$\frac{\Pr[T_{re} = \tau]}{\Pr[T_{id} = \tau]} \ge 1,$$

and thus $\Pr[T_{id} \in \mathcal{T}_{bad}]$ constitutes the final bound, i.e.,

$$\begin{aligned}&\left|\Pr[\mathcal{D}^\pi(\mathsf{Real}_B^1[\pi](IV, M)) \Rightarrow 1] - \Pr[\mathcal{D}^\pi(\mathsf{Ideal}^1(IV, M)) \Rightarrow 1]\right| \\ \le & \frac{(\ell+2)^2}{2^{c+1}} + \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, t^*, 2q_\pi) + \mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, t^*, q_\pi) + \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, t^*, q_\pi).\end{aligned}$$

When $\mathsf{t} = 0$, The event BadFinal relies on the assumption $\mathsf{uplsc}[p]$ instead of $\mathsf{upfi}[p]$. Thus we could drop the term $\mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, t^*, q_\pi)$ and obtain

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \le \frac{(\ell+2)^2}{2^{c+1}} + \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, t^*, 2q_\pi) + \mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, t^*, q_\pi).$$

The remaining is roughly the same, yielding $\Pr[T_{re} = \tau] \ge \Pr[T_{id} = \tau]$, and thus

$$\begin{aligned}&\left|\Pr[\mathcal{D}^\pi(\mathsf{Real}_B^0[\pi](IV, M)) \Rightarrow 1] - \Pr[\mathcal{D}^\pi(\mathsf{Ideal}^0(IV, M)) \Rightarrow 1]\right| \\ \le & \frac{(\ell+2)^2}{2^{c+1}} + \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, t^*, 2q_\pi) + \mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, t^*, q_\pi).\end{aligned}$$

Gathering the two cases yields Eq. (14). □

[9] For BadInit, the constructed $\mathcal{A}$ only outputs q_π guesses $\mathsf{ls}_n(S^{in})$ for $(S^{in}, S^{out}) \in \tau_\pi$; for BadFinal, $\mathcal{A}$ only outputs q_π guesses $\mathsf{ms}_{r+c+1-2n}(S^{out})$ for $(S^{in}, S^{out}) \in \tau_\pi$.

Summarizing: Leakage Eavesdropper security of Real. In the black-box setting, real/ideal indistinguishability suffices for the security. But with (message) leakages, the definition of "ideal" leakages is elusive, which is why we focus on the *leakage (left-or-right) eavesdropper security* setting,[10] and derive the following bound on $\mathsf{Real}[\pi]$. It appears like the bound in Lemma 1 plus the term $\ell\mathbf{Adv}^{\mathsf{LORL}[p]}$, which is due to masking the ℓ message blocks with ℓ independent random key stream blocks.

Theorem 5. *For every pair of messages M^0 and M^1 of equal-length, every (IV, A) such that $\lceil\frac{|A|}{r}\rceil + \lceil\frac{|M^0|}{r}\rceil \le \ell$, and every (q_π, t)-bounded adversary $\mathcal{A}^\pi$, the corresponding leakage eavesdropper advantage, denoted $\mathbf{Adv}^{\mathsf{eavl}[p]}_{\mathsf{Real}^{\mathsf{t}}}(q_\pi, t, \ell)$, is*

$$\begin{aligned}
&\left|\Pr[\mathcal{A}^\pi(\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^\pi(\mathsf{Real}^{\mathsf{t}}_B[\pi](IV, A, M^1)) \Rightarrow 1]\right| \\
\le &\frac{(\ell+2)^2}{2^c} + \ell \cdot \mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, O(t + p\ell t_l)) + 2\mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, O(t + p\ell t_l), q_\pi) \\
&+ 2\ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, O(t + p\ell t_l), 2q_\pi) + 2\mathsf{t} \cdot \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, O(t + p\ell t_l), q_\pi),
\end{aligned}$$

where t_l is as defined in Lemma 1. See appendix E.4 for its proof.

6.5 mu CCAmL security of S1P and S2P

From Theorem 5 we can establish the multi-user CCAmL advantages for S1P and S2P. We naturally define the leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{Enc}}, \mathsf{L}_{\mathsf{Dec}})$ of the AEAD's as follows:

- $\mathsf{L}_{\mathsf{Enc}}$ consists of the leakages that are generated during the encryption:
  - the leakages $\mathsf{L}^{in}_\pi(S^{in})$ and $\mathsf{L}^{out}_\pi(S^{out})$ generated by all the internal calls to $\pi(S^{in}) \to S^{out}$, and
  - the leakages $\mathsf{L}^{in}_{\widetilde{\mathsf{E}}}(K, T; X)$ and $\mathsf{L}^{out}_{\widetilde{\mathsf{E}}}(K, T; Y)$ generated by all the internal calls to $\widetilde{\mathsf{E}}^T_K(X) \to Y$ or $(\widetilde{\mathsf{E}}^T_K)^{-1}(Y) \to X$, 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 $\mathsf{H}[\pi]$ in S2P (i.e., keyless functions are non-protected, and leak everything).
- $\mathsf{L}_{\mathsf{Dec}}$ consists of the above that are generated during the decryption when $\mathsf{AEAD} = \mathsf{S2P}$, while $\mathsf{L}_{\mathsf{Dec}} = \emptyset$ when $\mathsf{AEAD} = \mathsf{S1P}$.

For $\overrightarrow{q} = (q_m, q_e, q_d, q_{\widetilde{\mathsf{IC}}}, q_\pi)$, we denote by $(\overrightarrow{q}, t, \sigma)$-adversaries those make $q_m, q_e, q_d, q_{\widetilde{\mathsf{IC}}}$, and q_π queries to the non-challenge LEnc, the challenge LEnc, Dec (for muCCAmL1) or LDec (for muCCAmL2), $\widetilde{\mathsf{IC}}$, and π resp., run in time t, and have σ blocks in all its (challenge & non-challenge) queries including associated data.

[10] Note that this setting is *weaker* than CPA with leakages, as the distinguisher doesn't have additional plaintext-ciphertext pairs.

Theorem 6. *Assume* $u \leq 2^{n_p}$, $n_p \leq n$, $n \geq 5$, $\sigma + q_e + q_d + q_m + q_\pi \leq \min\left\{2^n/4, 2^{r+c}/2\right\}$, *and* $\mathsf{S1P}$ *leakage* $\mathsf{L} = (\mathsf{L}_{\mathsf{Enc}}, \mathsf{L}_{\mathsf{Dec}})$ *is defined as above. Then in the ideal TBC and permutation model, for any* $(\overrightarrow{q}, t, \sigma)$*-adversary* $\mathcal{A}$*, it holds*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAmL1}}_{\mathcal{A},\mathsf{S1P},\mathsf{L},u} \leq & \frac{5u}{2^{n_p}} + \frac{49(q^*)^2}{2^c} + \frac{6(n+1)q^* + 2nq_d + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \sigma\mathbf{Adv}^{\mathsf{LORL}[1]}(q^*, t^*) \\ & + 2q_e\mathbf{Adv}^{\mathsf{uplsn}[1]}(q^*, t^*, q^*) + 2\sigma\mathbf{Adv}^{\mathsf{uplsc}[1]}(q^*, t^*, 2q^*) \\ & + 2q_e\mathbf{Adv}^{\mathsf{upfi}[1]}(q^*, t^*, q^*), \end{aligned} \tag{19}$$

where $\mathbf{Adv}^{\mathsf{LORL}[1]}$, $\mathbf{Adv}^{\mathsf{uplsn}[1]}$, $\mathbf{Adv}^{\mathsf{uplsc}[1]}$, *and* $\mathbf{Adv}^{\mathsf{upfi}[1]}$ *are defined in Eqs. (13), (9), (7), and (10) resp,* $q^* = \sigma + q_e + q_d + q_m + q_\pi$, $t^* = O(t + \sigma t_l)$, *and* t_l *is the total time for evaluating* L^{in} *and* L^{out}.

Theorem 7. *Assume* $u \leq 2^{n_p}$, $n_p \leq n$, $n \geq 5$, $2\sigma + 2(q_e + q_d + q_m) + q_\pi \leq \min\left\{2^n/4, 2^{r+c}/2\right\}$, *and* $\mathsf{S2P}$ *leakage* $\mathsf{L} = (\mathsf{L}_{\mathsf{Enc}}, \mathsf{L}_{\mathsf{Dec}})$ *is defined as above. Then in the ideal model, for any* $(\overrightarrow{q}, p-1, t, \sigma)$*-adversary* $\mathcal{A}$ *that makes* $p-1$ *queries to the challenge decryption leakage oracle* $\mathsf{L}_{\mathsf{Decch}}$ *besides the* $\overrightarrow{q}$ *queries, it holds*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{S2P},\mathsf{L},u} \leq & \frac{4u}{2^{n_p}} + \frac{49(q^*)^2}{2^c} + \frac{5q^* + 2nq_d + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \sigma\mathbf{Adv}^{\mathsf{LORL}[p]}(q^*, t^*) \\ & + 2q_e\mathbf{Adv}^{\mathsf{uplsn}[p]}(q^*, t^*, q^*) + 2\sigma\mathbf{Adv}^{\mathsf{uplsc}[p]}(q^*, t^*, 2q^*), \end{aligned} \tag{20}$$

where $q^* = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$, $t^* = O(t + p\sigma t_l)$, *and* t_l *is the total time for evaluating* L^{in} *and* L^{out}.

The proofs of the two theorems are very similar, and follow a standard hybrid argument of [34,25]. They rely on the feature that, for each challenge encryption query, since the nonce is used only once during its lifetime, the inner keyed duplex is seeded with an ephemeral key B that is somewhat independent from any other ephemeral key of the other encryption queries. See appendices E.5 and E.6 for the formal presentations.

Looking at the Bounds. We focus on the "non-obvious" terms since they are expected to limit the security. The term $\sigma\mathbf{Adv}^{\mathsf{LORL}[p]}(q^*, t^*)$ corresponds to the reduction to the "minimal" message manipulation, and the factor σ reflects a somewhat unavoidable leakage security loss (as discussed). The terms $2q_e\mathbf{Adv}^{\mathsf{uplsn}[p]}(q^*, t^*, q^*) + 2\sigma \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q^*, t^*, 2q^*) + 2q_e\mathbf{Adv}^{\mathsf{upfi}[p]}(q^*, t^*, q^*)$ capture the hardness of side-channel secret recovery, and they are roughly of some birthday type

$$O\left(q_e \cdot \frac{q_\pi + \sigma + t}{\mu_n \cdot 2^n}\right) + O\left(\sigma \cdot \frac{q_\pi + \sigma + t}{\mu_c \cdot 2^{c-2}}\right) + O\left(q_e \cdot \frac{q_\pi + \sigma + t}{\mu_f \cdot 2^{r+c+1-2n}}\right)$$

for some parameters μ_n, μ_c, and μ_f that depend on the concrete conditions. Yet, it is nowadays a common assumption that with such a small data complexity (2 leakage traces) μ_n, μ_c, and μ_f would be very small [35]. It is worth noting that:

– The birthday-type bounds are essentially tight w.r.t. our assumptions: a collision between the (internal) secret c-bit state values or between two n-bit

initial seed values allows the adversary to obtain more than 2 leakage traces about a single secret value, which is beyond our assumption (security with 2 traces). In all, our assumption Eq. (6), though a bit conservative, tightly results in a birthday-type bound.
– The influence of u the number of users on the security remains quite negligible: once $u \leq 2^{n_p}/5$, the sponges are secure up to the birthday complexity.
– Theorem 7 relies on the side-channel security assumptions against measuring leakages repeatedly, while Theorem 6 only relies on assumptions against a single measure. Yet, this doesn't mean S1P is more leakage secure than S2P, as the latter is (clearly) much more resistance to decryption leakages.

7 Discussion & related work

On reducing to PRP assumptions. An alternative solution to study the security of keyed sponge/duplex constructions is to reduce them to the PRP security of a "Partial-Key" Even-Mansour (PKEM) cipher [1]. As a result, it is natural to ask whether muCCAmL2 could also be reduced to the PRP security of PKEM with leakages. While this direction is in general an interesting open problem, we put forward two initial difficulties. First, the PKEM-based representations in [1] (see Figure 6 in Appendix B) contain a plenty of "imaginary" XORs that do not actually occur in reality. How to model the leakage of these XORs is not directly obvious (and we cannot simply assume that there is no leakage at all when making an hybrid to PKEM). Second, the PKEM cipher is *not re-keyed*, which further complicates the understanding of its leakage security.

Forward security issue. Keyed sponges are *not forward secure* [11,22], in the sense that the complete exposure of an $r + c$ bit state allows recovering the previous iterations. Consequently, the final permutation call must hide its output state to some extent. In S2P, this means the least significant c bits of S_ℓ (see e.g., line 9. in $\mathsf{S2P}[\pi, \widetilde{\mathsf{E}}].\mathsf{Enc}$) should not leak in full, which is in accordance with the uplsc assumption. In S1P this means that the final state $S_{\nu+\ell+1}$ should not leak. This is why we make a conservative assumption: we assume that $U\|V = \mathsf{ls}_{2n-1}(S_{\nu+\ell+1})$ is fully exposed from $\widetilde{\mathsf{E}}$ (see that $\mathsf{Real}^{\mathsf{t}}_B[\pi]$ returns $U\|V$ to the adversary), and the secrecy only relies on the remaining $r + c + 1 - 2n$ bits —as captured by upfi.

Comparison with Mennink's key predication security. Recently, Mennink *key predication security* in keyed sponges as [30] formalized

$$\mathbf{Adv}^{\mathsf{keypre}}(\mathcal{A}) := \Pr\left[Guesses \leftarrow \mathcal{A}^{\pi}, K \xleftarrow{\$} \{0,1\}^k : K \in Guesses\right]. \tag{21}$$

Equation (21) shares some similarities with our Equation (5): it also takes $K \in Guesses$ as the winning condition of $\mathcal{A}$. But the target secret K is picked *after* $\mathcal{A}$ stops in Equation (21), for which $\mathcal{A}$ does not receive any related information. This is clearly in sharp contrast with our definitions, in which $\mathcal{A}$ receives leakages about the target secret. As a result, it seems difficult to rely on Equation (21) in

a leakage setting, since leakages are not independent of the adversary's view. In order to capture side-channel attacks, our definitions therefore include leakages and allow measuring concrete adversary's advantage.

Related works. As mentioned in introduction, ISAP is a sponge-based AE mode of operation with (informal) SCA security arguments. In short, ISAP is an Encrypt-then-MAC composition of a sponge-based stream cipher and a sponge-based Hash-then-MAC. The "leak-free" re-keying functions in ISAP are instantiated with a rate-1 duplex, which makes it purely sponge-based. Compared with (the more comparable) S2P, ISAP does not achieve CIML2 security, which means there may be more possibilities of DPAs on ISAP in practice (see Appendix H for an extra discussion). The same holds in comparison with the 1-pass S1P, which is CIML2 and has the additional benefit of lightweightness.

In a series of papers [7,8,25], Berti et al. and Guo et al. formalized leakage security notions with challenge leakages, and eventually proposed a practical TBC-mode TEDT [6]. Roughly, TEDT is an Encrypt-then-MAC composition of a fresh-rekeying TBC-based stream cipher and a hash-then-TBC instance mentioned in Section 4. Its structure constitutes a starting point for TEDTSponge, and both designs achieve very close security guarantees. Yet, it is commonly believed that permutation-based designs are more efficient than modes based on (tweakable) block ciphers: their circuit sizes are comparable, but the former avoids the cost of frequent key scheduling.[11]

Finally, with respect to the black-box analysis of sponge-based designs, Daemen et al. showed that with a nonce-based KDF, the security bound of a keyed duplex is largely independent of the number of users u [16]. Yet, their duplex still suffers from the mu security degradation terms $u^2/2^k + ut/2^k$ when user keys are sampled with replacement (sampling without replacement reduces it to $ut/2^k$, but this seems only possible within multiple sessions of a single entity). While we could not get rid of these terms without increasing the key size, we did succeed by merely utilizing *public randomness* (which is likely less costly than secret key bits).

References

1. Andreeva, E., Daemen, J., Mennink, B., Assche, G.V.: Security of Keyed Sponge Constructions Using a Modular Proof Approach. In: FSE 2015. pp. 364–384 (2015)
2. Ashur, T., Dunkelman, O., Luykx, A.: Boosting Authenticated Encryption Robustness with Minimal Modifications. In: CRYPTO 2017, Part III. pp. 3–33 (2017)
3. Barwell, G., Martin, D.P., Oswald, E., Stam, M.: Authenticated Encryption in the Face of Protocol and Side Channel Leakage. In: ASIACRYPT 2017, Part I. pp. 693–723 (2017)

[11] Assuming that the circuit size is primarily depending on the domain of the primitives (see [26] for a discussion), we find the same $3n$ for both an (n, n, n)-TBC (when used in a re-keying setting) and a permutation with $r = n$ and $c = 2n$. As such, we expect TEDTSponge to be more efficient than TEDT.

4. Belaïd, S., Grosso, V., Standaert, F.: Masking and leakage-resilient primitives: One, the other(s) or both? Cryptography and Communications 7(1), 163–184 (2015), https://doi.org/10.1007/s12095-014-0113-6
5. Bellare, M., Tackmann, B.: The Multi-user Security of Authenticated Encryption: AES-GCM in TLS 1.3. In: CRYPTO 2016, Part I. pp. 247–276 (2016)
6. Berti, F., Guo, C., Pereira, O., Peters, T., Standaert, F.X.: TEDT, a leakage-resilient AEAD mode for high (physical) security applications. Cryptology ePrint Archive, Report 2019/137 (2019), https://eprint.iacr.org/2019/137
7. Berti, F., Koeune, F., Pereira, O., Peters, T., Standaert, F.: Ciphertext Integrity with Misuse and Leakage: Definition and Efficient Constructions with Symmetric Primitives. In: AsiaCCS 2018. pp. 37–50 (2018)
8. Berti, F., Pereira, O., Peters, T., Standaert, F.: On Leakage-Resilient Authenticated Encryption with Decryption Leakages. IACR Trans. Symmetric Cryptol. 2017(3), 271–293 (2017)
9. Bertoni, G., Daemen, J., Hoffert, S., Peeters, M., Assche, G.V., Keer, R.V.: CAESAR Submission: Keyak v2 (2015), https://keccak.team/obsolete/Keyak-2.0.pdf
10. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: On the Indifferentiability of the Sponge Construction. In: EUROCRYPT 2008. pp. 181–197 (2008)
11. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Sponge-Based Pseudo-Random Number Generators. In: CHES 2010. pp. 33–47 (2010)
12. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In: SAC 2011. pp. 320–337 (2011)
13. Biryukov, A., Perrin, L.: State of the art in lightweight symmetric cryptography. Cryptology ePrint Archive, Report 2017/511 (2017), https://eprint.iacr.org/2017/511
14. Chakraborti, A., Datta, N., Nandi, M., Yasuda, K.: Beetle Family of Lightweight and Secure Authenticated Encryption Ciphers. IACR Trans. Cryptogr. Hardw. Embed. Syst. 2018(2), 218–241 (2018)
15. Chen, S., Steinberger, J.P.: Tight Security Bounds for Key-Alternating Ciphers. In: EUROCRYPT 2014. pp. 327–350 (2014)
16. Daemen, J., Mennink, B., Assche, G.V.: Full-State Keyed Duplex with Built-In Multi-user Support. In: ASIACRYPT 2017, Part II. pp. 606–637 (2017)
17. Dobraunig, C., Eichlseder, M., Mangard, S., Mendel, F., Unterluggauer, T.: ISAP - Towards Side-Channel Secure Authenticated Encryption. IACR Trans. Symmetric Cryptol. 2017(1), 80–105 (2017)
18. Dodis, Y., Pietrzak, K.: Leakage-Resilient Pseudorandom Functions and Side-Channel Attacks on Feistel Networks. In: CRYPTO 2010. pp. 21–40 (2010)
19. Dziembowski, S., Pietrzak, K.: Leakage-Resilient Cryptography. In: FOCS 2008. pp. 293–302 (2008)
20. Faust, S., Pietrzak, K., Schipper, J.: Practical Leakage-Resilient Symmetric Cryptography. In: CHES 2012. pp. 213–232 (2012)
21. Fuller, B., Hamlin, A.: Unifying Leakage Classes: Simulatable Leakage and Pseudoentropy. In: ICITS. pp. 69–86 (2015)
22. Gazi, P., Tessaro, S.: Provably Robust Sponge-Based PRNGs and KDFs. In: EUROCRYPT 2016, Part I. pp. 87–116 (2016)
23. Goudarzi, D., Rivain, M.: How Fast Can Higher-Order Masking Be in Software? In: EUROCRYPT 2017, Part I. pp. 567–597 (2017)
24. Gueron, S., Lindell, Y.: Better Bounds for Block Cipher Modes of Operation via Nonce-Based Key Derivation. In: CCS 2017. pp. 1019–1036 (2017)

25. Guo, C., Pereira, O., Peters, T., Standaert, F.X.: Leakage-Resilient Authenticated Encryption with Misuse in the Leveled Leakage Setting: Definitions, Separation Results, and Constructions. Cryptology ePrint Archive, Report 2018/484 (2018)
26. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON Family of Lightweight Hash Functions. In: CRYPTO 2011. pp. 222–239 (2011)
27. Hazay, C., López-Alt, A., Wee, H., Wichs, D.: Leakage-Resilient Cryptography from Minimal Assumptions. J. Cryptology 29(3), 514–551 (2016)
28. Holenstein, T., Künzler, R., Tessaro, S.: The Equivalence of the Random Oracle Model and the Ideal Cipher Model, Revisited. In: STOC '11. pp. 89–98 (2011)
29. Jovanovic, P., Luykx, A., Mennink, B.: Beyond $2^{c/2}$ Security in Sponge-Based Authenticated Encryption Modes. In: ASIACRYPT 2014, Part I. pp. 85–104 (2014)
30. Mennink, B.: Key Prediction Security of Keyed Sponges. IACR Transactions on Symmetric Cryptology 2018(4), 128–149 (Dec 2018)
31. Mennink, B., Reyhanitabar, R., Vizár, D.: Security of Full-State Keyed Sponge and Duplex: Applications to Authenticated Encryption. pp. 465–489 (2015)
32. Micali, S., Reyzin, L.: Physically Observable Cryptography (Extended Abstract). In: TCC 2004. pp. 278–296 (2004)
33. O'Flynn, C., Chen, Z.D.: Side channel power analysis of an AES-256 bootloader. In: CCECE. pp. 750–755. IEEE (2015)
34. Pereira, O., Standaert, F., Vivek, S.: Leakage-Resilient Authentication and Encryption from Symmetric Cryptographic Primitives. In: CCS 2015. pp. 96–108 (2015)
35. Pietrzak, K.: A Leakage-Resilient Mode of Operation. In: EUROCRYPT 2009. pp. 462–482 (2009)
36. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: EUROCRYPT 2006. pp. 373–390 (2006)
37. Ronen, E., Shamir, A., Weingarten, A., O'Flynn, C.: Iot goes nuclear: Creating a zigbee chain reaction. IEEE Security & Privacy 16(1), 54–62 (2017)
38. Sasaki, Y., Yasuda, K.: How to Incorporate Associated Data in Sponge-Based Authenticated Encryption. In: CT-RSA 2015. pp. 353–370 (2015)
39. Unterluggauer, T., Werner, M., Mangard, S.: Meas: memory encryption and authentication secure against side-channel attacks. Journal of Cryptographic Engineering (Jan 2018), https://doi.org/10.1007/s13389-018-0180-2
40. Yu, Y., Standaert, F., Pereira, O., Yung, M.: Practical Leakage-Resilient Pseudorandom Generators. In: CCS 2010. pp. 141–151 (2010)

Supplementary Material

A TETSponge is not CCAmL2 secure

Following the notational convention of section 3, we could break the CCAmL2 security of TETSponge via the following steps.

(i) First, we fix a nonce N and λ distinct 1-block ciphertexts $c_1, \ldots, c_\lambda$. The parameter λ depends on the relative strength of the implementation.
(ii) Second, we make λ decryption queries $\mathsf{LDec}(N, A, c_1\|T), \ldots, \mathsf{LDec}(N, A, c_\lambda\|T)$ for $A = \bot$ and T arbitrary. This results in the state $S_1 = \pi(N\|PK\|0^*\|B)$, $B = \widetilde{\mathsf{E}}_K^{PK\|0^*}(N\|0^*)$, being derived λ times and the occurrence of the λ XORing $\mathsf{ms}_r(S_1) \oplus c_1, \ldots, \mathsf{ms}_r(S_1) \oplus c_\lambda$. Now a standard DPA allows recovering $\mathsf{ms}_r(S_1)$.

(iii) Then any challenge tuple (N, M_0, M_1) with $M_0[1] = M_1[1]$ is easily distinguished.

The crucial feature of TETSponge that allows this attack is that *processing invalid decryption queries requires to run with secrets*; as secrets are easier to be fixed during decryption, this allows DPAs. Note that all the decryption-leakage resilient designs including TEDTSponge tried to *process invalid decryption queries with keyless/non-secret primitives*.

B Additional Figures

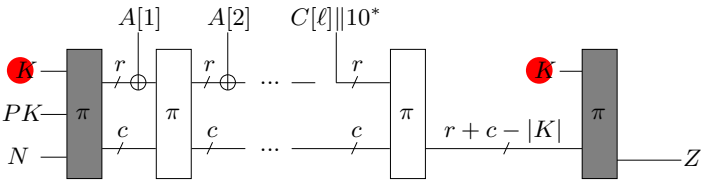

Fig. 5: Key-derivation and tag-generation from calls to π.

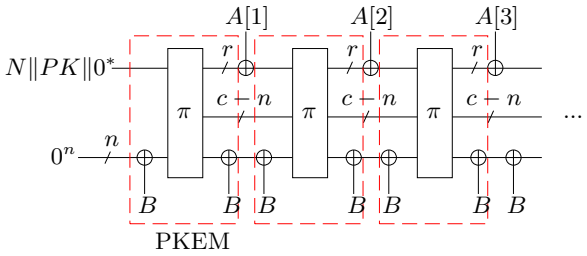

Fig. 6: The PKEM-based representation of a keyed sponge with n-bit key/initial seed. Inside the red dashed rectangles are the "partial-key" Even-Mansour cipher. It's easy to see after the internal actions of XORing B cancel, the construction turns basically the same as the keyed sponge.

C Proof of Theorem 3 (muCIML2 of S1P)

The proof proceeds in two steps:

(i) Below in appendix C.1, we transit the scheme $\mathsf{S1P}[\pi, \widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}}$ to its idealized version, via replacing the internal ideal TBC $\widetilde{\mathsf{IC}}$ by another "secret" ideal TBC $\widetilde{\mathsf{SIC}}$ that isn't accessible to the adversary $\mathcal{A}$. In appendix C.2, we show the real and idealized schemes are indistinguishable. The goal of this step is to argue that $\mathcal{A}$ cannot compromise the KDF- and TGF- calls.

(ii) Then in appendix C.3, we prove unforgability for the idealized scheme to complete the muCIML2 proof.

C.1 Idealizing S1P

Note that we can't simply replace the KDF- and TGF-calls of the u users by u independent tweakable random permutations, as otherwise the birthday term $\frac{u^2}{2^n}$ emerges.

Formally, we are to derive an upper bound on

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S1P}[\pi,\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S1P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u}$$

for any $\overrightarrow{q}$-bounded $\mathcal{D}$. For this, we rely on the H-coefficients technique [15]. We summarize the adversarial queries to the random permutation π in a list τ_π as defined in Eq. (15). Note that by our assumption, all the π queries made by S1P are completely leaked to $\mathcal{D}$. These queries also result in records of the form (S^{in}, S^{out}). To make a distinction, we denote by τ^*_π the union of these leakage records and the adversarial query transcript τ_π. It isn't hard to see:

- upon an encryption query $\mathsf{Enc}_{K_i,PK_i}(N,A,M)$, S1P makes at most $\lceil\frac{|A|}{r}\rceil+\lceil\frac{|M|}{r}\rceil+1$ queries to π;
- upon a decryption query $\mathsf{Dec}_{K_i,PK_i}(N,A,C)$, the number of internal π calls is (similarly) at most $\lceil\frac{|A|}{r}\rceil+\lceil\frac{|\mathbf{c}|}{r}\rceil+1$, with $C=\mathbf{c}\|Z$.

Therefore, when interacting with the idealized scheme $\mathsf{S1P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$, the number of internal π calls is at most $\sigma+q_e+q_d$, and thus

$$q^*_\pi := \left|\tau^*_\pi\right| \le \sigma + q_e + q_d + q_\pi. \tag{22}$$

Recall that the adversarial goal is to distinguish $\widetilde{\mathsf{IC}}_{K_1},\ldots,\widetilde{\mathsf{IC}}_{K_u}$ from $\widetilde{\mathsf{SIC}}_{K_1},\ldots,\widetilde{\mathsf{SIC}}_{K_u}$. In this respect, at the end of the interaction, we reveal all the internal calls to $\widetilde{\mathsf{IC}}$ (in the real world) and $\widetilde{\mathsf{SIC}}$ (in the ideal world) to $\mathcal{D}$. We summarize these calls in a list

$$\tau_{\widetilde{\mathsf{SIC}}} = \big((K_1,T_1,X_1,Y_1),(K_2,T_2,X_2,Y_2),\ldots\big).$$

In this set, the j-th tuple (K_j,T_j,X_j,Y_j) indicates that:

- interacting with the real scheme $\mathsf{S1P}[\pi,\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}}$, the j th query is either $\widetilde{\mathsf{IC}}^{T_j}_{K_j}(X_j)\to Y_j$ or $(\widetilde{\mathsf{IC}}^{T_j}_{K_j})^{-1}(X_j)\to Y_j$; and,
- interacting with the idealized scheme $\mathsf{S1P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$, the j th query is either $\widetilde{\mathsf{SIC}}^{T_j}_{K_j}(X_j)\to Y_j$ or $(\widetilde{\mathsf{SIC}}^{T_j}_{K_j})^{-1}(X_j)\to Y_j$.

Note that:

- these calls and their responses are secret in the black-box setting, but are leaked in our unbound leakage setting.
- yet, since we assume leak-freeness of KDF and TGF-calls, the secret user keys cannot be seen by the distinguisher, and don't appear in the true adversarial transcripts. The transcript $\tau_{\widetilde{\mathsf{SIC}}}$, in some sense, is a merge of the finally revealed secret keys and the information really leaked to $\mathcal{D}$.

Recall that in the unbounded leakage setting, we actually view the duplex as a sponge-based hash function. In this respect, we keep a list τ^*_h for the inputs and outputs of this "imaginary" hash function. Concretely, we denote by $((N,PK,B,A,\mathbf{c}),U\|V)$ an input-output pair of the hash, and further

$$\tau^*_h = \big(((N_1,PK_1,B_1,A_1,\mathbf{c}_1),U_1\|V_1),((N_2,PK_2,B_2,A_2,\mathbf{c}_2),U_2\|V_2),\ldots\big)$$

for the hash transcript. As we assumed all the internal π queries have been leaked and included in τ_π^*, this list is redundant, in the sense that it can be fully recovered from τ_π^*. But its presence eases the proof language.

In addition to the above, the "public-keys" $\mathbf{PK} = (PK_1, \ldots, PK_u)$ are also included in the transcript. Moreover, to simplify the definition of bad transcripts, we reveal to the distinguisher the user keys $\mathbf{K} = (K_1, \ldots, K_u)$ at the end of the interaction. This is wlog since $\mathcal{D}$ is free to ignore this additional information to compute its output bit. Formally, we append both $\mathbf{PK}$ and $\mathbf{K}$ to the tuple $(\tau_h^*, \tau_\pi^*, \tau_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\mathsf{SIC}}})$ and obtain what we call the *transcript*

$$\tau = (\tau_h^*, \tau_\pi^*, \tau_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\mathsf{SIC}}}, \mathbf{PK}, \mathbf{K}).$$

C.2 Gap between Real and Ideal

We start by defining bad transcripts. For a transcript τ, we define μ_{PK} and μ_V, the *maximum multiplicity of PK and V*, as

$$\begin{aligned} \mu_{PK} &:= \max_{pk \in \{0,1\}^{n_p}} \big|\{i \in \{1, \ldots, u\} : PK_i = pk\}\big|, \\ \mu_V &:= \max_{v \in \{0,1\}^{n-1}} \Big|\big\{((N, PK, B, A, \mathbf{c}), U \| V) \in \tau_h^* : V = v\big\}\Big|. \end{aligned} \tag{23}$$

Then it's defined as follows.

Definition 5 (Bad Transcripts for Idealizing $\mathsf{S1P}$, $\mathsf{muCIML2}$). *An attainable transcript τ is bad, if one of the following conditions is fulfilled:*

- *(B-1) $\mu_{PK} \geq n+1$, $\mu_V \geq n+1$.*
- *(B-2) there exists a query $(K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}$ such that $(K, T, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$.*

Otherwise τ is good. Denote by $\mathcal{T}_{bad}$ the set of bad transcripts.

We remark that this step concerns with the secrecy of the user secret keys. As such, the condition (B-2) captures the intuition that a contradiction appears between $\tau_{\widetilde{\mathsf{IC}}}$ and $\tau_{\widetilde{\mathsf{SIC}}}$. On the other hand, though crucial in the analyzes, τ_π^* doesn't appear in the conditions.

As $PK_1, \ldots, PK_u$ are uniformly distributed, it's easy to see

$$\Pr[\mu_{PK} \geq n+1] \leq \binom{u}{n+1} \cdot \frac{1}{(2^{n_p})^n} \leq \Big(\frac{u}{2^{n_p}}\Big)^{n+1} \cdot \frac{2^{n_p}}{(n+1)!} \leq \Big(\frac{u}{2^{n_p}}\Big)^{n+1},$$

where the last inequality comes from $(n+1)! \geq \big(\frac{n+1}{e}\big)^{n+1} \geq 2^{n+1} \geq 2^{n_p}$ since $n+1 \geq 6 > 2e$. Furthermore, when $u \leq 2^{n_p}$ and $n_p \leq n$, we have

$$\Pr[\mu_{PK} \geq n+1] \leq \Big(\frac{u}{2^{n_p}}\Big)^{n+1} \leq \frac{u}{2^{n_p}}. \tag{24}$$

To reason about μ_V, we analyze the multi-semicollision property of the sponge-based hash. In detail, we consider the game G_2 capturing the interaction of $\mathcal{D}$ with the ideal world $(\mathsf{S1P}[\pi, \widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}, \pi, \widetilde{\mathsf{IC}})$. We define several simple bad events during this interaction:

– (B-11) Right after a forward π query $\pi(S^{in}) \to S^{out}$ happens, there exists another π query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{in'})$ or $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{out'})$.
– (B-12) Right after a backward π query $\pi^{-1}(S^{out}) \to S^{in}$ happens,
  - there exists another π query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{in}) = \mathsf{ls}_{c-2}(S^{out'})$; or
  - there exists an $\widetilde{\mathsf{SIC}}$ query/a KDF query $(K, PK_i\|0, N\|0^*, B) \in \tau_{\widetilde{\mathsf{SIC}}}$ such that $S^{in} = N\|PK_i\|0^*\|B$.
– (B-13) At any time, there exists $n+1$ forward π queries $(S_1^{in}, S_1^{out}), \ldots, (S_{n+1}^{in}, S_{n+1}^{out})$ such that $\mathsf{ls}_{n-1}(S_1^{out}) = \ldots = \mathsf{ls}_{n-1}(S_{n+1}^{out})$.
– (B-14) Right after a (necessarily forward) $\widetilde{\mathsf{SIC}}/\mathsf{KDF}$ query $\widetilde{\mathsf{SIC}}_K^{PK\|0^*}(N\|0^*) \to B$ happens, there exists a π query (S^{in}, S^{out}) such that $S^{in} = N\|PK\|0^*\|B$.

Denote by q_1 the number of forward π queries, and by q_2 that of backward π queries. Clearly, $q_1 \le q_\pi^*$, $q_2 \le q_\pi$ (as $\mathsf{S1P}$ doesn't make backward π queries), and $q_1 + q_2 \le q_\pi^*$. With these, consider a forward query $\pi(S^{in}) \to S^{out}$. Its response S^{out} is uniformly distributed in a set of size at least $2^{r+c} - q_\pi^*$. Consider any "target" $(S^{in'}, S^{out'})$. To reach $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{in'})$, S^{out} shall be in a set of size at most 2^{r+2}. Therefore, when $q_\pi^* \le 2^{r+c}/2$, we have

$$\Pr[\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{in'})] \le \frac{2^{r+2}}{2^{r+c} - q_\pi^*} \le \frac{2^{r+3}}{2^{r+c}} = \frac{8}{2^c}.$$

This probability trick will be frequently used in the remaining analysis (without explicitly mentioned). Similarly, $\Pr[\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{out'})] \le \frac{8}{2^c}$. As the number of "targets" is at most q_π^*, we have

$$\Pr[\text{(B-11)}] \le q_1 \cdot q_\pi^* \cdot \left(\frac{8}{2^c} + \frac{8}{2^c}\right) \le \frac{16 q_1 q_\pi^*}{2^c}.$$

In a similar vein, it's easy to see (as argued, $(n+1)! \ge 2^{n+1}$)

$$\Pr[\text{(B-12)}] \le q_2 \cdot q_\pi^* \cdot \frac{8}{2^c} + q_2 \cdot (q_e + q_d) \cdot \frac{2}{2^{r+c}} \le \frac{10 q_2 q_\pi^*}{2^c}, \text{ and}$$
$$\Pr[\text{(B-13)}] \le \binom{q_1}{n+1} \cdot \left(\frac{2}{2^{n-1}}\right)^n \le \left(\frac{8q_\pi^*}{2^n}\right)^{n+1} \cdot \frac{1}{8(n+1)!} \le \left(\frac{4q_\pi^*}{2^n}\right)^{n+1} \cdot \frac{1}{8} \le \frac{q_\pi^*}{2^n}.$$

The last bound relies on $4q_\pi^* \le 2^n$.

For (B-14), we define a set

$$\tau_\pi^*\big[N, PK\big] := \big\{B \in \{0,1\}^n : (N\|PK\|0^*\|B, \star) \in \tau_\pi^*\big\}. \tag{25}$$

Then for a certain $\widetilde{\mathsf{SIC}}$ query $\widetilde{\mathsf{SIC}}_K^{PK\|0^*}(N\|0^*) \to B$, we have

$$\Pr\left[B \in \tau_\pi^*[N, PK]\right] \le \frac{\left|\tau_\pi^*[N, PK_i]\right|}{2^n - 2q_e - 2q_d} \le \frac{2\left|\tau_\pi^*[N, PK_i]\right|}{2^n}.$$

Summing over all the $\widetilde{\mathsf{SIC}}$ queries, we reach

$$\begin{aligned}\Pr[\text{(B-14)}] \le & \sum_{(K,PK\|0^*,N\|0^*,B)\in\tau_{\widetilde{\mathsf{SIC}}}} \frac{2\left|\tau_\pi^*\left[N,PK\right]\right|}{2^n} \\ \le & \sum_{i=1}^{u}\left(\frac{2\sum_{N\in\{0,1\}^{n_N}:(K_i,PK_i\|0^*,N\|0^*,\star)\in\tau_{\widetilde{\mathsf{SIC}}}}\left|\tau_\pi^*\left[N,PK_i\right]\right|}{2^n}\right) \\ \le & \mu_{PK}\cdot\sum_{N\in\{0,1\}^{n_N},PK\in\{0,1\}^{n_p}}\frac{2\left|\tau_\pi^*\left[N,PK\right]\right|}{2^n} \le \frac{2nq_\pi^*}{2^n},\end{aligned}$$

since $\sum_{N\in\{0,1\}^{n_N},PK\in\{0,1\}^{n_p}}\left|\tau_\pi^*\left[N,PK\right]\right| = |\tau_\pi^*| \le q_\pi^*$.

Define $\mathsf{Bad}_1 :=$ (B-11) $\vee$ (B-12) $\vee$ (B-13) $\vee$ (B-14). We now show that $\mu_V \le n$ conditioned on $\neg\mathsf{Bad}_1$, so that (recall that $q_1+q_2 \le q_\pi^*$)

$$\begin{aligned}\Pr[\mu_V \ge n+1] \le \Pr[\mathsf{Bad}_1] \le & \frac{16q_1q_\pi^*}{2^c}+\frac{10q_2q_\pi^*}{2^c}+\frac{q_\pi^*}{2^n}+\frac{2nq_\pi^*}{2^n} \\ \le & \frac{16(q_\pi^*)^2}{2^c}+\frac{(2n+1)q_\pi^*}{2^n}.\end{aligned}$$

For this, we define a notion of "$\mathsf{S1P}$ hash chain" corresponding to a tuple $(N,PK,A,\mathbf{c})$. Formally, this is a sequence of π queries (S_0^{in},S_0^{out}), (S_1^{in},S_1^{out}),..., $(S_\omega^{in},S_\omega^{out})$ such that:

- $S_0^{in} = N\|PK\|0^*\|B$, and
- With $\nu = \lceil |A|/r\rceil$ and $\ell = \lceil |M|/r\rceil$, it holds $\omega = \nu+\ell$, and
  - For $i = 1,\ldots,\nu-1$, $\mathsf{ls}_c(S_i^{in}) = \mathsf{ls}_c(S_{i-1}^{out})$, $\mathsf{ms}_r(S_i^{in}) \oplus \mathsf{ms}_r(S_{i-1}^{out}) = A[i]$;
  - When $|A[\nu]| < r$, $\mathsf{ls}_c(S_\nu^{in}) = \mathsf{ls}_c(S_{\nu-1}^{out})\oplus(0^r\|[1]_2\|0^{c-2})$, $\mathsf{ms}_r(S_\nu^{in})\oplus\mathsf{ms}_r(S_{\nu-1}^{out}) = A[\nu]\|10^*$; when $|A[\nu]| = r$, $\mathsf{ls}_c(S_\nu^{in}) = \mathsf{ls}_c(S_{\nu-1}^{out})$, and $\mathsf{ms}_r(S_\nu^{in}) \oplus \mathsf{ms}_r(S_{\nu-1}^{out}) = A[\nu]$;
  - If $\ell > 1$:
    - $\mathsf{ls}_c(S_{\nu+1}^{in}) = \mathsf{ls}_c(S_\nu^{out}) \oplus (0^r\|[2]_2\|0^{c-2})$, and $\mathsf{ms}_r(S_{\nu+1}^{in}) = C[1]$;
    - For $i = \nu+2,\ldots,\omega-1$, $\mathsf{ls}_c(S_i^{in}) = \mathsf{ls}_c(S_{i-1}^{out})$, $\mathsf{ms}_r(S_i^{in}) = C[i-\nu]$;
    - When $|C[\ell]| < r$, $\mathsf{ls}_c(S_\omega^{in}) = \mathsf{ls}_c(S_{\omega-1}^{out})\oplus(0^r\|[1]_2\|0^{c-2})$, $\mathsf{ms}_r(S_\omega^{in}) = C[\ell]\|10^*$; when $|C[\ell]| = r$, $\mathsf{ls}_c(S_\omega^{in}) = \mathsf{ls}_c(S_{\omega-1}^{out})$, and $\mathsf{ms}_r(S_\omega^{in}) = C[\ell]$.
  - If $\ell = 1$:
    - When $|C[\ell]| < r$, $\mathsf{ls}_c(S_\omega^{in}) = \mathsf{ls}_c(S_{\omega-1}^{out})\oplus(0^r\|[3]_2\|0^{c-2})$, $\mathsf{ms}_r(S_\omega^{in}) = C[\ell]\|10^*$; when $|C[\ell]| = r$, $\mathsf{ls}_c(S_\omega^{in}) = \mathsf{ls}_c(S_{\omega-1}^{out}) \oplus (0^r\|[2]_2\|0^{c-2})$, and $\mathsf{ms}_r(S_\omega^{in}) = C[\ell]$.

Note that conditioned on $\neg$(B-12) and $\neg$(B-14), the first record (S_0^{in},S_0^{out}) was necessarily resulted from a forward π query. Then, by iteratively applying $\neg$(B-12), it can be seen all the queries in such chains are forward.

Then, we show that, conditioned on $\neg\mathsf{Bad}_1$, distinct tuples $(N,PK,A,\mathbf{c})$ and $(N',PK',A',\mathbf{c}')$ necessarily induce distinct $\mathsf{S1P}$ hash chains (S_0^{in},S_0^{out}),..., $(S_\omega^{in},S_\omega^{out})$ and $(S_0^{in'},S_0^{out'})$,..., $(S_\omega^{in'},S_\omega^{out'})$, which further result in distinct "last calls", i.e., $S_\omega^{in} \ne S_\omega^{in'}$. Assume that $\lceil |A|/r\rceil = \nu$, $\lceil |M|/r\rceil = \ell$, $\lceil |A'|/r\rceil = \nu'$, and $\lceil |M'|/r\rceil = \ell'$. As argued, all these queries were due to forward π queries. We then consider several cases as follows.

Case 1: $(N,PK) \ne (N',PK')$. Then $S_0^{in} \ne S_0^{in'}$, i.e., the two chains are distinct from the first π queries. By $\neg$(B-11), we have $S_1^{in} \ne S_1^{in'}$; similarly, iteratively applying $\neg$(B-11) eventually results in the desired result $S_\omega^{in} \ne S_{\omega'}^{in'}$.

Case 2: $(N, PK) = (N', PK')$. This means $(A, \mathbf{c}) \neq (A', \mathbf{c}')$. We define $\overline{X}$ as:

- $\overline{X} = X$ when $|X|$ is a multiple of r, and
- $\overline{X} = X \| 10^*$ otherwise.

Then we have to further consider several subcases.

Subcase 2.1: $\overline{A}\|\overline{\mathbf{c}} \neq \overline{A'}\|\overline{\mathbf{c}'}$. Then it's clear that there exists an index i such that $S_i^{in} \neq S_i^{in'}$. By ¬(B-11), $S_j^{in} \neq S_j^{in'}$ for any $j > i$, and thus $S_\omega^{in} \neq S_\omega^{in'}$.

Subcase 2.2: $\overline{A}\|\overline{\mathbf{c}} = \overline{A'}\|\overline{\mathbf{c}'}$, and $\nu = \nu'$. Since $(A, \mathbf{c}) \neq (A', \mathbf{c}')$, it has to be $|A[\nu]| < r$ or $|C[\ell]| < r$ or $|A'[\nu']| < r$ or $|C'[\ell']| < r$. Now,

- If $|A[\nu]| < r \wedge |A'[\nu]| = r$ or $|A[\nu]| = r \wedge |A'[\nu]| < r$, then $S_\nu^{in'} \neq S_\nu^{in}$ due to the separation constant $[1]_2\|0^{c-2}$. Thus by ¬(B-11), $S_j^{in} \neq S_j^{in'}$ for any $j > \nu$ and further $S_\omega^{in} \neq S_\omega^{in'}$.
- Else, then either $|\mathbf{c}[\ell]| < r \wedge |\mathbf{c}'[\ell]| = r$ or $|\mathbf{c}[\ell]| = r \wedge |\mathbf{c}'[\ell]| < r$ since $(A, \mathbf{c}) \neq (A', \mathbf{c}')$. Then $S_\omega^{in'} \neq S_\omega^{in}$ due to the separation constant $[2]_2\|0^{c-2}$.

Subcase 2.3: $\overline{A}\|\overline{\mathbf{c}} = \overline{A'}\|\overline{\mathbf{c}'}$, and $\nu \neq \nu'$. Wlog assume $\nu > \nu'$: then it has to be $\ell' \geq 1$. Now,

- If $|A[\nu']| < r$, then $S_{\nu'}^{in'} \neq S_{\nu'}^{in}$ since the separation constant $[1]_2\|0^{c-2}$ is only XORed into $S_{\nu'-1}^{out'}$, and thus all the subsequent calls are distinct.
- Else, if $\ell' = 1$, then $S_{\nu'+1}^{in'} \neq S_{\nu'+1}^{in}$ since $S_{\nu'+1}^{in'}$ is obtained by XORing $[3]_2\|0^{c-2}$ with $S_{\nu'}^{out'}$ while $S_{\nu'+1}^{in}$ is obtained by XORing either $[1]_2\|0^{c-2}$ or 0^c (depending on whether $|A[\nu'+1]| < r$).
- Else, i.e., $\ell' > 1$, then $S_{\nu'+1}^{in'} \neq S_{\nu'+1}^{in}$ since $S_{\nu'+1}^{in'}$ is obtained by XORing $[2]_2\|0^{c-2}$ with $S_{\nu'}^{out'}$ while $S_{\nu'+1}^{in}$ is obtained by XORing either $[1]_2\|0^{c-2}$ or 0^c.

By the above, the $|\tau_h^*|$ hash records have $|\tau_h^*|$ distinct forward π queries as their final π queries. Conditioned on ¬(B-13), the number of semi-collisions on V within these final π queries is at most n. Therefore, the claim $\mu_V \leq n$ follows.

Now, conditioned on ¬(B-1), we analyze (B-2). Note that in the ideal world, for any $(K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}$, the key K is from the dummy key-tuple $\mathbf{K}$, and is uniformly distributed. Then, using an auxiliary set

$$\tau_{\widetilde{\mathsf{IC}}}[T] := \left\{K \in \{0,1\}^n : (K, T, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}\right\},$$

it's easy to see

$$\begin{aligned}\Pr[\text{(B-2)}] &\leq \sum_{(K,T,\star,\star)\in\tau_{\widetilde{\mathsf{SIC}}}} \Pr\left[K \in \tau_{\widetilde{\mathsf{IC}}}[T]\right] \\ &\leq \underbrace{\sum_{t\in\{0,1\}^{n-1}:(K,t\|0,\star,\star)\in\tau_{\widetilde{\mathsf{SIC}}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t\|0]\right|}{2^n}}_{C_1} + \underbrace{\sum_{V\in\{0,1\}^{n-1}:(K,V\|1,\star,\star)\in\tau_{\widetilde{\mathsf{SIC}}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[V\|1]\right|}{2^n}}_{C_2}.\end{aligned}$$

By the construction, the $\widetilde{\mathsf{SIC}}$ queries $(K, t\|0, \star, \star)$ are necessarily KDF queries, for which $K = K_i$ and $t\|0 = PK_i\|0^*$ for some user index i. Since $\mu_{PK} \le n$, we have

$$C_1 = \sum_{i=1}^{u} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[PK_i\|0^*]\right|}{2^n} \le \mu_{PK} \cdot \sum_{PK \in \{0,1\}^{n_p}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[PK\|0^*]\right|}{2^n} \le n \cdot \sum_{PK \in \{0,1\}^{n_p}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[PK\|0^*]\right|}{2^n}.$$

On the other hand, for any $\widetilde{\mathsf{SIC}}$ query $(K_i, V\|1, \star, \star)$, i.e., TGF query, there necessarily exists at least one hash record $((N, PK, B, A, \mathbf{c}), U\|V) \in \tau_h^*$ such that $PK = PK_i$. By this,

$$\begin{aligned} C_2 =& \sum_{i=1}^{u} \sum_{V:((\star, PK_i, \star, \star, \star), \star\|V) \in \tau_h^*} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[V\|1]\right|}{2^n} \\ \le& \mu_{PK} \cdot \sum_{PK \in \{0,1\}^{n_p}} \sum_{V:((\star, PK, \star, \star, \star), \star\|V) \in \tau_h^*} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[V\|1]\right|}{2^n} \\ \le& \mu_{PK} \cdot \mu_V \cdot \sum_{V \in \{0,1\}^{n-1}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[V\|1]\right|}{2^n} \;\le\; n^2 \cdot \sum_{V \in \{0,1\}^{n-1}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[V\|1]\right|}{2^n}. \end{aligned}$$

Therefore,

$$\begin{aligned} \Pr[\text{(B-2)} \mid \neg\text{(B-1)}] \le& n \cdot \sum_{PK \in \{0,1\}^{n_p}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[PK\|0^*]\right|}{2^n} + n^2 \cdot \sum_{V \in \{0,1\}^{n-1}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[V\|1]\right|}{2^n} \\ \le& n^2 \cdot \sum_{t \in \{0,1\}^n} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t]\right|}{2^n} \le \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}, \end{aligned}$$

which allows us to conclude

$$\begin{aligned} \Pr[T_{id} \in \mathcal{T}_{bad}] \le& \Pr[\text{(B-1)}] + \Pr[\text{(B-2)} \mid \neg\text{(B-1)}] \\ \le& \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \end{aligned}$$

Now consider a good transcript $\tau = (\tau_h^*, \tau_\pi^*, \tau_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\mathsf{SIC}}}, \mathbf{PK}, \mathbf{K})$. Define

$$\tau_{\widetilde{\mathsf{SIC}}}[K, T] := \left\{(X, Y) \in (\{0,1\}^n)^2 : (K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}\right\}.$$

With this notation, it's clear that

$$\Pr[T_{id} = \tau] = \Pr[\mathbf{K}, \mathbf{PK}] \cdot \Pr[\pi \vdash \tau_\pi^*] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \prod_{(K,T)} \frac{1}{(2^n)_{|\tau_{\widetilde{\mathsf{SIC}}}[K,T]|}}.$$

On the other hand,

$$\begin{aligned} \Pr[T_{re} = \tau] =& \Pr[\mathbf{K}, \mathbf{PK}] \cdot \Pr[\pi \vdash \tau_\pi^*] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{SIC}}} \mid \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \\ =& \Pr[\mathbf{K}, \mathbf{PK}] \cdot \Pr[\pi \vdash \tau_\pi^*] \\ & \cdot \Pr[\widetilde{\mathsf{IC}}_K^T(X) = Y \text{ for all } (K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}} \mid \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}], \end{aligned}$$

Since τ is good,

$$\begin{aligned}&\Pr[\widetilde{\mathsf{IC}}_K^T(X) = Y \text{ for all } (K,T,X,Y) \in \tau_{\widetilde{\mathsf{SIC}}} \mid \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}]\\ =&\Pr[\widetilde{\mathsf{IC}}_K^T(X) = Y \text{ for all } (K,T,X,Y) \in \tau_{\widetilde{\mathsf{SIC}}}] = \prod_{(K,T)} \frac{1}{(2^n)_{|\tau_{\widetilde{\mathsf{SIC}}}[K,T]|}}.\end{aligned}$$

Therefore, for any good transcript τ we have $\Pr[T_{re} = \tau] = \Pr[T_{id} = \tau]$, and thus

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S1P}[\pi,\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S1P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u}\\ \le&\frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \qquad (26)\end{aligned}$$

C.3 Unforgability of the Idealized S1P

The remaining devotes to analyze $\mathsf{S1P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$. Consider the muCIML2 game G_2 with $\mathsf{S1P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$. We define an event CHAIN: at any time, for the i th user there exists a hash record $((N,PK_i,B,A,\mathbf{c}),U\|V)$ and a $\widetilde{\mathsf{SIC}}$ query $(K_i,V^*\|1,U^*,Z)$ (i.e., a TGF relation) such that $U\|V = U^*\|V^*$, while there didn't exist any encryption query of the form $\mathsf{LEnc}(i,N,A,\star) \to \mathbf{c}\|Z$. It's easy to see that, it isn't possible to forge as long as CHAIN doesn't happen.

To ease the analysis, we "break" CHAIN into several simple bad events, then show that CHAIN isn't possible as long as these events didn't occur. Concretely,

- (C-1) There exists two user indexes j,ℓ such that $K_j\|PK_j = K_\ell\|PK_\ell$, or $\mu_{PK} \ge n+1$.
- (C-2) Right after a forward π query $\pi(S^{in}) \to S^{out}$ happens, if:
  - (C-21) there exists another π query $(S^{in'},S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{in'})$, $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{out'})$, or $\mathsf{ls}_{2n-1}(S^{out}) = \mathsf{ls}_{2n-1}(S^{out'})$; or
  - (C-22) there exists a $\widetilde{\mathsf{SIC}}$ query $(K,V\|1,U,Z)$ such that $\mathsf{ls}_{2n-1}(S^{out}) = U\|V$; or
- (C-3) At any time, there exists $n+1$ forward π queries $(S_1^{in},S_1^{out}),\ldots,(S_{n+1}^{in},S_{n+1}^{out})$ such that $\mathsf{ls}_{n-1}(S_1^{out}) = \ldots = \mathsf{ls}_{n-1}(S_{n+1}^{out})$.
- (C-4) Right after a backward π query $\pi^{-1}(S^{out}) \to S^{in}$ happens, if:
  - (C-41) there exists another π query $(S^{in'},S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{in}) = \mathsf{ls}_{c-2}(S^{out'})$, or
  - (C-42) there exists a $\widetilde{\mathsf{SIC}}$ query/a KDF relation $(K,PK\|0^*,N\|0^*,B)$ such that $S^{in} = N\|PK\|0^*\|B$.
- (C-5) Right after a (necessarily forward) KDF query $\widetilde{\mathsf{SIC}}_K^{PK\|0^*}(N\|0^*) \to B$ happens, there exists a π query (S^{in},S^{out}) such that $S^{in} = N\|PK\|0^*\|B$.
- (C-6) Right after an inverse TGF query $(\widetilde{\mathsf{SIC}}_K^{V\|1})^{-1}(Z) \to U$, there exists a π query (S^{in},S^{out}) such that $\mathsf{ls}_{2n-1}(S^{out}) = U\|V$.

Some of the conditions have been analyzed before. First, using $u \le 2^{n_p} \le 2^n$ we have

$$\Pr[\text{(C-1)}] \le \frac{u^2}{2^{n+n_p}} + \frac{u}{2^{n_p}} \le \frac{2u}{2^{n_p}}.$$

Second, (C-21) is the previous (B-11) enhanced with $\mathsf{ls}_{2n-1}(S^{out}) = \mathsf{ls}_{2n-1}(S^{out'})$, thus $\Pr[\text{(C-21)}] \leq \frac{16q_1q_\pi^*}{2^c} + \frac{2(q_\pi^*)^2}{2^{2n-1}}$ (q_1 being the number of forward π queries). And it's easy to see $\Pr[\text{(C-22)}] \leq \frac{2(q_e+q_d)q_\pi^*}{2^{2n-1}}$. Thus (using $q_e + q_d \leq q_\pi^*$)

$$\Pr[\text{(C-2)}] \leq \frac{16q_1q_\pi^*}{2^c} + \frac{4(q_\pi^*)^2 + 4(q_e+q_d)q_\pi^*}{2^{2n}} \leq \frac{16q_1q_\pi^*}{2^c} + \frac{q_\pi^* + q_e + q_d}{2^n} \leq \frac{16q_1q_\pi^*}{2^c} + \frac{2q_\pi^*}{2^n}.$$

The last inequality stems from $q_\pi^* \leq 2^n/4$.

The condition (C-3) is the same as the previous (B-13), thus $\Pr[\text{(C-3)}] \leq \frac{q_\pi^*}{2^n}$. The condition (C-4) is the previous (B-12), thus $\Pr[\text{(C-4)}] \leq \frac{10q_2q_\pi^*}{2^c}$. (C-5) is the previous (B-14), thus

$$\Pr[\text{(C-5)} \mid \neg\text{(C-1)}] \leq \frac{2nq_\pi^*}{2^n}. \tag{27}$$

For (C-6), by ¬(C-2) and ¬(C-3) and an analysis similar to the previous for μ_V, the number of distinct records $((N_1, PK_1, B_1, A_1, \mathbf{c}_1), U_1\|V), ((N_2, PK_2, B_2, A_2, \mathbf{c}_2), U_2\|V), \ldots$ (with the same V) in τ_h^* is at most n. Therefore, for each inverse query $(\widetilde{\mathsf{SIC}}_K^{V\|1})^{-1}(Z) \to U$, there are $\leq n$ "target" U values, and thus

$$\Pr[\text{(C-6)}] \leq \frac{nq_d}{2^n - q_{\widetilde{\mathsf{IC}}}} \leq \frac{2nq_d}{2^n}. \tag{28}$$

Define $\mathsf{Bad} :=$ (C-1) $\vee$ (C-2) $\vee \ldots \vee$ (C-6), then we have

$$\Pr[\mathsf{Bad}] \leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+3)q_\pi^* + 2nq_d}{2^n}.$$

Below we show $\Pr[\mathsf{CHAIN} \mid \neg\mathsf{Bad}] = 0$. Assume otherwise, then consider the last adversarial action before CHAIN happens:

Case 1: $\mathcal{A}$ makes a π query. If this query is forward, then it contradicts ¬(C-2); if this query is backward, then it contradicts ¬(C-4).

Case 2: $\mathcal{A}$ makes an encryption query $\mathsf{LEnc}_{\mathbf{K},\mathbf{PK}}(i, N, A, M)$. Note that KDF calls/$\widetilde{\mathsf{SIC}}$ queries of the form $(K_i, PK_i\|0^*, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}$ have nothing to do with the CHAIN event. Therefore, we further distinguish two subcases:

- Subcase 2.1: a subsequent (forward) π query causes CHAIN event. This again contradicts ¬(C-2).
- Subcase 2.2: the subsequent (new) TGF query $\widetilde{\mathsf{SIC}}_{K_i}^{V\|1}(U) \to Z$ causes CHAIN event. Assume that the involved hash record is $((N, PK_i, B, A, \mathbf{c}), U\|V)$ which corresponds to the i-th user. The assumption means there exists another hash record of the j-th user is $((N', PK_j, B', A', \mathbf{c}'), U'\|V')$ such that $U\|V = U'\|V'$. To reach a contradiction, we distinguish two cases:
  - Subcase 2.3.1: $(N, A, \mathbf{c}) \neq (N', A', \mathbf{c}')$. Then as argued before, conditioned on ¬(C-21), the two involved hash chains are different, and thus $U\|V = U'\|V'$ would contradict ¬(C-21);
  - Subcase 2.3.2: $(N, A, \mathbf{c}) = (N', A', \mathbf{c}')$. Then it has to be $i \neq j$. Now, if $K_i \neq K_j$, then the new $\widetilde{\mathsf{SIC}}$ query $\widetilde{\mathsf{SIC}}_{K_i}^{V\|1}(U) \to Z$ has nothing to do with the j-th user. Otherwise, it holds $PK_i \neq PK_j$ by ¬(C-1), which means the two involved hash chains are different, and thus $U\|V = U'\|V'$ would contradict ¬(C-21).

Case 3: $\mathcal{A}$ makes a decryption query. We further distinguish two subcases:

- Subcase 3.1: a subsequent (forward) π query causes CHAIN event. Then it again contradicts ¬(C-2).
- Subcase 3.2: the subsequent (new) TGF query $(\widetilde{\mathsf{SIC}}^{V\|1}_{K_i})^{-1}(Z) \to U$ causes CHAIN event. This contradicts ¬(C-6).

By the above, we have

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S1P}[\pi,\mathsf{TRPFamily}]_{\mathbf{PK}},\mathsf{L}^*,u} \le \Pr[\mathsf{Bad}] \le \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+3)q_\pi^* + 2nq_d}{2^n}. \quad (29)$$

This plus Eq. (26) yield Eq. (3) (note that $4 < 5 \le n$):

$$\begin{aligned}&\frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+3)q_\pi^* + 2nq_d}{2^n} \\ \le &\frac{3u}{2^{n_p}} + \frac{32(q_\pi^*)^2}{2^c} + \frac{5nq_\pi^* + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \end{aligned} \quad (30)$$

D Proof of Theorem 4 (muCIML2 of S2P)

The two-step proof resembles appendix C.

D.1 Idealizing S2P

We also idealize the scheme S2P first, via replacing the internal calls to $\widetilde{\mathsf{IC}}$ by calls to $\widetilde{\mathsf{SIC}}$. Denote by $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$ the resulted scheme. We then bound

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u}$$

for any $\overrightarrow{q}$-bounded $\mathcal{D}$, using H-coefficients. The transcripts τ_π, $\tau_{\widetilde{\mathsf{IC}}}$, $\tau_{\widetilde{\mathsf{SIC}}}$, and the extension τ_π^* are very similar to those defined in appendix C.1. It's easy to see that, when interacting with the idealized scheme $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$, the number of internal π calls is at most $2\sigma + 2(q_e + q_d)$, which means

$$q_\pi^* := \left|\tau_\pi^*\right| \le 2\sigma + 2(q_e + q_d) + q_\pi. \quad (31)$$

Besides the above, we also keep a (redundant) list for the hash records of $\mathsf{H}[\pi]$, i.e.,

$$\tau_h^* = \big(((A_1, \mathbf{c}_1, N_1, PK_1), U_1\|V_1), ((A_2, \mathbf{c}_2, N_2, PK_2), U_2\|V_2), \dots\big).$$

We also include the public-keys $\mathbf{PK}$ and reveal the secret keys $\mathbf{K}$. These yield

$$\tau = (\tau_h^*, \tau_\pi^*, \tau_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\mathsf{SIC}}}, \mathbf{PK}, \mathbf{K}).$$

For a transcript τ, we define the maximum multiplicity of V as

$$\mu_V := \max_{v \in \{0,1\}^{n-1}} \Big|\big\{((A, \mathbf{c}, N, PK), U\|V) \in \tau_h^* : V = v\big\}\Big|, \quad (32)$$

and μ_{PK} as before. Then bad transcripts are defined as follows.

Definition 6 (Bad Transcripts for Idealizing S2P, muCIML2). *An attainable transcript τ is bad, if one of the following conditions is fulfilled:*

- *(B-1)* $\mu_{PK} \geq n+1$, $\mu_V \geq n+1$.
- *(B-2) there exists a query* $(K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}$ *such that* $(K, T, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$.

They've been analyzed in appendix C.2 except for $\mu_V \geq n+1$. The reasoning about μ_V also relies on the semicollision properties gained before. In detail, we define several bad events during the game G_2 capturing the interaction of $\mathcal{D}$ with the ideal world:

- (B-11) Right after a forward π query $\pi(S^{in}) \to S^{out}$ happens, there exists another π query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{in'})$ or $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{out'})$;
- (B-12) Right after a backward query $\pi^{-1}(S^{out}) \to S^{in}$ happens, it holds $\mathsf{ls}_c(S^{in}) = 0^c$, or there exists another π query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{in}) = \mathsf{ls}_{c-2}(S^{out'})$.
- (B-13) At any time, there exists $n+1$ forward π queries $(S_1^{in}, S_1^{out}), \ldots, (S_{n+1}^{in}, S_{n+1}^{out})$ such that $\mathsf{ls}_{n-1}(S_1^{out}) = \ldots = \mathsf{ls}_{n-1}(S_{n+1}^{out})$.

Note that they are very similar as (more precisely, simpler than) (B-11)-(B-13) in appendix C.2, except that $\Pr[\text{(B-12)}] \leq \frac{2q_2}{2^c} + \frac{8q_2q_\pi^*}{2^c} \leq \frac{10q_2q_\pi^*}{2^c}$. Thus we could adapt the previous $\Pr[\mu_V \geq n+1]$. As before, define $\mathsf{Bad}_1 := \text{(B-11)} \vee \text{(B-12)} \vee \text{(B-13)}$. We show that $\mu_V \leq n$ conditioned on $\neg\mathsf{Bad}_1$, so that

$$\Pr[\mu_V \geq n] \leq \Pr[\mathsf{Bad}_1] \leq \frac{16q_1q_\pi^*}{2^c} + \frac{10q_2q_\pi^*}{2^c} + \frac{q_\pi^*}{2^n} \leq \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^*}{2^n}.$$

This roots at the following observations. First note that conditioned on $\neg\mathsf{Bad}_1$, the permutation queries in hash chains are necessarily all forward, and such chains necessarily result in distinct last block queries. By these, via an analysis similar to appendix C.2, distinct hash inputs $(A, \mathbf{c}, N, PK)$ necessarily result in distinct hash chains of $\mathsf{H}[\pi]$. Therefore, conditioned on $\neg\mathsf{Bad}_1$, there doesn't exist $n+1$ *distinct* hash inputs $(A_1, \mathbf{c}_1, N_1, PK_1), \ldots, (A_{n+1}, \mathbf{c}_{n+1}, N_{n+1}, PK_{n+1})$ that result in the same output $V_1 = \ldots = V_{n+1}$, i.e., it holds $\mu_V \leq n$.

As proved in appendix C.2, $\Pr[\mu_{PK} \geq n+1] \leq \frac{u}{2^{n_p}}$, and $\Pr[\text{(B-2)} \mid \neg\text{(B-1)}] \leq \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}$. Thus

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \leq \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^*}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}.$$

For any good transcript τ, we similarly have $\Pr[T_{re} = \tau] = \Pr[T_{id} = \tau]$, and thus

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u}\\ \leq &\frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^*}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \end{aligned} \tag{33}$$

D.2 Unforgability of the Idealized S2P

Consider the muCIML2 game G_2 with $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$. We define the CHAIN event: at any time, for the i th user there exists a hash record $((A, \mathbf{c}, N, PK_i), U\|V)$ and a $\widetilde{\mathsf{SIC}}$ query $(K_i, V^*\|1, U^*, Z^*)$ (i.e., a TGF query) such that $U = U^*$ and $V = V^*$, yet there didn't exist any encryption query of the form $\mathsf{LEnc}(i, N, A, M) \to \mathbf{c}\|Z^*$. We "break" CHAIN into several simple bad events as follows.

– (C-1) There exists two user indices j, ℓ such that $K_j \| PK_j = K_\ell \| PK_\ell$.
– (C-2) Right after a forward π query $\pi(S^{in}) \to S^{out}$ happens, if:
  • (C-21) there exists another π query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{in'})$, or $\mathsf{ls}_{c-2}(S^{out}) = \mathsf{ls}_{c-2}(S^{out'})$, or $\mathsf{ls}_{2n-1}(S^{out}) = \mathsf{ls}_{2n-1}(S^{out'})$; or
  • (C-22) there exists a TGF query $(K_i, V\|1, U, Z)$ such that $\mathsf{ls}_{2n-1}(S^{out}) = U\|V$.
– (C-3) At any time, there exists $n+1$ forward π queries $(S_1^{in}, S_1^{out}), \ldots, (S_{n+1}^{in}, S_{n+1}^{out})$ such that $\mathsf{ls}_{n-1}(S_1^{out}) = \ldots = \mathsf{ls}_{n-1}(S_{n+1}^{out})$.
– (C-4) Right after a backward π query $\pi^{-1}(S^{out}) \to S^{in}$ happens, if:
  • (C-41) there exists another $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-2}(S^{in}) = \mathsf{ls}_{c-2}(S^{out'})$, or
  • (C-42) $\mathsf{ls}_c(S^{in}) = 0^c$.
– (C-5) Right after an inverse TGF query $(\widetilde{\mathsf{SIC}}_{K_i}^{V\|1})^{-1}(Z) \to U^*$, there exists a hash record $((A, \mathbf{c}, N, PK_i), U\|V) \in \tau_h^*$ such that $U = U^*$.

With q_1 and q_2 denoting the number of forward and backward π queries, it's easy to see

$$\Pr[\text{(C-1)}] \le \frac{u}{2^{n_p}}, \quad \Pr[\text{(C-2)}] \le \frac{16q_1q_\pi^*}{2^c} + \frac{4(q_\pi^*)^2}{2^{2n}},$$
$$\Pr[\text{(C-3)}] \le \frac{q_\pi^*}{2^n}, \quad \Pr[\text{(C-4)}] \le \frac{8q_2q_\pi^*}{2^c} + \frac{2q_2}{2^c} \le \frac{10q_2q_\pi^*}{2^c}.$$

For (C-5), as argued before, conditioned on ¬(C-2), ¬(C-3), and ¬(C-4), the number of distinct hash records $((A_1, \mathbf{c}_1, N_1, PK_i), U_1\|V)$, $((A_2, \mathbf{c}_2, N_2, PK_i), U_2\|V), \ldots$ is at most n. These supply at most n "target values" $U_1, \ldots, U_n$ to each backward TGF query $(\widetilde{\mathsf{SIC}}_{K_i}^{V\|1})^{-1}(Z) \to U^*$. By this,

$$\Pr[\text{(C-5)}] \le \frac{2nq_d}{2^n}.$$

Define $\mathsf{Bad} :=$ (C-1) ∨ (C-2) ∨ (C-3) ∨ (C-4) ∨ (C-5), using $\frac{4(q_\pi^*)^2}{2^{2n}} \le \frac{q_\pi^*}{2^n}$ we have

$$\Pr[\mathsf{Bad}] \le \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{2q_\pi^* + 2nq_d}{2^n}.$$

Below we show $\Pr[\mathsf{CHAIN} \mid \neg\mathsf{Bad}] = 0$. Assume otherwise, then consider the last adversarial action before CHAIN happens:

Case 1: $\mathcal{A}$ makes a π query. If this query is forward, then it contradicts ¬(C-2); if this query is backward, then it contradicts ¬(C-3).

Case 2: $\mathcal{A}$ makes an encryption query $\mathsf{LEnc}_{\mathbf{PK}}(i, N, A, M) \to \mathbf{c}\|Z$. We further distinguish two subcases:

– Subcase 2.1: a subsequent (forward) π query causes CHAIN event. This again contradicts ¬(C-2).
– Subcase 2.2: the subsequent (new) TGF query $\widetilde{\mathsf{SIC}}_{K_i}^{V\|1}(U) \to Z$ causes CHAIN event. This means right before this query happens, there exists a hash record $\big((A', \mathbf{c}', N', PK_j), U'\|V'\big) \in \tau_h^*$ such that $(A', \mathbf{c}', N', PK_j) \ne (A, \mathbf{c}, N, PK_i)$, yet $U'\|V' = U\|V$. To reach a contradiction, we distinguish two cases:

- Subcase 2.3.1: $i \neq j$. Then if $PK_i \neq PK_j$, the two "hash chains" always have a different input blocks at the end, and thus $U\|V = U'\|V'$ contradicts ¬(C-11). If $PK_i = PK_j$, then $K_i \neq K_j$ by ¬(C-1), and thus the new TGF query $\widetilde{\mathsf{SIC}}^{V\|1}_{K_i}(U) \to Z$ has nothing to do with the j-th user.
- Subcase 2.3.2: $i = j$, yet $(N, A, \mathbf{c}) \neq (N', A', \mathbf{c}')$. Then as argued, the two hash chains necessarily have a different input blocks "in the middle", and thus $U\|V = U'\|V'$ contradicts ¬(C-11).

Case 3: $\mathcal{A}$ makes a decryption query. We distinguish two subcases:

– Subcase 3.1: a subsequent (forward) π query causes CHAIN event. Then it again contradicts ¬(C-2).
– Subcase 3.2: the subsequent (new) TGF query $(\widetilde{\mathsf{SIC}}^{V\|1}_{K_i})^{-1}(Z) \to U$ causes CHAIN event. This contradicts ¬(C-5).

By the above, we have

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} \leq \frac{u}{2^{n_p}} + \frac{16(q^*_\pi)^2}{2^c} + \frac{2q^*_\pi + 2nq_d}{2^n}. \tag{34}$$

This plus Eq. (33) yield Eq. (4):

$$\begin{aligned}&\frac{u}{2^{n_p}} + \frac{16(q^*_\pi)^2}{2^c} + \frac{q^*_\pi}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{u}{2^{n_p}} + \frac{16(q^*_\pi)^2}{2^c} + \frac{2q^*_\pi + 2nq_d}{2^n}\\ \leq&\frac{2u}{2^{n_p}} + \frac{32(q^*_\pi)^2}{2^c} + \frac{3q^*_\pi + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}.\end{aligned}$$

E Proofs for Leakage Privacy Lemmas

E.1 Tester for LORL Advantage

As stressed many times, depending on the context, the concrete value of $\mathbf{Adv}^{\mathsf{LORL}[p]}$ may not be negligible.

1: **Tester for LORL $\mathbf{Adv}^{\mathsf{LORL}[p]}$**
2: Let the challenging adversary $\mathcal{A}$ serve s and (m^0, m^1)
3: Pick the secret: $y_{ch} \xleftarrow{\$} \{0,1\}^r$, $b \xleftarrow{\$} \{0,1\}$
4: $c \leftarrow y_{ch} \oplus m^b$, and repeat $y_{ch} \oplus c$ for $p-1$ times
5: Generate the permutation leakages: repeating $S_{pre} \leftarrow \pi^{-1}(y_{ch}\|s)$ for p times
6: Serve $\mathcal{A}$ with c and the leakage traces resulted from steps 4 and 5. This gives $\mathcal{A}$ the tuple $(c, [\mathsf{L}^{out}_\pi(y_{ch}\|s)]^p, \mathsf{L}_\oplus(y_{ch}, m^b), [\mathsf{L}_\oplus(y_{ch}, c)]^{p-1})$
7: Let the challenging adversary $\mathcal{A}$ output the guess b', $\mathcal{A}$ wins as long as $b' = b$

E.2 Complete Description of $\mathsf{Ideal}^{\mathsf{t}}$

Description of $\mathsf{Ideal}^{\mathsf{t}}$:

– $\mathsf{Ideal}^{\mathsf{t}}(IV, A, M)$ proceeds in five steps:
  (1) Samples $B \xleftarrow{\$} \{0,1\}^n$ and initializes an empty list leak for the leakage;
  (2) Computes $S_0 \leftarrow IV\|B$ and samples $S_1 \xleftarrow{\$} \{0,1\}^{r+c}$, and adds the leakage traces $[\mathsf{L}^{in}_\pi(S_0), \mathsf{L}^{out}_\pi(S_1)]^p$ to leak;

(3) For $i = 1, \ldots, \nu = \lceil |A|/r \rceil$, samples $S_{i+1} \xleftarrow{\$} \{0,1\}^{r+c}$, and adds the leakage traces $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_i), A[i]), \mathsf{L}_{\pi}^{in}((A[i]\|0^c) \oplus S_i), \mathsf{L}_{\pi}^{out}(S_{i+1})]^p$ to leak;
(4) For $i = 1, \ldots, \ell = \lceil |M|/r \rceil$, $j = i + \nu$, samples $S_{j+1} \xleftarrow{\$} \{0,1\}^{r+c}$, computes the ciphertext $C[i] \leftarrow \mathsf{ms}_r(S_j) \oplus M[i]$, and adds the leakages $\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), M[i])$, $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), C[i])]^{p-1}$, $[\mathsf{L}_{\pi}^{in}((M[i]\|0^c) \oplus S_j), \mathsf{L}_{\pi}^{out}(S_{j+1})]^p$ to leak;
(5) If the tag $t = 1$ then defines $\mathbf{c} = C[1]\| \ldots \|C[\ell]\|\mathsf{ls}_{2n-1}(S_{j+1})$, else defines $\mathbf{c} = C[1]\| \ldots \|C[\ell]$.

$\mathsf{Ideal}^{\mathsf{t}}(IV, A, M)$ eventually returns $\mathbf{c}$.

E.3 A Useful Lemma: Leakage Eavesdropper Security of the Ideal Stream Cipher

The proof of Theorem 5 will rely on the leakage eavesdropper security of the ideal stream cipher $(\mathsf{Ideal}, \mathsf{L}_{\mathsf{Ideal}})$, which is related to the term $\mathbf{Adv}^{\mathsf{LORL}[p]}$ in Eq. (11). Formally,

Lemma 3. *For every pair of ℓ-block messages M^0 and M^1 and (q_π, t)-bounded adversary $\mathcal{A}^\pi$, it holds*

$$\begin{aligned}&\left|\Pr[\mathcal{A}^\pi(\mathsf{Ideal}_B^{\mathsf{t}}(IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^\pi(\mathsf{Ideal}_B^{\mathsf{t}}(IV, A, M^1)) \Rightarrow 1]\right| \\ \leq &\ell \cdot \mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, O(t + p\ell t_l)),\end{aligned}$$

where t_l is as defined in Lemma 1. See appendix E.3 for its proof.

Proof. Again we assume $|A| = 0$ for simplicity. Let $M^0 = M^0[1]\| \ldots \|M^0[\ell]$ and $M^1 = M^1[1]\| \ldots \|M^1[\ell]$. 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 till obtaining M^1. That is, $M_{h,i} := M^1[1]\| \ldots \|M^1[i]\|M^0[i+1]\| \ldots \|M^0[\ell]$. For any i, assuming a (q_π, t)-bounded adversary $\mathcal{A}^\pi$ against $\mathsf{Ideal}^{\mathsf{t}}(IV, M_{h,i-1})$ and $\mathsf{Ideal}^{\mathsf{t}}(IV, M_{h,i})$, we build a $(q_\pi, O(t + \ell p t_l))$-bounded adversary $\mathcal{A}_2^\pi$ against the distribution defined in Eq. (11). In detail, $\mathcal{A}_2^\pi$ proceeds in four steps:

(1) $\mathcal{A}_2^\pi$ samples $B \xleftarrow{\$} \{0,1\}^n$, initializes an empty list leak, and sets $S_0 \leftarrow IV\|B$;
(2) for $j = 1, \ldots, i-1$, $\mathcal{A}_2^\pi$ samples $S_j \xleftarrow{\$} \{0,1\}^{r+c}$, computes $C[j] \leftarrow \mathsf{ms}_r(S_j) \oplus M^1[j]$, and adds $[\mathsf{L}_{\pi}^{in}(S_{j-1}), \mathsf{L}_{\pi}^{out}(S_j)]^p$, $\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), M^1[j])$, and $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), C[j])]^{p-1}$ to leak;
(3) $\mathcal{A}_2^\pi$ samples $C \xleftarrow{\$} \{0,1\}^c$ and submits C to the $\mathsf{LORL}[p]$ challenger. Assume that the outputs are (c^b, leak_b) with

$$\mathsf{leak}_b = \left([\mathsf{L}_{\pi}^{out}(y_{ch}\|C)]^p, \mathsf{L}_{\oplus}(y_{ch}, m^b), [\mathsf{L}_{\oplus}(y_{ch}, c^b)]^{p-1}\right).$$

$\mathcal{A}_2^\pi$ then adds the traces $[\mathsf{L}_{\pi}^{in}((M^1[i-1]\|0^c) \oplus S_{i-1}), \mathsf{L}_{\pi}^{out}(y_{ch}\|C)]^p$, $\mathsf{L}_{\oplus}(y_{ch}, m^b)$, and $[\mathsf{L}_{\oplus}(y_{ch}, c^b)]^{p-1}$ to leak;
(4) $\mathcal{A}_2^\pi$ starts from $c^b\|C$ to emulate the remaining actions of $\mathsf{Ideal}^{\mathsf{t}}$ encrypting the tail $M^0[i+1]\| \ldots \|M^0[\ell]$ to obtain $C[i+1]\| \ldots \|C[\ell]$. Eventually, $\mathcal{A}_2^\pi$ serves the ciphertext $C[1]\| \ldots \|C[i-1]\|c^b\|C[i+1]\| \ldots \|C[\ell]$ (and $\mathsf{ls}_c(S_{\nu+\ell+1})$, when $\mathsf{t} = 1$) as well as all the generated simulated leakages to $\mathcal{A}^\pi$, and outputs whatever $\mathcal{A}^\pi$ outputs.

It can be seen depending on whether the input tuple received by $\mathcal{A}_2^\pi$ captures the $\mathsf{LORL}[p]$ challenger encrypting $M^0[i]$ or $M^1[i]$, the inputs to $\mathcal{A}^\pi$ capture Ideal encrypting $M_{h,i-1}$ or $M_{h,i}$. Moreover, $\mathcal{A}_2^\pi$ is $(q_\pi, O(t+\ell p t_l))$-bounded if $\mathcal{A}^\pi$ is (q_π, t)-bounded. Therefore,

$$\begin{aligned}&\big|\Pr[\mathcal{A}^\pi(\mathsf{Ideal}^{\mathsf{t}}(IV, M_{h,i-1})) \Rightarrow 1] - \Pr[\mathcal{A}^\pi(\mathsf{Ideal}^{\mathsf{t}}(IV, M_{h,i})) \Rightarrow 1]\big| \\ \leq &\mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, O(t+\ell p t_l))\end{aligned}$$

by Eq. (13). This along with a simple summation implies the main claim. □

E.4 Proof of Theorem 5 (Leakage Eavesdropper Security of Duplex)

$$\begin{aligned}&\big|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Real}_B^{\mathsf{t}}[\pi](IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Real}_B^{\mathsf{t}}[\pi](IV, A, M^1)) \Rightarrow 1]\big| \\ \leq &\underbrace{\big|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Ideal}_B^{\mathsf{t}}(IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Ideal}_B^{\mathsf{t}}(IV, A, M^1)) \Rightarrow 1]\big|}_{\leq \ell \cdot \mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, O(t+\ell p t_l)) \text{ (by Lemma 3)}} \\ &+ \sum_{b=0,1} \big|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Real}_B^{\mathsf{t}}[\pi](IV, A, M^b)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Ideal}_B^{\mathsf{t}}(IV, A, M^b)) \Rightarrow 1]\big|.\end{aligned}$$

For each b, Lemma 1 indicates

$$\begin{aligned}&\big|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Real}_B^{\mathsf{t}}[\pi](IV, A, M^b)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Ideal}_B^{\mathsf{t}}(IV, A, M^b)) \Rightarrow 1]\big| \\ \leq &\mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, O(t+p\ell t_l), q_\pi) + \ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, O(t+p\ell t_l), 2q_\pi) + \frac{(\ell+2)^2}{2^{c+1}} \\ &+ \mathsf{t} \cdot \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, O(t+p\ell t_l), q_\pi).\end{aligned}$$

Therefore,

$$\begin{aligned}&\big|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Real}_B^{\mathsf{t}}[\pi](IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{Real}_B^{\mathsf{t}}[\pi](IV, A, M^1)) \Rightarrow 1]\big| \\ \leq &\frac{(\ell+2)^2}{2^c} + \ell \cdot \mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi, O(t+\ell p t_l)) + 2\mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi, O(t+p\ell t_l), q_\pi) \\ &+ 2\ell \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi, O(t+p\ell t_l), 2q_\pi) + 2\mathsf{t} \cdot \mathbf{Adv}^{\mathsf{upfi}[p]}(q_\pi, O(t+p\ell t_l), q_\pi).\end{aligned}$$

E.5 Proof of Theorem 6 (muCCAmL1 of S1P)

As mentioned, the proof is built upon Theorem 5. We start by defining G_0 as the game $\mathsf{PrivK}_{\mathcal{A},\mathsf{S1P},\mathsf{L}}^{\mathsf{muCCAmL1},0}$, and G_0^* as the game $\mathsf{PrivK}_{\mathcal{A},\mathsf{S1P},\mathsf{L}}^{\mathsf{muCCAmL1},1}$. We say a decryption query $\mathsf{Dec}_{\mathbf{K},\mathbf{PK}}(i, N, A, C)$ is *trivial* if the action $\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i, N, A, \star) \to C$ happens before.

We then define two games G_1 and G_1^*: G_1, resp. G_1^*, is obtained from G_0, resp. G_0^*, via replacing the internal $\widetilde{\mathsf{IC}}$-calls by $\widetilde{\mathsf{SIC}}$-calls. By Eq. (26), we have

$$\big|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_0 \Rightarrow 1]\big| \leq \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}, \tag{35}$$

and (similarly)

$$\big|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_0^* \Rightarrow 1]\big| \leq \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}, \tag{36}$$

where $q_\pi^* = \sigma + q_e + q_d + q_m + q_\pi$.

We then prove

$$\begin{aligned}&\big|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_1^* \Rightarrow 1]\big| \\ \leq &\frac{3u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+4)q_\pi^* + 2nq_d}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(q_\pi^*, t^*, \ell_i), \end{aligned} \tag{37}$$

where ℓ_i is the number of blocks in the ith challenge message, and $t^* = O(t + \sigma t_l)$ for t_l defined in Lemma 1. By Theorem 5, the last term is bounded as

$$\begin{aligned}\textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(q_\pi^*, t^*, \ell_i) \leq &\sigma \mathbf{Adv}^{\mathsf{LORL}[1]}(q_\pi^*, t^*) + 2\sigma \mathbf{Adv}^{\mathsf{uplsc}[1]}(q_\pi^*, t^*, 2q_\pi^*) \\ &+ 2q_e \mathbf{Adv}^{\mathsf{uplsn}[1]}(q_\pi^*, t^*, q_\pi^*) + \underbrace{\textstyle\sum_{i=1}^{q_e} \frac{(\ell_i+2)^2}{2^c}}_{\leq \frac{\left(\sum_{i=1}^{q_e}(\ell_i+2)\right)^2}{2^c} \leq \frac{(q_\pi^*)^2}{2^c}} \\ &+ 2q_e \mathbf{Adv}^{\mathsf{upfi}[1]}(q_\pi^*, t^*, q_\pi^*).\end{aligned}$$

The above plus the gaps in Eq. (35) and Eq. (36) yield the claim. To this end, we denote the q_e challenge tuples by

$$(i_1, N_1, A_1, M_1^0, M_1^1), \ldots, (i_{q_e}, N_{q_e}, A_{q_e}, M_{q_e}^0, M_{q_e}^1).$$

Then, we use q_e hops to replace $M_1^0, \ldots, M_{q_e}^0$ by $M_1^1, \ldots, M_{q_e}^1$ 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

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

such that in the j-th system $\mathsf{G}_{2,j}$, the first j messages processed by the challenge encryption oracle are $M_1^0, \ldots, M_j^0$, while the remaining $q_e - j$ messages being processed are $M_{j+1}^1, \ldots, M_{q_e}^1$. In this vein, we have $\mathsf{G}_{2,q_e} = \mathsf{G}_1^*$.

We then show that for $j = 1, \ldots, q_e$, $\mathsf{G}_{2,j-1}$ and $\mathsf{G}_{2,j}$ are indistinguishable in the view of $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$. For this, from $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$ we build an adversary $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$, such that $|\Pr[\mathsf{G}_{2,i-1} \Rightarrow 1] - \Pr[\mathsf{G}_{2,i} \Rightarrow 1]|$ is related to $\mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}})$.

In detail, initially, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ samples two key vectors $\mathbf{K} = (K_1, \ldots, K_u)$ and $\mathbf{PK} = (PK_1, \ldots, PK_u)$ for the secret and public keys, and keeps a table SICTable to simulate the secret ideal TBC $\widetilde{\mathsf{SIC}}$ via lazy sampling. At this stage, we define a bad event $\mathsf{BadUserKey}$, which occurs if there exits two user indices ℓ_1, ℓ_2 such that $K_{\ell_1} \| PK_{\ell_1} = K_{\ell_2} \| PK_{\ell_2}$.

Assume that entries in the tables are of the form $\text{SICTable}(K, T, X) = Y$ and $\text{SICTable}^{-1}(K, T, Y) = X$. $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ runs $\mathcal{A}$: upon each query from $\mathcal{A}$, it reacts as follows.

Upon a query to $\widetilde{\mathsf{IC}}$ or π, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simply relays the query to its corresponding oracle and relays the response.

Upon a (non-challenge) encryption query (i^*, N^*, A^*, M^*), $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ distinguishes two cases:

– If $(K_{i^*}, PK_{i^*} \| 0^*, N^* \| 0^*) \notin \text{SICTable}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ samples an initial key B^* such that $(K_{i^*}, PK_{i^*} \| 0^*, B^*) \notin \text{SICTable}^{-1}$, defines $\text{SICTable}(K_{i^*}, PK_{i^*} \| 0^*, N^* \| 0^*) \leftarrow$

B^* and $\text{SICTable}^{-1}(K_{i^*}, PK_{i^*}\|0^*, B^*) \leftarrow N^*\|0^*$, and then runs the encryption $\mathsf{Real}^1_{B^*}[\pi](N^*\|PK_{i^*}\|0^*, A^*, M^*)$ to get the ciphertext $\mathbf{c}^*\|U^*\|V^*$ and leakages. $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ then computes $Z^* \leftarrow \text{SICTable}(K_{i^*}, V^*\|1, U^*)$ ($\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ defines the entry $\text{SICTable}(K_{i^*}, V^*\|1, U^*)$ to a newly sampled value Z^* if $(K_{i^*}, V^*\|1, U^*) \notin$ SICTable). For this entire process $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ has to make $\ell_i^* + 1$ queries to π and cost $O(\ell_i t_l)$ time. Finally, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ returns the outputs $\mathbf{c}^*\|Z^*$ and the leakages to $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$;

– If $(K_{i^*}, PK_{i^*}\|0^*, N^*\|0^*) \in$ SICTable, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simply runs the encryption process $\mathsf{Real}^1_{B^*}[\pi](N^*\|PK_{i^*}\|0^*, A^*, M^*)$ with $B^* = \text{SICTable}(K_{i^*}, PK_{i^*}\|0^*, N^*\|0^*)$, computes $Z^* \leftarrow \text{SICTable}(K_{i^*}, V^*\|1, U^*)$ on the obtained U^* and V^*, and returns $\mathbf{c}^*\|Z^*$ and the leakages to $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$. The cost is similar to the above case.

Upon a non-trivial decryption query (i, N, A, C), $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simply simulates the evaluation of $\mathsf{Dec}_{\mathbf{K},\mathbf{PK}}(i, N, A, C)$ and returns the result to $\mathcal{A}$. This requires $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ to make $\ell_i^* + 1$ queries to π. And if $\mathsf{Dec}_{\mathbf{K},\mathbf{PK}}(i, N, A, C) \neq \perp$, then we say another bad event BadCheck occurs.

Upon the ℓ-th challenge tuple $(i_j, N_j, A_j, M_j^0, M_j^1)$, it can be seen that, since BadUserKey didn't happen, it necessarily be $(K_{i_j}, PK_{i_j}\|0^*, N_j\|0^*) \notin$ SICTable by the challenge nonce-respecting restriction on $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$ on a single user. Therefore, depending on ℓ, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ reacts as follows:

– When $\ell < j$, it encrypts M_ℓ^0 and returns. In detail, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ samples B_ℓ, defines the entries $\text{SICTable}(K_{i_\ell}, PK_{i_\ell}\|0^*, N_\ell\|0^*) \leftarrow B_\ell$ and $\text{SICTable}^{-1}(K_{i_\ell}, PK_{i_\ell}\|0^*, B_\ell) \leftarrow N_\ell\|0^*$, and then runs $\mathsf{Real}^1_{B_\ell}[\pi](M_\ell^0) \rightarrow \mathbf{c}_\ell\|U_\ell\|V_\ell$, generates the tag Z_ℓ accordingly and returns $\mathbf{c}_\ell\|Z_\ell$ and the leakages to $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$. The cost is similar to the non-challenge encryption queries.
– When $\ell = j$, it relays M_ℓ^0 and M_ℓ^1 to the eavesdropper $\mathsf{eavl}[1]$ challenger of Real^1 to obtain $\mathbf{c}_\ell^b\|U_\ell\|V_\ell$ and leakages leak_{enc}, then generates the tag Z_ℓ accordingly and returns $\mathbf{c}_\ell^b\|Z_\ell$ to $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$. This means the relation $\text{SICTable}(K_{i_\ell}, PK_{i_\ell}\|0^*, N_\ell\|0^*) = B_{ch}$ is implicitly fixed, where B_{ch} is the secret n-bit initial seed picked inside the eavesdropper challenger. In this respect, we define an additional bad event BadInitKey, which happens if the entry $\text{SICTable}^{-1}(K_{i_\ell}, PK_{i_\ell}\|0^*, B_{ch})$ is defined before $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ terminates (i.e., a contradiction in the table SICTable occurs due to a collision within the initial keys).
– When $\ell > j$, it simply encrypts M_ℓ^1 and returns. The details are similar to the described case $\ell < j$.

Define $\mathsf{Bad} := \mathsf{BadUserKey} \vee \mathsf{BadInitKey} \vee \mathsf{BadCheck}$. It can be seen that as long as Bad never occurs, the whole process is the same as either $\mathsf{G}_{2,j-1}$ or $\mathsf{G}_{2,j}$ depending on whether $b = 0$ or 1. Clearly, $\Pr[\mathsf{BadUserKey}] \leq \frac{u^2}{2^{n+n_p}} \leq \frac{u}{2^{n_p}}$, while $\Pr[\mathsf{BadInitKey}] \leq \frac{q_e+q_d+q_m}{2^n} \leq \frac{q_\pi^*}{2^n}$ as $B_{ch} \xleftarrow{\$} \{0,1\}^n$ inside the challenger. For BadCheck we appeal to the intermediate results established in appendix C.3: in detail, Eq. (29) implies

$$\Pr[\mathsf{BadCheck}] \leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+3)q_\pi^* + 2nq_d}{2^n},$$

where $q_\pi^* = \sigma + q_e + q_d + q_m + q_\pi$.

By the remarks before, besides running $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ internally processes $q_m + q_e + q_d - 1$ queries (except for the query encrypted by its challenger). Therefore, for $\mathsf{G}_{2,j}$ and $\mathsf{G}_{2,j-1}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ makes at most $\sigma + q_m + q_e + q_d + q_\pi = q_\pi^*$ queries to π and spends $t^* = O(t + \sigma t_l)$ running time (the additional time is mainly spent on evaluating the leakage functions). By all the above, we have

$$\begin{aligned}&\Pr[\mathsf{G}_{2,j} \Rightarrow 1] - \Pr[\mathsf{G}_{2,j-1} \Rightarrow 1]\\ \le&\Pr[\mathsf{G}_{2,j} \Rightarrow 1 \wedge \mathsf{Bad} \text{ in } \mathsf{G}_{2,j}] - \Pr[\mathsf{G}_{2,j-1} \Rightarrow 1 \wedge \mathsf{Bad} \text{ in } \mathsf{G}_{2,j-1}] + \mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(q_\pi^*, t^*, \ell_i).\end{aligned}$$

Finally, wlog assume that when the event Bad happens during the interaction, $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$ could be aware and outputs 1; moreover, $\Pr[\mathsf{G}_{2,q_e} \Rightarrow 1] \ge \Pr[\mathsf{G}_{2,0} \Rightarrow 1]$. Then,

$$\begin{aligned}&\left|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_1 \Rightarrow 1]\right| \le \Pr[\mathsf{G}_{2,q_e} \Rightarrow 1] - \Pr[\mathsf{G}_{2,0} \Rightarrow 1]\\ \le&\underbrace{\Pr[\mathsf{G}_{2,q_e} \Rightarrow 1 \wedge \mathsf{Bad} \text{ in } \mathsf{G}_{2,q_e}] - \Pr[\mathsf{G}_{2,0} \Rightarrow 1 \wedge \mathsf{Bad} \text{ in } \mathsf{G}_{2,0}]}_{\le \Pr[\mathsf{G}_{2,q_e} \Rightarrow 1 \wedge \mathsf{Bad} \text{ in } \mathsf{G}_{2,q_e}]}\\ &+ \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(q_\pi^*, t^*, \ell_i)\\ \le&\frac{3u}{2^{n_p}} + \frac{q_\pi^*}{2^n} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+3)q_\pi^* + 2nq_d}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(q_\pi^*, t^*, \ell_i)\\ \le&\frac{3u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+4)q_\pi^* + 2nq_d}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[1]}_{\mathsf{Real}^1}(q_\pi^*, t^*, \ell_i),\end{aligned}$$

which is the claim in Eq. (37).

E.6 Proof of Theorem 7 (muCCAmL2 of S2P)

It resembles appendix E.5: the main modification is to add treatments for decryption leakage (as here we consider $\mathsf{muCCAmL2}$ rather than $\mathsf{muCCAmL1}$). Recall that a decryption query $\mathsf{Dec}_{\mathbf{K}}(i, N, A, C)$ is *trivial* if the action $\mathsf{Enc}_{\mathbf{K}}(i, N, A, M) \to C$ happens before. Although such trivial decryption queries are typically useless in the non-leaking setting, they may serve new information here, and thus require explicit considerations.

Concretely, we start by defining G_0 as the game $\mathsf{PrivK}^{\mathsf{muCCAmL2},0}_{\mathcal{A},\mathsf{S2P},\mathsf{L}}$ and G_0^* as the game $\mathsf{PrivK}^{\mathsf{muCCAmL2},1}_{\mathcal{A},\mathsf{S2P},\mathsf{L}}$.

We then replace the internal $\widetilde{\mathsf{IC}}$-calls by $\widetilde{\mathsf{SIC}}$-calls: this modifies G_0 to G_1 and G_0^* to G_1^*. By Eq. (33), we have $|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_0 \Rightarrow 1]| \le \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^*}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}$ and $|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_0^* \Rightarrow 1]| \le \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^*}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}$, where $q_\pi^* = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$. We then prove

$$\begin{aligned}&\left|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_1^* \Rightarrow 1]\right|\\ \le&\frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{3q_\pi^* + 2nq_d}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[p]}_{\mathsf{Real}^0}(q_\pi^*, t^*, \ell_i),\end{aligned} \tag{38}$$

where ℓ_i is the number of blocks in the ith challenge message, and $t^* = O(t + p\sigma t_l)$ for t_l defined in Lemma 1. By Theorem 5, the last term is bounded as

$$\begin{aligned}\textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[p]}_{\mathsf{Real}^0}(q_\pi^*, t^*, \ell_i) \le&\sigma \cdot \mathbf{Adv}^{\mathsf{LORL}[p]}(q_\pi^*, t^*) + 2q_e \mathbf{Adv}^{\mathsf{uplsn}[p]}(q_\pi^*, t^*, q_\pi^*)\\ &+ 2\sigma \cdot \mathbf{Adv}^{\mathsf{uplsc}[p]}(q_\pi^*, t^*, 2q_\pi^*) + \underbrace{\textstyle\sum_{i=1}^{q_e} \frac{(\ell_i+2)^2}{2^c}}_{\le \frac{(q_\pi^*)^2}{2^c}}.\end{aligned}$$

The above plus the gaps between G_0, G_1, G_0^*, and G_1^* yield the claim. The hybrid argument basically follows the same line as appendix E.5: we denote the q_e challenge tuples by $(i_1, N_1, A_1, M_1^0, M_1^1), \ldots, (i_{q_e}, N_{q_e}, A_{q_e}, M_{q_e}^0, M_{q_e}^1)$, and use q_e hops to replace $M_1^0, \ldots, M_{q_e}^0$ by $M_1^1, \ldots, M_{q_e}^1$ in turn. Consider the game $\mathsf{G}_{2,j}$ involved in the j-th hop: the first j messages processed by the challenge encryption oracle are $M_1^0, \ldots, M_j^0$, while the remaining $q_e - j$ messages being processed are $M_{j+1}^1, \ldots, M_{q_e}^1$. To bound the gap between $\mathsf{G}_{2,j-1}$ and $\mathsf{G}_{2,j}$ in the view of $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$, we build an adversary $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ such that $|\Pr[\mathsf{G}_{2,i-1} \Rightarrow 1] - \Pr[\mathsf{G}_{2,i} \Rightarrow 1]|$ is related to $\mathbf{Adv}^{\mathsf{eavl}[p]}_{\mathsf{Real}^0}(\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}})$.

Concretely, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ also samples $\mathbf{K} = (K_1, \ldots, K_u)$ and $\mathbf{PK} = (PK_1, \ldots, PK_u)$ for commence, and internally simulate $\widetilde{\mathsf{SIC}}$. Here the event $\mathsf{BadUserKey}$ occurs if there exits $K_{l_1}\|PK_{l_1} = K_{l_2}\|PK_{l_2}$. $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ then runs $\mathcal{A}$ and reacts as follows:

- Upon a query to $\widetilde{\mathsf{IC}}$ or π: simply relays.
- Upon a (non-challenge) encryption query (i^*, N^*, A^*, M^*) from $\mathcal{A}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simulates the process of $\mathsf{S2P}[\widetilde{\mathsf{IC}}, \widetilde{\mathsf{SIC}}].\mathsf{LEnc}_{K_{i^*},PK_{i^*}}(N^*, A^*, M^*)$ (which is similar to appendix E.5), and returns the resulted ciphertext and leakage to $\mathcal{A}$. For this entire process $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ makes $2\ell_i + 2$ queries to π and spends $O(p\ell_i t_l)$ time on evaluating the leakage functions.
- Upon a trivial decryption query (i^*, N^*, A^*, C^*) from $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ parses $C^* = \mathbf{c}^*\|Z^*$ and simply emulates $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}].\mathsf{LDec}_{K_{i^*},PK_{i^*}}(N^*, A^*, C^*)$, and relays the outputs *as well as the leakages* to $\mathcal{A}$. The cost is similar to the encryption case.
- Upon a non-trivial decryption query (i^*, N^*, A^*, C^*) from $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ parses $C^* = \mathbf{c}^*\|Z^*$, and computes $U^*\|V^* \leftarrow \mathsf{H}[\pi](A^*, \mathbf{c}^*, N^*, PK_{i^*})$. Then,
  - if $(K_{i^*}, V^*\|1, Z^*) \notin \text{SICTable}^{-1}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ samples V^{**} such that $(K_{i^*}, V^*\|1, V^{**}) \notin \text{SICTable}$, and defines the two entries $\text{SICTable}(K_{i^*}, V^*\|1, V^{**}) \leftarrow Z^*$ and $\text{SICTable}^{-1}(K_{i^*}, V^*\|1, Z^*) \leftarrow V^{**}$;
  - if $(K_{i^*}, V^*\|1, Z^*) \in \text{SICTable}^{-1}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ sets $V^{**} \leftarrow \text{SICTable}^{-1}(K_{i^*}, V^*\|1, Z^*)$.

  Now $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ returns $(\perp, V^{**})$ to $\mathcal{A}$. The bad event $\mathsf{BadCheck}$ occurs if $V^{**} = V^*$.
- Upon $\mathcal{A}$ submitting the ℓ-th challenge tuple $(i_\ell, N_\ell, A_\ell, M_\ell^0, M_\ell^1)$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simulates the process of $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}].\mathsf{LEnc}_{K_{i_\ell},PK_{i_\ell}}(N_\ell, A_\ell, M_\ell^0)$ when $\ell < j$, and the process of $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}].\mathsf{LEnc}_{K_{i_\ell},PK_{i_\ell}}(N_\ell, A_\ell, M_\ell^1)$ when $\ell > j$, and:
  - When $\ell = j$, it relays M_ℓ^0 and M_ℓ^1 to the eavesdropper $\mathsf{eavl}[p]$ challenger of Real^0 to obtain $\mathbf{c}_\ell^b$ and leakages leak_{enc} and $[\mathsf{leak}_{dec}]^{p-1}$, and then generate the tag Z_ℓ accordingly and returns $\mathbf{c}_\ell^b\|Z_\ell$ to $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$. Note that here $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ acquires $[\mathsf{leak}_{dec}]^{p-1}$ the decryption leakages. Also, we define the event $\mathsf{BadInitKey}$, which happens if the entry $\text{SICTable}^{-1}(K_{i_\ell}, PK_{i_\ell}\|0^*, B_{ch})$ is defined for the initial key B_{ch} sampled inside the eavesdropper challenger.
- Upon $\mathcal{A}$ making the λ-th query to $\mathsf{L}_{\mathsf{Decch}}(\ell)$ $(1 \le \lambda \le p-1)$,
  - When $\ell \neq j$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ performs the corresponding decryption and returns the obtained leakages to $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$;
  - When $\ell = j$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simply returns the λ-th trace in the aforementioned vector $[\mathsf{leak}_{dec}]^{p-1}$ as the answer.

The remaining analyzes are similar to appendix E.5: first,

$$\Pr[\mathsf{BadUserKey}] \le \frac{u^2}{2^{n+n_p}} \le \frac{u}{2^{n_p}}, \qquad \Pr[\mathsf{BadInitKey}] \le \frac{q_e + q_d + q_m}{2^n} \le \frac{q_\pi^*}{2^n}, \text{ and}$$

$$\Pr[\mathsf{BadCheck}] \le \frac{u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{2q_\pi^* + 2nq_d}{2^n},$$

where the last bound comes from Eq. (34) in appendix D, and $q_\pi^* = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$. Define $\mathsf{Bad} := \mathsf{BadUserKey} \vee \mathsf{BadInitKey} \vee \mathsf{BadCheck}$. Besides running $\mathcal{A}^{\pi,\widetilde{\mathsf{IC}}}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ internally processes $q_m + q_e + q_d - 1$ queries, and thus makes at most q_π^* queries to π and spends $t^* = O(t + p\sigma t_l)$ running time. Therefore,

$$\begin{aligned}&\left|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_1 \Rightarrow 1]\right| \;\le\; \Pr[\mathsf{Bad} \text{ in } \mathsf{G}_{2,q_e}] + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[p]}_{\mathsf{Real}^0}(q_\pi^*, t^*, \ell_i)\\ \le&\frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{3q_\pi^* + 2nq_d}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl}[p]}_{\mathsf{Real}^0}(q_\pi^*, t^*, \ell_i),\end{aligned}$$

which is the claim in Eq. (38).

F Proof of Theorem 1 (muCCAm$ of S1P)

Note that in the misuse resilience setting, schemes which achieve both CPA confidentiality and authenticity also achieve CCA confidentiality [2]:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathcal{A},\mathsf{AEAD},u} &= \left|\Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\mathsf{Dec}_{\mathbf{K},\mathbf{PK}},\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\$,\perp,\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right]\right|\\ &\le \underbrace{\left|\Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\mathsf{Dec}_{\mathbf{K},\mathbf{PK}},\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\perp,\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right]\right|}_{\mathbf{Adv}^{\mathsf{mu\text{-}INT\text{-}CTXT}}_{\mathcal{A},\mathsf{AEAD},u}\text{: mu INT-CTXT advantage of } \mathcal{A} \text{ on AEAD}}\\ &\quad + \underbrace{\left|\Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\perp,\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{PK}},\$,\perp,\pi,\widetilde{\mathsf{IC}}} \Rightarrow 1\right]\right|}_{\text{defined as } \mathbf{Adv}^{\mathsf{muCPAm\$}}_{\mathcal{A},\mathsf{AEAD},u}}.\end{aligned}$$

Clearly, $\mathbf{Adv}^{\mathsf{mu\text{-}INT\text{-}CTXT}}_{\mathcal{A},\mathsf{S1P},u} \le \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{S1P},u}$. Therefore, we focus on the CPA advantage $\mathbf{Adv}^{\mathsf{muCPAm\$}}_{\mathcal{A},\mathsf{S1P},u}$. Again we employ the H-coefficients technique, and present the two steps in two subsequent subsections.

F.1 Transcripts

We summarize to the queries to the challenge (second) encryption oracle in a set

$$\tau_e = \Big((i_1, N_1, A_1, M_1, C_1), \ldots, (i_{q_e}, N_{q_e}, A_{q_e}, M_{q_e}, C_{q_e})\Big),$$

where $C_j = \mathbf{c}_j \| Z_j$ for each $(i_j, N_j, A_j, M_j, C_j)$.

Besides, we reveal all the π queries underlying the non-challenge encryption queries (i.e., queries to the first encryption oracle). We merge all these additional π queries with the adversarial queries to obtain a set τ_π^*. It can be seen that

$$q_\pi^* := \left|\tau_\pi^*\right| \le \sigma_1 + q_m + q_\pi,$$

where σ_1 is the total number of blocks in the non-challenge encryption queries. At the end of the interaction, we also reveal the KDF- and TGF-calls underlying these non-challenge encryption queries, and organize them in a list $\tau_{\widetilde{\mathsf{SIC}}}$.

We also have the TBC query transcript $\tau_{\widetilde{\mathsf{IC}}}$. Similarly to appendix C, we organize τ_h^* from τ_π^*. Note that the queries underlying the challenge encryption queries are not revealed (unlike appendix C). As such, τ_h^* only contains the S1P hash records corresponding to the non-challenge encryption queries.

Recall that we've switched to the CPA setting, so these are "enough": transcripts are defined as

$$\tau = (\tau_e, \tau_h^*, \tau_\pi^*, \tau_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\mathsf{SIC}}}, \mathbf{PK}, \mathbf{K}).$$

For an encryption query $(i_j, N_j, A_j, M_j, C_j) \in \tau_e$, we denote

$$M_j = M_j[1]\| \dots \|M_j[\ell_j], \quad A_j = A_j[1]\| \dots \|A_j[\nu_j], \quad \text{and } C_j = C_j[1]\| \dots \|C_j[\ell_j]\|Z_j.$$

With these, the number of blocks in challenge encryption queries is $\sigma_2 = \sum_{j=1}^{q_e}(\ell_j + \nu_j)$. We write $\sigma_c = \sum_{j=1}^{q_e} \ell_j$. We further assume that $|M_j[\ell_j]| = r$ for any j: it can be seen this is wlog. Then it's easy to see

$$\Pr[T_{id} = \tau] = \Pr[\mathbf{K}, \mathbf{PK}] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \Pr[\widetilde{\mathsf{SIC}} \vdash \tau_{\widetilde{\mathsf{SIC}}}] \cdot \Pr[\pi \vdash \tau_\pi^*] \cdot \left(\frac{1}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}. \tag{39}$$

Below we write

$$q_\pi^{**} = \sigma_2 + q_e + q_\pi^* = \sigma + q_m + q_e + q_d + q_\pi,$$

which will be the total number of permutation-calls during the interaction.

F.2 Bad Transcripts

We then define bad transcripts as follows.

Definition 7 (Bad Transcripts for S1P, muCPAm$). *An attainable transcript τ is bad, if one of the following conditions is fulfilled:*

- *(B-1) there exists two users j, ℓ such that $K_j\|PK_j = K_\ell\|PK_\ell$.*
- *(B-2) $\mu_{PK} \geq n+1$, $\mu_V \geq n+1$.*
- *(B-3) There exists a query $(K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}$ such that $(K, T, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$.*
- *(B-4) There exists $(i_j, N_j, A_j, M_j, C_j) \in \tau_e$ such that $(K_{i_j}, PK_{i_j}\|0^*, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$.*
- *(B-5) Too many collisions within the q_e tags of the challenge encryption queries, i.e., $\left|\{(j,\ell) : Z_j = Z_\ell\}\right| \geq q_e$.*
- *(B-6) $\left|\{\big((i, N, A, M, \mathbf{c}\|Z), (K, V\|1, X, Y)\big) \in \tau_e \times (\tau_{\widetilde{\mathsf{IC}}} \cup \tau_{\widetilde{\mathsf{SIC}}}) : Z = Y\}\right| \geq q_e$.*

Otherwise τ is good. Denote by $\mathcal{T}_{bad}$ the set of bad transcripts.

First, $\Pr[\text{(B-1)}] \leq \frac{u^2}{2^{n+n_p}} \leq \frac{u}{2^{n_p}}$. Then, the conditions (B-2) and (B-3) $\vee$ (B-4) are essentially the same as Definition 5, and thus we recycle the corresponding results and obtain:

$$\Pr[\text{(B-1)} \vee \text{(B-2)} \vee \text{(B-3)} \vee \text{(B-4)}] \leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}.$$

For (B-5), using Markov's inequality yields

$$\Pr[\text{(B-5)}] \leq \frac{q_e^2}{2^n} \Big/ q_e \leq \frac{q_e}{2^n}.$$

Similarly,

$$\Pr[\text{(B-6)}] \leq \frac{q_e(q_{\widetilde{\mathsf{IC}}} + q_m)}{2^n} \Big/ q_e \leq \frac{q_{\widetilde{\mathsf{IC}}} + q_m}{2^n}.$$

In all,

$$\begin{aligned}\Pr[T_{id} \in \mathcal{T}_{bad}] &\leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + q_e + q_{\widetilde{\mathsf{IC}}} + q_m + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} \\ &\leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{4nq_\pi^* + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \end{aligned} \tag{40}$$

F.3 Ratio of Probabilities of Good Transcripts

For a good transcript τ, by $\neg$(B-3), for any $(i_j, N_j, A_j, M_j, C_j) \in \tau_e$ the initial key $B_j = \widetilde{\mathsf{IC}}_{K_{i_j}}^{PK_{i_j}\|0^*}(N_j\|0^*)$ is uniform. We define a predicate $\mathsf{BadKD}(\widetilde{\mathsf{IC}})$ to formalize the "badness" of this key, which is fulfilled if either of the following conditions is fulfilled:

- (C-1) there exists $(i_j, N_j, A_j, M_j, C_j) \in \tau_e$ such that $B_j = \widetilde{\mathsf{IC}}_{K_{i_j}}^{PK_{i_j}\|0^*}(N_j\|0^*)$ satisfies $(N_j\|PK_{i_j}\|0^*\|B_j, \star) \in \tau_\pi^*$.
- (C-2) there exists two queries $(i_j, N_j, A_j, M_j, C_j)$ and $(i_\ell, N_\ell, A_\ell, M_\ell, C_\ell)$ in τ_e such that $N_j = N_\ell$, $PK_{i_j} = PK_{i_\ell}$, and $B_j = B_\ell$.

Conditioned on $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}$ and $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{sIC}}}$, B_j remains random due to $\neg$(B-3) and $\neg$(B-4) (recall that the nonce values of non-challenge and challenge encryption queries have no overlap). Furthermore, even given $B_1, \ldots, B_{j-1}$, B_j remains random: because for any index $\ell < j$, if $i_\ell \neq i_j$ then $K_{i_\ell}\|PK_{i_\ell} \neq K_{i_j}\|PK_{i_j}$ due to $\neg$(B-1), and B_j is thus independent from B_ℓ; otherwise, we must have $N_\ell \neq N_j$ due to the nonce-respecting restriction, and thus B_j remains uniform in at least $2^n - q_e - q_m$ possibilities given B_ℓ, $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}$, and $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{sIC}}}$. Therefore, when $q_e + q_m \leq 2^n/2$, we have (the set $\tau_\pi^*[N_j\|PK_j]$ is from Eq. (25))

$$\begin{aligned}\Pr[\text{(C-1)}] \leq& \sum_{j=1}^{q_e} \Pr\left[B_j \in \tau_\pi^*[N_j\|PK_j]\right] \leq \sum_{j=1}^{q_e} \frac{\left|\tau_\pi^*[N_j\|PK_j]\right|}{2^n - q_e - q_m} \\ \leq& \mu_{PK} \cdot \sum_{N\|PK \in \{0,1\}^{n_N+n_p}} \frac{2\left|\tau_\pi^*[N_j\|PK_j]\right|}{2^n} \leq \frac{2nq_\pi^*}{2^n}.\end{aligned}$$

For (C-2), since nonce is never repeated in a specific user, it has to be $i_j \neq i_\ell$. Thus by the condition $PK_{i_j} = PK_{i_\ell}$ and $\neg$(B-2), the number of choices for such two queries is at most nq_e. For each of them, it holds $\Pr[B_j = B_\ell] \leq \frac{1}{2^n - q_e - q_m} \leq \frac{2}{2^n}$, thus

$$\Pr[\text{(C-2)}] \leq \frac{2nq_e}{2^n}.$$

In all,

$$\Pr_{\widetilde{\mathsf{IC}}}\left[\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \mid \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}\right] \leq \frac{2nq_\pi^* + 2nq_e}{2^n}. \tag{41}$$

Then, given a random permutation π such that $\pi \vdash \tau_\pi^*$, we define a bad predicate $\mathsf{Bad}(\pi)$. To this end, for any $j \in [1, \ldots, q_e]$, we define a sequence of values as follows:

$$S_{j,0}^{in} = N\|PK_{i_j}\|0^*\|B_j,\ S_{j,0}^{out} = \pi(S_{j,0}^{in}),\ S_{j,1}^{in} = S_{j,0}^{out} \oplus A_j[1]\|0^c,\ S_{j,1}^{out} = \pi(S_{j,1}^{in}), \ldots,$$

$S_{j,\nu_j}^{in} = S_{j,\nu_j-1}^{out} \oplus (A_j[\nu_j]\|1\|0^*) \oplus (0^r\|[1]_2\|0^{c-2})$ when $|A_j[\nu_j]| < r$, and $S_{j,\nu_j}^{in} = S_{j,\nu_j-1}^{out} \oplus (A_j[\nu_j]\|0^c)$ when $|A_j[\nu_j]| = r$, and

$$\begin{aligned}&S_{j,\nu_j}^{out} = \pi(S_{j,\nu_j}^{in}),\ S_{j,\nu_j+1}^{in} = C_j[1]\|\mathsf{ls}_c\left(S_{j,\nu_j}^{out} \oplus (0^r\|[2]_2\|0^{c-2})\right),\\ &S_{j,\nu_j+1}^{out} = \pi(S_{j,\nu_j+1}^{in}),\ S_{j,\nu_j+2}^{in} = C_j[2]\|\mathsf{ls}_c(S_{j,\nu_j+1}^{out}),\ S_{j,\nu_j+2}^{out} = \pi(S_{j,\nu_j+2}^{in}),\ \ldots,\end{aligned}$$

$S_{j,\nu_j+\ell_j}^{in} = C_j[\ell_j]\|\mathsf{ls}_c\left(S_{j,\nu_j+\ell_j-1}^{out} \oplus (0^r\|[1]_2\|0^{c-2})\right)$ when $|C_j[\ell_j]| < r$, and $S_{j,\nu_j+\ell_j}^{in} = C_j[\ell_j]\|\mathsf{ls}_c(S_{j,\nu_j+\ell_j-1}^{out})$ when $|C_j[\ell_j]| = r$, and finally $S_{j,\nu_j+\ell_j}^{out} = \pi(S_{j,\nu_j+\ell_j}^{in})$ and $U_j\|V_j = \mathsf{ls}_{2n-1}(S_{j,\nu_j+\ell_j}^{out})$. With the above, the predicate $\mathsf{Bad}(\pi)$ is fulfilled if:

– (C-3) there exists two indices $j \in [1,\ldots,q_e]$ and $\ell \in [1,\ldots,\nu_j+\ell_j]$ such that $\big(\star \| \mathsf{ls}_{c-2}(S^{in}_{j,\ell}), \star\big) \in \tau^*_\pi$.
– (C-4) there exists four indices $j_1, j_2 \in [1,\ldots,q_e]$ and $\ell_1 \in [1,\ldots,\nu_{j_1}+\ell_{j_1}], \ell_2 \in [1,\ldots,\nu_{j_2}+\ell_{j_2}]$ such that $(j_1,\ell_1) \neq (j_2,\ell_2)$, and $\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_1}) = \mathsf{ls}_{c-2}(S^{in}_{j_2,\ell_2})$.
– (C-5) there exists an index $j \in [1,\ldots,q_e]$ such that any of the following is fulfilled:
  • $(K_{i_j}, V_j \| 1, U_j, \star) \in (\tau_{\widetilde{\mathsf{lC}}} \cup \tau_{\widetilde{\mathsf{slC}}})$ or $(K_{i_j}, V_j \| 1, \star, Z_j) \in (\tau_{\widetilde{\mathsf{lC}}} \cup \tau_{\widetilde{\mathsf{slC}}})$, or
  • there exists another index $\ell \in [1,\ldots,q_e]$ such that $V_j \| U_j = V_\ell \| U_\ell$ or $V_j \| Z_j = V_\ell \| Z_\ell$.

For (C-3), we first consider the case $j=1, \ell=1$. Conditioned on $\pi \vdash \tau^*_\pi$, the new capacity value $\mathsf{ls}_{c-2}(S^{in}_{j,1})$ is uniform due to ¬(C-1). Therefore,

$$\Pr\left[(\star \| \mathsf{ls}_{c-2}(S^{in}_{j,1}), \star) \in \tau^*_\pi\right] \leq \frac{2|\tau^*_\pi|}{2^{c-2}} \leq \frac{8q^*_\pi}{2^c}.$$

Conditioned on $\big(\star \| \mathsf{ls}_{c-2}(S^{in}_{j,1})\big), \star\big) \notin \tau^*_\pi$, $\mathsf{ls}_{c-2}(S^{in}_{j,2})$, i.e., the "next" capacity value, is uniform. Therefore,

$$\Pr\left[(\star \| \mathsf{ls}_{c-2}(S^{in}_{j,2}), \star) \in \tau^*_\pi\right] \leq \frac{8q^*_\pi}{2^c}.$$

Therefore, via an iterative-style analysis, it can be seen all internal capacity values are uniform, and thus

$$\Pr[\text{(C-3)}] \leq \sum_{j=1}^{q_e} \big(\nu_j + \ell_j\big) \cdot \frac{8q^*_\pi}{2^c} \leq \frac{8\sigma_2 q^*_\pi}{2^c}.$$

For (C-4), when $j_1 = j_2$, then $\ell_1 \neq \ell_2$. By the above, both $\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_1})$ and $\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_2})$ are uniform, thus

$$\Pr\left[\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_1}) = \mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_2})\right] \leq \frac{8}{2^c}.$$

When $j_1 \neq j_2$, by the above analysis, both $\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_1})$ and $\mathsf{ls}_{c-2}(S^{in}_{j_2,\ell_2})$ are uniform. Moreover, we have $S^{in}_{j_1,0} \neq S^{in}_{j_2,0}$ by ¬(C-2), which means $\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_1})$ remains uniform given the value of $\mathsf{ls}_{c-2}(S^{in}_{j_2,\ell_2})$. Therefore,

$$\Pr\left[\mathsf{ls}_{c-2}(S^{in}_{j_1,\ell_1}) = \mathsf{ls}_{c-2}(S^{in}_{j_2,\ell_2})\right] \leq \frac{8}{2^c}.$$

By these, and as the number of choices for (j_1,ℓ_1) and (j_2,ℓ_2) is at most σ_2^2, we reach

$$\Pr[\text{(C-4)}] \leq \frac{8\sigma_2^2}{2^c}.$$

Finally, for (C-5), we have: (a) $\Pr[\exists j : (K_{i_j}, V_j \| 1, U_j, \star) \in (\tau_{\widetilde{\mathsf{lC}}} \cup \tau_{\widetilde{\mathsf{slC}}})] \leq q_e(q_{\widetilde{\mathsf{lC}}} + q_m) \cdot \frac{2}{2^{2n-1}}$ and $\Pr[\exists j, \ell : V_j \| U_j = V_\ell \| U_\ell] \leq q_e^2 \cdot \frac{2}{2^{2n-1}}$ as $U_j \| V_j$ is uniform for any j, and (b) $\Pr[\exists j : (K_{i_j}, V_j \| 1, \star, Z_j) \in (\tau_{\widetilde{\mathsf{lC}}} \cup \tau_{\widetilde{\mathsf{slC}}})] \leq \frac{2q_e}{2^{n-1}}$, and (c) by ¬(B-5), $\Pr[\exists j, \ell : V_j \| Z_j = V_\ell \| Z_\ell] \leq q_e \cdot \frac{2}{2^{n-1}}$. Therefore,

$$\Pr[\text{(C-5)}] \leq \frac{4q_e(q_{\widetilde{\mathsf{lC}}} + q_m)}{2^{2n}} + \frac{4q_e^2}{2^{2n}} + \frac{4q_e}{2^n} + \frac{4q_e}{2^n} \leq \frac{4q_e(q_{\widetilde{\mathsf{lC}}} + q_m + q_e)}{2^{2n}} + \frac{8q_e}{2^n} \leq \frac{2q_{\widetilde{\mathsf{lC}}} + 10q_e}{2^n},$$

where the last inequality follows from $q_e \leq 2^n/2$ and $q_m + q_e \leq 2^n/2$. Summing over the above, and further using $q_e + \sigma_2 + q^*_\pi \leq q^{**}_\pi$ yield

$$\Pr[\mathsf{Bad}(\pi)] \leq \frac{8\sigma_2 q^*_\pi}{2^c} + \frac{8\sigma_2^2}{2^c} + \frac{2q_{\widetilde{\mathsf{lC}}} + 10q_e}{2^n} \leq \frac{8\sigma q^{**}_\pi}{2^c} + \frac{2q_{\widetilde{\mathsf{lC}}} + 10q_e}{2^n}.$$

Finally, conditioned on that $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}$, $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{SIC}}}$, $\pi \vdash \tau_\pi^*$, $\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}})$, and $\neg\mathsf{Bad}(\pi)$, we analyze the probability that each produced key stream block equal $\Delta_j[l] = M_j[l] \oplus C_j[l]$, i.e., $\Pr[\mathsf{ms}_r(S^{in}_{j,\nu+l}) = M_j[l] \oplus C_j[l]]$. This means $\mathsf{ms}_r(S^{in}_{j,\nu+l}) = \Delta_j[l]\|s \notin \tau_\pi^*$ for some $s \in \{0,1\}^c$. Among the 2^c values of s, there are at most $|\tau_\pi^*| \leq q_\pi^*$ "bad" values s^* such that $\Delta_j[l]\|s^* \in \tau_\pi^*$. Therefore,

$$\Pr\left[\mathsf{ms}_r(S^{in}_{j,\nu+l}) = \Delta_j[l]\|s \notin \tau_\pi^*\right] \geq \frac{2^c - q_\pi^*}{2^{r+c}} \geq \frac{1 - \frac{q_\pi^*}{2^c}}{2^r}.$$

By this,

$$\begin{aligned}&\Pr[\forall j \in \{1,\ldots,q_e\} : \mathsf{S1P}[\pi,\widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j,N_j,A_j,M_j) = \mathbf{c}_j \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}]\\ \geq\ &\Pr[\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \wedge \neg\mathsf{Bad}(\pi) \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}} \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{SIC}}}] \cdot \left(\frac{1 - \frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c}. \qquad (42)\end{aligned}$$

It remains to analyze the involved tags. The event that the q_e tags equal $Z_1,\ldots,Z_{q_e}$ is equivalent to q_e equalities as follows:

$$\widetilde{\mathsf{IC}}^{V_1\|1}_{K_{i_1}}(U_1) = Z_1, \ldots, \widetilde{\mathsf{IC}}^{V_{q_e}\|1}_{K_{i_{q_e}}}(U_{q_e}) = Z_{q_e}.$$

Consider the first equality. The entries $\widetilde{\mathsf{IC}}^{V_1\|1}_{K_{i_1}}(U_1)$ and $(\widetilde{\mathsf{IC}}^{V_1\|1}_{K_{i_1}})^{-1}(Z_1)$ may be rendered non-random due to the condition $\mathsf{S1P}[\pi,\widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j,N_j,A_j,M_j) = \mathbf{c}_j$ for all $j \in \{1,\ldots,q_e\}$ or due to $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}$. Yet, the former condition only affects entries with the tweak $PK\|0^*$, while the latter won't affect $\widetilde{\mathsf{IC}}^{V_1\|1}_{K_{i_1}}(U_1)$ nor $(\widetilde{\mathsf{IC}}^{V_1\|1}_{K_{i_1}})^{-1}(Z_1)$ by $\neg$(C-5). Therefore, $\Pr[\widetilde{\mathsf{IC}}^{V_1\|1}_{K_{i_1}}(U_1) = Z_1] = \frac{1}{2^n}$.

In a similar vein, for any $j \in \{1,\ldots,q_e\}$, under the conditions that $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}$ and "$\forall j \in \{1,\ldots,q_e\} : \mathsf{S1P}[\pi,\widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j,N_j,A_j,M_j) = \mathbf{c}_j$", the ideal TBC entry $\widetilde{\mathsf{IC}}^{V_j\|1}_{K_{i_j}}(U_j)$ remains uniform. We need to additionally consider the condition "$\widetilde{\mathsf{IC}}^{V_\ell\|1}_{K_{i_\ell}}(U_\ell)$ for $\ell = 1,\ldots,j-1$". For this, for any $\ell < j$ we have $V_j\|Z_j \neq V_\ell\|Z_\ell$ and $U_j\|V_j \neq U_\ell\|V_\ell$ by $\neg$(C-5). Consequently, $\Pr[\widetilde{\mathsf{IC}}^{V_j\|1}_{K_{i_j}}(U_j) = Z_j] \geq \frac{1}{2^n}$, and thus

$$\Pr[\widetilde{\mathsf{IC}}^{V_j\|1}_{K_{i_j}}(U_j) = Z_j \text{ for } j = 1,\ldots,q_e] \geq \left(\frac{1}{2^n}\right)^{q_e}. \qquad (43)$$

In summary,

$$\begin{aligned}\frac{\Pr[T_{re} = \tau]}{\Pr[T_{id} = \tau]} \geq&\frac{\Pr[\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \wedge \neg\mathsf{Bad}(\pi) \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \left(\frac{1-\frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}}{\left(\frac{1}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}}\\ \geq&\Pr[\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \wedge \neg\mathsf{Bad}(\pi) \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \left(1 - \frac{q_\pi^*}{2^c}\right)^{\sigma_c}\\ \geq&\left(1 - \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma q_\pi^{**}}{2^c} - \frac{2q_{\widetilde{\mathsf{IC}}} + 10q_e}{2^n}\right) \cdot \left(1 - \frac{q_\pi^{**}}{2^c}\right)^{\sigma_c}\\ \geq&1 - \frac{2nq_\pi^* + 2nq_e + 2q_{\widetilde{\mathsf{IC}}} + 10q_e}{2^n} - \frac{8\sigma q_\pi^{**}}{2^c}.\end{aligned}$$

Gathering this and Eq. (40) and further the muCIML2 bound Eq. (30) (with that $q_\pi^* = \sigma + q_e + q_d + q_\pi$ replaced by $q_\pi^{**} = \sigma + q_e + q_d + q_m + q_\pi$) yield Eq. (1):

$$\begin{aligned}
&\frac{3u}{2^{n_p}} + \frac{32(q_\pi^*)^2}{2^c} + \frac{5nq_\pi^* + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} \\
&+ \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{4nq_\pi^* + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2nq_\pi^* + 2nq_e + 2q_{\widetilde{\mathsf{IC}}} + 10q_e}{2^n} + \frac{8\sigma q_\pi^{**}}{2^c} \\
\leq &\frac{5u}{2^{n_p}} + \frac{56(q_\pi^*)^2}{2^c} + \frac{23nq_\pi^* + 5n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} \qquad (2nq_e + 2nq_d \leq 2nq_\pi^{**}).
\end{aligned}$$

G Proof of Theorem 2 (muCCAm$ of S2P)

Similarly to appendix F, we mainly bound the CPA advantage $\mathbf{Adv}^{\mathsf{muCPAm\$}}_{\mathcal{A},\mathsf{S2P}}$ using H-coefficients. Since S2P and S1P process the messages in a very similar manner, the muCPAm$ proofs are also similar.

Bad Transcripts We summarize to the queries to the challenge (second) encryption oracle in $\tau_e = \Big((i_1, N_1, A_1, M_1, C_1), \ldots, (i_{q_e}, N_{q_e}, A_{q_e}, M_{q_e}, C_{q_e})\Big)$. For each of them $(i_j, N_j, A_j, M_j, C_j)$ where $C_j = \mathbf{c}_j \| Z_j$, we assume that the system makes the corresponding hash call $\mathsf{H}[\pi](A_j, \mathbf{c}_j, N_j, PK_{i_j})$ during the interaction, and that these π queries are known to the distinguisher. We also reveal all the π queries underlying the non-challenge encryption queries (i.e., to the first encryption oracle). We merge all these additional π queries with the adversarial queries to obtain τ_π^*. Here we have

$$q_\pi^* := \left|\tau_\pi^*\right| \leq q_\pi + (2q_m + 2\sigma_1) + (2q_e + \sigma_2) = q_\pi + 2(q_m + q_e) + 2\sigma_1 + \sigma_2$$

and

$$q_\pi^{**} = 2\sigma + 2(q_e + q_d + q_m) + q_\pi.$$

We also have the ideal TBC query transcript $\tau_{\widetilde{\mathsf{IC}}}$. Similarly to appendix D, we also organize the hash record transcript τ_h^* from τ_π^*, and the transcript $\tau_{\widetilde{\mathsf{SIC}}}$ for the KDF and TGF calls underlying non challenge encryption queries. In all, transcripts are defined as

$$\tau = (\tau_e, \tau_h^*, \tau_\pi^*, \tau_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\mathsf{SIC}}}, \mathbf{PK}, \mathbf{K}).$$

We then define bad transcripts.

Definition 8 (Bad Transcripts for S2P, muCCAm$). *An attainable transcript τ is bad, if one of the following conditions is fulfilled:*

- *(B-1) There exists two users j, ℓ such that $K_j \| PK_j = K_\ell \| PK_\ell$.*
- *(B-2) $\mu_{PK} \geq n + 1$, $\mu_V \geq n + 1$.*
- *(B-3) There exists a query $(K, T, X, Y) \in \tau_{\widetilde{\mathsf{SIC}}}$ such that $(K, T, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$.*
- *(B-4) There exists an encryption query $(i_j, N_j, A_j, M_j, C_j)$ such that:*
  - *$(K_{i_j}, PK_{i_j} \| 0^*, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$, or*
  - *$(K_{i_j}, V_j \| 1, \star, \star) \in \tau_{\widetilde{\mathsf{IC}}}$ for the corresponding hash record $((A_j, \mathbf{c}_j, N_j, PK_{i_j}), U_j \| V_j)$.*
- *(B-5) There exists a query $(i, N, A, M, \mathbf{c} \| Z) \in \tau_e$ with corresponding hash record $((A, \mathbf{c}, N, PK), U \| V) \in \tau_h^*$ such that:*
  - *contradiction-I: $(K_i, V \| 1, \star, Z) \in \tau_{\widetilde{\mathsf{SIC}}}$ or $(K_i, V \| 1, U, \star) \in \tau_{\widetilde{\mathsf{SIC}}}$; or*
  - *hash collision: there exists a hash record $((A', \mathbf{c}', N', PK'), U' \| V') \in \tau_h^*$ such that $(A, \mathbf{c}, N, PK) \neq (A', \mathbf{c}', N', PK')$ though $U \| V \neq U' \| V'$; or*

- *contradiction-II: there exists another query* $(i', N', A', M', \mathbf{c}' \| Z') \in \tau_e$ *such that* $V \| Z \neq V' \| Z'$.

(B-1) is obvious. Then, (B-2), (B-3), and (B-4) are essentially the same as Definition 6, and we recycle the bound (which already includes the hash collision event):

$$\Pr[\text{(B-1)} \vee \text{(B-2)} \vee \text{(B-3)} \vee \text{(B-4)} \vee \textit{hash collision}] \leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^*}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}.$$

For the remaining subconditions in (B-5), it's easy to see $\Pr[\textit{contradiction}] \leq \frac{2q_e(q_e+q_m)}{2^{2n-1}} + \frac{2q_e^2}{2^{2n-1}} = \frac{4q_e(2q_e+q_m)}{2^{2n}} \leq \frac{2q_e}{2^n}$ (since $2q_e + q_m \leq q_\pi^* \leq 2^n/2$). In all,

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \leq \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}} + 2q_e}{2^n}. \tag{44}$$

Ratio of Probabilities. This step could reuse the results in appendix F.3. Concretely, we define $\mathsf{BadKD}(\widetilde{\mathsf{IC}})$ and $\mathsf{Bad}(\pi)$ as those in appendix F.3 without the condition (C-5). As such, we could recycle Eqs. (42) and (43) in the following manner:

$$\begin{aligned}
&\Pr[\forall j \in \{1, \ldots, q_e\} : \mathsf{S2P}[\pi, \widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j, N_j, A_j, M_j) = \mathbf{c}_j \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \\
\geq &\left(1 - \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma q_\pi^{**}}{2^c}\right) \cdot \left(\frac{1 - \frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c}, \\
&\Pr[\forall j \in \{1, \ldots, q_e\} : \widetilde{\mathsf{IC}}_{K_{i_j}}^{V_j \| 1}(U_j) = Z_j] \geq \left(\frac{1}{2^n}\right)^{q_e}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\frac{\Pr[T_{re} = \tau]}{\Pr[T_{id} = \tau]} \geq &\frac{\left(1 - \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma q_\pi^{**}}{2^c}\right) \cdot \left(\frac{1 - \frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}}{\left(\frac{1}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}} \\
\geq &1 - \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma q_\pi^{**}}{2^c} - \frac{\sigma q_\pi^{**}}{2^c}.
\end{aligned}$$

Gathering this and Eq. (44) and further Eq. (4) (with that $q_\pi^* = 2\sigma + 2(q_e + q_d) + q_\pi$ replaced by $q_\pi^{**} = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$) yield Eq. (2):

$$\begin{aligned}
&\frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{q_\pi^* + n^2 q_{\widetilde{\mathsf{IC}}} + 2q_e}{2^n} \\
&+ \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma q_\pi^{**}}{2^c} - \frac{\sigma q_\pi^{**}}{2^c} + \frac{2u}{2^{n_p}} + \frac{32(q_\pi^{**})^2}{2^c} + \frac{3q_\pi^{**} + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} \\
\leq &\frac{4u}{2^{n_p}} + \frac{53(q_\pi^{**})^2}{2^c} + \frac{8nq_\pi^{**} + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}.
\end{aligned}$$

H Possibilities of DPAs on ISAP

ISAP is designed to achieve security against decryption leakages [17]. But as mentioned in the Introduction, ISAP does not achieve CIML2 when all the intermediate computations are leaked to the adversary but a long-term key. At the first glance this isn't

harmful since such a leakage model is overly strong. Yet, we'll show that this indeed brings in more possibilities of DPAs in practice.

We first outline a DPA against the integrity of ISAP. We omit the associated data A since it doesn't matter. Concretely,

(1) First, we fix a nonce N, a ciphertext C, and q arbitrary tags $T_1, \ldots, T_q$, where q depends on the relative SCA security of the implementation;
(2) Second, we make q queries to $\mathsf{LDec}(N, C, T_1), \ldots, \mathsf{LDec}(N, C, T_q)$. By this, the function $\textsc{IsapMac}(N, C) \to T$ is called q times. While this may not immediately increase the amount of leaked information about the right tag T, the q comparison operations "if $T = T_i$", $i = 1, \ldots, q$, *does increase* the possibility of recovering T via DPA. Thus a valid forgery (N, C, T) is derived.

Note that the above attack only queries the leaking decryption oracle. Therefore, it does not violate the nonce-respecting restriction, and it breaks both plaintext and ciphertext integrity.

We further show that leakage privacy can also be compromised. Concretely,

(1) First, we fix a nonce N, q ciphertext $C_1, \ldots, C_q$ of some fixed number of blocks, and q tags $T_1^*, \ldots, T_q^*$ in arbitrary, where q depends on the relative SCA security of the implementation;
(2) Second, using the q tags and the above integrity attack, we recover the q correct tags $T_1, \ldots, T_q$ and build q forgeries $(N, C_1, T_1), \ldots, (N, C_q, T_q)$;
(3) Third, we make q queries to $\mathsf{LDec}(N, C_1, T_1), \ldots, \mathsf{LDec}(N, C_q, T_q)$. Unlike the above invalid decryption queries, these (valid) queries would induce q internal calls to $\textsc{IsapDec}(N, C_1), \ldots, \textsc{IsapDec}(N, C_q)$. All these calls would compute the same sponge key stream (denoted $y_1, \ldots, y_\ell$) since they are with the same nonce N. As such, the key stream $y_1, \ldots, y_\ell$ can be recovered by DPA.
(4) Now, with the key stream $y_1, \ldots, y_\ell$, any message M' encrypted with the nonce N can be easily recovered (regardless of whether the leakage of encrypting M' is available or not).

Note that the above attack only queries the leaking decryption oracle and is thus nonce-respecting, but it seems to break *any* meaningful privacy notion, including Barwell et al.'s LAE notion (leakage resilient AE) [3] and Guo et al.'s CCAL2 notion (CCA with decryption leakage) [25].

Finally, we stress that the above only outlines *possibilities* of DPAs. Evaluation of the concrete strength of ISAP should be made w.r.t. to the implementations in question.