Towards Lighter Leakage-resilient Authenticated Encryption from the Duplex Construction

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

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

Abstract. The ongoing NIST lightweight standardization process explicitly puts forward a requirement of side-channel security, which has renewed the interest for Authenticated Encryption schemes (AEs) with light(er)-weight side-channel secure implementations. To address this challenge, we investigate the leakage-resilience of a generic duplex-based stream cipher, and prove the classical bound, i.e., $\approx 2^{c/2}$, under an assumption of non-invertible leakage. Based on this, we propose a new 1-pass AE mode TETSponge, which carefully combines a tweakable block cipher that must have strong protections against side-channel attacks and is scarcely used, and a duplex-style permutation that only needs weak side-channel protections and is used to frugally process the message and associated data. TETSponge offers: (*i*) provable resistance against side-channel attacks during both encryption and decryption, (*ii*) some level of nonce misuse robustness, and (*iii*) black-box AE security with good bounds in the multi-user setting as well. We conclude that TETSponge offers an appealing option for the implementation of lightweight AE in settings where side-channel attacks are an actual concern.

Our analysis offers the first rigorous methodology for the analysis of the leakage-resilience of sponge/duplex-based AEs. It can be easily adapted to others: we demonstrate this by showcasing brief analyzes of two other 1-pass AEs Ascon, GIBBON, and two 2-pass AEs TEDTSponge and ISAP. These provide various insights for both designs and implementations.

Keywords: Authenticated Encryption, Duplex Construction, Leakage-Resilience, Leveled Implementations.

1 Introduction

Problem statement. In 2013, the NIST initiated a lightweight cryptography project to understand the need for dedicated Authenticated Encryption with Associated Data (AEAD), 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 may be deployed in environments where they can be under physical control of an adversary, yet be responsible for critical 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 [39]). However, from a cryptographic viewpoint, NIST's lightweight physical security goal challenges the current understanding of side-channel countermeasures, which typically imply significant overheads. For example, the cycle counts of the (optimized) masked software implementations of block ciphers by Goudarzi and Rivain presented at Eurocrypt 2017 [26] 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. Significant overheads can also be observed in hardware as the number of shares increase (especially if the random generation of the shares is taken into account) [27].

This state of affairs raises the question of the design of AE modes allowing both efficient and lightweight implementation (e.g., supporting constant memory requirements in streaming applications) and embedding side-channel resistance features, so that the secure implementation of the mode can circumvent, at least in part, the costs associated to the protection of its underlying primitives. Besides, and if aiming at standardization, it is also desirable that such modes offer as many standard security features as possible. In this respect:

– Supporting good multi-user security is important in a context of large-scale deployment of lightweight devices, which may be subject to mass-surveillance and distributed adversaries [7].
– Security beyond the classical birthday bound is useful to increase key lifetime [28], which is particularly relevant in the case of lightweight devices that may not incorporate good key agreement mechanisms.

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

– Achieving some level of robustness against nonce reuse is in general a welcome option [38,4].

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 [23], which has then been the seed for the design of PRGs, PRFs and PRPs [43,24,22], with contrasted practical impact [6]. For leakage-resilient authentication, encryption and AE modes, we refer to the CCS 2015 work of Pereira et al. [37] and follow-ups [9,10,5]. Yet, significant gaps remain, particularly if we aim for a lighter single pass design.

Concretely, let's inspect two concrete proposals that get closer to the NIST requirements. The first is ISAP proposed at FSE 2017 by Dobraunig et al. as a potential solution for side-channel secure AE [19]. ISAP makes an important step in putting forward the good properties of sponge-based constructions for side-channel security — an observation that was also made 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, apart from the lack of a systematic analysis of leakage-resilience, which (to the best of our knowledge) is common to all sponge-based constructions, ISAP is a two-pass design, while we are seeking for some lighter solutions with a single pass.

Second, a recent AE mode TEDT [8] for Tweakable Block Ciphers (TBCs) was designed to encourage the so-called *leveled implementations*, where (expensive) protections against side-channel attacks are used in a minimal way, while the bulk of the computation can be executed by cheap and weakly protected circuits. The authors of TEDT argue that the leveled approach can bring significant performance gains compared to implementations where side-channel protections are uniformly used during encryption.[4] Regarding security, the leveled implementations of TEDT could ensure strong guarantees against leakages—corresponding to the top of the hierarchy of confidentiality and integrity definitions established in [29]. Namely, TEDT ensures Ciphertext Integrity with Misuse and Leakage in encryption and decryption (CIML2) in a liberal model where only the long-term key is "safe". TEDT also ensures security against Chosen Ciphertext Adversaries with misuse-resilience (meaning security for messages encrypted with *fresh, non-repeating* nonces,[5] in the presence of nonce-reuse) and leakage in encryption and decryption (CCAmL2), in two different leakage models. Yet, TEDT is also of two passes and rate 1/4, which can be expected to be more expensive than the sponge-based approach of ISAP.

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

Can we design a single-pass leakage resilient AE mode, and how do we demonstrate the form of leakage-resilience that it provides?

The goal of efficient AE in 1 pass pinpoints the duplex construction [12] as the natural starting point (leakage-resilient AE modes for block ciphers typically refresh their keying material for every message blocks, inducing a significant overhead). To this end, our first contribution is the first rigorous leakage security analysis of a general duplex-based stream encryption w.r.t. a leaky version of the classical eavesdropper security model. The analysis is made in the *ideal permutation model* since *there's no other choice at present* (see Section 3.5), and we naturally focus on *oracle-free* leakage functions—as in [43] and in the concurrent work [21]. As a compensation for this idealized analysis, all results are obtained under the weakest and easiest to validate leakage assumption, namely *non-invertibility* [25]. Building upon these, we prove security bounds that are expressive and easy-to-understand, translating to the classical $\approx 2^{c/2}$.

We then study how to extend the leakage-resilient duplex stream cipher into a 1-pass AEAD mode and what can be achieved. We propose a new AEAD mode TETSponge, standing for Tweakable (due to the use of TBC), (simultaneously) Encrypt & Tag (the natural feature of duplex) Sponge, as our result. It *enhances the duplex stream cipher with two calls to an n-bit TBC used as a key derivation function (*KDF*) and a tag generation function (*TGF*).* We also borrow the use of n_p-bit *public key* material from [8] to boost multi-user

[2]See https://competitions.cr.yp.to/caesar.html.

[3]See https://keccak.team/files/Keyakv2-doc2.2.pdf.

[4]A similar approach has been applied to the Ascon cipher in [1] – see Fig. 4 (a) for an illustration of the parts of the implementation they heavily protect.

[5]"Misuse-resilience" was due to [4]. As discussed in [29], misuse-*resistance* in the sense of [38] is believed impossible in many leakage settings: briefly, when the inputs including nonces are arbitrarily controlled, the adversary could (almost) control the leakage traces, and this enables identifying small changes in the encrypted messages.

security. We show that these modifications cinch nice leakage AE security: its leveled implementations, where only the two TBC-calls are protected from side-channels, achieve: (i) multi-user $\mathsf{CCAmL1}$ confidentiality up to $\approx 2^{n/2}$ queries and $\approx 2^{n_p}$ users, which corresponds to CCA security with misuse-resilience, in the presence of encryption leakages; (ii) multi-user $\mathsf{CIML2}$ authenticity up to $2^n/n^2$ queries and $\approx 2^{n_p}$ users, partly thanks to the invertibility of the TBC-based TGF.

For practical usability, we carefully designed $\mathsf{TETSponge}$ to achieve high black-box CCA security as well, i.e., up to $2^n/n^2$ queries and $\approx 2^{n_p}$ users. To keep this submission concise and easy to follow, the black-box results are only presented in the separate supporting document.

Our (intermediate) result on the duplex stream cipher as well as the methodology for the leakage security analysis of duplex/sponge-based AEs are general and can be easily adapted to other duplex/sponge AEs. To serve a brief demonstration, we showcase our method on two other 1-pass AEs Ascon and GIBBON that have a structure similar to $\mathsf{TETSponge}$. On the other hand, since the consensus reached by many [19,10,5,29] is that Encrypt-then-MAC designs with 2 passes are needed to achieve confidentiality against decryption leakages, we further consider two examples: the 1st is a natural 2-pass extension of $\mathsf{TETSponge}$ that we name $\mathsf{TEDTSponge}$, while the 2nd is ISAP. For all of them, we proved (roughly) $\approx 2^{c/2}$ leakage security under very conservative assumptions similar to those for $\mathsf{TETSponge}$. In all, our methodology could be applied to various duplex/sponge-based AEs, and this opens the way to formal leakage-resilience analyses of leveled implementations & possibly linking them to the side-channel cryptanalysis practice.

A concurrent and independent work of Dobraunig and Mennink (DM) [21] also analyzed the leakage-resilience of duplex. While their main result is a bit similar to our Lemma 1, we have slightly different positions: we focus more on how to build leakage secure AEs from duplex, while DM concentrated more on foundational aspects, and derived results on a duplex model more general than Lemma 1. Also we have different leakage assumptions to capture its limited knowledge: our non-invertible leakage assumption is strictly weaker than DM's high min-entropy assumption, and is directly connected to the side-channel cryptanalysis practice. DM's entropy assumption allows deriving tighter and simpler bounds, while our assumption of non-invertibility based on 2 leakages is *quite conservative*, and thus our bounds $2^{c/2}$ is likely to be pessimistic in most cases (we leave characterization for future work). In all, our works are complementary to each other, and we believe investigating a problem from two different perspectives greatly deepens the understanding.

Roadmap. In Section 2 we serve notations and the leakage security definitions used in this paper. Then Section 3 presents our analysis of the duplex-based stream cipher as well as the leakage assumptions in use. Building on this, Section 4 defines $\mathsf{TETSponge}$ and shows its security, while Section 5 demonstrates the applications of our methodology to some other candidates.

2 Preliminaries

Notations. 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 $(q_1, \dots, q_\omega, t)$-bounded adversary a probabilistic algorithm that has access to ω oracles, $O_1, \dots, O_\omega$, 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 $O_1, \dots, O_\omega$ 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 implementation 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 the 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.

Primitives. A random keyless permutation π, as used in sponge analyses, refers to a permutation of $\{0,1\}^b$ drawn uniformly at random among the set of all permutations of $\{0,1\}^b$. 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 $\{0,1\}^n$. In this paper we only focus on (n,n,n)-TBC. An ideal TBC $\widetilde{\mathsf{IC}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$, with the same spirit as ideal (block) ciphers, is a TBC sampled uniformly from all (n,n,n)-TBCs. 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* $C \in \mathcal{C}$*.*
- $\mathsf{Dec} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{C} \to \mathcal{M} \cup \{\bot\}$ *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* $\bot$ *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. 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.*

Multi-user leakage security. In general, leakage security definitions are stated w.r.t. *implementations* of some scheme AEAD in question, and both an encryption leakage function $\mathsf{L}_{\mathsf{Enc}}$ and a decryption leakage (dec-leakages) function $\mathsf{L}_{\mathsf{Dec}}$ are associated to the implementation(s). This models the real-world leaky implementations of the mathematical objects. Note that in theory, our leakage model is *non-adaptive*, as the leakages are a parameter of the to-be-studied implementations determined before the experiment starts rather than chosen by the adversary during the experiment. This restriction was motivated from the side-channel practice & the necessity for practical modes (see [43,24] for some discussion).

Pioneered by Rogaway and Shrimpton [38], nowadays black-box AE analyses typically follow all-in-one definitions that integrate both confidentiality and integrity. However, in front of a leakage adversary, separate definitions for integrity and confidentiality potentially offer more gradual degradations & clearer clarifications on which implementation-level properties are necessary/sufficient for which goal. In theory, this is in accordance with the important general feature of physically observable cryptography that *unpredictability is much easier to ensure than indistinguishability* [35], which naturally splits the level of confidence w.r.t. both notions. Also, different levels of robustness against nonce reuse may be achieved w.r.t. both notions: it was shown that when nonce is arbitrarily reused (in the same flavor as the *misuse-resistance* notion [38]), leakage integrity is achievable [9,10], yet leakage confidentiality may not [9,29] (see footnote [5]). This difference is also reflected in the separate definitions.

Integrity. In detail, regarding integrity, we rely on the *multi-user Ciphertext Integrity with Misuse*-resistance *and Leakage* ($\mathsf{muCIML2}$) defined in [8], which was built upon the single-user version $\mathsf{CIML2}$ introduced in [9,10]. The suffix 2 means two leakage sources, i.e., both encryption and decryption. In some sense, the definition is obtained by enhancing the traditional (multi-user) $\mathsf{INT\text{-}CTXT}$ security with leakages.

Definition 2 ($\mathsf{muCIML2}$ advantage). *Given the implementation of 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*

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

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

- $\mathsf{LEnc}_{\mathbf{K}}(i, N, A, M)$*: if* $1 \le i \le u$*, 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* $1 \le i \le u$*, outputs* $\big(\mathsf{Dec}_{K_i}(N, A, C), \mathsf{L}_{\mathsf{Dec}}(K_i, N, A, C)\big)$*;*
- *The event "*$\mathcal{A}$ *forges" means that any of* $\mathcal{A}$*'s query to* $\mathsf{LDec}_{\mathbf{K}}(i, N, A, C)$ *returns* $M \neq \bot$*, while* C *was not resulted from an earlier encryption query to* $\mathsf{LEnc}_{\mathbf{K}}(i, N, A, M)$*.*

CCA. Regarding confidentiality, we try to achieve weaker nonce robustness, i.e., *multi-user Chosen-Ciphertext Attack security with misuse*-**resilience** *and Leakage*. Informally, the notions allow the adversary $\mathcal{A}$ to query leaking encryption and (leaking) decryption oracles in arbitrary, and captures the confidentiality of several messages encrypted using **fresh, non-repeating nonces**. The latter goal is formalized via the "old-school" left-or-right paradigm, i.e., $\mathcal{A}$ shall not tell apart encrypting M^0 from encrypting M^1. As mentioned, the term misuse-**resilience** capturing confidentiality at fresh nonce was first introduced by Ashur et al. [4], and we follow.

To capture the real-world feature that the challenge messages will eventually be decrypted and give leakages that may cinch message recovering (relevant in applications such as secure bootloading [36]), an oracle $\mathsf{L}_{\mathsf{Decch}}$ providing these "challenge dec-leakages" may be given to $\mathcal{A}$. In all, following [29,8], the notion capturing 1 leakage source (i.e., encryption only) is abbreviated as $\mathsf{muCCAmL1}$, while that capturing the presence of 2 leakages sources (i.e., including decryption) is $\mathsf{muCCAmL2}$. See below for the formal definitions.

Definition 3 (muCCAmL2 & muCCAmL1 **advantages**). *Given the implementation of a nonce-based authenticated encryption* AEAD = (Enc, Dec) *with leakage function pair* L = (L$_{\mathsf{Enc}}$, L$_{\mathsf{Dec}}$)*, 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 $\mathsf{PrivK}^{\mathsf{muCCAmL2},d}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *is defined in Figure 1 and where the security game* $\mathsf{PrivK}^{\mathsf{muCCAmL1},d}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ *removes all the dec-leakages* leak_{dec} *and* leak^{d}_{dec} *from Figure 1.*

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

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

Leaking encryption queries: $\mathcal{A}$ 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 $\mathsf{leak}_{\mathsf{enc}} \leftarrow \mathsf{L}_{\mathsf{Enc}}(K_i,N,A,M)$, updates $\mathcal{E}_i \leftarrow \mathcal{E}_i \cup \{N\}$ and returns $(C,\mathsf{leak}_{\mathsf{enc}})$.

Leaking decryption queries: $\mathcal{A}$ 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 $\mathsf{leak}_{\mathsf{dec}} \leftarrow \mathsf{L}_{\mathsf{Dec}}(K_i,N,A,C)$ and returns $(M,\mathsf{leak}_{\mathsf{dec}})$;

Challenge queries: on possibly many occasions $\mathcal{A}$ 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 $C^d \leftarrow \mathsf{Enc}_{K_i}(N_{ch},A_{ch},M^d)$ and $\mathsf{leak}^d_{\mathsf{enc}} \leftarrow \mathsf{L}_{\mathsf{Enc}}(K_i,N_{ch},A_{ch},M^d)$, updates $\mathcal{E}_{ch} \leftarrow \mathcal{E}_{ch} \cup \{(i,N_{ch},A_{ch},C^d)\}$ and finally returns $(C^d,\mathsf{leak}^d_{\mathsf{enc}})$;

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

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

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

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

We'll also use a notion of *eavesdropper security with leakage*. Since it's a bit specific for "one-time" stream ciphers, we defer it to Section 3.

Challenge Leakages and Implications. It's worth noting that our definitions explicitly allow the challenge queries to leak, a feature that was believed impossible (and dismissed) by the theory community. However, we argue results in such models shed more light on real world implementations (i.e., their message processing leakages *shall* be somewhat "bounded"). A recent survey of Kalai and Reyzin [32] interpreted our results as using very strong assumption of "leak-free message processing". To certain extent we agree, but we remark that such assumptions are something the real-world implementations have to approach, i.e., the implementations shall be "theoretically prefect" as long as theoretically exponential security *is* the target. On the other hand, our results aren't limited to this confine. The real world is typically far from theoretical perfectness. For example, AES is sometimes assumed to be a 0.99-PRP rather than "exponentially secure" [33,41]. In a similar manner, one may implement the message manipulation to be 0.9999-secure, and we still have meaningful implications in this case.

We stress that we are *not* saying the classical models without challenge leakages are useless: they do fit into many scenarios. We are just saying that we need a *complementary model* and corresponding results highlighting relevant issues—as agreed by some of the relevant authors.[6] Technically, to have a bounded muCCAmL2 advantage, we'll assume the "basic" message manipulation (mainly a XOR action) has a bounded leakage distinguishing advantage, and then reduce the muCCAmL2 advantage to the "basic": see Section 3.3 for details.

3 Duplex stream cipher and its eavesdropper security with leakage (EavL)

In this section we'll investigate the leakage confidentiality of a duplex-based stream cipher that appears as a part of many AEs. Leakage confidentiality is in general hard to achieve (and analyze) and assumptions that "bound"

[6]Daniel Martin, author of [5]; personal communication.

the information in leakages are clearly necessary, as otherwise cleartext messages may leak in full. In this respect, we start with an introduction to the general leakage model (oracle-free probabilistic leakage functions) that we will employ (Section 3.1). Then we formalize our two assumptions that bound the information in leakages, i.e., the non-invertibility assumption in Section 3.2, and the bounded XOR leakage assumption in Section 3.3. Based on these, in Section 3.4 we formally define the duplex-based stream cipher and its leakages and prove its EavL security. We serve some discussion in Section 3.5.

3.1 Modeling leakages

Most analyses of sponge-based constructions rely on the ideal (permutation) model (see section 3.5 for an alternative). We follow that practice, which has the advantage of offering an easy compatibility with quite minimal leakage assumptions: we will assume that leakages resulting from each call to the permutation π is non-invertible. This assumption was previously used by Yu et al. [43] on a random oracle-based PRG. 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 section), and therefore might lead to a better understanding of how to implement and design modes from a real-world perspective.

Formally, for $\mathsf{F} \in \{\pi, \oplus\}$, i.e., the components of a duplex, each computation of F comes with leakages L_{F}, which is an *efficient probabilistic* function of the values involved in the computation. For the permutation π, we further split the leakage into an *input* and an *output* part, i.e., we write $(\mathsf{L}_{\pi}^{in}(S^{in}), \mathsf{L}_{\pi}^{out}(S^{out}))$ for the leakage due to evaluating $\pi(S^{in}) \to S^{out}$ or $\pi^{-1}(S^{out}) \to S^{in}$. This distinction between L_{π}^{in} and L_{π}^{out} allows to independently quantify the secrecy of the input and the output which better reflects the designers implementation goals for each functions/calls.

We insist again on the probabilistic nature of *all* the leakage functions. This 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 noise in devices for instance. For simplicity, we'll use superscript p to denote vectors of p repeated measures of the same computation: e.g., $[\mathsf{L}_{\pi}^{in}(S^{in}), \mathsf{L}_{\pi}^{out}(S^{out})]^p$ denotes the leakages resulted from measuring the evaluation $\pi(S^{in}) \to S^{out}$ p times.

Oracle-freeness. We enforce that for any F, the leakage function L_{F} must have *no access to the ideal oracle* π. This restriction effectively excludes the artificial "future computation attacks" from the model: it guarantees that L_{F} only leaks information about the computation that is happening in the device rather than the computation that may happen in "future" calls of π. Oracle-freeness restriction was first used by Yu et al. [43], and recently by the concurrent work of DM [21]. For the sake of space, we refer to [23] and [43] for detailed discussion about "future computation attacks".

3.2 Bounding the leakages: Non-invertibility restriction

Motivation. As mentioned before, we assume that (some of) the leakage functions are essentially non-invertible. In detail, note that the following sequence of actions appear in virtually all duplex-based AEs: (1) squeezing (the most significant bits, wlog) from a secret b-bit state S_i: $Y_i \leftarrow \mathsf{ms}_r(S_i)$; (2) modifying the state with a b-bit offset Δ: $S_i' \leftarrow S_i \oplus \Delta$; (3) deriving a new state: $S_{i+1} \leftarrow \pi(S_i')$. After the 3rd step, the subsequent computations are irrelevant to the state S_i. We therefore view this as a basic unit and bound its leakages on S_i.

Let's reconsider the above unit. Note that for the call $\pi(S_i')$ is step (3), in many cases the halve $\mathsf{ms}_r(S_i')$ is a non-secret ciphertext block, see Fig. 3 or 4 (on the other hand, the halve $\mathsf{ls}_c(\Delta)$ captures the possible XOR of domain separation bits). Therefore, the least significant bits $\mathsf{ls}_c(S_i')$ are the critical secret, and the subsequent actions are compromised as long as $\mathsf{ls}_c(S_i')$ is recovered. These c bits are involved in three actions: the 1st is of course the call $\pi(S_i')$, the others are: (a) the XOR of Δ, and (b) the "previous" π-call $\pi(\star) \to S_i$ that "produces" S_i–following the convention in duplex papers, we didn't present this action in the above unit, but it's clearly relevant. We shouldn't be restricted to $\mathsf{ls}_c(S_i')$, as we may also consider initializing the duplex with a key of $\kappa \neq c$ bits. Therefore, in our assumption we consider ω bits $\mathsf{ls}_\omega(S_i')$ for generality, and assume that the side-channel adversary cannot predicate the value of $\mathsf{ls}_\omega(S_i')$ within a limited number of guesses, even if all the other involved $(b-\omega)$-bit values are *chosen* by him (this simplification also emphasizes the crucial role of $\mathsf{ls}_\omega(S_i')$).

Definition. Formally, we define

$$\mathbf{Adv}^{\mathsf{Inv}[\omega]}(\mathcal{A}) := \Pr\big[s_{ch} \xleftarrow{\$} \{0,1\}^\omega, \mathcal{G} \leftarrow \mathcal{A}^{\pi}(\mathsf{leak}) : s_{ch} \in \mathcal{G}\big], \tag{1}$$

where $\mathcal{G}$ is a finite set of guesses, and $\mathcal{A}$'s input leak is a list of leakages depending on three values $y^{in}, y^{pre} \in \{0,1\}^{b-\omega}$, and $\delta \in \{0,1\}^{\omega}$ chosen by $\mathcal{A}$, i.e.,

$$\mathsf{leak} = \left[\mathsf{L}_\pi^{out}(y^{pre}\|s_{ch}), \mathsf{L}_\oplus(\delta, s_{ch}), \mathsf{L}_\pi^{in}(y^{in}\|(\delta \oplus s_{ch}))\right]^p. \tag{2}$$

Further Insights. To clarify, the random state s_{ch} is the secret that is to be Challenged by $\mathcal{A}$. $\mathcal{A}$ is required to choose y^{pre}, y^{in}, and δ (playing the role of the aforementioned $\mathsf{ls}_c(\Delta)$) to "fill in the gap" and get the leakages, as if $y^{pre}\|s_{ch}$ is the aforementioned current state S_i of the "unit", which is modified to $y^{in}\|(\delta \oplus s_{ch})$ and results in a call to $\pi(y^{in}\|(\delta \oplus s_{ch}))$. See Fig. 2 (Left) for illustration. As mentioned, s_{ch} is involved in three actions, which exactly correspond to the three leakages. To capture the possibility of repeated dec-leakages as formalized by $\mathsf{muCCAmL2}$ we allow $\mathcal{A}$ to measure the leakages for multiple p times, and thus leak contains p repeats: this may also enable an application to the rate-1 duplex in ISAP. 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 [42]). Namely, in practice, it is expected that the (SPA) advantage of Eq. (1) is still much smaller than that of a DPA against a block protected with similar countermeasures.

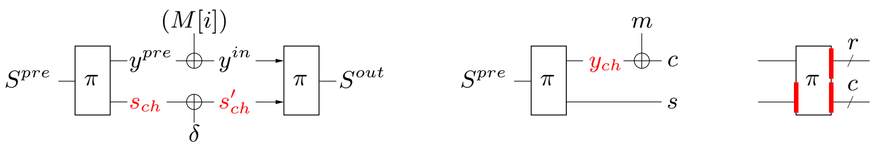

Fig. 2: (Left) Illustrating the $\mathsf{Inv}[\omega]$ assumption. The value in red (i.e., s_{ch}) is the critical secret. The involved values are as defined with Eq. (1), while the values S^{pre} and S^{out} are only mentioned in the subsequent security tester. (Middle) The "basic" message manipulating operation. The value y_{ch} in red is the secret. (Right) A summary: which values should be somehow ensured secret (i.e., the ls_ω bits of the input, and the entire output, as stressed by the red bold lines).

In such an invertibility game, the power of $\mathcal{A}$ is quantified along four dimensions, i.e., the number q_π of its queries to π, the number p of repeated leakage measures, the running time t, and the number N_G of allowed guesses (i.e., $|\mathcal{G}| \le N_G$; clearly the larger N_G, the higher $\mathbf{Adv}^{\mathsf{Inv}[\omega]}(\mathcal{A})$). To simplify, we further define

$$\mathbf{Adv}^{\mathsf{Inv}[\omega]}(p, q_\pi, t, N_G) := \max_{(p,q_\pi,t,N_G)\text{-}\mathcal{A}} \left\{\mathbf{Adv}^{\mathsf{Inv}[\omega]}(\mathcal{A})\right\}, \tag{3}$$

When $\omega = c$, the assumption captures the secrecy of the "capacity halve", which is in line with the intuition proposed for ISAP [19].

Tester: Measuring in practice. The concrete value of the advantage $\mathbf{Adv}^{\mathsf{Inv}[\omega]}$ can be measured for a specific implementation on a specific device by running the best known side-channel key/secret recovery adversary $\mathcal{A}$ against the following tester. This along with our Theorems later (e.g., Theorem 3) allows to determine how many plaintext blocks can be processed before key updating.

```
1: Tester for non-invertibility $\mathbf{Adv}^{\mathsf{Inv}[\omega]}(p, q_\pi, t, N_G)$
2: Let the challenging adversary $\mathcal{A}$ serve $b-\omega$ bit values $y^{pre}$ and $y^{in}$ and $\omega$-bit offset $\delta$
3: Pick the secret: $s_{ch} \xleftarrow{\$} \{0,1\}^\omega$
4: Repeating $S^{pre} \leftarrow \pi^{-1}(y^{pre}\|s_{ch})$, $s'_{ch} \leftarrow s_{ch} \oplus \delta$, and $S^{out} \leftarrow \pi^{-1}(y^{in}\|s'_{ch})$ for $p$ times
5: Serve $\mathcal{A}$ with the leakages resulted from step 4. Following our notations, this gives $\mathcal{A}$ the leakages $[\mathsf{L}_\pi^{out}(y^{pre}\|s_{ch})]^p$, $[\mathsf{L}_\oplus(s_{ch}, \delta)]^p$, and $[\mathsf{L}_\pi^{in}(y^{in}\|s'_{ch})]^p$.
6: Let $\mathcal{A}$ output $N_G$ guesses $\mathcal{G} = \{s_1, \ldots, s_{N_G}\}$, and $\mathcal{A}$ wins as long as $s_{ch} \in \mathcal{G}$
```

3.3 (In)Distinguishability of the XOR leakages

Motivation. As mentioned in Section 2, an implementation that well protects the sensitive messages *shall* offer some indistinguishability property in its message manipulations. To minimize assumptions & ease measuring in practice, we follow the methodology of [37,8]: we define such an assumption w.r.t. the information an adversary might extract from the "basic" message manipulation made in a keyed duplex (i.e., the aforementioned unit), and then reduce the confidentiality of the "bigger" encryption to this assumption.

Definition. Note that, when running the aforementioned unit, a part of the state—typically of r bit,—is extracted as a key stream block, see Fig. 2 (middle). Concerning the involved leakages, and with the goal of bounding LOR distinguishing advantages with Leakages in mind, we define

$$\mathbf{Adv}^{\mathsf{LORL}}(\mathcal{A}) := \Bigg| \Pr\Big[y_{ch} \xleftarrow{\$} \{0,1\}^r, c^0 \leftarrow y_{ch} \oplus m^0 : \mathcal{A}^{\pi}(c^0, \mathsf{leak}_0) \Rightarrow 1\Big] \\ - \Pr\Big[y_{ch} \xleftarrow{\$} \{0,1\}^r, c^1 \leftarrow y_{ch} \oplus m^1 : \mathcal{A}^{\pi}(c^1, \mathsf{leak}_1) \Rightarrow 1\Big] \Bigg| , \tag{4}$$

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

$$\mathsf{leak}_d = \Big(\big[\mathsf{L}_{\pi}^{out}(y_{ch} \| s)\big]^p, \mathsf{L}_{\oplus}(y_{ch}, m^d), \big[\mathsf{L}_{\oplus}(y_{ch}, c^d)\big]^{p-1} \Big). \tag{5}$$

Further Insights. Concretely, the sensitive is the key stream block y_{ch}. It's the output of a permutation-call, hence the presence of $\mathsf{L}_{\pi}^{out}(y_{ch}\|s)$. Then, y_{ch} is used to mask the message block, and thus $\mathsf{L}_{\oplus}(y_{ch}, m^d)$ comes. Presenting $\big[\mathsf{L}_{\pi}^{out}(y_{ch}\|s)\big]^p$, i.e., p repeats, as well as $[\mathsf{L}_{\oplus}(y_{ch}, c^d)]^{p-1}$ the $p-1$ leakages from the "decryption direction", stem from the (multiple) challenge dec-leakages requirements in the $\mathsf{muCCAmL2}$ game: this resembles Eq. (2). We also define

$$\mathbf{Adv}^{\mathsf{LORL}}(p, q_{\pi}, t) := \max_{\mathcal{A}} \Big\{ \mathbf{Adv}^{\mathsf{LORL}}(\mathcal{A}) \Big\}. \tag{6}$$

This quantity can again be measured by a tester (given in Appendix F.1). As discussed in [30,37], if a single XOR of the message leaks a single bit, then no confidentiality 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, the advantages of this methodology are two-fold: in theory, it could help establish somewhat *best possible* mode-level leakage security for relevant applications; in practice, it allows us to faithfully reduce the confidentiality to very simplified one-time components, for a single computation from a fixed random string, which further makes it easier to study and to protect as isolated components.

To ease understanding, a summary of the critical values and their leakages is given in Fig. 2 (right).

3.4 Leakage EavL security of duplex stream ciphers

We'll first formally define the duplex-based stream cipher $\mathsf{DuStr}[\pi]$ in question. We then define a security notion of *(left-or-right) Eavesdropper security with Leakage* (EavL) as a tool to characterize the leakage confidentiality of $\mathsf{DuStr}[\pi]$: briefly, it states that two distinct messages encrypted by $\mathsf{DuStr}[\pi]$ are indistinguishable. It's hard to prove it "directly": therefore we follow [37] and define an idealized stream cipher IdealS for "relay". Technically, we first prove that $\mathsf{DuStr}[\pi]$ and IdealS are indistinguishable with leakages. The gap between IdealS encrypting two messages is easier to analyze, and can be bounded to $O(\ell) \cdot \mathbf{Adv}^{\mathsf{LORL}}$: these enable us to conclude on $\mathsf{DuStr}[\pi]$.

The leaking duplex stream cipher and its ideal reference. In detail, we consider the stream cipher defined by the following pseudocode, which constitutes the basis of many duplex-based AEs (e.g., Ascon, and TETSponge that will be introduced). For simplicity we assume the size of the inputs $|A|$ and $|M|$ are always multiples of r (otherwise it would be too complicated to follow!), and we use an offset δ_1 to model the commonly used domain separation bits in duplex AEs, which depends on the concrete schemes and is typically of only 1 or 2 bits. For preciseness we also make the leakages of each step explicit. Note that, to fit into the (multiple) challenge dec-leakages requirements of $\mathsf{muCCAmL2}$, the leakages from the "decryption direction" are allowed to be repeated $p-1$ times (e.g., see the vector $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), C[i])]^{p-1}$ in step (4) below).

```
The duplex stream cipher $\mathsf{DuStr}_B[\pi](IV, A, M)$, $|B| = \kappa$:
(1) Computes $S_0' \leftarrow IV\|B$, $S_1 \leftarrow \pi(S_0')$. **The leakages** of this step are $[\mathsf{L}_{\pi}^{in}(S_0')]^p$ and $[\mathsf{L}_{\pi}^{out}(S_1)]^p$;
(2) For $i = 1, \ldots, \nu$, $\nu = |A|/r$, computes $S_i' \leftarrow (A[i]\|0^c) \oplus S_i$ and $S_{i+1} \leftarrow \pi(S_i')$. **The leakages** of this step are
    $[\mathsf{L}_{\pi}^{in}(S_i'), \mathsf{L}_{\pi}^{out}(S_{i+1}), \mathsf{L}_{\oplus}(\mathsf{ms}_r(S_i), A[i])]^p$;
(3) $S_{\nu+1} \leftarrow S_{\nu+1} \oplus (0^r\|\delta_1)$ for a fixed offset $\delta_1$. **The leakages** are $[\mathsf{L}_{\oplus}(\mathsf{ls}_c(S_{\nu+1}), \delta_1)]^p$;
(4) For $i = 1, \ldots, \ell$, $\ell = |M|/r$, computes $j \leftarrow i+\nu$, $C[i] \leftarrow \mathsf{ms}_r(S_j) \oplus M[i]$, $S_j' \leftarrow C[i]\|\mathsf{ls}_c(S_j)$, $S_{j+1} \leftarrow \pi(S_j')$. **Leakages**
    are $\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), M[i])$, $[\mathsf{L}_{\oplus}(\mathsf{ms}_r(S_j), C[i])]^{p-1}$, $[\mathsf{L}_{\pi}^{in}(S_j'), \mathsf{L}_{\pi}^{out}(S_{j+1})]^p$;
```

(5) Returns $\mathbf{c} = C[1]\| \ldots \|C[\ell]$.

Its ideal reference IdealS is obtained via replacing all the internal actions $S_{i+1} \leftarrow \pi(S'_i)$ in $\mathsf{DuStr}[\pi]$ by sampling $S_{i+1} \xleftarrow{\$} \{0,1\}^b$. Formally,

```
The ideal stream cipher $\mathsf{IdealS}(IV, A, M)$:
(1) Samples $B \xleftarrow{\$} \{0,1\}^\kappa$;
(2) Computes $S'_0 \leftarrow IV\|B$ and samples $S_1 \xleftarrow{\$} \{0,1\}^b$. **The leakages** of this step are $[\mathsf{L}_\pi^{in}(S'_0)]^p$ and $[\mathsf{L}_\pi^{out}(S_1)]^p$;
(3) For $i = 1, \ldots, \nu$, $\nu = |A|/r$, computes $S'_i \leftarrow (A[i]\|0^c) \oplus S_i$ and samples $S_{i+1} \xleftarrow{\$} \{0,1\}^b$. **The leakages** are $[\mathsf{L}_\pi^{in}(S'_i), \mathsf{L}_\pi^{out}(S_{i+1}), \mathsf{L}_\oplus(\mathsf{ms}_r(S_i), A[i])]^p$;
(4) $S_{\nu+1} \leftarrow S_{\nu+1} \oplus (0^r\|\delta_1)$. **The leakages** are $[\mathsf{L}_\oplus(\mathsf{ls}_c(S_{\nu+1}), \delta_1)]^p$;
(5) For $i = 1, \ldots, \ell$, $\ell = |M|/r$, computes $j \leftarrow i + \nu$, $C[i] \leftarrow \mathsf{ms}_r(S_j) \oplus M[i]$, $S'_j \leftarrow C[i]\|\mathsf{ls}_c(S_j)$, and samples $S_{j+1} \xleftarrow{\$} \{0,1\}^b$. **The leakages** are $\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), M[i])$, $[\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), C[i])]^{p-1}$, $[\mathsf{L}_\pi^{in}(S'_j), \mathsf{L}_\pi^{out}(S_{j+1})]^p$;
(6) Returns $\mathbf{c} = C[1]\| \ldots \|C[\ell]$.
```

EavL: Eavesdropper security with leakage. The EavL notion can be seen as the leaky version of the black-box eavesdropper security, or an extremely weakened $\mathsf{muCCAmL2}$ setting that *only allows the adversary to make a single query to the challenge leaking encryption oracle and have its ciphertext, encryption leakages, and several dec-leakages* (as such, it can also be viewed as "single-query leakage CCA"). Formally,

$$\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(\mathcal{A}) := \big| \Pr[\mathcal{A}^\pi(\mathsf{LDuStr}_B(IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^\pi(\mathsf{LDuStr}_B(IV, A, M^1)) \Rightarrow 1] \big|. \tag{7}$$

Throughout this subsection, we call an adversary (p, q_π, t)*-bounded*, if it makes q_π permutation queries, measures the dec-leakages for $p - 1$ times, and runs in time t.

Indistinguishability of the real and ideal. The leakage functions $\mathsf{L}_{\mathsf{DuStr}}$, resp. $\mathsf{L}_{\mathsf{IdealS}}$, include all the leakages mentioned with the above definitions of $\mathsf{DuStr}_B[\pi]$, resp. IdealS. Recall that $\mathsf{LDuStr}_B[\pi](IV, A, M) := (\mathsf{DuStr}_B[\pi](IV, A, M), \mathsf{L}_{\mathsf{DuStr}}(B, IV, A, M))$ and $\mathsf{LIdealS}(IV, A, M) := (\mathsf{IdealS}(IV, A, M), \mathsf{L}_{\mathsf{IdealS}}(IV, A, M))$ according to our conventions. We show that the leaking objects $\mathsf{LDuStr}_B[\pi]$ and $\mathsf{LIdealS}$ are indistinguishable upon processing a single input (IV, A, M). The crux is to bound certain "bad events" during the execution of IdealS, for which we leverage the non-invertible leakage assumption.

Lemma 1. *For every (p, q_π, t)-bounded distinguisher $\mathcal{D}^\pi$ and every adversary-chosen triple $\mathbf{tpl} = (IV, A, M)$ such that (A, M) has ℓ blocks in total, it holds*

$$\begin{aligned} &\big| \Pr[\mathcal{D}^\pi(\mathsf{LDuStr}_B[\pi](\mathbf{tpl})) \Rightarrow 1] - \Pr[\mathcal{D}^\pi(\mathsf{LIdealS}(\mathbf{tpl})) \Rightarrow 1] \big| \\ \le &\frac{(\ell+2)^2}{2^{c+1}} + \mathbf{Adv}^{\mathsf{Inv}[\kappa]}(p, q_\pi, t^*, 2q_\pi) + (\ell+1) \cdot \mathbf{Adv}^{\mathsf{Inv}[c]}(p, q_\pi, t^*, 2q_\pi), \end{aligned} \tag{8}$$

where $t^* = O(t + p\ell t_l)$, and t_l is the total time needed for evaluating L_π^{in}, L_π^{out}, $\mathsf{L}_\oplus$, and the xor of two r-bit values.

Proof. Wlog we consider the case of $|A| = 0$ for simplicity.

Preparations. Denote $\mathsf{G}_1(\mathcal{D}, \mathsf{DuStr}_B[\pi], \pi)$ and $\mathsf{G}_2(\mathcal{D}, \mathsf{IdealS}, \pi)$ the games capturing the interactions between $\mathcal{D}$ and the real $(\mathsf{DuStr}_B[\pi], \pi)$ and the ideal (IdealS, π) 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 coins of the leakage functions, and these could only be summarized by extended transcripts.

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 $\tau_\pi = \big((S_1^{in}, S_1^{out}), \ldots, (S_{q_\pi}^{in}, S_{q_\pi}^{out})\big)$ 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 values $\mathbf{S} = (S'_0, S_1, S'_1, \ldots)$ to $\mathcal{D}$, and append it to the transcript. Clearly, this doesn't reduce its advantage.

Moreover, note that the interactions with the (real or ideal) stream cipher 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 $\mathsf{G}_1(\mathcal{D}, \mathsf{DuStr}_B[\pi_1], \pi_1)$ and $\mathsf{G}_2(\mathcal{D}, \mathsf{IdealS}, \pi_2)$, 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.

It's because all the internal actions in $\mathsf{G}_1(\mathcal{D}, \mathsf{DuStr}_B[\pi_1], \pi_1)$ and $\mathsf{G}_2(\mathcal{D}, \mathsf{IdealS}, \pi_2)$ 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 $\mathsf{G}_2(\mathcal{D}, \mathsf{IdealS}, \pi)$ 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 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 $\pi \vdash \tau_\pi$, $\pi(S'_i) = S_{i+1}$ for every S'_i, S_{i+1} 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 $i, j \in [0, \ldots, \ell]$ such that $S'_i = S'_j \wedge S_{i+1} \neq S_{j+1}$, or $S'_i \neq S'_j \wedge S_{i+1} = S_{j+1}$;
– (B-2) *exposure of secret state*: if any of the following is fulfilled:
  - there exits $i \in [0, \ldots, \ell]$ such that $(S'_i, \star) \in \tau_\pi$; or
  - there exits $i \in [1, \ldots, \ell+1]$ such that $(\star, S_i) \in \tau_\pi$.

As long as the c-bit (uniformly distributed) capacity halves of $S'_0, S_1, S_2, \ldots, S_{\ell+1}$ don't collide, the condition (B-1) cannot be fulfilled. Therefore,

$$\Pr[\text{(B-1)}] \leq \binom{\ell+2}{2} \cdot \frac{1}{2^c} = \frac{(\ell+2)^2}{2^{c+1}}. \tag{9}$$

To bound Pr[(B-2)], we need the non-invertible leakage assumption. Consider an execution of G_2 with the inputs (IV, M). We divide (B-2) into three subevents:

(1) $\mathsf{BadInner}$: that occurs when there exists an index $i \in [1, \ldots, \ell]$ such that $(\star, S_i) \in \tau_\pi$ or $(S'_i, \star) \in \tau_\pi$;
(2) $\mathsf{BadInit}$: that occurs when $(S'_0, \star) \in \tau_\pi$;
(3) $\mathsf{BadFinal}$: that occurs when $(\star, S_{\ell+1}) \in \tau_\pi$.

Consider $\mathsf{BadInner}$ first. We follow Yu et al. [43, Appendix A] (i.e., their argument for an event Query_a with somewhat similar meaning): given an adversary $\mathcal{D}^\pi$, we construct an adversary $\mathcal{A}^\pi$ such that

$$\mathbf{Adv}^{\mathsf{Inv}[c]}(\mathcal{A}) \leq \Pr_{\mathbf{r}, \mathbf{S}, \pi}[\mathsf{BadInner} \text{ in } \mathcal{D}^\pi(\mathsf{IdealS}(IV, M))]. \tag{10}$$

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 a key $B \xleftarrow{\$} \{0,1\}^\kappa$, sets $S'_0 \leftarrow IV \| B$, and initializes a list leak with the leakages $[\mathsf{L}_\pi^{in}(S'_0)]^p$;

(2) For $j = 1, \dots, i-1$, $\mathcal{A}^\pi$ samples the new state $S_j \xleftarrow{\$} \{0,1\}^b$, computes $C[j] \leftarrow \mathsf{ms}_r(S_j) \oplus M[j]$, and:
   – computes $S'_j \leftarrow C[j] \| (\mathsf{ls}_c(S_j) \oplus \delta_1)$ and adds the leakages $[\mathsf{L}_\oplus(\mathsf{ls}_c(S_j), \delta_1)]^p$ to leak when $j = 1$, and
   – computes $S'_j \leftarrow C[j] \| \mathsf{ls}_c(S_j)$ otherwise.

   $\mathcal{A}^\pi$ further adds the leakages $\left[\mathsf{L}_\pi^{out}(S_j), \mathsf{L}_\pi^{in}(S'_j)\right]^p$, $\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), M[j])$, and $[\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), C[j])]^{p-1}$ 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 $y, C[i]$, and $\delta = \delta_1$ (when $i = 1$) or $\delta = 0$ (when $i > 1$) to its $\mathsf{Inv}[c]$ challenger and obtains the leakages

$$\mathsf{leak}_{ch} = \left[\mathsf{L}_\pi^{out}(y \| s_{ch}), \mathsf{L}_\oplus(s_{ch}, \delta), \mathsf{L}_\pi^{in}\big(C[i] \| (s_{ch} \oplus \delta)\big)\right]^p$$

   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 completes the remaining encrypting: for $j = i+1, \dots, \ell$, $\mathcal{A}^\pi$ samples $S_j \xleftarrow{\$} \{0,1\}^b$, computes $C[j] \leftarrow \mathsf{ms}_r(S_j) \oplus M[j]$, $S'_j \leftarrow C[j] \| \mathsf{ls}_c(S_j)$, and adds $\left[\mathsf{L}_\pi^{out}(S_j), \mathsf{L}_\pi^{in}(S'_j)\right]^p$, $\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), M[j])$, and $[\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), C[j])]^{p-1}$ to leak.

(5) Finally, $\mathcal{A}^\pi$ samples $S_{\ell+1} \xleftarrow{\$} \{0,1\}^b$, adds $[\mathsf{L}_\pi^{out}(S_{\ell+1})]^p$ to leak, returns $(C[1] \| \dots \| C[\ell], \mathsf{leak})$ 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 $\mathcal{G}$.

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 $\mathsf{IdealS}(IV, M)$ and provides the leakages to $\mathcal{D}$, except for the i index, for which the leakages are replaced by those obtained from an Inv 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 with s_{ch}, then outputting the aforementioned set $\mathcal{G}$ (based on τ_π) would break the Inv game. So we have

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

Now, we observe that

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

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

$$\begin{aligned}\Pr[\mathsf{BadInner} \text{ in } \mathsf{G}_2(\mathcal{D}, \mathsf{IdealS}, \pi)] &\leq \ell \cdot \Pr[s_{ch} \in \mathcal{G}] \\ &\leq \ell \cdot \mathbf{Adv}^{\mathsf{Inv}[c]}(\mathcal{A}) \quad \text{(Eq. 1)} \\ &\leq \ell \cdot \mathbf{Adv}^{\mathsf{Inv}[c]}(p, q_\pi, t^*, 2q_\pi). \quad \text{(Eq. 3)}\end{aligned}$$

These finish the analysis of $\mathsf{BadInner}$. For the events $\mathsf{BadInit}$ and $\mathsf{BadFinal}$, similar arguments could establish

$$\Pr[\mathsf{BadInit}] \leq \mathbf{Adv}^{\mathsf{Inv}[\kappa]}(p, q_\pi, t^*, 2q_\pi), \quad \Pr[\mathsf{BadFinal}] \leq \mathbf{Adv}^{\mathsf{Inv}[c]}(p, q_\pi, t^*, 2q_\pi).$$

The three terms plus Eq. (9) yield

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \;\leq (\ell+1) \cdot \mathbf{Adv}^{\mathsf{Inv}[c]}(p, q_\pi, t^*, 2q_\pi) + \mathbf{Adv}^{\mathsf{Inv}[\kappa]}(p, q_\pi, t^*, 2q_\pi) + \frac{(\ell+2)^2}{2^{c+1}}. \tag{11}$$

Summarizing. For any good transcript $\tau = (\tau_\pi, \mathbf{S}, \mathbf{r})$ we have

$$\Pr[T_{id} = \tau] = \Pr[\pi \vdash \tau_\pi] \cdot \Pr[\mathbf{r}] \cdot \Pr[\mathbf{S}] = \Pr[\pi \vdash \tau_\pi] \cdot \Pr[\mathbf{r}] \cdot \frac{1}{2^\kappa} \cdot \frac{1}{2^{(\ell+1)b}},$$

as $\Pr[S'_0] = \Pr[B] = \frac{1}{2^\kappa}$, while $\Pr[S_i] = \frac{1}{2^b}$ for $i = 1, \dots, \ell+1$. Whereas

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

Conditioned on ¬(B-1) and ¬(B-2), it can be seen $\Pr[\forall i \in [0,\ell] : \pi(S'_i) = S_{i+1} \mid \pi \vdash \tau_\pi \mid \forall j < i : \pi(S'_j) = S_{j+1}] \geq \frac{1}{2^{(\ell+1)b}}$. Therefore, $\Pr[T_{re} = \tau] \geq \Pr[T_{id} = \tau]$, and thus $\Pr[T_{id} \in \mathcal{T}_{bad}]$ constitutes the final bound, i.e.,

$$\begin{aligned}&\left|\Pr[\mathcal{D}^\pi(\mathsf{LDuStr}_B[\pi](IV,M)) \Rightarrow 1] - \Pr[\mathcal{D}^\pi(\mathsf{LIdealS}(IV,M)) \Rightarrow 1]\right| \\ \leq &\frac{(\ell+2)^2}{2^{c+1}} + (\ell+1)\cdot \mathbf{Adv}^{\mathsf{Inv}[c]}(p,q_\pi,t^*,2q_\pi) + \mathbf{Adv}^{\mathsf{Inv}[\kappa]}(p,q_\pi,t^*,2q_\pi).\end{aligned}$$

as claimed. □

Summarizing: EavL security of DuStr. The bound appears like twice the terms in Lemma 1 plus the term $\ell\mathbf{Adv}^{\mathsf{LORL}}$, which is due to masking the ℓ message blocks with ℓ independent random key stream blocks.

Theorem 1. *For every (p,q_π,t)-bounded adversary $\mathcal{A}$, every pair of messages M^0 and M^1 of equal-length, and every (IV,A) such that $\lceil\frac{|A|}{r}\rceil + \lceil\frac{|M^0|}{r}\rceil \leq \ell$, we have*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(\mathcal{A}) \leq &\frac{(\ell+2)^2}{2^c} + \ell\cdot\mathbf{Adv}^{\mathsf{LORL}}(p,q_\pi,t^*) + 2\mathbf{Adv}^{\mathsf{Inv}[\kappa]}(p,q_\pi,t^*,2q_\pi) \\ &+ 2(\ell+1)\cdot\mathbf{Adv}^{\mathsf{Inv}[c]}(p,q_\pi,t^*,2q_\pi), \qquad (12)\end{aligned}$$

where t_l is as defined in Lemma 1, and $t^* = O(t + p\ell t_l)$. See appendix F.3 for its (simple hybrid-based) proof.

Interpretations. Theorem 1 doesn't give rise to an encryption scheme since it concerns with encrypting only 1 message. But as will be seen, it can be used to establish AE security in an almost modular manner.

Regarding bounds, the term $\ell\cdot\mathbf{Adv}^{\mathsf{LORL}}(p,q_\pi,t^*)$ reflects the reduction to the "minimal" message manipulation (as discussed), and the factor ℓ reflects a (seemingly) unavoidable leakage security loss. The terms $2\mathbf{Adv}^{\mathsf{Inv}[\kappa]}(p,q_\pi,t^*,2q_\pi) + 2(\ell+1)\cdot\mathbf{Adv}^{\mathsf{Inv}[c]}(p,q_\pi,t^*,2q_\pi)$ capture the hardness of side-channel secret recovery, and they are roughly of some birthday type

$$O\Big(\frac{q_\pi+\ell+t}{\mu_\kappa\cdot 2^\kappa}\Big) + O\Big(\ell\cdot\frac{q_\pi+\ell+t}{\mu_c\cdot 2^c}\Big)$$

for some parameters μ_κ and μ_c that depend on the concrete conditions. It is nowadays a common assumption that with such a small data complexity (2 or 3 leakage traces) μ_κ and μ_c would be very small [19], so we "restore" the classical $2^{c/2}$ security. The birthday bounds are essentially tight w.r.t. our assumptions: a collision between the (internal) secret c-bit state values allows the adversary to obtain more than 2 leakages about a single secret value, which is beyond our assumption (security with 2 leakages). In all, our assumption Eq. (2), though a bit conservative, helps us tightly restore the classical $2^{c/2}$ birthday bound.

3.5 Discussion

Where does Leakage Resilience Come from? Note that in (the model of) a duplex, the internal state is kept evolving via calling a perfectly random permutation. And since the adversary can't recover the secret, he can't predicate what would be the next internal permutation-call. Therefore, the subsequent state is derived via a fresh permutation-call and thus random and secret.

On reducing to PRP assumptions. An alternative solution to study keyed sponge/duplex is to reduce them to the PRP security of a "Partial-Key" Even-Mansour (PKEM) cipher [3]. It's thus natural to ask whether EavL could also be reduced to the PRP security of PKEM with leakages. While this direction is in general an interesting open problem, the tricky issue is that the PKEM-based representations in [3] (see Figure 5 in Appendix C) contain a plenty of "imaginary" XORs that do not actually happen in reality. How to model the leakage of these XORs is not directly obvious.

Keyed sponges are not forward secure [11], in the sense that the exposure of a b bit state enables recovering the earlier states. Consequently, the final permutation call must hide its output state to some extent. This is captured by the bad condition $\mathsf{BadFinal}$ in the proof of Lemma 1.

4 The new mode TETSponge/S1P

Considerations. By Theorem 1, duplex is a nice starting point for efficient 1-pass leakage-resilient AEs. Indeed, the final duplex state is typically truncated as the tag. But typically, in the leveled implementations only the initial keyed part is protected against side-channels, and this causes the concern of dec-leakages. First, dec-leakages enable recovering the internal state of (leveled implementations of) duplex [1], after which universal forgery is possible. Second, verifying the tag requires to perform the entire computation to recover the valid tag to compare it with the one included in the ciphertext, and this valid tag may be leaked, constituting trivial forgeries and breaking CIML2—see appendix A for a realistic DPA path. It's natural to ask how to remedy and what can be achieved.

Towards the first issue, our (new) observation is that an additional *keyed* finalization function helps a lot. Namely, even if the internal state has been recovered in complete, the construction collapses to a Hash-then-MAC authenticator, with the duplex being a keyless hash and the keyed finalization being a fixed length MAC. Thus, if the finalization is carefully protected, strong integrity with nonce-misuse and leakages is restored.

Towards the second issue, assuming n-bit tag size, then an approach to cope with the leaking tag is to rely on an n-bit invertible primitive (e.g., a block cipher was used in [10]), 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 duplex permutation size b is typically much larger than any ordinary tag size n. Due to the presence of leakages, it is hard to build a "small" invertible primitive from the large permutations (Feistel-based solution consumes $O(n)$ rounds [22]).

With these considerations, we use a tweakable block cipher (TBC) for the finalization to increase robustness against dec-leakages. In detail, we derive a $2n$-bit (hash) digest from the keyed duplex, and then use an (n, n, n)-TBC to absorb this digest and generate the (n-bit) tag. This solves all the above. 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 [8].

This TBC shall be well protected from side-channel attacks. Therefore, following [37,19,8], we also use it to derive the duplex-key from the nonce and "setup" the duplex. Now, 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; even if the state is leaked in full, the scheme collapses to Hash-then-TBC authenticator, and integrity is retained (so that severe malware attacks like [39] won't be possible). The use of "public keys" in order to offer multi-user security beyond $|K|/2$ is also inherited from [8]. Briefly, once properly used, the public keys create a strong independence between different users and thus reduce the effectiveness of offline computation against the multiple user keys. Besides, the actions involving these public keys do not need any additional physical protection. We refer to [8] for more information.

Remark. At this stage, it's tempting to ask why bother integrity with *nonce misuse-resistance* and *dec-leakages* in a 1-pass scheme that isn't designed for nonce-reuse at all & loses security (precisely, confidentiality) against dec-leakages? The reason is fundamental: when designing an AE, we choose 1-pass not because we don't desire robustness against nonce-reuse & dec-leakages, but because we prefer a trade-off leaning towards efficiency rather than security. Still, it's desirable to have more robustness with minimal overhead—which is a general direction [4,28]. Concretely, *what* level of robustness can be achieved by this sort of efficient designs and *how* to achieve? We thus characterize the remaining.

Specification.

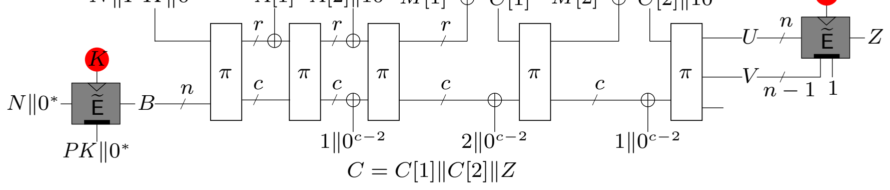

Fig. 3: $\mathsf{TETSponge}[\pi, \widetilde{\mathsf{E}}]_{K,PK}$ AEAD for $\nu = 2$ blocks of associated data and $\ell = 2$ message blocks. Dark squares indicate the "leak-free" TBC where the triangle denotes the key input and the small black rectangle denotes the tweak input. The value $1\|0^{c-2}$ is inserted only if $|A[\nu]| < r$, resp. $|M[\ell]| < r$.

Parameters. $\mathsf{TETSponge}[\pi, \widetilde{\mathsf{E}}]$ is built on an (n, n, n)-TBC $\widetilde{\mathsf{E}}$ and a $b = r + c$ bit permutation π. The key 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 is hope to be *unique per session* for a separation. We feel that the easiest way to ensure this uniqueness is to pick PK uniformly from $\{0,1\}^{n_p}$, and thus we focus on this case. Let $n_N = |N|$ be the fixed length of the nonces. We require that $n_p \le r$, $n_N + n_p + n \le r + c$, and $2n \le r + c + 1$. 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 (black-box CCA & $\mathsf{muCIML2}$) security up to $2^n/n^2$ queries. There is no recommendation for n_N, but when $n = 128$ one could take $n_N = 96$ which is a standard choice.

The encryption. As shown in Fig. 3, 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 seed B is then used as the key of the 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 most significant $2n-1$ bits of the final state with $|U| = n$. As discussed, 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 recovers the initial seed B via $B = \widetilde{\mathsf{E}}_K^{PK\|0^*}(N\|0^*)$, and then runs the duplex on A and $C[1]\|\ldots\|C[\ell]$ to recover 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 increases robustness against dec-leakages at the mode level (formally, helps achieve $\mathsf{CIML2}$).

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

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

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

Note that since we expect the duplex to act as a keyless hash when the internal state has been leaked, we cannot follow the more efficient design approaches that encroach the capacity of the duplex/sponge — including *full-state* [34,17] and *concurrent absorption* [40].

Leakage integrity $\mathsf{muCIML2}$**.** For $\mathsf{TETSponge}$, even very weak implementations could ensure integrity: as long as the protected TBC calls are secure against key recovery attacks (e.g., DPAs exploiting multiple queries to the TBCs), integrity is ensured even if *all the other intermediate values (in the duplex) are leaked in complete.* Such a leakage assumption is called "unbounded" [9]. 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. With these, we have the following result for $\mathsf{TETSponge}$.

Theorem 2. *Assume $u \le 2^{n_p}$, $n_p \le n$, $n \ge 5$, $Q = \sigma + q_e + q_d + q_\pi \le \min\{2^n/4, 2^b/2\}$, 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*

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{TETSponge},\mathsf{L}^*,u} \le \frac{3u}{2^{n_p}} + \frac{32Q^2}{2^c} + \frac{7nQ + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{13}$$

See appendix D for its proof—its idea is simple, while its complexity stems from the analysis of the non-standard sponge hash. Regarding the bounds, $q_\pi, q_{\widetilde{\mathsf{IC}}} = t$ 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 high security up to 2^{124} users, 2^{114} computations, and roughly 2^{120} message blocks.

Leakage confidentiality $\mathsf{muCCAmL1}$**.** For confidentiality the implementations shall satisfy the "bounded" leakages assumptions in Section 3, i.e., non-invertibility & bounded XOR leakages; and we'll apply Theorem 1. Following the notations of Section 3, we naturally define the leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{Enc}}, \mathsf{L}_{\mathsf{Dec}})$ of its implementation 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 $\mathsf{L}_\oplus(a, b)$ generated by all the internal actions $a \oplus b$.
- $\mathsf{L}_{\mathsf{Dec}} = \emptyset$ since $\mathsf{muCCAmL1}$ is CCA with encryption leakages only.

For the TBC $\widetilde{\mathsf{E}}$, we simply assume its leakage function $\mathsf{L}_{\widetilde{\mathsf{E}}}$ returns nothing, i.e., $\widetilde{\mathsf{E}}$ is leak-free. All the analyses can be easily modified to incorporate non-empty $\mathsf{L}_{\widetilde{\mathsf{E}}}$ (indeed, our earlier version addressed), but we eschew for simplicity.

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, $\widetilde{\mathsf{IC}}$, and π resp., run in time t, and have σ blocks in all its (challenge & non-challenge) queries including associated data.

Theorem 3. *Assume $u \le 2^{n_p}$, $n_p \le n$, $n \ge 5$, $\sigma + q_e + q_d + q_m + q_\pi \le \min\{2^n/4, 2^b/2\}$, and 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*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAmL1}}_{\mathcal{A},\mathsf{TETSponge},\mathsf{L},u} \le & \frac{5u}{2^{n_p}} + \frac{49Q^2}{2^c} + \frac{6(n+1)Q + 2nq_d + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \sigma \mathbf{Adv}^{\mathsf{LORL}}(1, Q, t^*) \\ & + 2q_e \mathbf{Adv}^{\mathsf{Inv}[n]}(1, Q, t^*, 2Q) + 2(\sigma + q_e)\mathbf{Adv}^{\mathsf{Inv}[c]}(1, Q, t^*, 2Q),\end{aligned}$$

where $\mathbf{Adv}^{\mathsf{LORL}}$ and $\mathbf{Adv}^{\mathsf{Inv}[\omega]}$ are defined in Eqs. (6) and (3) 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}.

The proof follows a quite standard hybrid argument: from $\mathcal{A}$ we build a EavL adversary $\mathcal{A}_2$ against the duplex stream cipher $\mathsf{DuStr}_B[\pi]$, which simulates all but one challenge encryption queries and relays the exceptional to its EavL challenger. The core feature enabling the hybrid is that, for each challenge encryption query, since the nonce is used only once during its lifetime, $\mathsf{DuStr}_B[\pi]$ is seeded with an initializing key B that is somewhat independent from any other initializing key of the other encryption queries. See appendix F.4 for the formal presentations. Note that: (a) since the $\mathsf{muCCAmL1}$ adversary $\mathcal{A}$ can't require "challenge dec-leakages", $\mathcal{A}_2$ doesn't need to measure them on $\mathsf{DuStr}_B[\pi]$ either, and thus we use the terms in Theorem 1 with $p = 1$; (b) in $\mathsf{TETSponge}$, the initializing key B is of n bits, and thus the term $\mathbf{Adv}^{\mathsf{Inv}[n]}(1, Q, t^*, 2Q)$.

The concrete security is mainly limited by the terms due to Theorem 1. As discussed before, the terms $2q_e\mathbf{Adv}^{\mathsf{Inv}[n]}(1, Q, t^*, 2Q) + 2(\sigma + q_e)\mathbf{Adv}^{\mathsf{Inv}[c]}(1, Q, t^*, 2Q)$ are $O\Big(q_e \cdot \frac{q_\pi+\sigma+t}{\mu_n \cdot 2^n}\Big) + O\Big(\sigma \cdot \frac{q_\pi+\sigma+t}{\mu_c \cdot 2^c}\Big)$ for some specific parameters μ_n and μ_c. Though, the influence of u the number of users on the security remains negligible: once $u \le 2^{n_p}/5$, $\mathsf{TETSponge}$ is secure up to the birthday $2^{n/2}$ complexity—it's smaller than $2^{c/2}$ due to the shorter initial seed B.

Discussion. We remark again that the leakage assumptions needed for the two theorems are different, and integrity is ensured by much weaker implementations. This not only coincides with the theory separation between unpredictability and pseudorandomness [35] but also unveils which goal is easier in the real world. In addition, for both theorems, the strong assumption on the implementation of $\widetilde{\mathsf{E}}$ versus the weak assumption on π provides the separation of duties that enables leveled implementations.

Also we remark that, as a first step, we concentrate on a simplified model of "unbounded" leakage plus ideal permutation for integrity. There are complicated gaps between this model and the real world situation, e.g., lightweight AEs tend to use weak permutations far from ideal and may admit (hash collision-based) attacks in our model. Since duplex state can indeed be recovered via side-channels [1], this "attack" seems something real rather than an artifact of our model, and may be worth noting. We leave these for future investigations.

5 Applications to some other AEs

Duplex-based AEs follow the same general structure, but deviate on many details. For 1-pass modes, here we consider two examples Ascon and GIBBON whose structures are very similar to $\mathsf{TETSponge}$. Note that this also eases understanding. We leave the analyses of other 1-pass duplex such as full-state duplex-based AEs [17] and Beetle [14] as open questions. We then extend the discussion to two 2-pass modes $\mathsf{TEDTSponge}$ and ISAP.

Ascon. Ascon was by Dobraunig et al. [16], and is among the final portfolio of CAESAR competition. Here we consider its NIST lightweight submission Ascon v1.2 [20]. Its structure is depicted in Fig. 4 (a)—as mentioned, this in fact represents its leveled implementation in [1]. It uses two permutations π_1 and π_2 that differ in the number of underlying rounds. Let $n := |K| = |Z|$. Assume π_1 and π_2 are two independent random permutations. Then Ascon's structure resembles a variant of $\mathsf{TETSponge}$ with $\mathsf{KDF}_K(N) := \pi_1(IV\|K\|N) \oplus (0^{b-n}\|K)$ and $\mathsf{TGF}_K(S) := \mathsf{ls}_n\big(\pi_1(S \oplus (0^r\|K\|0^{c-n}))\big) \oplus K$.

As discussed in Section 4 (also see Appendix A), merely protecting $\mathsf{KDF}_K(N)$ and $\mathsf{TGF}_K(S)$ against side-channel key recovery isn't enough. Concretely, we have to assume leak-freeness on the sequence of actions $\mathsf{IntCheck}_K(S, Z)$ below.

```
1: Integrity checking $\mathsf{IntCheck}_K(S, Z)$
2: $Z_c \leftarrow \mathsf{TGF}_K(S)$
3: If $Z = Z_c$ then return 1 else return 0
```

Then, the "leak-freeness" of $\mathsf{KDF}_K(N)$ and $\mathsf{IntCheck}_K(S, Z)$ is already sufficient for Ascon implementation to ensure integrity, which resembles $\mathsf{TETSponge}$. On the other hand, for confidentiality it also has to meet the non-invertible assumption in Section 3.2 and the bounded XOR leakage assumption in Section 3.3. The results are as follows.

Theorem 4 (informal). *For the leveled implementation of Ascon depicted in Fig. 4 (a), if $\mathsf{KDF}_K(N)$ and $\mathsf{IntCheck}_K(S, Z)$ are "leak-free", then for any adversary $\mathcal{A}$ making q queries to its oracles and running in time*

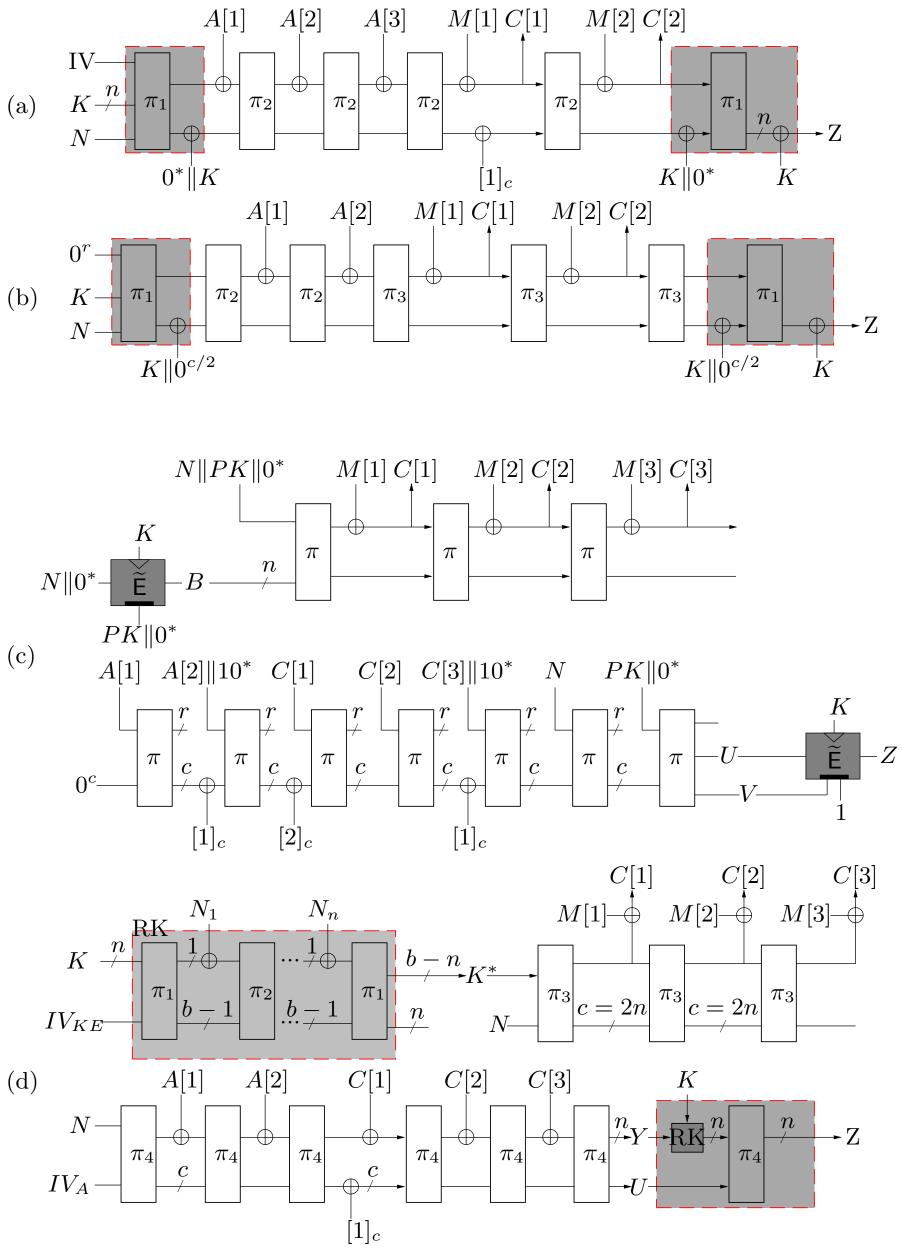

Fig. 4: Duplex-based AEs considered in section 5 and their leveled implementations. (a) Ascon v1.2; (b) GIBBON; (c) TEDTSponge; (d) ISAP v2.0. For TEDTSponge, the rule of adding separation bits is the same as TETSponge. For ISAP v2.0, the structure named RK (short for ISAPRK) in the 1st gray rectangle is a rate-1 duplex, and its input values $N_1, \ldots, N_n$ are the 1st,..., n-th bits of the nonce N (i.e., π_2 is called $n-1=127$ times). This structure was used again in the 2nd pass (the dark square "RK").

t, we have

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{Ascon},\mathsf{L}^*,u} = O\Big(\frac{u^2}{2^n}\Big) + O\Big(\frac{uq}{2^n}\Big) + O\Big(\frac{(\sigma+q)^2}{2^c}\Big) + O\Big(\frac{q}{2^n}\Big), \qquad (14)$$

$$\mathbf{Adv}^{\mathsf{muCCAmL1}}_{\mathcal{A},\mathsf{Ascon},\mathsf{L},u} = O\Big(\frac{u^2}{2^n}\Big) + O\Big(\frac{uq}{2^n}\Big) + O\Big(\frac{(\sigma+q)^2}{2^c}\Big) + O(\sigma)\cdot\mathbf{Adv}^{\mathsf{LORL}}(1,O(q),O(t)) + O(\sigma)\cdot\mathbf{Adv}^{\mathsf{Inv}[c]}(1,O(q),O(t),O(q)) + O\Big(\frac{q}{2^n}\Big), \qquad (15)$$

where L^* *is "unbounded" as in Theorem 2, while* L *is "bounded" as Theorem 3.*

The proof is the same as the proof of Theorem 2 modulo minor changes. For verifying, a sketch is given in Appendix G. The bounds are comparable to TETSponge. The terms $O\big(\frac{u^2}{2^n}\big) + O\big(\frac{uq}{2^n}\big)$—more precisely, $\frac{u^2}{2^{|K|}} + \frac{uq}{2^{|K|}}$,—reflect the well-known multi-user security degradation, and can't be avoided,[7] and it may be interesting to investigate whether it can be reduced to $\frac{u}{2^{|K|}} + \frac{q}{2^{|K|}}$ by employing public randomness (like TETSponge).

We remark that while the equality checking line 3 in $\mathsf{IntCheck}_K(S,Z)$ may not be difficult to protect (the value Z_c output by a masked implementation of $\mathsf{TGF}_K(S)$ is already in a secret shared form), this requirement shall be emphasized for future implementations—as we've clarified several times.

GIBBON. GIBBON was a member of the submission PRIMATEs [2] to the CAESAR competition. As shown in Fig. 4 (b), its structure is very similar to Ascon, except that one more permutation is used. Therefore, our analysis for Ascon can be transited without essential change and lead to similar conclusions.

2-pass mode TEDTSponge. As mentioned in introduction, the consensus reached by many [5,19,10] seems to be that Encrypt-then-MAC style designs with *2 passes* is the only choice for achieving dec-leakages confidentiality with leveled implementations. This was also a limitation of the discussed 1-pass designs. We thus consider $\mathsf{TEDTSponge}[\pi,\widetilde{\mathsf{E}}]$, a natural 2-pass extension of $\mathsf{TETSponge}[\pi,\widetilde{\mathsf{E}}]$: see Fig. 4 (c). More clearly, instead of deriving the tag from the final duplex state of the 1st pass, we leverage the 2nd leak-free TBC-call for a Hash-then-TBC authenticator and make an Encrypt-then-MAC composition. The keyless hash function for Hash-then-TBC is, of course, built upon the sponge function. In summary, $\mathsf{TEDTSponge}[\pi,\widetilde{\mathsf{E}}]$ can be seen as a (more efficient) duplex-based variant of the TEDT TBC-mode [8], or an ISAP variant using a (masked) TBC instead of the rate-1 duplex (this will be introduced below).

For the sake of space, we defer its formal description to Appendix E.1. With similar assumptions and follow the proof for TETSponge, it can be proved that the leveled implementation of TEDTSponge offers muCIML2 (see Theorem 5 in Appendix E.1) and muCCAmL2 security (see Theorem 6 in Appendix F.5) with bounds almost the same as TETSponge. The muCCAmL2 security proof is similar to the muCCAmL1 proof of TETSponge, except for the added handling of dec-leakages. For TEDTSponge, it's feasible to handle invalid dec-leakages since they are far less informative than the 1-pass modes, while valid (challenge) dec-leakages have been included in Theorem 1.

2-pass mode ISAP. Finally, we consider ISAP, and more precisely, its NIST submission ISAP v2.0 [18], which is more robust against state exposure than the original [19]. An illustration of ISAP v2.0 is in Fig. 4 (d). It may be seen as a TEDTSponge variant (but note that ISAP was designed earlier) with the leak-free TBCs replaced by primitives built upon ISAPRK, which is a rate-1 duplex as shown in the 1st gray rectangle in Fig. 4 (d). Thus it's purely sponge-based. The use of rate-1 duplex is to exclude DPA & minimize the effectiveness of SPA, even if the rate-1 duplex is called for polynomial times.

As mentioned in the Introduction, the original version only came with informal side-channel security arguments. DM has given a proof for its "encryption part" under the high min-entropy leakage assumption [21]. Here we show our methodology and assumptions could establish AE security muCIML2 and muCCAmL2. For this, define the two functions $\mathsf{KDF}_K(N)$ and $\mathsf{TGF}_K(U,Y)$ as well as the corresponding action $\mathsf{IntCheck}_K(Y\|U,Z)$ as follows.

```
algorithm KDF_K(N)
1. $N_1,\ldots,N_n \leftarrow N$
2. $S_0 \leftarrow K\|IV_{KE}$, $S_1 \leftarrow \pi_1(S_0)$
3. for $i = 1$ to $n-1$ do
4.     $S'_i \leftarrow S_i \oplus (N_i\|0^{b-1})$
5.     $S_{i+1} \leftarrow \pi_2(S'_i)$
6. $S'_n \leftarrow S_n \oplus (N_n\|0^{b-1})$
7. $S_f \leftarrow \pi_1(S'_n)$
```

[7]The term $\frac{u^2}{2^{|K|}}$ corresponds to user key collision, while the term $\frac{uq}{2^{|K|}}$ is matched by the naive attack seeking for collision between user encryption and offline key guesses.

```
8. $K^* \leftarrow \mathsf{ms}_{b-n}(S_f)$
9. return $K^*$

   algorithm $\mathsf{IntCheck}_K(Y \| U, Z)$
1. $Z_c \leftarrow \mathsf{TGF}_K(U, Y)$
2. If $Z = Z_c$ then return 1
3. return 0

   algorithm $\mathsf{TGF}_K(U, Y)$
1. $Y_1, \ldots, Y_n \leftarrow Y$
2. $S_0 \leftarrow K \| IV_{KA}$, $S_1 \leftarrow \pi_1(S_0)$
3. for $i = 1$ to $n-1$ do
4.     $S_i' \leftarrow S_i \oplus (Y_i \| 0^{b-1})$
5.     $S_{i+1} \leftarrow \pi_2(S_i')$
6. $S_n' \leftarrow S_n \oplus (N_n \| 0^{b-1})$
7. $S_f \leftarrow \pi_1(S_n')$
8. $K^* \leftarrow \mathsf{ms}_{b-n}(S_f)$

9. $S_Z \leftarrow \pi_4(K^* \| U)$
10. $Z \leftarrow \mathsf{ms}_n(S_Z)$
11. return $Z$
```

Then, for an ISAP implementation such that: (a) calls to $\mathsf{KDF}_K(N)$ and $\mathsf{IntCheck}_K(Y \| U, Z)$ are "leak-free", and (b) leakages are non-invertible as in Section 3.2, and (c) XOR leakages are bounded as in Section 3.3, the muCIML2 and muCCAmL2 security can be established similarly to TEDTSponge. For any adversary $\mathcal{A}$ making q queries to its oracles and running in time t, assuming $b - n \geq c$ for simplicity, then we have

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{ISAP},\mathsf{L}^*,u} = O\Big(\frac{u^2}{2^n}\Big) + O\Big(\frac{u(\sigma+q)}{2^n}\Big) + O\Big(\frac{(\sigma+q)^2}{2^c}\Big) + O\Big(\frac{n(\sigma+q)}{2^n}\Big), \tag{16}$$

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{ISAP},\mathsf{L},u} &= O\Big(\frac{u^2}{2^n}\Big) + O\Big(\frac{u(\sigma+q)}{2^n}\Big) + O\Big(\frac{(\sigma+q)^2}{2^c}\Big) + O\Big(\frac{n(\sigma+q)}{2^n}\Big) + O(\sigma)\cdot\mathbf{Adv}^{\mathsf{LORL}}(p, O(q), O(t)) \\ &\quad + O(\sigma)\cdot\mathbf{Adv}^{\mathsf{Inv}[c]}(p, O(q), O(t), O(q)).\end{aligned} \tag{17}$$

The proof is similar to the proof for TEDTSponge, Appendix F.5: we serve a sketch in Appendix G. Note that ISAP doesn't use duplex for message encryption (i.e., ciphertext blocks are not fed into the next permutations), but Theorem 1 can be easily adapted for this case. Regarding bounds, again we encounter the unavoidable multi-user degradation terms $O(\frac{u^2}{2^n}) + O(\frac{u(\sigma+q)}{2^n})$, and it may be interesting to study whether it can be reduced to $\frac{u}{2^n}$ using public randomness.

Our technique (more precisely, the proof of Lemma 1) can be used to prove that the rate-1 duplex is a PRF resilient to non-adaptive leakages. This shows the leak-freeness assumption of KDF_K and TGF_K can be replaced by SPA security assumption on the permutations. But we have to strengthen the non-invertibility assumption, by both allowing $O(q)$ repeated measures and increasing the number of involved permutation calls. We defer the discussion to Appendix H.1.

Summary. Our analyses mainly aimed at demonstrating the power and applicability of our methodology, and we view this as a success. In summary, the results indicate that 1-pass modes with keyed initializing and finalizing functions could have nice highly secure leveled implementations. As non-invertibility leakage assumption is easily connected to side-channel cryptanalysis practice, we hope that this could bridge the two sides.

Compared to Ascon, TETSponge has a *higher mode-level robustness* against dec-leakages (for integrity muCIML2, as discussed in Section 4) and avoid protecting the equality checking. But Ascon is *inverse-free* and may be tweaked to support *variable-length tags*. We thus feel that the complete characterization can only be left to the future several years of practice.

Finally, we restress that the motivation for studying integrity with *nonce misuse-resistance* flavor is to squeeze more security/robustness without efficiency penalty: see the *Remark* in Section 4.

References

1. Adomnicai, A., Fournier, J.J., Masson, L.: Masking the Lightweight Authenticated Ciphers ACORN and Ascon in Software. Cryptology ePrint Archive, Report 2018/708 (2019), appeared at BalkanCryptSec 2018.
2. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mendel, F., Mennink, B., Mouha, N., Wang, Q., Yasuda, K.: PRIMATEs v1.02. Submission to the CAESAR Competition (2016), Available: https://competitions.cr.yp.to/round2/primatesv102.pdf.
3. 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)
4. Ashur, T., Dunkelman, O., Luykx, A.: Boosting Authenticated Encryption Robustness with Minimal Modifications. In: CRYPTO 2017, Part III. pp. 3–33 (2017)
5. 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)

6. 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
7. 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)
8. 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
9. 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)
10. 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)
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. Christoph Dobraunig, Maria Eichlseder, F.M.M.S.: Ascon v1.2. Submission to the CAESAR Competition (2016), Available: https://competitions.cr.yp.to/round3/asconv12.pdf.
17. 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)
18. Dobraunig, C., Eichlseder, M., Mangard, S., Mendel, F., Mennink, B., Primas, R.: ISAP v2.0. Submission to NIST (2019), Available: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/ISAP-spec.pdf.
19. 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)
20. Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M.: Ascon v1.2. Submission to NIST (2019), Available: https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/ascon-spec.pdf.
21. Dobraunig, C., Mennink, B.: Leakage Resilience of the Duplex Construction. Cryptology ePrint Archive, Report 2019/225 (2019)
22. Dodis, Y., Pietrzak, K.: Leakage-Resilient Pseudorandom Functions and Side-Channel Attacks on Feistel Networks. In: CRYPTO 2010. pp. 21–40 (2010)
23. Dziembowski, S., Pietrzak, K.: Leakage-Resilient Cryptography. In: FOCS 2008. pp. 293–302 (2008)
24. Faust, S., Pietrzak, K., Schipper, J.: Practical Leakage-Resilient Symmetric Cryptography. In: CHES 2012. pp. 213–232 (2012)
25. Fuller, B., Hamlin, A.: Unifying Leakage Classes: Simulatable Leakage and Pseudoentropy. In: ICITS. pp. 69–86 (2015)
26. Goudarzi, D., Rivain, M.: How Fast Can Higher-Order Masking Be in Software? In: EUROCRYPT 2017, Part I. pp. 567–597 (2017)
27. Groß, H., Mangard, S., Korak, T.: An Efficient Side-Channel Protected AES Implementation with Arbitrary Protection Order. In: CT-RSA 2017. pp. 95–112 (2017)
28. Gueron, S., Lindell, Y.: Better Bounds for Block Cipher Modes of Operation via Nonce-Based Key Derivation. In: CCS 2017. pp. 1019–1036 (2017)
29. 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)
30. Hazay, C., López-Alt, A., Wee, H., Wichs, D.: Leakage-Resilient Cryptography from Minimal Assumptions. J. Cryptology 29(3), 514–551 (2016)
31. 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)
32. Kalai, Y.T., Reyzin, L.: A Survey of Leakage-Resilient Cryptography. Cryptology ePrint Archive, Report 2019/302 (2019)
33. Maurer, U.M., Tessaro, S.: Computational Indistinguishability Amplification: Tight Product Theorems for System Composition. In: CRYPTO 2009. pp. 355–373 (2009)
34. Mennink, B., Reyhanitabar, R., Vizár, D.: Security of Full-State Keyed Sponge and Duplex: Applications to Authenticated Encryption. In: ASIACRYPT 2015, Part II. pp. 465–489 (2015)
35. Micali, S., Reyzin, L.: Physically Observable Cryptography (Extended Abstract). In: TCC 2004. pp. 278–296 (2004)

36. O'Flynn, C., Chen, Z.D.: Side channel power analysis of an AES-256 bootloader. In: CCECE. pp. 750–755. IEEE (2015)
37. Pereira, O., Standaert, F., Vivek, S.: Leakage-Resilient Authentication and Encryption from Symmetric Cryptographic Primitives. In: CCS 2015. pp. 96–108 (2015)
38. Rogaway, P., Shrimpton, T.: A Provable-Security Treatment of the Key-Wrap Problem. In: EUROCRYPT 2006. pp. 373–390 (2006)
39. 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)
40. Sasaki, Y., Yasuda, K.: How to Incorporate Associated Data in Sponge-Based Authenticated Encryption. In: CT-RSA 2015. pp. 353–370 (2015)
41. Tessaro, S.: Security Amplification for the Cascade of Arbitrarily Weak PRPs: Tight Bounds via the Interactive Hardcore Lemma. In: TCC 2011. pp. 37–54 (2011)
42. 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
43. Yu, Y., Standaert, F., Pereira, O., Yung, M.: Practical Leakage-Resilient Pseudorandom Generators. In: CCS 2010. pp. 141–151 (2010)

A A DPA Path on General "Tag-then-Compare" Designs

Most AEs and MACs designed for black-box security follows the "tag-then-compare" approach for integrity checking. Concretely, take the nonce-based AE as the example, and assume that an encryption query $\mathsf{Enc}_K(N, A, M) \to C$, $C = \mathbf{c} \| Z$ is the concatenation of a "main ciphertext" $\mathbf{c}$ and a tag Z. Assume that $Z = \mathsf{tag}(N, A, \mathbf{c})$ is a deterministic function of $(N, A, \mathbf{c})$: this is the case in most AEs. In such AEs, the decryption $\mathsf{Dec}_K(N, A, \mathbf{c}\|Z)$ would first derive the "correct" tag $Z_c = \mathsf{tag}(N, A, \mathbf{c})$ and then make a comparison to see if $Z_c =^? Z$. This equality checking creates a new DPA path. Concretely, consider the following CCA forgery attack:

(i) First, we fix a tuple $(N, A, \mathbf{c})$ and q arbitrary tags $T_1, \ldots, T_q$;
(ii) Second, we make q chosen-ciphertext queries to $\mathsf{LDec}(N, A, \mathbf{c}, T_1), \ldots, \mathsf{LDec}(N, A, \mathbf{c}, T_q)$. By this, the tag function $\mathsf{tag}(N, A, \mathbf{c}) \to T_c$ is called q times, and the equality checking is made w.r.t. q distinct known faked tags, i.e., $Z_c =^? Z_1, \ldots, Z_c =^? Z_q$.

Now the above second step could produce power traces that allow us to take Z_c as the target and perform a DPA secret recovery. If the implementation didn't pay a sufficient amount of attention on protecting the equality checking, then Z_c may be recovered after several thousands chosen-ciphertext queries, giving rise to a valid forgery $(N, A, \mathbf{c}\|Z_c)$.

We stress that the above path only applies to certain **implementations** rather than the corresponding mathematical objects (i.e., algorithms themselves). Yet, we feel that this (more practice-relevant) issue shall be emphasized. Previously, it was only noticed by the theory community: Barwell et al. used bilinear map-based secret sharing implementation to protect against the attack [5], while Berti et al. used the inverse of the blockcipher to completely avoid computing Z_c in the integrity checking [10]. We follow the latter (as we are not using bilinear maps).

Arguably, chosen-ciphertext attacks are not so realistic. However, we stress that: first, *it has been practically carried out* in some settings, e.g., see [39], the authors were able to insert ciphertexts into the flash memory of Philips Hue smart lamps and trigger decrypting; second, we shall be conservative during the designing phase, and this requires to consider CCAs.

Furthermore, to minimize the amount of decryption leakages, many leakage resilient AEs have chosen Encrypt-then-MAC composition to stop invalid decryption right after the (supposed to be failed) integrity checking. Since we have created valid forgeries via the above process, we can further trigger the "hidden" main decryption process to gather more leakages and break confidentiality.

B TETSponge is not CCAmL2 secure

Following the notational convention of section 4, we show a DPA path breaking the CCAmL2 security of TETSponge.

(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 $A = \perp$ 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 to achieve confidentiality against dec-leakages, all designs including TEDTSponge tried to *process invalid decryption queries with keyless/non-secret primitives.*

C PKEM-based Representation of Keyed Sponges

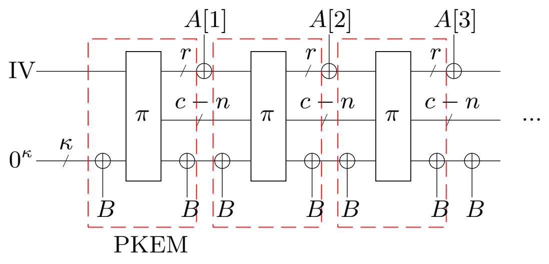

Fig. 5: The PKEM-based representation of a keyed sponge with κ-bit key/initial seed B. Inside the red dashed rectangles are the "partial-key" Even-Mansour cipher. It's easy to see after the internal actions of XORing B cancel, the construction turns basically the same as the keyed sponge.

D Proof of Theorem 2 (muCIML2 of S1P)

For brevity, from now on we refer to TETSponge and TEDTSponge as S1P and S2P, standing for *Sponge with 1/2 Pass(es)*.

The proof proceeds in two steps:

(i) Below in appendix D.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 D.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 D.3, we prove unforgability for the idealized scheme to complete the muCIML2 proof.

D.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

$$\tau_\pi = \big((S_1^{in}, S_1^{out}), \dots, (S_{q_\pi}^{in}, S_{q_\pi}^{out})\big).$$

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

$$Q := \big|\tau_\pi^*\big| \le \sigma + q_e + q_d + q_\pi. \tag{18}$$

Recall that the adversarial goal is to distinguish $\widetilde{\mathsf{IC}}_{K_1}, \dots, \widetilde{\mathsf{IC}}_{K_u}$ from $\widetilde{\mathsf{SIC}}_{K_1}, \dots, \widetilde{\mathsf{SIC}}_{K_u}$. 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

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

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

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

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" $\mathbf{PK} = (PK_1, \dots, PK_u)$ are also included in the transcript. Moreover, to simplify the definition of bad transcripts, we reveal to the distinguisher the user keys $\mathbf{K} = (K_1, \dots, K_u)$ 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*

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

D.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

$$\mu_{PK} := \max_{pk \in \{0,1\}^{n_p}} \big|\{i \in \{1, \dots, u\} : PK_i = pk\}\big|,$$
$$\mu_V := \max_{v \in \{0,1\}^{n-1}} \Big|\Big\{((N, PK, B, A, \mathbf{c}), U\|V) \in \tau_h^* : V = v\Big\}\Big|. \tag{19}$$

Then it's defined as follows.

Definition 4 (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 analyses, τ_π^* doesn't appear in the conditions.

As $PK_1, \dots, PK_u$ 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

$$\Pr[\mu_{PK} \geq n+1] \leq \Big(\frac{u}{2^{n_p}}\Big)^{n+1} \leq \frac{u}{2^{n_p}}. \tag{20}$$

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 $(S_1^{in}, S_1^{out}), \dots, (S_{n+1}^{in}, S_{n+1}^{out})$ such that $\mathsf{mid}(S_1^{out}) = \ldots = \mathsf{mid}(S_{n+1}^{out})$, where $\mathsf{mid}(S_i^{out}) = \mathsf{ms}_{2n-1}(\mathsf{ls}_{n-1}(S_i^{out}))$, i.e., extracting $n-1$ bits from the middle.

– (B-14) Right after a (necessarily forward) $\widetilde{\mathsf{SIC}}$/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, $q_1 \le Q$, $q_2 \le q_\pi$ (as $\mathsf{S1P}$ doesn't make backward π queries), and $q_1 + q_2 \le Q$. 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 $2^b - Q$. 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 $Q \le 2^b/2$, we have

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

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, we have

$$\Pr[\text{(B-11)}] \le q_1 \cdot Q \cdot \Big(\frac{8}{2^c} + \frac{8}{2^c}\Big) \le \frac{16 q_1 Q}{2^c}.$$

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

$$\Pr[\text{(B-12)}] \le q_2 \cdot Q \cdot \frac{8}{2^c} + q_2 \cdot (q_e + q_d) \cdot \frac{2}{2^b} \le \frac{10 q_2 Q}{2^c}, \text{ and}$$
$$\Pr[\text{(B-13)}] \le \binom{q_1}{n+1} \cdot \Big(\frac{2}{2^{n-1}}\Big)^n \le \Big(\frac{8Q}{2^n}\Big)^{n+1} \cdot \frac{1}{8(n+1)!} \le \Big(\frac{4Q}{2^n}\Big)^{n+1} \cdot \frac{1}{8} \le \frac{Q}{2^n}.$$

The last bound relies on $4Q \le 2^n$.

For (B-14), we define a set

$$\tau_\pi^*[N, PK] := \big\{B \in \{0,1\}^n : (N\|PK\|0^*\|B, \star) \in \tau_\pi^*\big\}. \tag{21}$$

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

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

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

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

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

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 $q_1 + q_2 \le Q$)

$$\Pr[\mu_V \ge n+1] \le \Pr[\mathsf{Bad}_1] \le \frac{16 q_1 Q}{2^c} + \frac{10 q_2 Q}{2^c} + \frac{Q}{2^n} + \frac{2nQ}{2^n} \le \frac{16 Q^2}{2^c} + \frac{(2n+1)Q}{2^n}.$$

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 $i = 1, \dots, \nu - 1$, $\mathsf{ls}_c(S_i^{in}) = \mathsf{ls}_c(S_{i-1}^{out})$, $\mathsf{ms}_r(S_i^{in}) \oplus \mathsf{ms}_r(S_{i-1}^{out}) = A[i]$;

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

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 $\neg$(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 $\neg$(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 $\neg$(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 $\neg$(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

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

it's easy to see

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

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,

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

which allows us to conclude

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

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

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D}, \mathsf{S1P}[\pi, \widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}}, \mathsf{L}^*, u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D}, \mathsf{S1P}[\pi, \widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}, \mathsf{L}^*, u} \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{(2n+1)Q + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \quad (22)$$

D.3 Unforgability of the Idealized S1P

The remaining devotes to analyze $\mathsf{S1P}[\pi, \widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}}$. Consider the $\mathsf{muCIML2}$ 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 indices j, ℓ such that $K_j \| PK_j = K_\ell \| PK_\ell$, or $\mu_{PK} \geq n+1$.
- (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 $\mathsf{ms}_{2n-1}(S^{out}) = \mathsf{ms}_{2n-1}(S^{out'})$; or
  - (C-22) there exists a $\widetilde{\mathsf{SIC}}$ query $(K, V\|1, U, Z)$ such that $\mathsf{ms}_{2n-1}(S^{out}) = U\|V$; 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 $\mathsf{mid}(S_1^{out}) = \ldots = \mathsf{mid}(S_{n+1}^{out})$.
- (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 $\mathsf{ms}_{2n-1}(S^{out}) = U\|V$.

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

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

Second, (C-21) is the previous (B-11) enhanced with $\mathsf{ms}_{2n-1}(S^{out}) = \mathsf{ms}_{2n-1}(S^{out'})$, thus $\Pr[\text{(C-21)}] \leq \frac{16q_1Q}{2^c} + \frac{2Q^2}{2^{2n-1}}$ (q_1 being the number of forward π queries). And it's easy to see $\Pr[\text{(C-22)}] \leq \frac{2(q_e+q_d)Q}{2^{2n-1}}$. Thus (using $q_e + q_d \leq Q$)

$$\Pr[\text{(C-2)}] \leq \frac{16q_1Q}{2^c} + \frac{4Q^2 + 4(q_e+q_d)Q}{2^{2n}} \leq \frac{16q_1Q}{2^c} + \frac{Q + q_e + q_d}{2^n} \leq \frac{16q_1Q}{2^c} + \frac{2Q}{2^n}.$$

The last inequality stems from $Q \leq 2^n/4$.

The condition (C-3) is the same as the previous (B-13), thus $\Pr[\text{(C-3)}] \leq \frac{Q}{2^n}$. The condition (C-4) is the previous (B-12), thus $\Pr[\text{(C-4)}] \leq \frac{10q_2Q}{2^c}$. (C-5) is the previous (B-14), thus

$$\Pr[\text{(C-5)} \mid \neg\text{(C-1)}] \leq \frac{2nQ}{2^n}. \tag{23}$$

For (C-6), by $\neg$(C-2) and $\neg$(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

$$\Pr[\text{(C-6)}] \leq \frac{nq_d}{2^n - q_{\widetilde{\mathsf{IC}}}} \leq \frac{2nq_d}{2^n}. \tag{24}$$

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

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

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 $\neg$(C-2); if this query is backward, then it contradicts $\neg$(C-4).

Case 2: $\mathcal{A}$ makes an encryption query $\mathsf{LEnc}_{\mathbf{K},\mathbf{PK}}(i, N, A, M)$. Note that the 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 $\neg$(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 $((N', PK_j, B', A', \mathbf{c}'), U'\|V')$ 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 $\neg$(C-21), the two involved hash chains are different, and thus $U\|V = U'\|V'$ would contradict $\neg$(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 $\neg$(C-1), which means the two involved hash chains are different, and thus $U\|V = U'\|V'$ would contradict $\neg$(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 $\neg$(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 $\neg$(C-6).

By the above, we have

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S1P}[\pi,\mathsf{TRPFamily}]_{\mathbf{PK}},\mathsf{L}^*,u} \le \Pr[\mathsf{Bad}] \le \frac{2u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{(2n+3)Q + 2nq_d}{2^n}. \tag{25}$$

This plus Eq. (22) yield Eq. (13) (note that $4 < 5 \le n$):

$$\frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{(2n+1)Q + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{(2n+3)Q + 2nq_d}{2^n} \le \frac{3u}{2^{n_p}} + \frac{32Q^2}{2^c} + \frac{5nQ + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{26}$$

E Proof of Theorem 5 (muCIML2 of S2P)

E.1 Definition of TEDTSponge/S2P

Formally, with the same conventions $n_p \le r$, $n_N + n_p + n \le r + c$, and $2n \le r + c + 1$ as in Section 4, the mode $\mathsf{TEDTSponge}[\pi, \widetilde{\mathsf{E}}]/\mathsf{S2P}[\pi, \widetilde{\mathsf{E}}]$ is described as follows. We again recommend the parameters $n_p = n - 1$ and $c = 2n$.

```
algorithm S2P[$\pi, \widetilde{\mathsf{E}}$].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 $1 \le |M[\ell]| \le r$
3. if $\ell > 0$ then
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.   $S_0 \leftarrow IV\|B$
8.   for $i = 1$ to $\ell$ do
9.     $S_i \leftarrow \pi(S_{i-1})$
10.    $C[i] \leftarrow \mathsf{ms}_{|M[i]|}(S_i) \oplus M[i]$
11.    $S_i \leftarrow C[i]\|\mathsf{ls}_c(S_i)$
12.  $\mathbf{c} \leftarrow C[1]\|\ldots\|C[\ell]$
13. else $\mathbf{c} \leftarrow \perp$
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 S2P[$\pi, \widetilde{\mathsf{E}}$].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 $1 \le |C[\ell]| \le r$
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 $U \ne U^*$ then return $\perp$
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 $\ell$ 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 $M[1]\|\ldots\|M[\ell]$

algorithm 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 $M[1]\|\ldots\|M[\ell]$, 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. $S_0 \leftarrow 0^{r+c}$
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 $U\|V$
```

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 a consistent c parameter.

With the "unbounded" leakage function L^* similar to that defined for Theorem 2, $\mathsf{TEDTSponge}$ delivers $2^n/n^2$ $\mathsf{muCIML2}$ security as follows.

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

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{TEDTSponge},\mathsf{L}^*,u} \le \frac{2u}{2^{n_p}} + \frac{32Q^2}{2^c} + \frac{3Q + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{27}$$

The two-step proof resembles appendix D.

E.2 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 D.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

$$Q := |\tau_\pi^*| \le 2\sigma + 2(q_e + q_d) + q_\pi. \tag{28}$$

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

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

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

$$\mu_V := \max_{v \in \{0,1\}^{n-1}} \Big|\big\{((A, \mathbf{c}, N, PK), U\|V) \in \tau_h^* : V = v\big\}\Big|, \tag{29}$$

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

Definition 5 (Bad Transcripts for Idealizing S2P, muCIML2). *An attainable transcript τ is bad, if one of the following conditions is fulfilled:*

- *(B-1) $\mu_{PK} \ge n+1$, $\mu_V \ge n+1$.*
- *(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 D.2 except for $\mu_V \ge n+1$. 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 D.2, except that $\Pr[\text{(B-12)}] \le \frac{2q_2}{2^c} + \frac{8q_2 Q}{2^c} \le \frac{10q_2 Q}{2^c}$. Thus we could adapt the previous $\Pr[\mu_V \ge n+1]$. As before, define $\mathsf{Bad}_1 := \text{(B-11)} \vee \text{(B-12)} \vee \text{(B-13)}$. We show that $\mu_V \le n$ conditioned on $\neg\mathsf{Bad}_1$, so that

$$\Pr[\mu_V \ge n] \le \Pr[\mathsf{Bad}_1] \le \frac{16q_1 Q}{2^c} + \frac{10q_2 Q}{2^c} + \frac{Q}{2^n} \le \frac{16Q^2}{2^c} + \frac{Q}{2^n}.$$

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 D.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 $(A_1, \mathbf{c}_1, N_1, PK_1)$, ..., $(A_{n+1}, \mathbf{c}_{n+1}, N_{n+1}, PK_{n+1})$ that result in the same output $V_1 = \ldots = V_{n+1}$, i.e., it holds $\mu_V \le n$.

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

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

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

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{Q}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{30}$$

E.3 Unforgability of the Idealized S2P

Consider the $\mathsf{muCIML2}$ 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

$$\Pr[\text{(C-1)}] \le \frac{u}{2^{n_p}}, \quad \Pr[\text{(C-2)}] \le \frac{16q_1Q}{2^c} + \frac{4Q^2}{2^{2n}}, \quad \Pr[\text{(C-3)}] \le \frac{Q}{2^n}, \quad \Pr[\text{(C-4)}] \le \frac{8q_2Q}{2^c} + \frac{2q_2}{2^c} \le \frac{10q_2Q}{2^c}.$$

For (C-5), as argued before, conditioned on $\neg$(C-2), $\neg$(C-3), and $\neg$(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) $\vee$ (C-2) $\vee$ (C-3) $\vee$ (C-4) $\vee$ (C-5), using $\frac{4Q^2}{2^{2n}} \le \frac{Q}{2^n}$ we have

$$\Pr[\mathsf{Bad}] \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{2Q + 2nq_d}{2^n}.$$

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 $\neg$(C-2); if this query is backward, then it contradicts $\neg$(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 $\neg$(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: $i \ne j$. Then if $PK_i \ne PK_j$, the two "hash chains" always have a different input blocks at the end, and thus $U\|V = U'\|V'$ contradicts $\neg$(C-11). If $PK_i = PK_j$, then $K_i \ne K_j$ by $\neg$(C-1), and thus the new TGF query $\widetilde{\mathsf{SIC}}_{K_i}^{V\|1}(U) \to Z$ has nothing to do with the j-th user.
  - Subcase 2.3.2: $i = j$, yet $(N, A, \mathbf{c}) \ne (N', A', \mathbf{c}')$. Then as argued, the two hash chains necessarily have a different input blocks "in the middle", and thus $U\|V = U'\|V'$ contradicts $\neg$(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 $\neg$(C-2).
- Subcase 3.2: the subsequent (new) TGF query $(\widetilde{\mathsf{SIC}}_{K_i}^{V\|1})^{-1}(Z) \to U$ causes CHAIN event. This contradicts $\neg$(C-5).

By the above, we have

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{S2P}[\pi,\widetilde{\mathsf{SIC}}]_{\mathbf{K},\mathbf{PK}},\mathsf{L}^*,u} \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{2Q + 2nq_d}{2^n}. \tag{31}$$

This plus Eq. (30) yield Eq. (27):

$$\frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{Q}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{2Q + 2nq_d}{2^n} \le \frac{2u}{2^{n_p}} + \frac{32Q^2}{2^c} + \frac{3Q + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}.$$

F Proofs for Leakage Privacy Lemmas

F.1 Tester for LORL Advantage

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

```
1: Tester for LORL $\mathbf{Adv}^{\mathsf{LORL}}$
2: Let the challenging adversary $\mathcal{A}$ serve $s$ and $(m^0, m^1)$
3: Pick the secret: $y_{ch} \xleftarrow{\$} \{0,1\}^r$, $d \xleftarrow{\$} \{0,1\}$
4: $c \leftarrow y_{ch} \oplus m^d$, 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. According to our notational convention, this gives
   $\mathcal{A}$ the tuple $(c, [\mathsf{L}_\pi^{out}(y_{ch}\|s)]^p, \mathsf{L}_\oplus(y_{ch}, m^d), [\mathsf{L}_\oplus(y_{ch}, c)]^{p-1})$
7: Let $\mathcal{A}$ output the guess $d'$, $\mathcal{A}$ wins as long as $d' = d$.
```

F.2 A Useful Lemma: EavL Security of the Ideal Stream IdealS

The proof of Theorem 1 will rely on the EavL security of the ideal stream cipher $\mathsf{LIdealS}$, which is related to the term $\mathbf{Adv}^{\mathsf{LORL}}$ in Eq. (4). Formally,

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

$$\left|\Pr[\mathcal{A}^\pi(\mathsf{IdealS}(IV, A, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^\pi(\mathsf{IdealS}(IV, A, M^1)) \Rightarrow 1]\right| \le \ell \cdot \mathbf{Adv}^{\mathsf{LORL}}(p, q_\pi, O(t + p\ell t_l)),$$

where t_l is as defined in Lemma 1.

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 (p, q_π, t)-bounded adversary $\mathcal{A}^\pi$ against $\mathsf{IdealS}(IV, M_{h,i-1})$ and $\mathsf{IdealS}(IV, M_{h,i})$, we build a $(p, q_\pi, O(t + p\ell t_l))$-bounded adversary $\mathcal{A}_2^\pi$ against the distribution defined in Eq. (4). In detail, $\mathcal{A}_2^\pi$ proceeds in four steps:

(i) $\mathcal{A}_2^\pi$ samples $B \xleftarrow{\$} \{0,1\}^\kappa$, initializes an empty list leak, and sets $S'_0 \leftarrow IV\|B$;

(ii) for $j = 1, \ldots, i-1$, $\mathcal{A}_2^\pi$ samples $S_j \xleftarrow{\$} \{0,1\}^b$, computes $C[j] \leftarrow \mathsf{ms}_r(S_j) \oplus M^1[j]$, $S'_j \leftarrow C[j]\|\mathsf{ls}_c(S_j)$ ($S'_j \leftarrow C[j]\|(\mathsf{ls}_c(S_j) \oplus \delta_1)$ when $j = 1$), 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 $[\mathsf{L}_\oplus(\mathsf{ms}_r(S_j), C[j])]^{p-1}$ (and $\mathsf{L}_\oplus(\mathsf{ls}_c(S_j), \delta_1)$) to leak;

(iii) $\mathcal{A}_2^\pi$ samples $s \xleftarrow{\$} \{0,1\}^c$ and submits s to the LORL challenger. Assume that the outputs are (c^b, leak_b) with

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

$\mathcal{A}_2^\pi$ then adds the traces $[\mathsf{L}_\pi^{in}((M^1[i-1]\|0^c) \oplus S_{i-1}), \mathsf{L}_\pi^{out}(y_{ch}\|s)]^p$, $\mathsf{L}_\oplus(y_{ch}, m^b)$, and $[\mathsf{L}_\oplus(y_{ch}, c^b)]^{p-1}$ to leak;

(iv) $\mathcal{A}_2^\pi$ starts from $c^b\|C$ to emulate the remaining actions of IdealS 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 $=1$) 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 LORL challenger encrypting $M^0[i]$ or $M^1[i]$, the inputs to $\mathcal{A}^\pi$ capture IdealS encrypting $M_{h,i-1}$ or $M_{h,i}$. Moreover, $\mathcal{A}_2^\pi$ is $(p, q_\pi, O(t + p\ell t_l))$-bounded if $\mathcal{A}^\pi$ is (p, q_π, t)-bounded. Therefore,

$$\left|\Pr[\mathcal{A}^\pi(\mathsf{IdealS}(IV, M_{h,i-1})) \Rightarrow 1] - \Pr[\mathcal{A}^\pi(\mathsf{IdealS}(IV, M_{h,i})) \Rightarrow 1]\right| \le \mathbf{Adv}^{\mathsf{LORL}}(p, q_\pi, O(t + p\ell t_l))$$

by Eq. (6). This along with a simple summation implies the main claim. □

F.3 Proof of Theorem 1 (EavL Security of DuStr)

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

For each b, Lemma 1 indicates

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

Therefore,

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

F.4 Proof of Theorem 3 (muCCAmL1 of S1P)

As mentioned, the proof is built upon Theorem 1. For ease of use, we define

$$\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(p,q_\pi,t,\ell):=\max\big\{\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(\mathcal{A})\big\},$$

with the maximal taken over all $\mathcal{A}$ measuring the decryption leakages for $p-1$ times, making q_π permutation queries, running in time t, and choosing inputs with ℓ blocks. Eq. (12) thus gives a bound for $\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(p,q_\pi,t,\ell)$.

We then begin the $\mathsf{muCCAmL1}$ proof of $\mathsf{TETSponge}$. We start by defining G_0 as the game $\mathsf{PrivK}^{\mathsf{muCCAmL1},0}_{\mathcal{A},\mathsf{S1P},\mathsf{L}}$, and G_0^* as the game $\mathsf{PrivK}^{\mathsf{muCCAmL1},1}_{\mathcal{A},\mathsf{S1P},\mathsf{L}}$. We say a decryption query $\mathsf{Dec}_{\mathbf{K},\mathbf{PK}}(i,N,A,C)$ is *trivial* if the action $\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i,N,A,\star)\to C$ happened 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. (22), we have

$$\big|\Pr[\mathsf{G}_1\Rightarrow 1]-\Pr[\mathsf{G}_0\Rightarrow 1]\big|\leq\frac{u}{2^{n_p}}+\frac{16Q^2}{2^c}+\frac{(2n+1)Q+n^2q_{\widetilde{\mathsf{IC}}}}{2^n},\tag{32}$$

and (similarly)

$$\big|\Pr[\mathsf{G}_1^*\Rightarrow 1]-\Pr[\mathsf{G}_0^*\Rightarrow 1]\big|\leq\frac{u}{2^{n_p}}+\frac{16Q^2}{2^c}+\frac{(2n+1)Q+n^2q_{\widetilde{\mathsf{IC}}}}{2^n},\tag{33}$$

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

We then prove

$$\big|\Pr[\mathsf{G}_1\Rightarrow 1]-\Pr[\mathsf{G}_1^*\Rightarrow 1]\big|\leq\frac{3u}{2^{n_p}}+\frac{16Q^2}{2^c}+\frac{(2n+4)Q+2nq_d}{2^n}+\textstyle\sum_{i=1}^{q_e}\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(1,Q,t^*,\ell_i),\tag{34}$$

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 1, the last term is bounded as

$$\begin{aligned}\textstyle\sum_{i=1}^{q_e}\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(1,Q,t^*,\ell_i)\leq&\sigma\mathbf{Adv}^{\mathsf{LORL}}(1,Q,t^*)+2(\sigma+q_e)\mathbf{Adv}^{\mathsf{Inv}[c]}(1,Q,t^*,2Q)\\ &+2q_e\mathbf{Adv}^{\mathsf{Inv}[n]}(1,Q,t^*,2Q)+\underbrace{\textstyle\sum_{i=1}^{q_e}\frac{(\ell_i+2)^2}{2^c}}_{\leq\frac{\left(\sum_{i=1}^{q_e}(\ell_i+2)\right)^2}{2^c}\leq\frac{Q^2}{2^c}}.\end{aligned}$$

The above plus the gaps in Eq. (32) and Eq. (33) 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 $\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{DuStr}^1}(\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}})$.

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 $\mathrm{SICTable}(K,T,X)=Y$ and $\mathrm{SICTable}^{-1}(K,T,Y)=X$. $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ 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 $\mathsf{DuStr}^1_{B^*}[\pi](N^*\|PK_{i^*}\|0^*, A^*, M^*)$ 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 $(K_{i^*}, V^*\|1, U^*) \notin \text{SICTable}$). 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 $(K_{i^*}, PK_{i^*}\|0^*, N^*\|0^*) \in \text{SICTable}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ simply runs the encryption process $\mathsf{DuStr}^1_{B^*}[\pi](N^*\|PK_{i^*}\|0^*, A^*, M^*)$ 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 $\mathsf{Dec}_{\mathbf{K},\mathbf{PK}}(i, N, A, C) \neq \bot$, then we say another bad event $\mathsf{BadCheck}$ 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 $\mathsf{BadUserKey}$ didn't happen, it necessarily be $(K_{i_j}, PK_{i_j}\|0^*, N_j\|0^*) \notin \text{SICTable}$ 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 $\text{SICTable}(K_{i_\ell}, PK_{i_\ell}\|0^*, N_\ell\|0^*) \leftarrow B_\ell$ and $\text{SICTable}^{-1}(K_{i_\ell}, PK_{i_\ell}\|0^*, B_\ell) \leftarrow N_\ell\|0^*$, and then runs $\mathsf{DuStr}^1_{B_\ell}[\pi](M_\ell^0) \rightarrow \mathbf{c}_\ell\|U_\ell\|V_\ell$, 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 EavL challenger of DuStr^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 simulated $\widetilde{\mathsf{SIC}}$ 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 $\mathsf{BadInitKey}$, 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 $\Pr[\mathsf{BadInitKey}] \leq \frac{q_e+q_d+q_m}{2^n} \leq \frac{Q}{2^n}$ as $B_{ch} \xleftarrow{\$} \{0,1\}^n$ inside the challenger. For $\mathsf{BadCheck}$ we appeal to the intermediate results established in appendix D.3: in detail, Eq. (25) implies

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

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

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 $\sigma + q_m + q_e + q_d + q_\pi = Q$ 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

$$\begin{aligned}
&\Pr[\mathsf{G}_{2,j} \Rightarrow 1] - \Pr[\mathsf{G}_{2,j-1} \Rightarrow 1] \\
\leq &\Pr[\mathsf{G}_{2,j} \Rightarrow 1 \wedge \mathsf{Bad}\text{ in }\mathsf{G}_{2,j}] - \Pr[\mathsf{G}_{2,j-1} \Rightarrow 1 \wedge \mathsf{Bad}\text{ in }\mathsf{G}_{2,j-1}] + \mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(1, Q, t^*, \ell_i).
\end{aligned}$$

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, $\Pr[\mathsf{G}_{2,q_e} \Rightarrow 1] \geq \Pr[\mathsf{G}_{2,0} \Rightarrow 1]$. Then,

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

which is the claim in Eq. (34).

F.5 muCCAmL2 Security of TEDTSponge/S2P

We also need the non-invertibility & bounded XOR leakage assumptions. Following the notations of Section 3, we define the leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{Enc}}, \mathsf{L}_{\mathsf{Dec}})$ of its implementation as follows:

– $\mathsf{L}_{\mathsf{Enc}}$ consists of the leakages that are generated during the encryption:
  - the leakages $\mathsf{L}_\pi^{in}(S^{in})$ and $\mathsf{L}_\pi^{out}(S^{out})$ generated by all the internal calls to $\pi(S^{in}) \to S^{out}$, and
  - the leakages $\mathsf{L}_\oplus(a, b)$ generated by all the internal actions $a \oplus b$.
  - all the intermediate values involved in the computations of the hash $\mathsf{H}[\pi]$ (i.e., keyless functions are non-protected, and leak everything).

– $\mathsf{L}_{\mathsf{Dec}}$ consists of the above that are generated during the decryption.

We also use $\overrightarrow{q} = (p, q_m, q_e, q_d, q_\pi, q_{\widetilde{\mathsf{IC}}})$ to characterize the query power of a muCCAmL2 adversary.

Theorem 6. *Assume $u \le 2^{n_p}$, $n_p \le n$, $n \ge 5$, $2\sigma + 2(q_e + q_d + q_m) + q_\pi \le \min\{2^n/4, 2^b/2\}$, and* 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*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{S2P},\mathsf{L},u} \le & \frac{4u}{2^{n_p}} + \frac{49Q^2}{2^c} + \frac{5Q + 2nq_d + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \sigma \mathbf{Adv}^{\mathsf{LORL}}(p, Q, t^*) \\ & + 2q_e \mathbf{Adv}^{\mathsf{Inv}[n]}(p, Q, t^*, 2Q) + 2\sigma \mathbf{Adv}^{\mathsf{Inv}[c]}(p, Q, t^*, 2Q),\end{aligned} \tag{35}$$

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}.

Proof. The proof resembles appendix F.4: the main modification is to add treatments for decryption leakage (as here we consider muCCAmL2 rather than muCCAmL1). Recall that a decryption query $\mathsf{Dec}_{\mathbf{K},\mathbf{PK}}(i, N, A, C)$ is *trivial* if the action $\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i, N, A, M) \to C$ 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. (30), we have $|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_0 \Rightarrow 1]| \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{Q}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}$ and $|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_0^* \Rightarrow 1]| \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{Q}{2^n} + \frac{n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}$, where $Q = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$. We then prove

$$\Big|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_1^* \Rightarrow 1]\Big| \le \frac{2u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{3Q + 2nq_d}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(p, Q, t^*, \ell_i), \tag{36}$$

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 1, the last term is bounded as

$$\textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{LDuStr}}(p, Q, t^*, \ell_i) \le \sigma \cdot \mathbf{Adv}^{\mathsf{LORL}}(p, Q, t^*) + 2q_e \mathbf{Adv}^{\mathsf{Inv}[n]}(p, Q, t^*, 2Q) + 2\sigma \cdot \mathbf{Adv}^{\mathsf{Inv}[c]}(p, Q, t^*, 2Q) + \underbrace{\textstyle\sum_{i=1}^{q_e} \frac{(\ell_i+2)^2}{2^c}}_{\le \frac{Q^2}{2^c}}.$$

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 F.4: 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 $\mathbf{Adv}^{\mathsf{EavL}}_{\mathsf{DuStr}^0}(\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}})$.

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 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 $\mathcal{A}$, $\mathcal{A}_2^{\pi,\widetilde{\mathsf{IC}}}$ proceeds to simulate the encryption process of $\mathsf{S2P}[\widetilde{\mathsf{IC}}, \widetilde{\mathsf{SIC}}].\mathsf{LEnc}_{K_{i^*}, PK_{i^*}}(N^*, A^*, M^*)$ (which is similar to appendix F.4), 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 the decryption $\mathsf{S2P}[\pi, \widetilde{\mathsf{SIC}}].\mathsf{LDec}_{K_{i^*}, PK_{i^*}}(N^*, A^*, C^*)$, 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 leaking encryption 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 EavL challenger of DuStr^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 analyses are similar to appendix F.4: first,

$$\Pr[\mathsf{BadUserKey}] \le \frac{u^2}{2^{n+n_p}} \le \frac{u}{2^{n_p}}, \qquad \Pr[\mathsf{BadInitKey}] \le \frac{q_e + q_d + q_m}{2^n} \le \frac{Q}{2^n}, \text{ and}$$
$$\Pr[\mathsf{BadCheck}] \le \frac{u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{2Q + 2nq_d}{2^n},$$

where the last bound comes from Eq. (31) in appendix E, and $Q = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$. 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 queries to π and spends $t^* = O(t + p\sigma t_l)$ running time. Therefore,

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

which is the claim in Eq. (36). □

G Proof Sketch for Theorem 4 (Leakage Security of Ascon)

For muCIML2, the idea is simple: again if we reveal all the intermediate values to the adversary, Ascon collapses to a Hash-then-MAC instance, and the integrity follows.

More concretely, following the flow in appendix D, we first modify Ascon as follows to obtain an idealized scheme:

– for each user index i, we replace $\mathsf{KDF}_{K_i}(N) := \pi_1(IV \| K_i \| N) \oplus (0^{b-n} \| K_i)$ by a secret random function $F_i(N)$ that maps N to b-bit uniform values, and
– for each user index i, we replace $\mathsf{TGF}_{K_i}(S) := \mathsf{ls}_n\big(\pi_1(S \oplus (0^c \| K_i \| 0^{c/2}))\big) \oplus K_i$ by a secret Tweakable Random Function (TRF) $G_i(\mathsf{ms}_r(S), \mathsf{ls}_c(S))$ that maps b-bit inputs to c-bit uniform values. As will be seen, using a TRF helps unify the integrity analyses of Ascon and ISAP.

While appearing complicated, analysis of the gap due to these modifications follows the multi-instance PRF security proof of the partial-key Even-Mansour cipher, and produces a bound

$$\frac{u^2}{2^n} + O\Big(\frac{uq}{2^n}\Big) + O\Big(\frac{q^2}{2^b}\Big) = \frac{u^2}{2^n} + O\Big(\frac{uq}{2^n}\Big) + O\Big(\frac{(q+\sigma)^2}{2^c}\Big). \tag{37}$$

We remark that, in the single-user setting, the gap due to this step can be bounded to $O\big(\frac{q}{2^n}\big)$ via a sophisticated, multicollision-based argument. Eq. (37) thus has demonstrated the multi-user degradation.

This idealized Ascon variant collapses to a Hash-then-TRP variant shown in Fig. 6, with $\mathbf{c}_L$ denoting $C[1] \| \ldots \| C[\ell-1]$, i.e., the ciphertext except for the last block, and tw denoting the last $C[\ell]$.

Then, we consider the "CHAIN" event: at any time, for the i th user there exists a chain of permutation queries from the F_i query record $F_i(N) = S_0$ to the G_i query record $G_i(\mathsf{ms}_r(S'), \mathsf{ls}_c(S'))$, such that there didn't exist any encryption query of the form $\mathsf{LEnc}(i, N, A, \star) \to \mathbf{c} \| Z$ for the corresponding A and $\mathbf{c}$. To bound $\Pr[\mathsf{CHAIN}]$, we adapt the bad conditioned in appendix D.3 as follows:

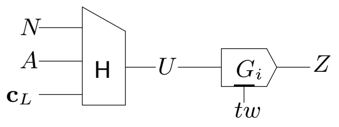

Fig. 6: The simplified Hash-then-TRF authenticator for Ascon and ISAP.

- (C-1) There exists two user indices j, ℓ such that $K_j = K_\ell$.
- (C-2) Right after a forward π_2 query $\pi_2(S^{in}) \to S^{out}$ happens, if:
  - (C-21) there exists another π_2 query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-1}(S^{out}) = \mathsf{ls}_{c-1}(S^{in'})$, $\mathsf{ls}_{c-1}(S^{out}) = \mathsf{ls}_{c-1}(S^{out'})$; or
  - (C-22) there exists a G_i query $G_i(tw, U)$ such that $\mathsf{ls}_c(S^{out}) = U$; or
- (C-4) Right after a backward π_2 query $\pi_2^{-1}(S^{out}) \to S^{in}$ happens, if:
  - (C-41) there exists another π_2 query $(S^{in'}, S^{out'})$ such that $\mathsf{ls}_{c-1}(S^{in}) = \mathsf{ls}_{c-1}(S^{out'})$, or
  - (C-42) there exists a F_i query record $F_i(N) = S_0$ such that $\mathsf{ls}_{c-1}(S^{in}) = \mathsf{ls}_{c-1}(S_0)$.
- (C-5) Right after a F_i query $F_i(N) \to S_0$ happens, there exists a π_2 query (S^{in}, S^{out}) such that $\mathsf{ls}_{c-1}(S_0) = \mathsf{ls}_{c-1}(S^{in})$.

Via an analysis similar to appendix D.3, we could have $\Pr[\mathsf{Bad}] = O\big(\frac{u^2}{2^n}\big) + O\big(\frac{(q+\sigma)^2}{2^c}\big)$. Below we show that the event CHAIN cannot be triggered by adversarial permutation and encryption queries. 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 $\neg$(C-2); if this query is backward, then it contradicts $\neg$(C-4).

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

- Subcase 2.1: a subsequent (forward) π query causes CHAIN event. This again contradicts $\neg$(C-2).
- Subcase 2.2: the subsequent (new) TGF query $G_i(C[\ell], U) \to Z$ causes CHAIN event. Assume that the involved input tuple record is $(N, A, C[1]\|\ldots\|C[\ell])$. The assumption means there exists another input tuple record of the i-th user $(N', A', \mathbf{c}')$ such that $\mathsf{H}(N, A, C[1]\|\ldots\|C[\ell-1]) = U = U' = \mathsf{H}(N', A', C[1]\|\ldots\|C'[\ell'-1])$. We further have two cases:
  - Case 1: $(N, A, C[1]\|\ldots\|C[\ell-1]) \neq (N', A', C'[1]\|\ldots\|C'[\ell'-1])$. Then due to the suffix-free padding in Ascon, and due to the domain separation bits between the two processing phases, the two tuples necessarily result in distinct computation chains, which implies $U \neq U'$ by $\neg$(C-2);
  - Case 2: $\ell = \ell'$, and $(N, A, C[1]\|\ldots\|C[\ell-1]) \neq (N', A', C'[1]\|\ldots\|C'[\ell'-1])$. Then the condition $C[\ell] \neq C'[\ell]$ means the query $G_i(C[\ell], U) \to Z$ cannot create a chain w.r.t. $(N', A', \mathbf{c}')$.

Thus the impossibility.

Finally, we bound the probability that a decryption query returns non-empty conditioned on that CHAIN was never triggered by π and encryption queries. Consider a non-trivial decryption query $\mathsf{LDec}_{\mathbf{K}}(i, N, A, \mathbf{c}\|Z)$, where $\mathbf{c} = C[1]\|\ldots\|C[\ell]$. If there has been a chain corresponding to $(N, A, \mathbf{c})$, then it was created by a previous encryption query $\mathsf{LEnc}_{\mathbf{K}}(i, N, A, M) \to \mathbf{c}\|Z^*$. Since $\mathsf{LDec}_{\mathbf{K}}(i, N, A, C)$ is non-trivial, it has to be $Z^* \neq Z$, and thus $\mathsf{LDec}_{\mathbf{K}}(i, N, A, C)$ must return $\perp$. Otherwise, conditioned on the previous query-response transcript, G_i was never called on the input $(C[\ell], U)$, U is the formatted input corresponding to $(N, A, C[1]\|\ldots\|C[\ell-1])$: the argument is similar to the previous argument for encryption queries don't trigger CHAIN event. Therefore, the output $G_i(C[\ell], U)$ is necessarily uniform, and thus $\Pr[Z^* = G_i(C[\ell], U) = Z] \leq \frac{2}{2^n}$. Taking a union over all the q decryption queries yields the term $\frac{q}{2^n}$. Summing over this term, $\Pr[\mathsf{Bad}]$, and Eq. (37) yield Eq. (14).

For muCCAmL1, the first step is also to replace the $\mathsf{KDF}_{K_i}(N)$ and $\mathsf{TGF}_{K_i}(S)$ calls by $F_i(N)$ and $G_i(tw, U)$, i.e., in the same way as mentioned. Now, assume that the muCCAmL1 adversary $\mathcal{A}$ only makes challenge encryption queries with non-empty associated data: it's clear that this assumption can only amplifies $\mathcal{A}$'s advantage. With this, in the subsequent hybrid argument, the constructed muCCAmL1 adversary $\mathcal{A}_2$ samples b-bit values to emulate the random functions $F_1, \ldots, F_u$. For the challenge encryption query $\mathsf{LEnc}_{\mathbf{K}}(i, N, A, M)$, assume that the sampled b-bit value of $F_i(N)$ is S_0, then $\mathcal{A}_2$ takes $A[1] \oplus \mathsf{ms}_r(S_0)$ as the IV to query his EavL challenger of the duplex-bases stream cipher DuStr: the subsequent argument is basically the same as appendix F.4. In all, similarly to appendix F.4, the final muCCAmL1 bound is the muCIML2 bound plus the terms $O(\sigma) \cdot \Big(\mathbf{Adv}^{\mathsf{LORL}}(1, O(q), O(t)) + \mathbf{Adv}^{\mathsf{Inv}[c]}(1, O(q), O(t), O(q))\Big)$. Note that here the size of the initial key of DuStr is c rather than n, and thus we don't have the (n-bit non-invertibility) term $O(\sigma) \cdot \mathbf{Adv}^{\mathsf{Inv}[n]}(1, O(q), O(t), O(q))$ any more.

H Proof Sketch for Leakage Security of ISAP

For muCIML2, we again follow the flow in appendix D and first modify ISAP as follows to obtain an idealized scheme:

– for each user index i, we replace $\mathsf{KDF}_{K_i}(N)$ by a secret random function $F_i(N)$ that maps N to $(b-n)$-bit uniform values, and
– for each user index i, we replace $\mathsf{TGF}_{K_i}(U,Y)$ by a secret Tweakable Random Function (TRF) $G_i(U,Y)$ that maps an n-bit block Y and a $b-n$ bit tweak U to n-bit uniform values.

To obtain a good bound, less straightforward technique has to be used. Concretely, we use two steps. We first replace ISAPRK by random functions: the ISAPRK instance with the initial vector IV_{KE} by $F_i(N)$ (thus this covers the transition $\mathsf{KDF}_{K_i}(N) \longrightarrow F_i(N)$), while the ISAPRK instance with the initial vector IV_{KA} by $F_i^*(N)$. The gap is the standard multi-instance PRF security of the inner keyed duplex, and is (such a bound can be found in [17], Theorem 1, in the case of sampling key *with replacement*)

$$O\Big(\frac{u^2}{2^n}\Big)+O\Big(\frac{u(q+\sigma)}{2^n}\Big)+O\Big(\frac{(q+\sigma)^2}{2^b}\Big)+O\Big(\frac{(q+\sigma)^2}{2^{b-n}}\Big). \tag{38}$$

We then use the H-coefficients technique to further bound the gap due to the transition $\mathsf{TGF}_{K_i}(U,Y) \longrightarrow G_i(U,Y)$. The conditions for bad transcripts roughly consist of the following:

– There exists n hash records $\mathsf{Hash}(N_1,A_1,\mathbf{c}_1)=Y_1\|U_1,\ldots,\mathsf{Hash}(N_n,A_n,\mathbf{c}_n)=Y_n\|U_n$ such that $Y_1=\ldots=Y_n$. Following the analysis for TEDTSponge, the probability is $O\big(\frac{(q+\sigma)^2}{2^c}\big)+O\big(\frac{q+\sigma}{2^n}\big)$.
– For the i-th user, there exists a hash record $\mathsf{Hash}(N,A,\mathbf{c})=Y\|U$ such that $F_i^*(Y)\|U\in\tau_{\pi_4}$. Since multi-collisions on Y has been limited to n by the 1st condition, we are able to obtain a probability of $O\big(\frac{n(q+\sigma)}{2^n}\big)$.[8]

With these, a simple analysis bounds the gap due to $\mathsf{TGF}_{K_i}(U,Y) \longrightarrow G_i(U,Y)$ to

$$O\Big(\frac{(q+\sigma)^2}{2^c}\Big)+O\Big(\frac{q+\sigma}{2^n}\Big)+O\Big(\frac{n(q+\sigma)}{2^n}\Big). \tag{39}$$

The resulted idealized ISAP collapses to a Hash-then-TRP MAC, i.e., it consists of two steps: (1) $Y\|U \leftarrow \mathsf{Hash}(N,A,\mathbf{c})$, (2) $Z\leftarrow G_i(U,Y)$. It's similar to Fig. 6—unlike Ascon, now the tweak input tw is an output halve of the hash rather than something controlled by the adversary. It's not hard to prove that the sponge-based hash function $\mathsf{Hash}(N,A,\mathbf{c})$ is collision resistant with the bound $O\big(\frac{(q+\sigma)^2}{2^c}\big)$: we are not aware of existing work addressing this non-standard use, but the proof is simple (follow the ideas in appendix E) and we omit. As long as no hash collision occurs during the interaction, every non-trivial decryption query $\mathsf{LDec}_{\mathbf{K}}(i,N,A,\mathbf{c}\|Z)$ results in a new call to $G_i(U,Y)\to Z'$ (as every early query necessarily results in $Y'\|U'\neq Y\|U$), and thus $\Pr[Z'=Z]=O\big(\frac{1}{2^n}\big)$. Taking a union bound over the q decryption queries, and further summing over all the terms including Eq. (38) and Eq. (39) yield the bound (note that we assumed $b-n\geq c$).

For its muCCAmL2 security, we also need the implementation to satisfy the non-invertibility & bounded XOR leakage assumptions. Following the notations of Section 3, we define the leakage function $\mathsf{L}=(\mathsf{L}_{\mathsf{Enc}},\mathsf{L}_{\mathsf{Dec}})$ of its implementation as follows:

– $\mathsf{L}_{\mathsf{Enc}}$ consists of the leakages that are generated during the encryption:
  • the leakages $\mathsf{L}_\pi^{in}(S^{in})$ and $\mathsf{L}_\pi^{out}(S^{out})$ generated by all the internal calls to $\pi(S^{in})\to S^{out}$, and
  • the leakages $\mathsf{L}_\oplus(a,b)$ generated by all the internal actions $a\oplus b$.
  • all the intermediate values involved in the computations of the hash $\mathsf{H}[\pi]$ (i.e., keyless functions are non-protected, and leak everything).
– $\mathsf{L}_{\mathsf{Dec}}$ consists of the above that are generated during the decryption.

Then, assuming leak-freeness of $\mathsf{KDF}_K(N)$ and $\mathsf{IntCheck}_K(Y\|U,Z)$, the proof flow is the same as that for TEDTSponge (Appendix F.5)—note that we've demonstrated the transitions $\mathsf{KDF}_{K_i}(N)\longrightarrow F_i(N)$ and $\mathsf{TGF}_{K_i}(U,Y)\longrightarrow G_i(U,Y)$ before, which is helpful for the argument.

H.1 Leakage-resilience of ISAPRK (sketch)

Formally, we show that $\mathsf{ISAPRK}_K(Y)$ (defined as follows) is a leakage-resilient PRF, in the sense of indistinguishability from the following leaking random function LF upon less than q queries. Note that LF contains a uniformly distributed "faked" key K, but it only affects the leakages rather than the function output.

[8]Otherwise it's the inferior $O\big(\frac{(q+\sigma)^2}{2^n}\big)$, i.e., birthday $2^{n/2}$ security even in the single-user setting.

```
algorithm ISAPRK$_K(Y)$
1. $Y_1, \ldots, Y_n \leftarrow Y$
2. $S_0 \leftarrow K \| IV$, $S_1 \leftarrow \pi(S_0)$
3. for $i = 1$ to $n-1$ do
4.     $S'_i \leftarrow S_i \oplus (Y_i \| 0^{b-1})$
5.     $S_{i+1} \leftarrow \pi(S'_i)$
6. $S'_n \leftarrow S_n \oplus (Y_n \| 0^{b-1})$
7. $S_f \leftarrow \pi(S'_n)$
8. $K^* \leftarrow \mathsf{ms}_{b-n}(S_f)$
9. return $K^*$
```

```
algorithm $LF_K(Y)$
1. $Y_1, \ldots, Y_n \leftarrow Y$
2. $S_0 \leftarrow K \| IV$, $S_1 \xleftarrow{\$} \{0,1\}^b$
3. for $i = 1$ to $n-1$ do
4.     $S'_i \leftarrow S_i \oplus (Y_i \| 0^{b-1})$
5.     $S_{i+1} \xleftarrow{\$} \{0,1\}^b$
6. $S'_n \leftarrow S_n \oplus (Y_n \| 0^{b-1})$
7. $S_f \xleftarrow{\$} \{0,1\}^b$
8. $K^* \leftarrow \mathsf{ms}_{b-n}(S_f)$
9. return $K^*$
```

Assume that the total number of queries to the oracle is at most q. Then we extend the non-invertibility assumption of Eq. (1) to the following:

$$\mathbf{Adv}^{\mathsf{sSPA}}(\mathcal{A}) := \Pr\big[s_{ch} \xleftarrow{\$} \{0,1\}^{b-1}, \mathcal{G} \leftarrow \mathcal{A}^{\pi}(\mathsf{leak}) : s_{ch} \in \mathcal{G}\big], \tag{40}$$

where $\mathcal{G}$ is a finite set of guesses, and $\mathcal{A}$'s input leak is a list of leakages:

$$\mathsf{leak} = \big[\mathsf{L}_{\pi}^{out}(0\|s_{ch}), \mathsf{L}_{\pi}^{out}(1\|s_{ch}), \mathsf{L}_{\pi}^{in}(0\|s_{ch}), \mathsf{L}_{\pi}^{in}(1\|s_{ch})\big]^{q}. \tag{41}$$

The superscript sSPA stands for "super SPA", meaning that it assumes SPA security but allows an arbitrary number of repeated measuring. We also introduce the simpler notation

$$\mathbf{Adv}^{\mathsf{sSPA}}(q, t, N_G) := \max_{(q,t,N_G)\text{-}\mathcal{A}} \Big\{\mathbf{Adv}^{\mathsf{Inv}[\omega]}(\mathcal{A})\Big\}, \tag{42}$$

with the maximal taken over all adversaries running in time t, outputting N_G guesses, and making at most q queries to its permutation oracle π.

With this assumption, and following the proof of Lemma 1, it can be shown that for any (q,t)-adversary $\mathcal{A}$, it holds

$$\Big|\Pr[\mathcal{A}^{\mathsf{ISAPRK}_K,\pi,\pi^{-1}} \Rightarrow 1] - \Pr[\mathcal{A}^{LF,\pi,\pi^{-1}} \Rightarrow 1]\Big| \le nq \cdot \mathbf{Adv}^{\mathsf{sSPA}}\big(O(q), O(t), O(q)\big) + O\Big(\frac{n^2q^2}{2^b}\Big).$$

To wit, the reduction is possible because during the lifetime of the rate-1 duplex ISAPRK_K, with probability $\ge 1 - O\Big(\frac{n^2q^2}{2^b}\Big)$ (excluding state collisions), each $b-1$ bit secret state can only be involved in 2 different outputs and 2 different inputs, which are all presented in Eq. (41). So the reduction simply picks the corresponding leakage from this list for simulating.

Comparison to our modes. The super SPA assumption shall be compared to the leak-free TBC in our modes TETSponge and TEDTSponge, but the discussion isn't feasible here (a huge amount of experiments is expected) and is left for future work.

Comparison to DM. DM [21] gave a similar analysis using their high min-entropy leakage assumption. They assumed deterministic leakage function, so it's not beneficial at all to repeat measuring a leakage for Q times. This could be seen as a simplified and more concise model of leakages. We feel that our "$[\mathsf{L}_{\pi}]^{Q}$" model emphasizes the details in the SPA assumption (i.e., it shall withstand repeated measuring during the whole lifetime of the main key), with the expense of a higher rotational-level complexity.

I Black-box CCA Security of TETSponge and TEDTSponge

At the end, let's fill in the missed piece, i.e., the black-box CCA security results of our two new modes TETSponge and TEDTSponge. We'll first define the CCA notion in question, then present the two theorems, and then their proofs.

I.1 CCA with misuse-resilience

Ashur et al. [4] 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* [38], we will refer to *misuse-resilience* as 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 6 **(muCCAm\$ advantage).** *Given a nonce-based authenticated encryption* AEAD = (Enc, Dec)*, the* multi-user chosen ciphertext misuse resilience advantage *of an adversary $\mathcal{A}$ against* AEAD *with u users is*

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

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 $1 \le i \le u$, 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.*

Note that this notion has included a notion of integrity since decryption are required to always return $\perp$. This is of the same spirit with the CCA3 notion suggested in [38].

I.2 muCCAm\$ security of our modes

We can then deliver the results. To ease comparison, we defer their proofs. 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 7. *Assume $u \le 2^{n_p}$, $n_p \le n$, $n \ge 5$, and $Q = \sigma + q_e + q_d + q_m + q_\pi \le \min\{2^n/4, 2^b/2\}$. Then in the ideal TBC and permutation model, for any $(\overrightarrow{q}, \sigma)$-adversary $\mathcal{A}$ it holds*

$$\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathcal{A},\mathsf{TETSponge},u} \le \frac{5u}{2^{n_p}} + \frac{56Q^2}{2^c} + \frac{23nQ + 5n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{43}$$

The proof is deferred to Appendix I.3.

Theorem 8. *Assume $u \le 2^{n_p}$, $n_p \le n$, $n \ge 5$, and $Q = 2\sigma + 2(q_e + q_d + q_m) + q_\pi \le \min\left\{2^n/4, 2^b/2\right\}$. 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:*

$$\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathcal{A},\mathsf{TEDTSponge},u} \le \frac{4u}{2^{n_p}} + \frac{53Q^2}{2^c} + \frac{8nQ + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{44}$$

The proof is offered in Appendix I.4. The bounds are comparable to their muCIML2 bounds, i.e., implying high 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 I.3, 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 $2^{c/2}$ [31,14], we will see that the term $2^{c/2}$ constitutes the muCIML2 security limit and cannot be improved. Thus, better muCCAm\$ bounds cannot result in a smaller choice of c when muCIML2 security is a concern.

I.3 Proof of Theorem 7 (TETSponge)

Note that in the misuse resilience setting, schemes which achieve both CPA confidentiality and authenticity also achieve CCA confidentiality [4]:

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

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.

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

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

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 D, we organize τ_h^* from τ_π^*. Note that the queries underlying the challenge encryption queries are not revealed (unlike appendix D). As such, τ_h^* only contains the $\mathsf{S1P}$ 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

$$\Pr[T_{id} = \tau] = \Pr[\mathbf{K}, \mathbf{PK}] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \Pr[\widetilde{\mathsf{SIC}} \vdash \tau_{\widetilde{\mathsf{SIC}}}] \cdot \Pr[\pi \vdash \tau_\pi^*] \cdot \left(\frac{1}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}. \tag{45}$$

Below we write

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

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

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) $\mu_{PK} \ge n+1$, $\mu_V \ge n+1$.*
- *(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., $\left|\{(j,\ell) : Z_j = Z_\ell\}\right| \ge q_e$.*
- *(B-6) $\left|\{\big((i, N, A, M, \mathbf{c}\|Z), (K, V\|1, X, Y)\big) \in \tau_e \times (\tau_{\widetilde{\mathsf{IC}}} \cup \tau_{\widetilde{\mathsf{SIC}}}) : Z = Y\}\right| \ge q_e$.*

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

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

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

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

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

Similarly,

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

In all,

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \le \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{(2n+1)q_\pi^* + q_e + q_{\widetilde{\mathsf{IC}}} + q_m + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} \le \frac{2u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{4nQ + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{46}$$

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. (21))

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

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,

$$\Pr_{\widetilde{\mathsf{IC}}}\left[\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \mid \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}\right] \leq \frac{2nq_\pi^* + 2nq_e}{2^n}. \tag{47}$$

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

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

$S_{j,\nu_j+\ell_j}^{in} = C_j[\ell_j]\|\mathsf{ls}_c\big(S_{j,\nu_j+\ell_j-1}^{out} \oplus (0^r\|[1]_2\|0^{c-2})\big)$ 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 $U_j\|V_j = \mathsf{ms}_{2n-1}(S_{j,\nu_j+\ell_j}^{out})$. 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 $\big(\star\|\mathsf{ls}_{c-2}(S_{j,\ell}^{in}), \star\big) \in \tau_\pi^*$.
– (C-4) there exists four indices $j_1, j_2 \in [1, \ldots, q_e]$ and $\ell_1 \in [1, \ldots, \nu_{j_1} + \ell_{j1}], \ell_2 \in [1, \ldots, \nu_{j_2} + \ell_{j2}]$ such that $(j_1, \ell_1) \neq (j_2, \ell_2)$, and $\mathsf{ls}_{c-2}(S_{j1,\ell_1}^{in}) = \mathsf{ls}_{c-2}(S_{j2,\ell_2}^{in})$.
– (C-5) there exists an index $j \in [1, \ldots, q_e]$ such that any of the following is fulfilled:
  - $(K_{i_j}, V_j\|1, U_j, \star) \in (\tau_{\widetilde{\mathsf{IC}}} \cup \tau_{\widetilde{\mathsf{SIC}}})$ or $(K_{i_j}, V_j\|1, \star, Z_j) \in (\tau_{\widetilde{\mathsf{IC}}} \cup \tau_{\widetilde{\mathsf{SIC}}})$, 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 $\mathsf{ls}_{c-2}(S_{j,1}^{in})$ is uniform due to $\neg$(C-1). Therefore,

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

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

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

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

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

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 $\neg$(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) $\Pr[\exists j : (K_{i_j}, V_j\|1, U_j, \star) \in (\tau_{\widetilde{\mathsf{IC}}} \cup \tau_{\widetilde{\mathsf{SIC}}})] \leq q_e(q_{\widetilde{\mathsf{IC}}} + q_m) \cdot \frac{2}{2^{2n-1}}$ 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) $\Pr[\exists j : (K_{i_j}, V_j\|1, \star, Z_j) \in (\tau_{\widetilde{\mathsf{IC}}} \cup \tau_{\widetilde{\mathsf{SIC}}})] \leq \frac{2q_e}{2^{n-1}}$, and (c) by $\neg$(B-5), $\Pr[\exists j,\ell : V_j\|Z_j = V_\ell\|Z_\ell] \leq q_e \cdot \frac{2}{2^{n-1}}$. Therefore,

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

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 $q_e + \sigma_2 + q_\pi^* \leq Q$ yield

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

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 $\varDelta_j[l] = M_j[l] \oplus C_j[l]$, i.e., $\Pr[\mathsf{ms}_r(S^{in}_{j,\nu+l}) = M_j[l] \oplus C_j[l]]$. This means $\mathsf{ms}_r(S^{in}_{j,\nu+l}) = \varDelta_j[l]\|s \notin \tau_\pi^*$ 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 $\varDelta_j[l]\|s^* \in \tau_\pi^*$. Therefore,

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

By this,

$$\begin{aligned}
&\Pr[\forall j \in \{1,\ldots,q_e\} : \mathsf{S1P}[\pi, \widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j, N_j, A_j, M_j) = \mathbf{c}_j \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \\
&\geq \Pr[\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \wedge \neg\mathsf{Bad}(\pi) \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}} \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{SIC}}}] \cdot \left(\frac{1 - \frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c}.
\end{aligned} \tag{48}$$

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 index $j \in \{1,\ldots,q_e\}$, under the condition that $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}$ and the condition that "$\forall j \in \{1,\ldots,q_e\} : \mathsf{S1P}[\pi, \widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j, N_j, A_j, M_j) = \mathbf{c}_j$", 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

$$\Pr[\widetilde{\mathsf{IC}}^{V_j\|1}_{K_{i_j}}(U_j) = Z_j \text{ for } j = 1,\ldots,q_e] \geq \left(\frac{1}{2^n}\right)^{q_e}. \tag{49}$$

In summary,

$$\frac{\Pr[T_{re}=\tau]}{\Pr[T_{id}=\tau]} \geq \frac{\Pr[\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \wedge \neg\mathsf{Bad}(\pi) \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \left(\frac{1-\frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}}{\left(\frac{1}{2^r}\right)^{\sigma_c} \cdot \left(\frac{1}{2^n}\right)^{q_e}}$$

$$\geq \Pr[\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \wedge \neg\mathsf{Bad}(\pi) \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \cdot \left(1-\frac{q_\pi^*}{2^c}\right)^{\sigma_c}$$

$$\geq \left(1 - \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma Q}{2^c} - \frac{2q_{\widetilde{\mathsf{IC}}} + 10q_e}{2^n}\right) \cdot \left(1 - \frac{Q}{2^c}\right)^{\sigma_c}$$

$$\geq 1 - \frac{2nq_\pi^* + 2nq_e + 2q_{\widetilde{\mathsf{IC}}} + 10q_e}{2^n} - \frac{8\sigma Q}{2^c}.$$

Gathering this and Eq. (46) and further the muCIML2 bound Eq. (26) yield Eq. (43):

$$\frac{3u}{2^{n_p}} + \frac{32(q_\pi^*)^2}{2^c} + \frac{5nq_\pi^* + 2nq_d + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2u}{2^{n_p}} + \frac{16(q_\pi^*)^2}{2^c} + \frac{4nq_\pi^* + 2n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2nq_\pi^* + 2nq_e + 2q_{\widetilde{\mathsf{IC}}} + 10q_e}{2^n} + \frac{8\sigma Q}{2^c}$$

$$\leq \frac{5u}{2^{n_p}} + \frac{56Q^2}{2^c} + \frac{23nQ + 5n^2 q_{\widetilde{\mathsf{IC}}}}{2^n} \qquad (2nq_e + 2nq_d \leq 2nQ).$$

I.4 Proof of Theorem 8 (TEDTSponge)

Similarly to appendix I.3, 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 the distinguisher's queries to the challenge (second) encryption oracle in the list $\tau_e = \Big((i_1, N_1, A_1, M_1, C_1), \ldots, (i_{q_e}, N_{q_e}, A_{q_e}, M_{q_e}, C_{q_e})\Big)$. 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

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

and

$$Q = 2\sigma + 2(q_e + q_d + q_m) + q_\pi.$$

We also have the ideal TBC query transcript $\tau_{\widetilde{\mathsf{IC}}}$. Similarly to appendix I.3, 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 5, and we recycle the bound (which already includes the hash collision event):

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

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

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \le \frac{2u}{2^{n_p}} + \frac{16Q^2}{2^c} + \frac{Q + n^2 q_{\widetilde{\mathsf{IC}}} + 2q_e}{2^n}. \tag{50}$$

Ratio of Probabilities. This step could reuse the results in appendix I.3. Concretely, we define $\mathsf{BadKD}(\widetilde{\mathsf{IC}})$ and $\mathsf{Bad}(\pi)$ as those in appendix I.3 without the condition (C-5). As such, we could recycle Eqs. (48) and (49) in the following manner:

$$\Pr[\forall j \in \{1,\dots,q_e\} : \mathsf{S2P}[\pi, \widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{PK}}(i_j, N_j, A_j, M_j) = \mathbf{c}_j \mid \pi \vdash \tau_\pi^* \wedge \widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\mathsf{IC}}}] \ge \left(1 - \frac{2nq_\pi^* + 2nq_e}{2^n} - \frac{8\sigma Q}{2^c}\right) \cdot \left(\frac{1 - \frac{q_\pi^*}{2^c}}{2^r}\right)^{\sigma_c},$$

$$\Pr[\forall j \in \{1,\dots,q_e\} : \widetilde{\mathsf{IC}}_{K_{i_j}}^{V_j\|1}(U_j) = Z_j] \ge \left(\frac{1}{2^n}\right)^{q_e}.$$

Therefore,

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

Gathering this and Eq. (50) and further Eq. (27) (with that $q_\pi^* = 2\sigma + 2(q_e + q_d) + q_\pi$ replaced by $Q = 2\sigma + 2(q_e + q_d + q_m) + q_\pi$) yield Eq. (44):

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