TEDT, a Leakage-Resistant AEAD Mode for High Physical Security Applications

Francesco Berti[1], Chun Guo[1,2], Olivier Pereira[1], Thomas Peters[1] and François-Xavier Standaert[1]

[1] ICTEAM/ELEN/Crypto Group, UCL, Louvain-la-Neuve, Belgium
[2] School of Cyber Science and Technology and Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University

Abstract. We propose TEDT, a new Authenticated Encryption with Associated Data (AEAD) mode leveraging Tweakable Block Ciphers (TBCs). TEDT provides the following features: (i) It offers full leakage-resistance, that is, it limits the exploitability of physical leakages via side-channel attacks, even if these leakages happen during every message encryption and decryption operation. Moreover, the leakage integrity bound is asymptotically optimal in the multi-user setting. (ii) It offers nonce misuse-resilience, that is, the repetition of nonces does not impact the security of ciphertexts produced with fresh nonces. (iii) It can be implemented with a remarkably low energy cost when strong resistance to side-channel attacks is needed, supports online encryption and handles static and incremental associated data efficiently. Concretely, TEDT encourages so-called leveled implementations, in which two TBCs are implemented: the first one needs strong and energy demanding protections against side-channel attacks but is used in a limited way, while the other only requires weak and energy-efficient protections and performs the bulk of the computation. As a result, TEDT leads to more energy-efficient implementations compared to traditional AEAD schemes, whose side-channel security requires to uniformly protect every (T)BC execution.

Keywords: Authenticated encryption · leakage-resilience/resitance · beyond-birthday bound · multi-user security · masking countermeasure · leveled implementations

1 Introduction

The development of Authenticated Encryption with Associated Data (AEAD) schemes has been an area of extremely active research since the beginning of this millennium. Numerous prominent designs have emerged and, on top of the traditional confidentiality and integrity requirements [KY01, BN08], a number of desirable functional and (sometimes conflicting) security properties emerged. This paper proposes TEDT, a new AEAD mode for tweakable block ciphers that primarily aims at a high efficiency when a strong resistance to side-channel attacks is needed, which are among the most practical threats against cryptographic implementations, as highlighted in a recent white paper [ABB+, chapter 1.1] – see also [EKM+08, MBKP11, ZYSQ13, BGRV15, GPT15, GST17, DK18].

Every time an encryption or a decryption operation takes place, some side-effects may be observable, which can leak information on the internal state of a computing device, including keys: these can be timing, power consumption and electromagnetic radiation measurements. Such attacks can be mounted in two main flavors which, in the context of power consumption, are called Simple Power Analysis (SPA) and Differential Power Analysis (DPA) [MOP07]. In an SPA, an attacker takes advantage of the leakages resulting from a single input (message) provided for encryption, with measurements that are possibly repeated multiple times in order to remove the noise in measurements. A DPA exploits

Licensed under Creative Commons License CC-BY 4.0.
IACR Transactions on Cryptographic Hardware and Embedded Systems ISSN 2569-2925, Vol. 2020, No. 1, pp. 256–320
DOI:10.13154/tches.v2020.i1.256-320

the leakages resulting from multiple inputs, which all provide new information about the internal state of the device, reducing the computational secrecy of this state at a rate that is exponential in the number of distinct inputs.

Standard AEAD are typically susceptible to the mounting of a DPA attack: for instance, prominent modes like OCB [RBBK01], CCM [WHF03], or GCM [MV04] all evaluate a block cipher used with a single key on a distinct input for each message block, which is the exact setting in which DPA applies.

The protection against side-channel attacks is then typically left to engineers who will implement the block ciphers and other components of the AEAD in a way that limits the leakages as much as possible. One of the most popular countermeasures is masking [CJRR99], in which the internal state of the device is secret-shared into a number of pieces (leading to so-called higher-order security under some noise and independence assumptions [DFS19]), which are then used for the computation. The strong protection of a block cipher against DPA usually decreases the standard performance metrics of both software and hardware implementations by orders of magnitude compared to non-protected implementations [GR17, GMK17]. The overheads for the full AEAD mode are of the same order, since all message blocks are processed by the strongly protected block cipher.

Another approach consists in designing leakage-resilient or leakage-resistant modes of operations [BMOS17, BKP+18, BPPS17, DEM+17].[1] These modes, which often come with some computational overheads in the black-box setting, e.g., require more block cipher calls than a standard AEAD or require more keying material, aim at considerably reducing the effect of leakages and the possibility to mount a DPA. A first classical ingredient is to use some form of key update or re-keying [DP08, YSPY10] in order to make sure that each execution of a block cipher leaks about a different key, hence effectively leaving the adversary with the possibility to mount an SPA only. A second common ingredient, required in modes aiming at integrity properties, is that the decryption/verification of the validity of a ciphertext/MAC does not require computing the correct value of the authentication tag [BMOS17, BKP+18]. This prevents attacks in which an adversary uses a verification oracle, which it repeatedly queries with a forged message or ciphertext and an invalid tag in order to obtain leakages about the correct tag.

Leakage-resistant modes can also be designed in such a way that they are amenable to so-called leveled implementations, in which different implementations of the mode components (e.g., block ciphers) are used. On the one hand, a leveled implementation relies on the limited use of highly protected components. In practice, these components are at least expected to be secure against DPA, thanks to protections like masking. Theoretically, they are usually modeled as leak-free for simplicity. On the other hand, it will be tolerated that the rest of the components, which would perform the bulk of the computation, continuously leak a certain amount of information to the adversary every time they are used, hence requiring very limited protections, or even no specific protection at all depending on the platform. For example, shuffled implementations could be considered in case of mid-range devices [MSGR10, VMKS12], and plain unprotected (parallel) implementations could even be sufficient in case of hardware devices [BGSD10]. The expected benefits of this leakage-resistant approach are twofold.

First, they can lead to more efficient implementations for a given level of resistance to side-channel attacks. Indeed, even if these modes come with apparently heavier requirements in the black-box world (TEDT requires 4 calls of a TBC per message block), this cost is expected to be largely compensated by the more limited use of side-channel countermeasures that is required. For example, using the cycle counts of the higher-order masked implementations in ARM 32-bit devices from the recent work of Goudarzi and

[1] We use the term leakage-resistant for security definitions of confidentiality that allow all the computations (including the computation of the "challenge ciphertext") to leak, and the term leakage-resilient for security definitions that exclude it, following the terminology in [GPPS19a, Sta19].

Rivain at Eurocrypt 2017 [GR17], we show that TEDT leads to reduced cycle counts (hence more energy efficiency) compared to a uniformly protected implementation of OCB already with two shares, and that the factor of gain approximately reaches $\frac{\ell+2}{2}$ for messages of ℓ blocks as the number of shares in the masking schemes increases. For "reasonable" number of shares (given the high security goal of TEDT), like four to eight, the gains can reach factors larger than ten for moderate size messages (like $\ell = 100$). We expect similar energy gains to be observed in hardware (since the cost of masking is in general quadratic).

Second, the security reductions that come with the definition of leakage-resistant modes bring clear requirements on the specific blocks to implement. This considerably simplifies the task of designers and evaluation laboratories, and can then also increase the confidence that can be placed in the result of these evaluations. The reductions also clarify the effect of the failure of some components. For instance, in the case of TEDT, we show that weakly protected components that would leak their internal state in full through an SPA would break confidentiality, but would have no impact on ciphertext integrity. In the context of confidentiality, we reduce the security requirements on these weakly protected components, in encryption and decryption, to two simple security games that require evaluating the effect of at most two leakages of a single TBC, a task that is considerably simpler than evaluating what can be derived from millions of leakages of a full mode.

Besides, TEDT also comes with the following important benefits:

- TEDT avoids the so-called multi-user security degradation, i.e., its security bounds do not degrade significantly as the number of users increases. Such a feature is typically handy to avoid massively parallel attacks that could occur under a mass-surveillance adversary [BT16], but are also potentially applicable in an IoT context where an adversary tries to take control of some physical infrastructure. While it is known that increasing the length of secret keys and randomizing nonces can help avoid such degradation, we achieve this using a public key (used in addition to the usual secret key). This approach might be of independent interest.

- Using a TBC with n-bit blocks, TEDT offers asymptotically optimal multi-user (mu) integrity against $2^n/n^2$ queries up to $\approx 2^n$ users, even with nonce misuse and full side-channel leakages. Note that such a "Beyond Birthday Bound" (BBB, beyond $2^{n/2}$) security was highlighted in [GL17] and [ABB+, chapter 1.2].

- TEDT offers nonce misuse-resilience in the sense of Ashur et al. [ADL17]. Misuse-resilience guarantees that repeated nonces do not have an impact on the security of messages that are encrypted with fresh nonces, a property that is not satisfied by many standard modes of operation. We do not aim for nonce misuse-resistance [RS06], a stronger form of protection that requires that security is maintained even for ciphertexts produced with repeated nonces, provided that distinct messages are encrypted. Misuse-resistance requires two successive passes on messages for encryption, which creates additional latency and memory requirements, and is expected to be infeasible to achieve in most settings in which the encryption process leaks [GPPS19a] (as also discussed in supplementary material, Appendix A).

- Under very conservative leakage assumptions (see later), TEDT is proved CCA secure with misuse-resilience and decryption leakages, under the definitions in [GPPS19a]. While the conservative assumptions only enable a birthday-type $n/2$-bit security, the security is again retained up to $\approx 2^n$ users (i.e., no degradation).

TEDT analysis. TEDT runs a *keyless hash function* from the underlying TBC. By this, the TBC has to be as strong as an *ideal TBC*, i.e., a randomly selected TBC. Strong dedicated TBC algorithms, e.g., Skinny-128-256, and Deoxys-BC-256, could be used to instantiate TEDT. On the other hand, many common block cipher-based TBCs such as the

XEX construction are not sufficient (as justified in supplementary material, Appendix B). While as debated as the random oracle model [CGH98], this ideal model also gives a clear and rigorous way to measure the offline computation in terms of the number of adversarial queries to the ideal TBC, as argued in previous works [BT16, BHT18]. In fact, non-degrading mu security results typically rely on the ideal model [Tes15], and for some constructions (e.g., XGCM [BT16] and its underlying FX-key length extension [KR01]) it can only be proved effective in the ideal model. Still, most of the security properties of TEDT can also be proven in the standard model. This analysis then leads to weaker security bounds, which is a well-known artifact of the proof techniques on re-keying designs and rarely relates to actual weaknesses [ST16, Men17] (this constitutes another equally important reason for relying on ideal model analysis).

In terms of leakages, the ideal cipher model prompts us to make very simple assumptions. Our weakly protected components might even leak their full state as far as authentication is concerned, which follows [BKP+18], while they only need to satisfy hard-to-invert leakages for confidentiality, which follows [YSPY10] and appears theoretically minimal & practically measurable (which we believe is essential for modes in use). Our strongly protected components are modeled as leak-free, that is, hiding their key and, in the case of confidentiality, also mildly hiding their output, which follows [PSV15]. We additionally provide analyzes in the standard model in supplementary material, Appendix C: they require to make a stronger (physical) assumption of leakage simulatability [SPY13] regarding the weakly protected components. Interestingly, these analyzes lead to comparable bounds.

Organization. Preliminaries and security definitions with leakage in the multi-user setting are in Section 2. We detail and motivate our techniques step by step in Section 3, which leads to the specification of TEDT in Section 4. We give a thorough analysis of the leakage integrity (resp. confidentiality) of TEDT in Section 5 (resp. Section 6). We conclude the paper with a discussion of performances and related works attempting to model and realize secure authenticated encryption in a leakage setting in Section 7.

To better explain and to further illustrate the advantage of TEDT we provide a black-box CCA Ssecurity in supplementary material, Appendix D.

2 Preliminaries

The size or the length of a bit string $x \in \{0,1\}^*$, denoted $|x|$, is the integer a such that $x \in \{0,1\}^a$; an a-bit string has length a. Let n be a non-negative integer so that, when n is clear in the context, given a bit string $x \in \{0,1\}^*$, the padding $x\|0^*$ is the smallest bit string containing the prefix x only followed, if necessary, by 0's and whose length is a multiple of n. Given $x \in \{0,1\}^n$ and $1 \leq a \leq n$, $x[:a]$ is the a-bit prefix of x. We denote by $[\mathsf{num}]_{\mathsf{size}}$ the size-bit binary encoding of the integer num. We denote by a $(q_1, \ldots, q_\omega, t)$-bounded adversary a probabilistic algorithm that has access to ω oracles, $O_1, \ldots, O_\omega$, can make at most q_i queries to its i-th oracle O_i, and can perform computation bounded by running time t. A leaking version of an algorithm Algo is denoted LAlgo. It runs both Algo and a *leakage function* $\mathsf{L}_{\mathsf{algo}}$ which captures the information given by an implementation of Algo during its execution. LAlgo returns the outputs of both Algo and $\mathsf{L}_{\mathsf{algo}}$ which all take the same input.

2.1 Primitives

A Tweakable Block Cipher (TBC) with key space $\{0,1\}^\kappa$, tweak space $\{0,1\}^t$, and domain $\{0,1\}^n$ is a mapping $\mathsf{TE} : \{0,1\}^\kappa \times \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$ such that for any key $K \in \{0,1\}^\kappa$ and any tweak $T \in \{0,1\}^t$, $X \mapsto \mathsf{TE}(K,T,X)$ is a permutation of $\{0,1\}^n$. We call such TBCs (κ,t,n)-TBC. Similarly, we denote (κ,n)-block cipher those with

κ-bit keys and n-bit blocks. Note that we focus on (n, n, n)-TBC in this paper. A block cipher which is sampled uniformly at random from the set of all block ciphers with corresponding key space and domain is called an ideal cipher. Similarly, an ideal TBC $\mathsf{IC} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a TBC sampled uniformly at random from all (n, n, n)-TBCs. In this case, 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*. Throughout the remaining, we simply use the notation E for TBCs (instead of TE); and in our ideal TBC-based security proofs, we use the notation IC.

In this paper we focus on *nonce-based authenticated encryption schemes with associated data* (AEAD), which is defined as 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 selected from $\mathcal{K}$, a nonce from $\mathcal{N}$, blocks of associated data from $\mathcal{AD}$, and a message from $\mathcal{M}$ to a ciphertext in $\mathcal{C}$.
• $\mathsf{Dec} : \mathcal{K} \times \mathcal{N} \times \mathcal{AD} \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$ maps a key from $\mathcal{K}$, a nonce from $\mathcal{N}$, blocks associated data from $\mathcal{AD}$, and a ciphertext from $\mathcal{C}$ to a message in $\mathcal{M}$ that is the decryption of the ciphertext, or to a special symbol $\perp$ if integrity checking fails.

We consider AEAD schemes with constant stretch, for which the message size ℓ_m uniquely determines the ciphertext size $\ell_c = \ell_m + \mathsf{oh}$, where the constant oh is the overhead. $\mathcal{C}_{\ell_m}$ denotes the set of all the ciphertexts encrypting ℓ_m-size messages. Given a key $k \leftarrow \mathcal{K}$, $\mathsf{Enc}_k(N, A, M) := \mathsf{Enc}(k, N, A, M)$ and $\mathsf{Dec}_k(N, A, C) := \mathsf{Dec}(k, N, A, C)$ are deterministic functions whose implementations may be probabilistic. Nonce-based AEAD must be *correct* meaning that for any key $k \leftarrow \mathcal{K}$ and any triple $(N, A, M) \in \mathcal{N} \times \mathcal{AD} \times \mathcal{M}$, $\mathsf{Dec}_k(N, A, \mathsf{Enc}_k(N, A, M)) = M$. Since we only focus on correct nonce-based authenticated encryption with associated data in this paper, we will often simply refer to it as *authenticated encryption* in the following.

2.2 Leakage Security of AEAD in the Multi-User Setting

As our model, we extend important existing leakage integrity notions of Berti et al. [BPPS17] and confidentiality notions of Guo et al. [GPPS19a] to the multi-user setting.

In face of a leakage adversary, separate definitions for integrity and confidentiality potentially offer more gradual degradation. This relies on the feature of physically observable cryptography that *unpredictability is much easier to ensure than indistinguishability* [MR04], which naturally splits the level of confidence we might expect to achieve both notions. Here, we will focus on leaking AEAD with nonce-misuse-*resistant* integrity and nonce-misuse-*resilient* confidentiality (as mentioned earlier, nonce-misuse-*resistant* confidentiality seems unachievable, see Appendix A). To formalize the leakage depending on an implementation, AEAD is associated to both an encryption leakage function $\mathsf{L}_{\mathsf{enc}}$ and a decryption leakage function $\mathsf{L}_{\mathsf{dec}}$. Berti et al. defined a leakage integrity notion *Ciphertext Integrity with Misuse*-resistance *and (encryption & decryption) Leakage* in [BKP+18, BPPS17], which is denoted $\mathsf{CIML2}$. In some sense, the definition is obtained by enhancing the traditional $\mathsf{INT\text{-}CTXT}$ security game with encryption and decryption leakage. Here we further extend it to multi-user setting, denoted $\mathsf{muCIML2}$. Formally, given a nonce-based authenticated encryption $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$ with leakage function pair $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$, the multi-user ciphertext integrity advantage with misuse-*resistance* and leakage of an adversary $\mathcal{A}$ against AEAD with u users is

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

where the probability is taken over the u user keys $\mathbf{K} = (K_1, \ldots, K_u)$, with $K_i \leftarrow \mathcal{K}$, over $\mathcal{A}$'s random tape and the ideal TBC IC,[2] and where (for $1 \le i \le u$):
• $\mathsf{LEnc}_{\mathbf{K}}(i, N, A, M)$: outputs the cipher $\mathsf{Enc}_{K_i}(N, A, M)$ and the corresponding leakage trace $\mathsf{L}_{\mathsf{enc}}(K_i, N, A, M)$;

[2] We follow Bellare and Tackmann [BT16] and highlight IC in the definition.

$\mathsf{PrivK}^{\mathsf{muCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u}$ is the output of the following experiment, which starts (resp. ends) with *Initialization* (resp. *Finalization*), while the other types of queries can be made in any order and possibly multiple times:

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

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

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

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

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

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

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

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

To capture an achievable level of nonce robustness in confidentiality, Guo et al. formalized CCAmL2, after chosen-ciphertext security with misuse-*resilience* and leakage. In a high level, this notion is a leakage CCA security with "fresh nonces" [GPPS19a]. Such a level of nonce robustness was borrowed from Ashur et al. who formalized *nonce-misuse resilience* for integrity, CPA, and CCA security [ADL17].[3] Below we define the mu extension muCCAmL2: given an authenticated encryption $\mathsf{AEAD} = (\mathsf{Enc},\mathsf{Dec})$ with leakage function pair $\mathsf{L} = (\mathsf{L_{enc}},\mathsf{L_{dec}})$, the *multi-user chosen-ciphertext advantage with misuse-resilience and leakage* of an adversary $\mathcal{A}$ against AEAD with u users is

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

where the security game $\mathsf{PrivK}^{\mathsf{muCCAmL2},b}_{\mathcal{A},\mathsf{AEAD},\mathsf{L}}$ is defined in Figure 1.

When the context is clear, we will refer to the muCCAmL2 advantage (also in the ideal TBC model) with the next less formal notation

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{AEAD},\mathsf{L},u} \stackrel{\text{def}}{=} \Big|\Pr[\mathcal{A}^{\mathsf{LEnc}_{\mathbf{K}},\mathsf{LEnc}^0_{\mathbf{K}},\mathsf{L}_{\mathsf{decch}},\mathsf{LDec}_{\mathbf{K}},\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{LEnc}_{\mathbf{K}},\mathsf{LEnc}^1_{\mathbf{K}},\mathsf{L}_{\mathsf{decch}},\mathsf{LDec}_{\mathbf{K}},\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1]\Big|,$$

which allow us to talk about the 1st oracle, the 2nd oracle, and so on.

The CCAmL2 notion is more or less the traditional CCA notion enhanced with encryption and decryption leakage. The presence of $\mathsf{L_{decch}}$, which provides decryption leakage to challenge queries, follows the single-user version of Guo et al. [GPPS19a]: their intention was to capture the informativeness of valid decryption leakage, and it was motivated by applications such as secure bootloading [OC15], in which decrypting valid ciphertexts may leak crucial information. It was proved in [GPPS19a] that the combination

[3] We prove the black box nonce misuse-resilience CCA security of TEDT in Appendix D.

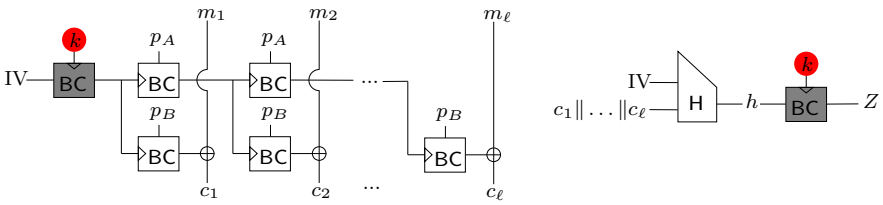

Figure 2: The AE mode EDT. E denotes a *classical block cipher*: EDT is a block cipher-based AEAD mode. The two dark blocks indicate two "leak-free" executions of E, while the others are leaky. p_A and p_B could be any two distinct constants, while in section 3.1 we consider the simplest case $p_A = 0$ and $p_B = 1$.

CIML2 + CCAmL2 is not implied by other combinations of any natural relaxation of these two notions. To provide a high leakage-resistant guarantee, an AEAD should satisfy both CIML2 and CCAmL2 (while in this paper we consider the mu extensions of both).

Note that leakage CCA is not defined in the "real-or-random" form due to the conceptual difficulty to define the leakage of the idealized (random) objects that are used in such definitions [GPPS19a, Sta19].

3 Background and Design Considerations

Starting from EDT [BPPS17], a nonce-based authenticated encryption mode *without* associated data (see Figure 2), we detail the different techniques to concretize our TEDT mode and improve the level of integrity and confidentiality.

3.1 Overview of the Starting Point EDT

While composing leakage-resistant building blocks does not necessarily lead to a leakage resistant scheme, EDT is an Encrypt-then-MAC (EtM) composition of a re-keying encryption due to [PSV15] and a Hash-then-MAC, where the hash function is keyless and the fixed-length MAC is a Strong PseudoRandom Permutation (SPRP).

While going one step further in the direction of a fully analyzed practical mode with tight security bounds maintained in a multi-user setting, TEDT inherits some of the core ideas of EDT for achieving security under leakage.[4]

Re-keying against Side-Channel Key-Recovery. Re-keying prevents DPA by deriving session keys and by consistently refreshing the key, so that the attacker cannot collect side-channel leakage on the key in use during cryptographic operations with different inputs. Concretely, for EDT, upon each encryption, a new session key k_0 is computed via an IV-based Key-Derivation Function (KDF), i.e., $k_0 \leftarrow \mathsf{KDF}_K(N)$. This KDF is heavily protected, so that the master key K is kept safe. Therefore, as long as the IV is fresh, the resulting encryption process generates a new internal state that has not been affected by the previously observed leakage. Then, the key k_0 is served to a block cipher E to produce a key stream block $y_1 \leftarrow \mathsf{E}_{k_0}(1)$. In parallel, k_0 is refreshed into a new ephemeral key $k_1 \leftarrow \mathsf{E}_{k_0}(0)$. This process is repeated until the number of key stream blocks matches the number of message blocks. In all, upon a message of ℓ blocks, ℓ keys are produced during the encryption, and E is re-keyed ℓ times. But for each of these keys, only 2 different leakage traces of E are produced (i.e., with inputs 0 and 1), and this typically makes side-channel attacks (mainly DPAs) hard to mount (if not impossible). Re-keying is usually

[4] EDT was proved CIML2 in [PSV15], but its CCAmL2 security was not formally analyzed.

deemed as costly. However, as mentioned in the introduction, this cost is expected to be largely compensated by the more limited use of side-channel countermeasures (i.e., when targeting high side-channel security – see Section 7).[5]

Minimal Message Manipulation for Side-Channel Confidentiality. Encryption schemes have to perform operations on sensitive messages. Concretely, the more operations manipulate messages, the more information leaks about them, and this leakage is very hard to completely avoid [PSV15]. To remedy this situation, in EDT each message block is involved in *only a single and easy to protect XOR operation*. This is arguably the minimum that cannot be avoided in the current state-of-the-art.

EtM & Hash-then-MAC for Side-Channel Integrity. As efficiency remains the primary concern for crypto designs it may seem surprising, at a first sight, not to simply adopt a good black-box design and add leakage protections to it. Concretely, for AE, integrated designs that perform encryption and tag generation in a "single run" seem the most efficient and attractive: examples include IAPM [Jut08], OCB [RBBK01], TAE [LRW11], OTR [Min14], COFB [CIMN17], and sponge-based proposals [BDPV12, BDH+15, CDNY18, BBLT18]. However, integrated designs typically employ the Decrypt-then-Verify style decryption, which (as observed by Barwell et al. [BMOS17]) would leak unverified plaintext and alter the side-channel security. In contrast, as independently observed in [DEM+17, BPPS17, BMOS17], the plain Encrypt-then-MAC (EtM) paradigm grants resistance to decryption leakage: since its decryption is Verify-then-Decrypt, invalid decryption queries are prevented to step into the decryption process and thus cannot produce much leakage (i.e., it only needs to secure the tag verification). As a result, EtM seems the most suitable classical solution to mitigate concrete side-channel leakages.

For MAC designs, a similar situation appears: during the input-absorbing phase, keyless crypto hash functions are preferred to the typically more efficient universal hash functions. The motivation is simply to *minimize the number of calls to keyed primitives*. Used in the plain Hash-then-MAC paradigm $T = \mathsf{TGF}_K(\mathsf{H}(U))$ ($U = IV\|C$ in EDT as well as in all our examples: this maximizes the resistance to invalid decryption queries), side-channel protections are only needed for the Tag Generation Function TGF_K, which is much easier. In fact, leakage security of Hash-then-MAC has been extensively analyzed [MOSW15, BKP+18, DEM+17]. Interestingly, if we replace the IV by a nonce, then such designs naturally achieve full nonce-robustness on the decryption side, since resisting invalid (i.e., inherently *nonce-misuse*) decryption means resisting misuse.

The elegant and provably secure idea used in EDT is to use an invertible block cipher for TGF, and to define the integrity checking as "For input (U, Z), If $\mathsf{H}(U) = \mathsf{TGF}_K^{-1}(Z)$ then accept Else reject". In this vein, decryption only leaks a useless pseudorandom value $\mathsf{TGF}_K^{-1}(Z)$ rather than the right tag of U, excluding the obvious forgery.

Shortages of EDT. EDT did not consider handling Associated Data (AD), rendering it insufficient for practical use. Note that existing approaches for enhancing an authenticated encryption scheme with efficient AD handling [Rog02] was only proved secure in the black-box setting, and thus may not suffice in the leakage setting.

Another drawback of EDT is the concrete security, which contains the birthday terms $\sigma^2/2^n + t\sigma/2^n + t^2/2^n$ *even in the black-box setting*, where t denotes the time complexity of the adversary, σ denotes the number of processed message blocks, and n is the block-size of the block cipher in use. While the term $\sigma^2/2^n$, concerning with the security of the encryption part, could be overcome by using counters (like the standard approach in the counter mode, or in GCM) instead of two fixed constants, the terms $t\sigma/2^n$ and $t^2/2^n$, resulting from the hash-then-SPRP authentication, are more problematic. Indeed the

[5] This re-keying-per-block stream cipher was first analyzed in [AB00, BY03] for black-box security, but its leakage security was only later proved in [YSPY10, SPY13]. These were actually the basis of EDT.

term $t\sigma/2^n$ is matched by a simple attack.[6] In detail, the attacker first pre-computes t hash digests $\mathsf{H}(IV, c_1) = d_1, \ldots, \mathsf{H}(IV, c_t) = d_t$ for arbitrary IV's and $c_1, \ldots, c_t$, and then encrypts σ messages $m_1^*, \ldots, m_\sigma^*$ to obtain $(IV_1, c_1, Z_1), \ldots, (IV_\sigma, c_\sigma, Z_\sigma)$. Now, as long as a hash collision $\mathsf{H}(IV_i, c_i) = d_j$ is found—the probability is $t\sigma/2^n$—the tuple (IV, c_j, Z_i) constitutes a valid forgery. As the standard NIST recommendation is to stop using a key when the probability of some leakage exceeds 2^{-32}, this means (assuming that the adversary could afford t computations) that the key of EDT has to be updated after processing $\approx 2^{96}/t$ message blocks. Nowadays it is likely to have $t \geq 2^{64}$ [SBK+17], meaning that the number of message blocks that can be securely processed by EDT is less than 2^{32}, which is less than 64GB when $n = 128$. In all, using a (tweakable) block cipher with n-bit blocks, we would like to achieve beyond $n/2$-bit security which is usually called "beyond-birthday-bound" (see, e.g., [PS15]).

In addition, unlike modes such as GCM-SIV [BHT18], the multi-user security of EDT cannot be boosted by simply increasing the key length of the block cipher. This constitutes another limitation for its practical use. In fact, it is not even clear how to use a $(2n, n)$-bloc kcipher in a re-keying encryption mode while keeping the rate 1/2.

3.2 Our New Ideas

First, it is quite obvious that the random IV in EDT can be replaced by a nonce, and this increases its robustness against randomness misuse (see [Rog04c]). Then, besides replacing the two constants in EDT by GCM-style counters to achieve BBB encryption (as mentioned), TEDT uses four main new ideas that we describe next.

Efficient Handling of AD. Our first observation is that, to securely handle the AD, we can simply insert the AD as a part of the input to the Hash-then-MAC authenticator.

Hash-then-TBC for Efficient BBB Authentication. After many failed trials, it appears that the simplest and most efficient approach to BBB secure leakage-resistant authentication is to increase the output of the hash to $2n$ bits, and then use an (n, n, n)-TBC for the tag generation function TGF to absorb this digest.

Concretely, upon tagging U, we apply a $2n$-bit hash function H, i.e., $V\|W = \mathsf{H}(U)$, where V and W are the two n-bit halves. Then, the tag is $Z = \mathsf{TGF}_K(W, V) = (\mathsf{E}_K^W)(V)$. To resist verification/ decryption leakage, we utilize the inverse, i.e., upon (U, Z), "If $(\mathsf{E}_K^W)^{-1}(Z) = V$ for $V\|W = \mathsf{H}(U)$ then accept Else reject".

If one insists on using classical (n, n)-block ciphers, then it seems two calls are necessary for the BBB (leakage-resistant) authentication. Both of them as well as the additional internal wires have to be well protected, which could be problematic. We believe a single protected TBC-call would be more efficient/secure than two protected classical block cipher calls. Moreover, the (n, n, n)-TBC also cinches the mu security strengthening trick.

Making the Hash Function Concrete. To make the mode more concrete (and practical), we instantiate the hash function H from the same TBC E, as in Figure 3. For this purpose, we (have to) view E as an ideal TBC IC. An ideal (n, n, n)-TBC is essentially the same as an ideal $(2n, n)$-block cipher—both are 2^{2n} independent random permutations. With the consideration that Hirose's double-block-length (DBL) construction is XOR-only and does not require re-keying within each invocation, we select it to instantiate a $3n$-to-$2n$-bit compression function Hir[E] [Hir06], and then use Hir[E] in the strengthened Merkle-Damgård to build H[E]. The formal description is given later in Fig 5. In this vein, the obtained mode is purely TBC-based which may reduce implementation costs. Yet, the DBL hash does *not* trivially offer security: see the end of this section.

Public Randomness to Remedy mu Security Degradation. Roughly, mu security

[6] The term $t^2/2^n$ is more severe as it indicates low-data attacks with $2^{n/2}$ computations. But this is not matched by any attack, and might be an artifact of the standard model proof approach.

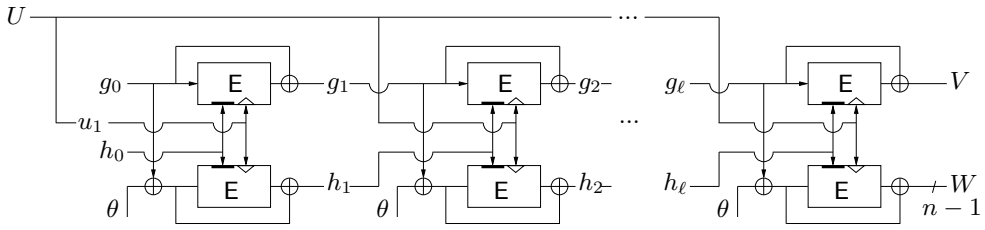

Figure 3: The hash function H upon a message $U = u_1 \| \dots \| u_{\ell+1}$ of $\ell + 1$ blocks, with initial vector $g_0 \| h_0$. For each square, the input to the triangle denotes the key input. θ is a domain separation constant used by the Hir[E] DBL compression function.

degradation stems from (i) collision between user keys, and (ii) multiple user keys that increase the effectiveness of offline computations.

We illustrate both with examples. Consider u users. For issue (i), the probability to have two users i and j such that $K_i = K_j$ is $\frac{u^2}{2^{n+1}}$. With this, for any (N, A, M), $\mathsf{Enc}_{K_i}(N, A, M)$ is a valid forgery for user j. For issue (ii), the adversary could compute $\mathsf{Enc}_{K^*}(N, A, M) \to C$ for a guess K^*, and as long as K^* equals K_i for some user i the adversary could have a chance to detect and notice $K_i = K^*$: the probability of such a collision is $\frac{1}{2^n}$ in the su setting, yet balloons to $\frac{u}{2^n}$ in the mu setting.

Clearly, increasing the secret key length solves both (just boosting the denominator). We show that they can be overcome by *properly using public key bits*. Concretely, after we replace all E-calls in the EDT encryption by TBC-calls, we could simply use the "public key" PK for the tweak input. This simple trick does not work for the authentication since the tweak input of TGF has been "occupied" by a half V of the hash digest. Yet, once we append PK to the hash input U, we achieve some separation between users.

Roughly speaking, now two encryption instances collide only if a collision occurs between *both their secret and public keys*. Thus, the probability of user key collision is decreased and issue (i) is solved. For issue (ii), while PK cannot immediately enlarge the denominator, it makes each guess $K^* = SK^* \| PK^*$ less effective: the guess K^* hits a key $K_i = SK_i \| PK_i$ only if $PK^* = PK_i$. Therefore, if the maximal multiplicity of the PK value is small $\mu \ll u$ (which can be achieved by ensuring distinct PK values, or picking PK at random), for issue (ii) the additional public randomness reduces the probability to $\frac{\mu}{2^n} \ll \frac{u}{2^n}$.[7] While more random bits are required, the *secret* key remains of n bits. In this respect, we note that it is easier to generate "*public* keys" than *secret* ones: for the latter a key agreement protocol is needed, while for the former one could uniformly pick and send it to the other user *in (authenticated) plaintext form*.

Summary. Our technical contribution is focused on solving the two following problems:

(1) Proving that public randomness does avoid the aforementioned multi-user degradation (both for encryption and authentication).

(2) Proving that the hash-then-TBC authentication does achieve BBB mu security against verification leakage when the hash function is a strengthened Merkle-Damgård iteration of Hirose DBL compression function. We remark this is highly non-trivial, since we aim at $2^n/n^2$ security (so far beyond birthday).[8]

A simpler mode AEDT. If we only apply the efficient AD handling technique to EDT, then we'll obtain an AEAD mode for block ciphers that achieve the same birthday-bound

[7] This clarification is simplified. We have to rely on some non-standard collision properties of H.

[8] If H was (indifferentiable from) a random oracle with good $2^n/n^2$ bounds, then the result would be much easier to obtain. Yet, neither "plain" Merkle-Damgård [CDMP05] nor Hir[IC] [Men13] is indifferentiable.

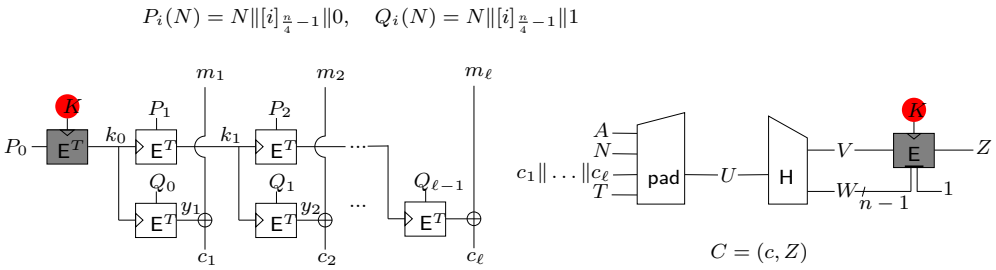

Figure 4: The TEDT AEAD. The constants P_i and Q_i are derived from the nonce N, i.e., $P_i = N\|[i]_{\frac{n}{4}-1}\|0$, and $Q_i = N\|[i]_{\frac{n}{4}-1}\|1$, where $i \in [0, \ldots, 2^{\frac{n}{4}-1} - 1]$. The two dark blocks are KDF_K and TGF_K, and for side-channel security they need heavy protection to be "leak-free". The other TBC-calls are leaking. For each square, the input to the triangle denotes the key input. The tweak $T = PK\|0$ is the public-key PK padded with 0. H is instantiated as in Figure 3.

security as EDT. We may name it AEDT, standing for EDT with AD handling. In this paper, we concentrate on the more secure mode TEDT.

4 Specification for TEDT

Parameters. Built upon an (n, n, n)-TBC, the key of TEDT is written as $K\|PK$, with $|K| = n$ and $|PK| = n - 1$. While K is secret, PK can be *public*.

The secret key K is picked *uniformly* (from now on we eschew the notation SK used in section 3.2). On the other hand, PK could be either *uniformly picked* or *ensured distinct* for each user. See section 3.2: this requirement avoids mu security degradation. TEDT accepts $\frac{3n}{4}$-bit nonce and results in n-bit stretch. From a nonce N, it generates two sequences of distinct constants for encryption / decryption, i.e., $P_i(N) = N\|[i]_{\frac{n}{4}-1}\|0$, and $Q_i(N) = N\|[i]_{\frac{n}{4}-1}\|1$, where the integer $i \in [0, \ldots, 2^{\frac{n}{4}-1} - 1]$.[9] The index i corresponds to the message block index, and the first sequence $P_i(N)$ is used for re-keying, while $Q_i(N)$ is used to generate key stream blocks: see Figure 4. By this, a single message cannot exceed $n \cdot 2^{\frac{n}{4}-1}$ bits. In Table 1, we list the TEDT parameters for the general n-bit case, and for the primary use case $n = 128$.

Table 1: TEDT parameters.

	General n	$n = 128$
Key size	$2n - 1$:	255 bits:
	n secret, $n - 1$ public	128 private, 127 public
Nonce size	$\frac{3n}{4}$ bits	96 bits
Maximal message	$2^{n/4-1}$ blocks	2^{35} bytes
Maximal AD	$2^{n/2} - 1$ bits	$2^{61} - 1$ bytes
Stretch	n bits	16 bytes

Hash and Padding. Our TBC-based hash function H[E] is a Merkle-Damgård iteration of the Hirose DBL compression function Hir[E]. The domain separation constant in Hir[E] is $\theta = 1$, and the initial vector for the Merkle-Damgård is all-zero $[0]_{2n}$. The fact that $P_i(N) = Q_i(N) \oplus \theta$ will help reducing some constant factors in the security proof. Formally, H[E] is described in Figure 5. For our purpose, the hash input is a 4-tuple $(A, N, \mathbf{c}, T)$, with A first to handle static and some type of incremental associated data. A suffix-free

[9] Any function ensures $P_i(N) \neq P_{i'}(N')$, $Q_i(N) \neq Q_{i'}(N')$ for any $(i, N) \neq (i', N')$, and $P_i(N) \neq Q_i(N)$ for any $(i, N), (i', N')$, would suffice. But our choice, motivated by GCM, is an obvious simple approach.

padding $\mathsf{pad}(A, N, \mathbf{c}, T)$ is needed for the Merkle-Damgård iteration. This padding must prevent the attacks trying to confuse the A and $\mathbf{c}$ fields of variable length. For this, we define $\mathsf{pad}(A, N, \mathbf{c}, T) := A\|N\|\mathbf{c}\|T\|0^*\|[|A|]_{n/2}\|[|\mathbf{c}|]_{n/2}$. This padding upper-bounds the length of A by $2^{n/2} - 1$, as shown in Table 1. We remark that this padding is part of the AEAD scheme and not of the hash function H.

The Encryption is an EtM composition of the counter-based variant of Pereira et al.'s re-keying encryption [PSV15] and the Hash-then-TBC tag generation, as discussed in section 3.2. Both KDF (in the encryption) and TGF (in the authentication) are instantiated with a single TBC-call. To achieve a separation (so that we can use the same secret key K for both), we reserve 1 bit in the tweak input: concretely, we derive $T = PK\|0$ by padding 0 and use it for the tweak of KDF_K, and chop the output of H by 1 bit to obtain an $n-1$ bit half W and use $W\|1$ for the tweak of TGF_K. The whole process is described by the algorithm $\mathsf{TEDT}[\mathsf{E}].\mathsf{Enc}_{K,PK}(N, A, M)$ in Figure 5. We separate KDF and TGF from the other TBC-calls (though algorithmically equal) for conceptual convenience: both highlighting the heavily protected calls and simplifying language. Since $T = PK\|0$, in the remainder of this paper, we also call T a "public-key".

The Decryption is of Verify-then-Decrypt type: it first invokes the Hash-then-TBC verification to check the integrity, and decrypts only if the input is decided as authentic. To ensure this verification does not leak the right tag, the inverse of TGF is invoked. The whole process is described by the algorithm $\mathsf{TEDT}[\mathsf{E}].\mathsf{Dec}_{K,PK}(N, A, C)$ in Figure 5.[10]

Remark. The public-key T in the KDF call is also crucial for avoiding the mu security degradation term $2^n/u$, so it cannot be replaced by a constant. Furthermore, unlike most TBC modes [PS15, IMPS17], we do *not* use domain separation to make TBC-calls during encryption and authentication independent. This avoids many issues (such as unusual message block size) discussed in [IMPS17, section 6.1].

We are not aware of any comprehensive analysis of how symmetric cryptosystems shall manage "public keys". In particular, it is not clear if PK shall be updated with the secret part K during key updating, which we recommend in order to be conservative.

5 Leakage Integrity of TEDT

We now investigate the $\mathsf{muCIML2}$ security of TEDT in the "unbounded leakage" setting [BKP+18, BPPS17], which means all the intermediate values completely leak, except the master key K of KDF_K and TGF_K (which remains secret). Formally, we define the leakage function $\mathsf{L}^* = (\mathsf{L}^*_{\mathsf{enc}}, \mathsf{L}^*_{\mathsf{dec}})$, where:

- $\mathsf{L}^*_{\mathsf{enc}}$ consists of the following information appearing during the encryption:
  - $\{K, T, X, Y\}$ for each leaking TBC-call to $\mathsf{E}^T_K(X) \to Y$ or $(\mathsf{E}^T_K)^{-1}(Y) \to X$,
  - $\{T, I, S\}$ for each leak-free KDF-call to $\mathsf{KDF}(K, T, I) \to S$,
  - $\{W, V, Z\}$ for each leak-free TGF-call to $\mathsf{TGF}(K, W, V) \to Z$, or inverse leak-free TGF-call $\mathsf{TGF}^{-1}(K, W, Z) \to V$,
  - $\{a, b\}$ for each XOR action $a \oplus b$.
- $\mathsf{L}^*_{\mathsf{dec}}$ consists of the above that are generated during the decryption.

Note that the above means KDF- and TGF-calls do not leak any information about K: the same assumption will be used in Section 6. Of course it is too strong to be fulfilled in reality. Yet, it nicely abstracts the details in the TBC and allows to focus on the *mode-level leakages*. So we use it as a first step and leave relaxations for further research.

[10] There are two decryption times for the ciphertexts of any given length, corresponding to invalid & valid ciphertexts (the difference of which is not critical since available through black box queries).

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

1. $\ell \leftarrow \lceil |M|/n \rceil$, $r \leftarrow |M| - n(\ell - 1)$
2. parse M as $m_1 \| \ldots \| m_\ell$, with $|m_1| = \ldots = |m_{\ell-1}| = n$ and $1 \le |m_\ell| = r < n$
3. $T \leftarrow PK \| 0$, $m_\ell \leftarrow m_\ell \| 0^*$
4. **if** $\ell > 0$ **then**
5. $\quad k_0 \leftarrow \mathsf{KDF}(K, T, P_0(N))$
   // $P_i(N) = N \| [i]_{\frac{n}{4}-1} \| 0$
6. **for** $i = 1$ **to** ℓ **do**
7. $\quad y_i \leftarrow \mathsf{E}^T_{k_{i-1}}(Q_{i-1}(N))$
   // $Q_i(N) = N \| [i]_{\frac{n}{4}-1} \| 1$
8. $\quad c_i \leftarrow y_i \oplus m_i$
9. $\quad k_i \leftarrow \mathsf{E}^T_{k_{i-1}}(P_i(N))$
   // line 9. omitted for $i{=}\ell$
10. $\mathbf{c} \leftarrow c_1 \| \ldots \| c_\ell[:r]$
11. $U \leftarrow \mathsf{pad}(A, N, \mathbf{c}, T)$
12. $V \| W \leftarrow \mathsf{H}[\mathsf{E}](U)$
13. $Z \leftarrow \mathsf{TGF}(K, W, V)$, $C \leftarrow \mathbf{c} \| Z$
14. **return** C

algorithm $\mathsf{H}[\mathsf{E}](U)$

1. parse U as $u_1 \| \ldots \| u_\ell$ with $|u_1| {=} \ldots {=} |u_\ell| = n$
2. $g_0 \| h_0 \leftarrow [0]_{2n}$ // IV
3. **for** $i = 1$ **to** ℓ **do**
4. $\quad g_i \| h_i \leftarrow \mathsf{Hir}[\mathsf{E}](u_i \| g_{i-1} \| h_{i-1})$
5. $V \leftarrow g_\ell$, $W \leftarrow h_\ell[:n-1]$
6. **return** $V \| W$

algorithm $\mathsf{pad}(A, N, \mathbf{c}, T)$

1. $u \leftarrow A \| N \| \mathbf{c} \| T$
2. $\ell \leftarrow |u|$
3. $\Delta \leftarrow \lceil \frac{\ell}{n} \rceil \cdot n - \ell$
4. **return** $u \| [0]_\Delta \| [|A|]_{n/2} \| [|\mathbf{c}|]_{n/2}$

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

1. $\ell \leftarrow \lceil |C|/n \rceil - 1$, $r \leftarrow |C| - n\ell$
2. parse C as $\mathbf{c} \| Z$, with $\mathbf{c} = c_1 \| \ldots \| c_\ell$, $|c_1| = \ldots = |c_{\ell-1}| = |Z| = n$, and $1 \le |c_\ell| = r < n$
3. $T \leftarrow PK \| 0$, $c_\ell \leftarrow c_\ell \| 0^*$
4. $U \leftarrow \mathsf{pad}(A, N, \mathbf{c}, T)$
5. $V \| W \leftarrow \mathsf{H}[\mathsf{E}](U)$
6. $V^* \leftarrow \mathsf{TGF}^{-1}(K, W, Z)$
7. **if** $V \neq V^*$ **then return** $\perp$
8. **if** $\ell > 0$ **then**
9. $\quad k_0 \leftarrow \mathsf{KDF}(K, T, P_0(N))$
10. **for** $i = 1$ **to** ℓ **do**
11. $\quad y_i \leftarrow \mathsf{E}^T_{k_{i-1}}(Q_{i-1}(N))$
12. $\quad m_i \leftarrow y_i \oplus c_i$
13. $\quad k_i \leftarrow \mathsf{E}^T_{k_{i-1}}(P_i(N))$
    // line 13. omitted for $i{=}\ell$
14. $M \leftarrow m_1 \| \ldots \| m_\ell[:r]$
15. **return** M

algorithm $\mathsf{KDF}(K, T, I)$

1. **return** $\mathsf{E}^T_K(I)$

algorithm $\mathsf{TGF}(K, W, V)$

1. **return** $\mathsf{E}^{W\|1}_K(V)$

algorithm $\mathsf{TGF}^{-1}(K, W, Z)$

1. **return** $(\mathsf{E}^{W\|1}_K)^{-1}(Z)$

algorithm $\mathsf{Hir}[\mathsf{E}](X)$

1. parse X as $u \| g \| h$, $|u| = |g| = |h| = n$
2. $g' \leftarrow \mathsf{E}^h_u(g) \oplus g$
3. $h' \leftarrow \mathsf{E}^h_u(g \oplus 1) \oplus g \oplus [1]_n$ // $\theta = 1$
4. **return** $g' \| h'$

Figure 5: Definition of the TEDT mode, using a TBC E.

Since we will analyze TEDT in the ideal TBC model, we can prove information theoretic security that only requires bounded amount of adversarial queries. This means the adversarial power is characterized by $\mathbf{q} = (q_e, q_d, q_{\mathsf{IC}})$, where q_e, q_d, and q_{IC} denotes the number of adversarial queries to the leaking encryption oracle, the leaking decryption oracle, and the ideal TBC oracle, respectively. To simplify the notations, using the definition given in Section 2.2 we define

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT},\mathsf{L}^*}(u, \mathbf{q}, \sigma) \stackrel{\text{def}}{=} \max \left\{ \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{TEDT},\mathsf{L}^*,u} \right\},$$

where the maximum is taken over all $\mathbf{q}$-bounded adversaries against u users that have at most σ blocks in all their queried plaintexts and ciphertexts including associated data. Then our main claim is as follows:

Theorem 1. *Assume that the u public-keys $T_1, \ldots, T_u$ are uniformly distributed, $n \geq 6$, $2\sigma + 3(q_e + q_d) + q_{\mathsf{IC}} \leq 2^n/8$, and leakage L^* is "unbounded" as above. Then*

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT},\mathsf{L}^*}(u, \mathbf{q}, \sigma) \leq \frac{(2n^2+17)(4\sigma + 6(q_e+q_d) + 2q_{\mathsf{IC}})}{2^n} + \frac{u^2}{2^{2n}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n \quad (1)$$

in the ideal TBC model.

The bound means that, as long as we carefully protect the key of KDF_K and TGF_K, the muCIML2 bound is asymptotically optimal $O\left(\frac{u^2}{2^{2n}} + \frac{n^2\sigma + n^2 q_{\mathsf{IC}}}{2^n}\right)$. Concretely, when $n = 128$, the integrity security is up to $u \approx 2^{126}$ users, $\sigma \approx 2^{114}$ blocks, and $q_{\mathsf{IC}} \approx 2^{114}$ offline computation. If the u public-keys are ensured to be *distinct*, then we could drop the terms $\frac{u^2}{2^{2n}}$ and $\frac{1}{2n!} \cdot (4u/2^n)^n$, but the bound does not substantially improve. In the black-box model, similar asymptotically optimal bounds can be proved: see Appendix D.[11]

The remaining of this section is filled by the proof of Theorem 1. We first serve an outline of the proof and some hints on how we obtain the involved terms in subsection 5.1. The two core steps of the proof will be in subsection 5.3 and 5.4, but before them we prove some crucial properties for the Hirose compression function and the pad scheme in subsection 5.2. Finally, as mentioned, the TBC-call in line 6 of Dec (Fig. 5) has to be backward to resist decryption leakages. To give more insights, at the end in subsection 5.5 we discuss where our proof approach fails for the design without the backward TBC-call.

5.1 Outline of the Proof and Hints on the Terms

We proceed in two main steps. First, below in section 5.3, we idealize the scheme $\mathsf{TEDT}[\mathsf{IC}]_{\mathbf{K},\mathbf{T}}$ via replacing calls to $\mathsf{KDF}_{K_1}, \mathsf{TGF}_{K_1}, \ldots, \mathsf{KDF}_{K_u}, \mathsf{TGF}_{K_u}$ by several tweakable random permutations $\widetilde{\pi}_1, \ldots, \widetilde{\pi}_u$, such that $\widetilde{\pi}_i$ *and* $\widetilde{\pi}_j$ *are independent if and only if* $K_i \neq K_j$. These permutations are secret, i.e., they cannot be queried by $\mathcal{A}$.

We then show the real and idealized schemes are indistinguishable. The actual goal of this step is to argue that during the interaction, the secret keys $\mathbf{K}$ cannot be recovered from information gained by the adversary. The indistinguishability is proved using the H-coeffcient technique [CS14]. For this step, note that all the internal values of the TBC-calls, except K, are given to the adversary $\mathcal{A}$. This additional information increases the offline computation power of $\mathcal{A}$. Therefore, conceptually, the situation is that the adversary $\mathcal{A}$ makes $O(\sigma + q_{\mathsf{IC}})$ queries to IC and tries to recover the secret user keys.

Typically, the naive proof idea would be: since each offline IC query will "hit" one of the u user secret keys with probability $u/2^n$, a union bound over all the offline queries yields $O\left(\frac{u(\sigma+q_{\mathsf{IC}})}{2^n}\right)$. But our goal is to avoid the factor u, i.e., the multi-user security degradation. To have a better bound, we rely on the fact that these $O(\sigma + q_{\mathsf{IC}})$ offline queries are "separated" by the tweak inputs. In detail,

- The KDF-calls make IC-calls of the form $\mathsf{IC}_K^{PK\|0}(I)$, and this is related to the distribution of the public keys. When the public keys are uniformly picked, the number of users sharing a same public key value cannot exceed n, except with a sufficiently small probability of $O(u/2^n)$. The TBC ensures independence between calls with different tweaks, i.e., for a user with public key $PK' \neq pk$ and secret key K', an offline IC query of the form $\mathsf{IC}_k^{pk\|0}(x)$ is independent from its KDF-calls $\mathsf{IC}_{K'}^{PK'\|0}(I)$, even if $k = K'$. By this, an offline IC query of the form $\mathsf{IC}_k^{pk\|0}(x)$ can

[11] In an ongoing lightweight cryptography standardization effort, the NIST requires new proposals to securely process more than 2^{50} bytes of data (before key updating) and resist 2^{112} computations *in the single-user setting* [NIS19]. What TEDT achieves is clearly far better.

have its key k "hit" at most n users (i.e., the users using pk as the public key). This decreases the term to $O\Big(\frac{n(\sigma+q_{\mathsf{IC}})}{2^n}\Big)$ and avoids the factor u.[12]

- The TGF-calls make IC-calls of the form $\mathsf{IC}_K^{W\|1}(V)$ or $(\mathsf{IC}_K^{W\|1})^{-1}(Z)$, in which the tweak W is always (half of) the output of the hash function. Given distinct inputs, the number of semi-collisions among the outputs of H cannot exceed n, except with a sufficiently small probability of $O(\frac{\sigma+q_{\mathsf{IC}}}{2^n})$. This means for each W, TEDT makes at most n calls of the form $\mathsf{IC}_K^{W\|1}(V)$ or $(\mathsf{IC}_K^{W\|1})^{-1}(Z)$ for each user. But note that, due to the collision among the public keys, TEDT may make the same hash evaluation for different users: as argued, this situation appears at most n times. The two effects indicate that each offline IC query of the form $\mathsf{IC}_k^{W\|1}(x)$ simultaneously "targets" at most n^2 TGF calls, and this creates the term $O\Big(\frac{n^2(\sigma+q_{\mathsf{IC}})}{2^n}\Big)$. Although the factor n^2 is worse, the factor u is still avoided.

Next, in subsection 5.4, we prove unforgeability for the idealized scheme to complete the $\mathsf{muCIML2}$ proof. To this end, we show that various query records gained by the adversary during the interaction will not form certain "chains", except with a sufficiently small probability. As will be seen in the detailed analysis, the adversary has two ways to create such "chains": either trying to create a collision between two different hash evaluations, or trying to create a collision between a hash digest and an inverse TGF-call. Since we are using a DBL hash, the probability of hash collision is $O\Big(\frac{(\sigma+q_{\mathsf{IC}})^2}{2^{2n}}\Big)$; since it is hard to create n-collisions on the W values, it can be shown that the probability of collisions between a hash digest and an inverse TGF-call is $O\Big(\frac{nq_d}{2^n}\Big)$.

As an additional remark and hint, note that if we compute the tag from the digest $V\|W = \mathsf{H}[\mathsf{E}](\mathsf{pad}(A,N,\mathbf{c}))$ without using the public key T, then a user secret key collision $K_i = K_j$ enables forgery since $\mathsf{TGF}(K_i,W,V) = \mathsf{TGF}(K_j,W,V)$. But since the public keys are injected, for $V\|W = \mathsf{H}[\mathsf{E}](\mathsf{pad}(A,N,\mathbf{c},T_i))$ and $V'\|W' = \mathsf{H}[\mathsf{E}](\mathsf{pad}(A,N,\mathbf{c}),T_j)$ we have $\mathsf{TGF}(K_i,W,V) = \mathsf{TGF}(K_j,W',V')$ only if both $K_i = K_j$ and $T_i = T_j$, the probability of which is reduced to the small enough term $u^2/2^{2n}$.

5.2 Properties of Hirose Compression Function and pad

Denote by $\mathsf{Hir}[\mathsf{IC}]$ the Hirose compression function built upon the ideal TBC IC. Note that any adversary $\mathcal{A}$ against $\mathsf{Hir}[\mathsf{IC}]$ can be normalized to an adversary $\mathcal{A}'$ that only makes pairs of *Hirose matching queries*: $\mathcal{A}'$ runs $\mathcal{A}$, and

- each time $\mathcal{A}$ makes a forward query $\mathsf{IC}_K(T,X)$, $\mathcal{A}'$ makes a query $\mathsf{IC}_K(T,X\oplus\theta)\to Y'$ right after relaying $\mathsf{IC}_K(T,X)\to Y$, and
- each time $\mathcal{A}$ makes a backward query $\mathsf{IC}_K^{-1}(T,Y)$, $\mathcal{A}'$ makes a query $\mathsf{IC}_K(T,X\oplus\theta)\to Y'$ right after relaying $\mathsf{IC}_K^{-1}(T,Y)\to X$.

Therefore, we could concentrate on adversaries that only make such pairs of matching queries. In this vein, the function $\mathsf{Hir}[\mathsf{IC}]$ has the following collision-related properties.

Lemma 1. *For any $\mathcal{A}$ making q pairs of matching queries to IC with $1\le q\le 2^n/4$, it holds*

$$\Pr\left[(X,X')\leftarrow\mathcal{A}^{\mathsf{IC}} : \mathit{chop}(\mathsf{Hir}[\mathsf{IC}](X)) = \mathit{chop}(\mathsf{Hir}[\mathsf{IC}](X'))\right]\le\frac{6q}{2^n}.$$

[12] When the public keys are ensured distinct, then following the same idea, the factor n is decreased to 1.

Proof. The collision resistance of $\mathsf{Hir}[\mathsf{IC}]$ does not necessarily imply the collision resistance of $\mathsf{chop}\circ\mathsf{Hir}[\mathsf{IC}]$. To show the latter, consider any two pairs of matching queries $((v,h,g,y),(v\oplus\theta,h,g\oplus\theta,z)$ and $((v',h',g',y'),(v'\oplus\theta,h',g'\oplus\theta,z')$. We now bound the probability of collision due to these queries. We distinguish two cases:

Case 1: $(v,h,g,y)=(v'\oplus\theta,h',g'\oplus\theta,z')$**.** This means the two pairs are actually the same two IC queries in different orders. Then the collision indicates $\mathsf{chop}(g\oplus y)=\mathsf{chop}(g\oplus\theta\oplus z)$. Without loss of generality, assume (v,h,g,y) is made after $(v\oplus\theta,h,g\oplus\theta,z)$. Then regardless of whether the query is forward or backward, either g or y is uniform in a set of size at least 2^n-2q. Therefore,

$$\Pr[\mathsf{chop}(g\oplus y)=\mathsf{chop}(g\oplus\theta\oplus z)]=\Pr[g\oplus y=b\mid \mathsf{chop}(g\oplus\theta\oplus z)\text{ for }b=0\text{ or }1]\le\frac{2}{2^n-2q},$$

and with q matching pairs, we have

$$\Pr[\text{Case 1}]\le\frac{2q}{2^n-2q}.$$

Case 2: $(v,h,g,y)\neq(v'\oplus\theta,h',g'\oplus\theta,z')$**.** Then it can be seen the involved queries are four different ones, and a collision indicates

$$g\oplus y=g'\oplus y'\wedge\mathsf{chop}(g\oplus\theta\oplus z)=\mathsf{chop}(g'\oplus\theta\oplus z') \tag{2}$$

or

$$g\oplus y=g'\oplus\theta\oplus z'\wedge\mathsf{chop}(g\oplus\theta\oplus z)=\mathsf{chop}(g'\oplus y'). \tag{3}$$

Similarly to Case 1, in any subcase, we have

$$\Pr[g\oplus y=g'\oplus y']\le\frac{1}{2^n-2q}\text{ and }\Pr[\mathsf{chop}(g\oplus\theta\oplus z)=\mathsf{chop}(g'\oplus\theta\oplus z')]\le\frac{2}{2^n-2q},$$

so that $\Pr[\text{Eq. (2)}]\le\frac{2}{(2^n-2q)^2}$. A similar reasoning shows $\Pr[\text{Eq. (3)}]\le\frac{2}{(2^n-2q)^2}$. Therefore, when $q\le 2^n/4$ we have

$$\Pr[\text{Case 2}]\le\binom{q}{2}\cdot\frac{4}{(2^n-2q)^2}\le\left(\frac{4q}{2^n}\right)^2\le\frac{4q}{2^n}.$$

In summary,

$$\Pr[\text{collision after chopping}]=\Pr[\text{Case 1}]+\Pr[\text{Case 2}]\le\frac{6q}{2^n}$$

as claimed. □

We also need the multi-collision resistance of the chopped Davies-Meyer.

Lemma 2. *Consider the Davies-Meyer function* $\mathsf{DM}[\mathsf{IC}](K\|T,h)=\mathsf{IC}_K^T(h)\oplus h$ *built upon the ideal TBC* IC*. Then, for any adversary* $\mathcal{A}$ *making at most* $q\le 2^n/2$ *queries to* IC *and any integer* λ*, it holds*

$$\Pr\left[(X_1,\ldots,X_\lambda)\leftarrow\mathcal{A}^{\mathsf{IC}}:\mathit{chop}(\mathsf{DM}[\mathsf{IC}](X_1))=\ldots=\mathit{chop}(\mathsf{DM}[\mathsf{IC}](X_\lambda))\right]\le\frac{(4q)^\lambda}{\lambda!2^{(\lambda-1)n}}.$$

Proof. Consider any λ IC queries $(K_1\|T_1,h_1,y_1),\ldots,(K_\lambda\|T_\lambda,h_\lambda,y_\lambda)$ listed according the order they were made, and let $z_i=h_i\oplus y_i$ for each i. Then, since $q\le 2^n/2$, for any $i\ge 2$ we have $\Pr[\mathsf{chop}(z_i)=\mathsf{chop}(z_1)]\le\frac{2}{2^n-q}\le\frac{4}{2^n}$. Therefore,

$$\Pr[\lambda\text{ collisions}]\le\prod_{i=2}^{\lambda}\Pr[\mathsf{chop}(z_i)=\mathsf{chop}(z_1)]\le\binom{q}{\lambda}\cdot\left(\frac{4}{2^n}\right)^{\lambda-1}\le\frac{(4q)^\lambda}{\lambda!2^{(\lambda-1)n}}$$

as claimed. □

Initialization(u):

1. Choose u random keys $\mathbf{K}^* = (K_1^*, \ldots, K_u^*) \xleftarrow{\$} (\mathcal{K})^u$, and tweakable random permutations $\widetilde{\pi}_1, \ldots, \widetilde{\pi}_u$ from $\{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$, under the constraint that $\widetilde{\pi}_i = \widetilde{\pi}_j$ if and only if $K_i^* = K_j^*$.

Interface $\mathsf{TRPFamily}(i, T, X)$

1. Return $\widetilde{\pi}_i\ (T, X)$

Interface $\mathsf{TRPFamily}^{-1}(i, T, Y)$

1. Return $\widetilde{\pi}_i^{-1}\ (T, Y)$

Finalization(u):

1. Choose u random keys $\mathbf{K} = (K_1, \ldots, K_u) \xleftarrow{\$} (\mathcal{K})^u$ under the constraint that $K_i = K_j$ if and only if $K_i^* = K_j^*$.

Figure 6: Definition of the primitive $\mathsf{TRPFamily}$.

Last, the pad scheme is effective in distinguishing between inputs with different semantics.

Lemma 3. *It holds* $\mathsf{pad}(A, N, \mathbf{c}, T) \neq \mathsf{pad}(A', N', \mathbf{c}', T')$ *for any two distinct tuples* $(A, N, \mathbf{c}, T)$ *and* $(A', N', \mathbf{c}', T')$.

Proof. Note that $\mathsf{pad}(A, N, \mathbf{c}, T) \neq \mathsf{pad}(A', N', \mathbf{c}', T')$ unless $|A| = |A'|$ & $|\mathbf{c}| = |\mathbf{c}'|$, as otherwise the values appended in the length field are different. But if $|A| = |A'|$ and $|\mathbf{c}| = |\mathbf{c}'|$ (note $|N| = |N'|$ and $|T| = |T'|$ always hold), then $(A, N, \mathbf{c}, T) \neq (A', N', \mathbf{c}', T')$ indicates at least one of them has different values, and we thus have $A\|N\|\mathbf{c}\|T \neq A'\|N'\|\mathbf{c}'\|T'$ and the resulting values are different after padded. □

5.3 Idealizing TEDT

To formally define the idealized scheme, we define an ideal primitive $\mathsf{TRPFamily}$ in Fig. 6. Briefly, $\mathsf{TRPFamily}$ captures the behavior of an ideal TBC instantiated with u uniformly picked keys $(K_1^*, \ldots, K_u^*)$, i.e., u tweakable random permutations $\widetilde{\pi}_1, \ldots, \widetilde{\pi}_u$. Yet, the u keys $(K_1^*, \ldots, K_u^*)$ are actually not used by $\mathsf{TRPFamily}$: they only define a "pattern" for the permutations $\widetilde{\pi}_1, \ldots, \widetilde{\pi}_u$.[13] The definition and use of $\mathsf{TRPFamily}$ are similar to Gueron and Lindell [GL17]. Based on this, the ideal scheme is obtained by replacing every call to $\mathsf{KDF}(K_i, t, X)/\mathsf{TGF}(K_i, t, X)$ and $\mathsf{KDF}^{-1}(K_i, t, Y)/\mathsf{TGF}^{-1}(K_i, t, Y)$ by $\mathsf{TRPFamily}(i, t, X)$ and $\mathsf{TRPFamily}^{-1}(i, t, Y)$ for $i = 1, \ldots, u$. Denote the obtained idealized scheme by $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$.

5.3.1 Preparation for the Indistinguishability Proof: Notations

Denote by $\mathsf{LEnc}_{\mathbf{K},\mathbf{T}}$ and $\mathsf{LDec}_{\mathbf{K},\mathbf{T}}$ the leaking encryption and decryption oracles of the real scheme $\mathsf{TEDT}[\mathsf{IC}]_{\mathbf{K},\mathbf{T}}$, and by $\mathsf{ILEnc}_{\mathbf{T}}$ and $\mathsf{ILDec}_{\mathbf{T}}$ the leaking encryption and decryption oracles of the idealized scheme $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$ (the keys have been made explicit). Formally, we are to derive an upper bound for

$$\left| \Pr\left[\mathcal{A}^{\mathsf{LEnc}_{\mathbf{K},\mathbf{T}}, \mathsf{LDec}_{\mathbf{K}}, \mathsf{IC}, \mathsf{IC}^{-1}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{ILEnc}_{\mathbf{T}}, \mathsf{ILDec}_{\mathbf{T}}, \mathsf{IC}, \mathsf{IC}^{-1}} \Rightarrow 1\right] \right|$$

for any $\mathbf{q}$-bounded $\mathcal{D}$—since this is essentially a distinguishing advantage, we use the notation $\mathcal{D}$ instead of $\mathcal{A}$. For this, we employ Patarin's H-coefficient technique [CS14]. We first define the transcripts of the distinguisher when interacting with the real/ideal scheme, and recall the technique.

[13] As mentioned, we cannot always use independent $\widetilde{\pi}_1, \ldots, \widetilde{\pi}_u$, otherwise the u keys have to be collision-free, resulting in the undesired birthday term $\frac{u^2}{2^n}$.

In detail, we summarize the adversarial queries to the ideal TBC IC in a set

$$\tau_{\mathsf{IC}} = ((k_1, t_1, x_1, y_1), \ldots, (k_{q_{\mathsf{IC}}}, t_{q_{\mathsf{IC}}}, x_{q_{\mathsf{IC}}}, y_{q_{\mathsf{IC}}})),$$

which indicates the i-th query is either forward $\mathsf{IC}_{k_1}^{t_1}(x_1) \to y_1$ or backward $\mathsf{IC}_{k_1}^{t_1}(y_1) \to x_1$.

Note that by our assumption, all the IC queries (except for KDF and TGF) made by TEDT are leaked to $\mathcal{D}$. Such information also gives rise to records of the form (k_i, t_i, x_i, y_i). To make a distinction, we denote by τ_{IC}^* the union of these records and the adversarial query transcript τ_{IC}. For the number of internal IC calls made by the scheme, we have:

- upon encrypting (N, A, M), TEDT makes at most $2 \cdot 2 \cdot \lceil \frac{|M|}{n} \rceil + 2 \cdot \lceil \frac{|A|}{n} \rceil \leq 4(\lceil \frac{|M|}{n} \rceil + \lceil \frac{|A|}{n} \rceil)$ IC queries during encrypting and hashing A and M, and 2×3 queries during hashing the additional 3 blocks due to N, T, and the padded length field. Therefore, the number of internal IC calls is at most $4(\lceil \frac{|M|}{n} \rceil + \lceil \frac{|A|}{n} \rceil) + 6$;
- upon decrypting (N, A, C) with $C = \mathbf{c} \| Z$, the number of internal IC calls is at most $4(\lceil \frac{|\mathbf{c}|}{n} \rceil + \lceil \frac{|A|}{n} \rceil) + 6$.

Therefore, when interacting with the idealized scheme $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$, the number of internal IC calls is at most $4\sigma + 6(q_e + q_d)$, and thus

$$\left|\tau_{\mathsf{IC}}^*\right| \leq 4\sigma + 6(q_e + q_d) + q_{\mathsf{IC}}. \tag{4}$$

We also summarize the calls to KDF and TGF in a list

$$\tau_{\widetilde{\pi}} = ((K_1, t_1, x_1, y_1), (K_2, t_2, x_2, y_2), \ldots).$$

In this set, the i-th tuple (K_i, t_i, x_i, y_i) indicates:

- When interacting with the real AEAD scheme $\mathsf{TEDT}[\mathsf{IC}]_{\mathbf{K},\mathbf{T}}$, the i-th query is either $\mathsf{KDF}/\mathsf{TGF}(K_i, t_i, x_i) \to y_i$ or $\mathsf{KDF}^{-1}/\mathsf{TGF}^{-1}(K_i, t_i, y_i) \to x_i$; and,
- When interacting with the idealized scheme $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$, the i-th query is either $\mathsf{TRPFamily}(j, t_i, x_i) \to y_i$ or $\mathsf{TRPFamily}^{-1}(j, t_i, y_i) \to x_i$ for some user index j such that $K_j = K_i$, where K_j is the key sampled by the procedure *Finalization*(u) (see Table 6).

Note that 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. One could imagine that we append these keys to the true adversarial $\mathsf{KDF}/\mathsf{TGF}$ transcripts at the end of the interaction for conceptual simplicity (see below), and this enables reconstruct the above $\tau_{\widetilde{\pi}}$.

For convenience, we also keep a list

$$\tau_{\mathsf{H}}^* = \big((U_1, V_1 \| W_1), (U_2, V_2 \| W_2), \ldots\big)$$

for the appeared inputs and outputs of the hash function $\mathsf{H}[\mathsf{IC}]$. Note that by the specification of TEDT, none of the U_i can be empty. As we assumed all the underlying IC queries have been leaked and included in τ_{IC}^*, this list contains redundant information: it can be recovered from τ_{IC}^*. But it simplifies the language: for example, see the simple form of Eq. (5).

In addition to the above, the "public-keys" $\mathbf{T} = (T_1, \ldots, T_u)$, where $T_i = PK_i \| 0$, are also included in the transcript. Moreover, to simplify the arguments (in particular, the definition of bad transcripts), we reveal to the distinguisher the user keys $\mathbf{K} = (K_1, \ldots, K_u)$ *at the end of the interaction.* In detail,

- in the real world, we reveal the keys $\mathbf{K}$ in use, and

- in the ideal world, we reveal the keys $\mathbf{K}$ sampled by the procedure *Finalization*(u) (see Table 6).

This is without loss of generality since $\mathcal{D}$ is free to ignore this additional information to compute its output bit. Formally, we append both $\mathbf{T}$ and $\mathbf{K}$ to the tuple $(\tau^*_{\mathsf{H}}, \tau^*_{\mathsf{IC}}, \tau_{\widetilde{\pi}})$ and obtain what we call the *transcript*

$$\tau = (\tau^*_{\mathsf{H}}, \tau^*_{\mathsf{IC}}, \tau_{\widetilde{\pi}}, \mathbf{T}, \mathbf{K}).$$

It may seem exotic that transcripts for an AEAD scheme do not include encryption and decryption query transcripts τ_e and τ_d. This is due to the unbound leakage assumption: since during encryption and decryption, almost all the underlying queries to IC are leaked, the transcripts τ^*_{IC} and $\tau_{\widetilde{\pi}}$ allow completely recover the encryption/decryption queries and answers. Therefore, there is no need for their presence in this section.

With respect to some fixed distinguisher $\mathcal{D}$, a transcript τ is called *attainable* if there exist oracles $(\mathsf{IC}, \mathsf{TRPFamily})$ such that the interaction of $\mathcal{D}$ with the ideal scheme $(\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}, \mathsf{IC})$ yields τ. 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^*_{IC} and an ideal TBC IC, we say that IC *extends* τ^*_{IC}, denoted $\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$, if $\mathsf{IC}^t_k(x) = y$ for all $(k, t, x, y) \in \tau^*_{\mathsf{IC}}$. Given a set $\tau_{\widetilde{\pi}}$,

- in the real world, we say IC extends $\tau_{\widetilde{\pi}}$, denoted $\mathsf{IC} \vdash \tau_{\widetilde{\pi}}$, if $\mathsf{IC}^t_K(x) = y$ for all $(K, t, x, y) \in \tau_{\widetilde{\pi}}$;
- in the ideal world, we say $\mathsf{TRPFamily}$ extends $\tau_{\widetilde{\pi}}$, denoted $\mathsf{TRPFamily} \vdash \tau_{\widetilde{\pi}}$, if $\mathsf{TRPFamily}(i, t, x) = y$ for all $(K, t, x, y) \in \tau_{\widetilde{\pi}}$ and all user index i such that $K_i = K$.

It's easy to see that for any attainable transcript $\tau = (\tau^*_{\mathsf{H}}, \tau^*_{\mathsf{IC}}, \tau_{\widetilde{\pi}}, \mathbf{T}, \mathbf{K})$, the interaction of $\mathcal{D}$ with oracles $(\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}, \mathsf{IC})$ produces τ if and only if $\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$ and $\mathsf{TRPFamily} \vdash \tau_{\widetilde{\pi}}$.

With the above, the main lemma of H-coefficient technique is as follows.

Lemma 4. *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 $\mathbf{Adv}(\mathcal{D}) \leq \varepsilon_1 + \varepsilon_2$.

A proof could be found in [CS14].

5.3.2 Completing the Transition

Following the above framework, we start by defining bad transcripts. For a transcript τ, we define μ_T and μ_W, the *maximum multiplicity of T and W*, as

$$\mu_T := \max_{t \in \{0,1\}^n} \left|\{i \in \{1, \ldots, u\} : T_i = t\}\right|,$$
$$\mu_W := \max_{w \in \{0,1\}^{n-1}} \left|\left\{(U, V \| W) \in \tau^*_{\mathsf{H}} : W = w\right\}\right|. \tag{5}$$

Then, an attainable transcript τ is bad, if either of the following conditions is fulfilled:

- (B-1) $\mu_T \geq n$, $\mu_W \geq n$.
- (B-2) there exists a KDF/TGF query $(K, t, x, y) \in \tau_{\widetilde{\pi}}$ such that $(K, t, x, \star) \in \tau^*_{\mathsf{IC}}$ or $(K, t, \star, y) \in \tau^*_{\mathsf{IC}}$.

The bound on μ_T depends on the distribution PKDistribution. If $T_1, \ldots, T_u$ are uniformly distributed, then it's easy to see

$$\Pr[\mu_T \geq n] \leq \binom{u}{n} \cdot \frac{1}{2^{(n-1)(n-1)}} \leq \frac{2^{n-1} \cdot u^n}{n! 2^{(n-1)n}} \leq \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n. \tag{6}$$

Whereas if $T_1, \ldots, T_u$ are distinct, then it trivially holds $\mu_T = 1$ and $\Pr[\mu_T \geq n] = 0$.

Reasoning about μ_W requires the multi-semicollision resistance of H. Formally, we rely on the following lemma, which claims (multi-semi) collision resistance on H.

Lemma 5. *Consider the interaction between an* muCIML2 $\mathbf{q}$*-adversary* $\mathcal{A}$ *and the scheme* $\mathsf{TEDT}[\mathsf{IC}, \mathit{TRPFamily}]_{\mathbf{T}}$*, and* $\mathcal{A}$ *has at most* σ *blocks in all its queried plaintext, ciphertext, and associated data. Define*

$$q^*_{\mathsf{IC}} := 4\sigma + 6(q_e + q_d) + 2q_{\mathsf{IC}}. \tag{7}$$

Then with probability at least $1 - \frac{8q^*_{\mathsf{IC}}}{2^n}$*, it holds:*

(1) H *is collision-free: any two distinct* $(U, V\|W)$ *and* $(U^*, V^*\|W^*)$ *in* τ^*_{H} *necessarily have* $V\|W \neq V^*\|W^*$*;*

(2) H *is multi semi-collision-free:* $\mu_W < n$*.*

Proof. We denote by G_1 the game that captures the interaction between the muCIML2 adversary $\mathcal{A}$ and $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$. Following Hirose [Hir06], we normalize the game: for each IC query either made by $\mathcal{A}$ or made by the TEDT mode, we assume the system makes its Hirose matching query immediately (see section 5.2). Denote by G_2 the obtained normalized game. We now count the number of IC queries in G_2:

- Since $Q_i(N) = P_i(N) \oplus 1$ for any i, the $2\sigma - 1$ IC queries made during the encryption pass only result in an increase of at most 1 new IC queries for the first constant $P_0(N)$ (since $P_0(N)$ is the input to KDF, it wouldn't appear in τ^*_{IC});
- the $2\sigma + 6(q_e + q_d)$ IC queries made by H would not result in new queries, since their matching queries have been included in these $2\sigma + 6(q_e + q_d)$ queries, and
- the q_{IC} queries made by $\mathcal{A}$ may give rise to $\leq q_{\mathsf{IC}}$ new queries.

Therefore, in G_2, the number of IC queries does not exceed $q^*_{\mathsf{IC}} = 4\sigma + 6(q_e + q_d) + 2q_{\mathsf{IC}}$ as in Eq. (7), and the upper bound on the number of matching IC query pairs is $q^*_{\mathsf{IC}}/2$.

Then consider G_2. We define three bad conditions during its execution:

- (C-1) Collision occurs between compression function outputs. Formally, there exists two $3n$-bit values $u\|g\|h$ and $u'\|g'\|h'$ such that:
  - $(u, h, g, g \oplus g'), (u, h, g \oplus \theta, g \oplus \theta \oplus h'), (u, h, g, g \oplus g''), (u, h, g \oplus \theta, g \oplus \theta \oplus h'') \in \tau^*_{\mathsf{IC}}$ for some g', h', g'', h'', and
  - $\mathsf{chop}(g'\|h') = \mathsf{chop}(g''\|h'')$.
- (C-2) Initial-vector is hit: $\exists u, g, h : \mathsf{Hir}[\tau^*_{\mathsf{IC}}](u\|g\|h) = [0]^{2n}$. Formally, there exists a $3n$-bit value $u\|g\|h$ such that $(u, h, g, g), (u, h, g \oplus \theta, g \oplus \theta) \in \tau^*_{\mathsf{IC}}$;

- (C-3) n-collision occurs: there exists n records $(u_1, h_1, g_1, z_1), \ldots, (u_n, h_n, g_n, z_n) \in \tau^*_{\mathsf{IC}}$ such that $\mathsf{chop}(g_1 \oplus z_1) = \ldots = \mathsf{chop}(g_n \oplus z_n)$.

For (C-1), from Lemma 1 we have (when $q^*_{\mathsf{IC}}/2 \le 2^n/4$, i.e., $q^*_{\mathsf{IC}} \le 2^n/2$)

$$\Pr[\text{(C-1)}] \le \frac{6}{2^n} \cdot \frac{q^*_{\mathsf{IC}}}{2} = \frac{3q^*_{\mathsf{IC}}}{2^n}.$$

For (C-2), consider any pair of queries (k, t, x, y) and $(k, t, x \oplus \theta, y')$. Clearly, regardless of the directions of these two queries, it holds $\Pr[y = x \wedge y' = x \oplus \theta] \le \frac{1}{(2^n - q^*_{\mathsf{IC}})^2} \le \frac{4}{2^{2n}}$ when $q^*_{\mathsf{IC}} \le 2^n/2$, and thus

$$\Pr[\text{(C-2)}] \le \frac{4}{2^{2n}} \cdot \frac{q^*_{\mathsf{IC}}}{2} = \frac{2q^*_{\mathsf{IC}}}{2^{2n}} \le \frac{q^*_{\mathsf{IC}}}{2^n}. \qquad (n \ge 2)$$

For (C-3), an application of Lemma 2 (also requires $q^*_{\mathsf{IC}} \le 2^n/2$) yields $\Pr[\text{(C-3)}] \le \frac{(4q^*_{\mathsf{IC}})^n}{n! 2^{n(n-1)}}$. Using $n! \ge \left(\frac{n}{e}\right)^n \ge 2^n$ (since $n \ge 6 > 2e$), we reach

$$\Pr[\text{(C-3)}] \le \frac{(4q^*_{\mathsf{IC}})^n}{2^{n \cdot n}} \le \frac{4q^*_{\mathsf{IC}}}{2^n}. \qquad (4q^*_{\mathsf{IC}} \le 2^n)$$

A union bound yields

$$\Pr[\text{(C-1)} \vee \text{(C-2)} \vee \text{(C-3)}] \le \frac{8q^*_{\mathsf{IC}}}{2^n}.$$

Then, conditioned on $\neg$(C-1) $\wedge$ $\neg$(C-2) $\wedge$ $\neg$(C-3), we show that the two claims hold. We first consider claim (1). For any two records $(U, V\|W)$ and $(U^*, V^*\|W^*)$, assume that *tail* is the maximum common suffix of U and U^*, i.e., $U = header\|u\|tail$, $U^* = header^*\|u^*\|tail$, $|u| = |u^*| = n$, $u \ne u^*$, and $|header|, |header^*|$, and $|tail|$ are multiples of n. Then we distinguish two cases:

Case 1: either $header\|u$ **or** $header^*\|u^*$ **is empty.** Without loss of generality, we assume $header\|u$ is empty. Since U isn't empty, this means *tail* isn't empty. Then we have $\mathsf{H}(header^*\|u^*) \ne [0]_{2n}$ by $\neg$(C-2), i.e., in $\mathsf{H}(U^*)$, the hash-chain value after absorbing u^* is different from the initial vector $[0]_{2n}$. So the two "first-block" calls in absorbing *tail* in $\mathsf{H}(U)$ and $\mathsf{H}(U^*)$ are different. By $\neg$(C-1), this means the resulted hash-chain values are different. Then by iteratively applying $\neg$(C-1), it can be seen the "last-block calls" in $\mathsf{H}(U)$ and $\mathsf{H}(U^*)$ are different, and further $V\|W \ne V^*\|W^*$.

Case 2: neither $header\|u$ **or** $header^*\|u^*$ **is empty.** Then by $\neg$(C-1), $\mathsf{H}(header\|u) \ne \mathsf{H}(header^*\|u^*)$, i.e., the two hash-chain values after absorbing $u \ne u^*$ are different. If *tail* is empty, then as $u \ne u^*$ the "last-block calls" in $\mathsf{H}(U)$ and $\mathsf{H}(U^*)$ are clearly different and further $V\|W \ne V^*\|W^*$; otherwise, the two claims follow by iteratively applying $\neg$(C-1).

We then prove claim (2). The above show that distinct hash inputs U and U^* necessarily result in distinct "last-block-calls". By this, any n hash inputs $(U_1, \ldots, U_n)$ necessarily result in n distinct "last-block-calls" denoted $\mathsf{Hir}(u_1, g_1, h_1), \ldots, \mathsf{Hir}(u_n, g_n, h_n)$. By the definition of Hir, it can be seen such an n-semicollision correspond to an n-collision within n chopped Davies-Meyer function calls. Concretely, assume that for

$$g'_1\|h'_1 = \mathsf{Hir}(u_1\|g_1\|h_1), \ldots, g'_n\|h'_n = \mathsf{Hir}(u_n\|g_n\|h_n),$$

it holds $\mathsf{chop}(h'_1) = \ldots = \mathsf{chop}(h'_n)$, then it essentially holds

$$\mathsf{chop}\big(\mathsf{IC}^{h_1}_{u_1}(g_1 \oplus \theta) \oplus (g_1 \oplus \theta)\big) = \ldots = \mathsf{chop}\big(\mathsf{IC}^{h_n}_{u_n}(g_n \oplus \theta) \oplus (g_n \oplus \theta)\big),$$

contradicting $\neg$(C-3). These complete the proof. □

Lemma 5 indicates

$$\Pr[\mu_W \geq n] \leq \frac{8q^*_{\mathsf{IC}}}{2^n}. \tag{8}$$

We continue analyzing bad transcripts, and consider (B-2). Note that by our definition, for every $(K,t,x,y) \in \tau_{\widetilde{\pi}}$ in the ideal world, the key K is actually given by the "dummy" key vector $\mathbf{K}$. We thus need to argue that:

- The "dummy" key vector $\mathbf{K}$ is uniformly distributed, i.e., for any "target" $\mathbf{K}^\dagger = (K_1^\dagger, \ldots, K_u^\dagger)$, the keys $\mathbf{K}$ picked during the *Finalization*(u) process of $\mathsf{TRPFamily}$ satisfies $\Pr[\mathbf{K} = \mathbf{K}^\dagger] = \frac{1}{2^{un}}$.

For this, define a set of keys $\mathcal{K}^\circ$ as

$$\mathcal{K}^\circ := \left\{\mathbf{K}^\circ = (K_1^\circ, \ldots, K_u^\circ) : \forall (i,j), K_i^\dagger = K_j^\dagger \text{ if and only if } K_i^\circ = K_j^\circ\right\}.$$

Then, the sufficient and necessary condition for $\mathsf{TRPFamily}$ to finally reach $\mathbf{K}^\dagger$ is: (i) a key vector $\mathbf{K}^{\circ\circ} \in \mathcal{K}^\circ$ is sampled during *Initialization*(u), and (ii) the key vector $\mathbf{K}$, which is essentially sampled from $\mathcal{K}^\circ$, hits $\mathbf{K}^\dagger$ during *Finalization*(u). Therefore,

$$\Pr[\mathbf{K} = \mathbf{K}^\dagger] = \frac{\left|\mathcal{K}^\circ\right|}{2^{un}} \cdot \frac{1}{\left|\mathcal{K}^\circ\right|} = \frac{1}{2^{un}}$$

as expected. For the remaining analysis, we introduce an auxiliary set

$$\tau^*_{\mathsf{IC}}[t] := \left\{k \in \{0,1\}^n : (k,t,x,y) \in \tau^*_{\mathsf{IC}} \text{ for some } x, y\right\}.$$

The sets $\tau^*_{\mathsf{IC}}[0], \tau^*_{\mathsf{IC}}[1], \ldots$ constitutes a partition of τ^*_{IC}, and thus $\sum_{t\in\{0,1\}^n} |\tau^*_{\mathsf{IC}}[t]| = q^*_{\mathsf{IC}}$.

Then, the condition (B-2) can be divided into two subconditions, i.e.,

- (B-21) there exists a KDF query $(K, pk\|0, x, y) \in \tau_{\widetilde{\pi}}$ such that $(K, pk\|0, x, \star) \in \tau^*_{\mathsf{IC}}$ or $(K, pk\|0, \star, y) \in \tau^*_{\mathsf{IC}}$;
- (B-22) there exists a TGF query $(K, W\|0, x, y) \in \tau_{\widetilde{\pi}}$ such that $(K, W\|0, x, \star) \in \tau^*_{\mathsf{IC}}$ or $(K, W\|0, \star, y) \in \tau^*_{\mathsf{IC}}$.

A necessary condition (not yet sufficient) for (B-21) is that there exists a user index i such that $K_i \in \tau^*_{\mathsf{IC}}[PK_i\|0]$. As argued, K_i is uniformly distributed. Therefore, conditioned on $\neg$(B-1), we have

$$\begin{aligned}\Pr[\text{(B-21)}] &= \sum_{i=1}^{u} \Pr\left[K_i \in \tau^*_{\mathsf{IC}}[PK_i\|0]\right] \\ &\leq \mu_T \sum_{pk\in\{0,1\}^{n-1}} \frac{\left|\tau^*_{\mathsf{IC}}[pk\|0]\right|}{2^n} \leq \sum_{pk\in\{0,1\}^{n-1}} \frac{n\left|\tau^*_{\mathsf{IC}}[pk\|0]\right|}{2^n}.\end{aligned}$$

A necessary condition (not yet sufficient) for (B-22) is that there exists a user index i such that, there exists a corresponding encryption query $\mathsf{ILEnc}(i,N,A,M) \to \mathbf{c}\|Z$ or decryption query $\mathsf{ILDec}(i,N,A,\mathbf{c}\|Z)$ that results in a hash evaluation of the form $\mathsf{H}[\mathsf{IC}](\mathsf{pad}(A,N,\mathbf{c},T_i)) \to U\|W$, and that $K_i \in \tau^*_{\mathsf{IC}}[W\|0]$. Therefore, conditioned on $\neg$(B-1), we have

$$\begin{aligned}\Pr[\text{(B-22)}] \le & \sum_{i=1}^{u} \sum_{W:(\mathsf{pad}(A,N,\mathbf{c},T_i),*\|W)\in\tau_{\mathsf{H}}^*} \frac{\left|\tau_{\mathsf{IC}}^*[W\|1]\right|}{2^n} \\ \le & \mu_T \cdot \sum_{T\in\{0,1\}^n} \sum_{W:(\mathsf{pad}(A,N,\mathbf{c},T),*\|W)\in\tau_{\mathsf{H}}^*} \frac{\left|\tau_{\mathsf{IC}}^*[W\|1]\right|}{2^n} \\ \le & \mu_T \cdot \mu_W \cdot \sum_{W\in\{0,1\}^{n-1}} \frac{\left|\tau_{\mathsf{IC}}^*[W\|1]\right|}{2^n} \\ \le & n^2 \cdot \sum_{W\in\{0,1\}^{n-1}} \frac{\left|\tau_{\mathsf{IC}}^*[W\|1]\right|}{2^n}. \end{aligned} \tag{9}$$

Summing over the two subconditions, we obtain

$$\Pr[\text{(B-2)} \mid \neg\text{(B-1)}] \le n^2 \cdot \sum_{t\in\{0,1\}^n} \frac{\left|\tau_{\mathsf{IC}}^*[t]\right|}{2^n} \le \frac{n^2|\tau_{\mathsf{IC}}^*|}{2^n} \le \frac{n^2 q_{\mathsf{IC}}^*}{2^n}. \tag{10}$$

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

$$\tau_{\widetilde{\pi}}[K,t] := \left\{(x,y) \in (\{0,1\}^n)^2 : (K,t,x,y) \in \tau_{\widetilde{\pi}}\right\}.$$

With this notation, the ideal world probability is calculated as

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

where $\Pr(\mathbf{K},\mathbf{T})$ stands for the probability that the u user keys take the values $(\mathbf{K},\mathbf{T})$. For the real world distribution, we have

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

Since τ is good, $(K,t,*,*) \notin \tau_{\mathsf{IC}}^*$ for all $(K,t,x,y) \in \tau_{\widetilde{\pi}}$ (otherwise (B-2) is fulfilled). Therefore,

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

Therefore, for any good transcript τ we have

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

and thus

$$\begin{aligned}&\left|\Pr\left[\mathcal{A}^{\mathsf{LEnc}_{\mathbf{K},\mathbf{T}},\mathsf{LDec}_{\mathbf{K}},\mathsf{IC},\mathsf{IC}^{-1}} \Rightarrow 1\right] - \Pr\left[\mathcal{A}^{\mathsf{ILEnc}_{\mathbf{T}},\mathsf{ILDec}_{\mathbf{T}},\mathsf{IC},\mathsf{IC}^{-1}} \Rightarrow 1\right]\right| \\ \le& \Pr[T_{id} \in \mathcal{T}_{bad}] \le \Pr[\text{(B-2)} \mid \neg\text{(B-1)}] + \Pr[\text{(B-1)}] \\ \le& \frac{(n^2+8)q_{\mathsf{IC}}^*}{2^n} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n. \quad \text{(from Eqs. (6), (8), and (10))}\end{aligned} \tag{11}$$

5.4 Unforgeability of the Idealized TEDT

Denote by G_1 (as before) the game that captures the interaction between the muCIML2 adversary $\mathcal{A}$ and $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$. We divide the unforgeability argument into two substeps in two paragraphs below: first, we define and bound several simple bad conditions that may be fulfilled during an execution of the game G_1; then, we show that certain chains of queries never appear during the interaction between the muCIML2 adversary $\mathcal{A}$ and $\mathsf{TEDT}[\mathsf{IC}, \mathsf{TRPFamily}]_{\mathbf{T}}$. As a decryption query returning non-empty message is necessarily validated by such chains of queries, the above indicates that $\mathcal{A}$ is unable to forge in G_1 as long as none of these conditions is fulfilled.

5.4.1 Bad Conditions for Unforgeability

We identify the following conditions during an execution of G_1:

- (C-1) there exists two user indices i, j such that $K_i \| T_i = K_j \| T_j$;
- (C-2) one of the three claims in Lemma 5 is fulfilled, i.e.,
  - *collision*: there exists $(U, V\|W) \neq (U^*, V^*\|W^*) \in \tau^*_{\mathsf{H}}$ with $V\|W = V^*\|W^*$;
  - *multi semi-collision*: $\mu_W \leq \lambda$.

For (C-1), since $\mathbf{T}$ is uniform, we have $\Pr[\text{(C-1)}] \leq \binom{u}{2} \cdot \frac{1}{2^{2n-1}} = \frac{u^2}{2^{2n}}$. The other bound $\Pr[\text{(C-2)}] \leq \frac{8q^*_{\mathsf{IC}}}{2^n}$ follows from Lemma 5, where q^*_{IC} is defined in Eq. (7). For simplicity let $\mathsf{Bad} = \text{(C-1)} \vee \text{(C-2)}$, then

$$\Pr[\mathsf{Bad}] \leq \frac{u^2}{2^{2n}} + \frac{8q^*_{\mathsf{IC}}}{2^n}.$$

5.4.2 Unforgeability unless Bad

We call a decryption query $\mathsf{ILDec}_{\mathbf{T}}(i, N, A, C)$ *non-trivial*, if the corresponding encryption query $\mathsf{ILEnc}_{\mathbf{T}}(i, N, A, M) \to C$ never happened before. Conditioned on $\neg\mathsf{Bad}$, we argue all such non-trivial decryption queries result in $\perp$ except with a bounded probability. Let $C = \mathbf{c}\|Z$. If this query does not give rise to $\perp$, then right after this query is processed, for some user index i, there exists a hash record $(\mathsf{pad}(A, N, \mathbf{c}, T_i), V\|W)$ and a $\mathsf{TRPFamily}$ query $(K_i, W^*\|1, V^*, Z)$ in the history, such that $W^* = W$ and $V^* = V$. This means at some time during the execution, the following queries exist in the history:

$$(a, h, g, g \oplus V) \in \tau^*_{\mathsf{IC}},\ (a, h, g \oplus \theta, g \oplus \theta \oplus W\|b) \in \tau^*_{\mathsf{IC}},\ (K_i, W^*\|1, V^*, Z) \in \tau_{\widetilde{\pi}},$$

where a is the last block of $\mathsf{pad}(A, N, \mathbf{c}, T_i)$, b is either 0 or 1, and $W^* = W$ and $V^* = V$. We distinguish two cases:

Case 1: $(K_i, W^*\|1, V^*, Z)$ is created After the pair of IC queries. As $W^* = W$, we simplify the notation as $(K_i, W\|1, V^*, Z)$. We argue this query $(K_i, W\|1, V^*, Z)$ cannot be forward. For this, assume otherwise, then it's due to an earlier encryption query $\mathsf{ILEnc}_{\mathbf{T}}(j, N', A', M') \to \mathbf{c}'\|Z$, and that $\mathsf{H}(\mathsf{pad}(A', N', \mathbf{c}', T_j)) = V\|W$ (i.e., it collides with the pair of IC queries in question). Now,

- if $j = i$ and $(N, A, \mathbf{c}) = (N', A', \mathbf{c}')$, then since we forbid trivial decryption queries, the tag produced by $\mathsf{ILEnc}_{\mathbf{T}}(j, N', A', M')$ cannot be Z, and hence cannot create the query $(K_i, W\|1, V^*, Z)$;
- if $j = i$ while $(N, A, \mathbf{c}) \neq (N', A', \mathbf{c}')$, then by Lemma 3 we have $\mathsf{pad}(A, N, \mathbf{c}, T_i) \neq \mathsf{pad}(A', N', \mathbf{c}', T_i)$, which then implies a hash collision and contradicts $\neg$(C-2);

- if $j \neq i$ and $K_j \neq K_i$, then the query $\mathsf{ILEnc}_{\mathbf{T}}(j, N', A', M')$ cannot lead to creating $(K_i, W\|1, V^*, Z)$;
- finally, if $j \neq i$ yet $K_j = K_i$, then we have $T_j \neq T_i$ by ¬(C-1), which by Lemma 3 further implies $\mathsf{pad}(A, N, \mathbf{c}, T_i) \neq \mathsf{pad}(A', N', \mathbf{c}', T_j)$ and contradicts ¬(C-2).

In all, $(K_i, W\|1, V^*, Z)$ has to be backward. During the execution of G_1, the number of such backward queries is at most q_d. Conditioned on ¬(C-2), the number of records $(U^\dagger, V^\dagger\|W^\dagger) \in \tau^*_{\mathsf{H}}$ with $W^\dagger = W$ is at most n. This implies that the number of "target" $V^\dagger$ values is also at most n. For each such "target" $V^\dagger$ and each backward query $(K_i, W\|1, V^*, Z)$, we have

$$\Pr[V^* = V^\dagger] \leq \frac{1}{2^n - q_e - q_d} \leq \frac{2}{2^n}.$$

Therefore,

$$\Pr[\text{Case 1} \mid \neg\mathsf{Bad}] \leq \frac{2nq_d}{2^n}. \tag{12}$$

Case 2: $(K_i, W^*\|1, V^*, Z)$ is created Before the pair of IC queries. We consider the query $(v, h, g, g \oplus V)$ first. Regardless of its direction, $V = g \oplus (g \oplus V)$ is uniform in $\geq 2^n - q^*_{\mathsf{IC}}$ possibilities. So

$$\Pr[V = V^*] \leq \frac{1}{2^n - q^*_{\mathsf{IC}}} \leq \frac{2}{2^n}.$$

As argued before,

$$\Pr[W = W^*] \leq \frac{2}{2^n - q^*_{\mathsf{IC}}} \leq \frac{4}{2^n}.$$

Therefore, for each such triple of queries, the probability of collision is at most $\frac{8}{2^{2n}}$. We have at most $q_d + q_e$ choices for $(K_i, W^*\|1, V^*, Z)$, and q^*_{IC} choices for the pair of IC queries.[14] Therefore,

$$\Pr[\text{Case 2}] \leq \frac{8q^*_{\mathsf{IC}}(q_d + q_e)}{2^{2n}}. \tag{13}$$

Note that these arguments are significantly simplified by the normalization of the game: without the normalization, $(K_i, W^*\|1, V^*, Z)$ may be created between the two matching IC queries, which gives rise to many additional cases. Gathering (12) and (13) yields

$$\Pr[\mathcal{A} \text{ forge in } \mathsf{G}_2 \mid \neg\mathsf{Bad}] \leq \frac{2nq_d}{2^n} + \frac{8q^*_{\mathsf{IC}}(q_d + q_e)}{2^{2n}}.$$

This plus $\Pr[\mathsf{Bad}]$ yield

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\mathsf{IC},\mathsf{TRPFamily}]_{\mathbf{T}},\mathsf{L}^*,u} &\leq \frac{u^2}{2^{2n}} + \frac{8q^*_{\mathsf{IC}}}{2^n} + \frac{2nq_d}{2^n} + \frac{8q^*_{\mathsf{IC}}(q_d + q_e)}{2^{2n}} \\ &\leq \frac{u^2}{2^{2n}} + \frac{(n^2+9)q^*_{\mathsf{IC}}}{2^n},\end{aligned} \tag{14}$$

since $q^*_{\mathsf{IC}} \leq 4\sigma + 6(q_e + q_d) + 2q_{\mathsf{IC}}$, $\frac{8q^*_{\mathsf{IC}}(q_d+q_e)}{2^{2n}} \leq \frac{8(q^*_{\mathsf{IC}})^2}{6\cdot 2^{2n}} \leq \frac{q^*_{\mathsf{IC}}}{2^n}$, and $\frac{2nq_d}{2^n} \leq \frac{n^2 q^*_{\mathsf{IC}}}{2^n}$. Then, Eq. (14) and Eq. (11) yield Eq. (1):

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\mathsf{IC}]_{\mathbf{K},\mathbf{T}},\mathsf{L}^*,u} \leq \frac{u^2}{2^{2n}} + \frac{(2n^2+17)(4\sigma + 6(q_e+q_d) + 2q_{\mathsf{IC}})}{2^n} + \frac{1}{2n!}\cdot\left(\frac{4u}{2^n}\right)^n.$$

[14] Note that the pair $\big((v, h, g, g \oplus V), (v, h, g \oplus \theta, g \oplus \theta \oplus W\|b)\big)$ should be deemed as an *ordered* pair.

5.5 The Necessity of the Inverse

Upon each decryption query $\mathsf{Dec}_{K,PK}(N, A, \mathbf{c}\|Z)$, after deriving the digest $V\|W = \mathsf{H}(\mathsf{pad}(A, N, \mathbf{c}, PK\|0))$, if we use the inverse-free checking "If $\mathsf{TGF}(K, W, V) \neq Z$ then return $\perp$; else decrypt", then the forgery attack turns clear: as mentioned, invalid decryption queries leak the correct tag $\mathsf{TGF}(K, W, V)$. Let's see where our proof approach fails for this design. Concretely, when the proof proceeds into the analogue of Case 1 in Section 5.4, the query $(K_i, W\|1, V^*, Z)$ may be *a forward query created during decryption* (which is exactly the case of the aforementioned obvious forgery attack), and we are unable to utilize the uniformness of V^* to complete this argument as we did in section 5.4. Overall, in our model, the inverse of the underlying primitives seems necessary to achieve integrity against decryption/verification leakage. In practice, this means the use of inverse increases robustness against decryption leakage.

6 Leakage CCA Security of TEDT

We now detail the leakage-resistant CCA security of TEDT. The leakage model and assumption are much more complicated than those in Section 5. Therefore, we first state the model in subsection 6.1, then specify the (mild) assumptions in subsections 6.2 and 6.3, present the security claims in subsection 6.4, and give the proof in subsection 6.5.

6.1 Modeling Leakage Functions

We model the leakage as *probabilistic* efficient functions manipulating and/or computing (partially) secret values. In TEDT, each computation of E (resp. $\oplus$) comes with some additional (internal) information given by L_E (resp. $\mathsf{L}_\oplus$). However, we make a distinction between the leakages given by KDF and TGF and those given by the less protected calls to E. Indeed, while KDF and TGF both use E, the implementation of these algorithms might offer different levels of protection compared to all the other calls to E in TEDT, including those of the hash function in Figure 3.

We split the leakage trace resulting from the leaking execution of the TBC E between its *input* and *output* parts: if $\mathsf{E}_K^T(X) \to Y$, $\mathsf{L}_\mathsf{E}(K, T, X, Y) := (\mathsf{L}_\mathsf{E}^{in}(K, T; X), \mathsf{L}_\mathsf{E}^{out}(K, T; Y))$ with semicolon. While this distinction appears a bit theoretical at first, it better reflects the designers' implementation goals for each functions/calls and it allows interpreting the security bounds based on cryptanalytic experience (as explained later in section 6.2).

Finally, we insist one more time on the probabilistic feature of the leakage functions $\mathsf{L}_\mathsf{E}^{in}$, $\mathsf{L}_\mathsf{E}^{out}$, and $\mathsf{L}_\oplus$ (which is indeed likely in practice): measuring p times the leakage from the same computation would *not* result in completely identical traces. Therefore, we will write $[\mathsf{L}_\oplus]^p$ for the vector of p leakage traces of $\oplus$ (and use similar notations for the other operations). Because of the plentiful possible uses of E, we will next denote its input-output leakage function pair as $(\mathsf{L}^{in}, \mathsf{L}^{out})$ for simplicity.

Oracle-free leakage function. An artifact of modeling leakages as probabilistic functions is that L_E might contain "future" calls to E [DP08]. While benign as a first sight, the possibility for an adversary to call such a leakage function gives him the ability to mount a "future computation attack" [YSPY10]. For instance, if the leakage resulting from $\mathsf{E}(k_0, T, P_1(N)) \to k_1$ in TEDT might also already contain $y_2 = \mathsf{E}(k_1, T, Q_1(N))$, the value y_2 cannot remain unpredictable in the next block. Preventing $\mathsf{L}_\mathsf{E}(k_0, *, *)$ from calling $\mathsf{E}(k_1, *, *)$ cannot be achieved only from the tweakable pseudorandom permutation in the standard model. Hence, leakage security (mainly confidentiality) cannot follow. Consequently, for the natural single-pass re-keying encryption used in TEDT, there are only two reliable existing proof approaches which are (i) the leakage simulatability assumption in the standard model [SPY13], and (ii) the non-invertible leakage assumption (or variants)

in the ideal model [YSPY10].[15] Since we focus here on the ideal TBC model, and since the non-invertible leakages can be more easily measured by cryptanalytic practice (which we believe is important for designing modes), we start following this approach of Yu et al. Nevertheless, we also provide a standard model analysis (of muCCAmL2) based on the former approach in Appendix C.

With this in mind, and to prevent future computation attacks in the ideal model, we assume oracle-free leakage functions [YSPY10]: they cannot make any call to IC, since it is natural for an implementation not to evaluate computations that are unrelated to its current state — e.g., $\mathsf{E}(k_0, *, *)$ in our above example. Therefore, we will say that the leakage function associated to E is oracle-free, if $\tau(\mathsf{L}_{\mathsf{E}}^{in}) = \tau(\mathsf{L}_{\mathsf{E}}^{out}) = \emptyset$, where $\tau(\mathsf{L}_{\mathsf{E}}^{*})$ is the transcript of queries and answers made by $\mathsf{L}_{\mathsf{E}}^{*}$ to IC when $\mathsf{L}_{\mathsf{E}}^{*}$ is evaluated on its inputs.

As discussed in [YSPY10], this model appears to have a natural correspondence with concrete attacks on circuits implementing (tweakable) block ciphers, where the measured leakages can be interpreted as a simple function of the cipher's input and key during the first few rounds of the computation, and/or as a simple function of the cipher's output and key during the last few rounds of the computation, but where any useful function of the cipher input and output remains elusive (or is the sign of a completely broken implementation). Also, the use of ideal models does not result in trivial results, as the bounds essentially match simple side-channel attacks to some extent.

6.2 Non-Invertible Leakage Assumption

To achieve confidentiality, the leakages have to be somewhat "bounded"—unlike Section 5. To capture this, we follow [YSPY10] and require that the ephemeral TBC keys remain secret in the following sense: the probability that an adversary recovers an ephemeral key before it is being refreshed should be small.

Yu et al. required the adversary $\mathcal{A}$ to *precisely output the secret* [YSPY10, Definition 2]. We are more generous as we allow $\mathcal{A}$ to *output a set of q guesses* instead, and $\mathcal{A}$ wins as long as *the secret is in this set*. Clearly, this weakens the assumption. Yet, interestingly, this *weaker* assumption results in *better* bounds than Yu et al. [YSPY10] (which should be an artifact of the proof technique). More formally, we define

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{2\text{-}up}[q]}(\mathcal{A}) \qquad (15)\\ &\quad \stackrel{\text{def}}{=\!=} \Pr_{\mathsf{IC},s_1}\left[s_2 \leftarrow \mathsf{IC}_{s_1}^{T}(P_A), z \leftarrow \mathsf{IC}_{s_1}^{T}(P_B), \text{Guesses} \leftarrow \mathcal{A}^{\mathsf{IC}}(s_2, z, \mathsf{leak}) : s_1 \in \text{Guesses}\right],\end{aligned}$$

where $|\text{Guesses}| = q$, and $\mathcal{A}$'s input leak is a list of leakages depending on values T, P_A, P_B, s_0 specified by $\mathcal{A}$:[16]

$$\mathsf{leak} = \left[\mathsf{L}^{out}(s_0, T; s_1), \mathsf{L}^{in}(s_1, T; P_A), \mathsf{L}^{out}(s_1, T; s_2), \mathsf{L}^{in}(s_1, T; P_B), \mathsf{L}^{out}(s_1, T; z)\right]^{p}. \quad (16)$$

Note that the challenge secret s_1 is uniformly picked. This emulates the real scenario, in which the target secret is (nearly) uniform in all the possibilities. We further define

$$\mathbf{Adv}^{\mathsf{2\text{-}up}[q]}(p, q_{\mathsf{IC}}, t) \stackrel{\text{def}}{=\!=} \max\left\{\mathbf{Adv}^{\mathsf{2\text{-}up}[q]}(\mathcal{A})\right\}, \quad (17)$$

where the maximum is taken over all adversaries that repeat their measurements p times, makes q_{IC} IC-queries, and runs in time t.

[15] Using the standard model and PPT leakage functions, existing provably secure re-keying schemes typically require a more complicated alternating structure [DP08, Pie09], or public randomness [YSPY10, FPS12] in order to overcome the "future computation" attack.

[16] This is a simplified description of a challenge-response process. Granting $\mathcal{A}$ the freedom to choose this s_0 is for composability purpose (which appears in the proof): the same holds for s in Equation (19).

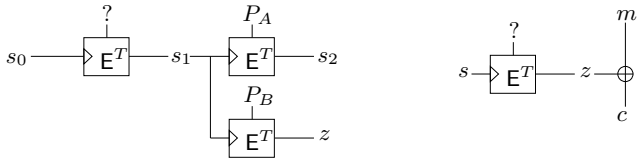

Figure 7: (Left) Illustration on the 2-up assumption; (Right) Basic message manipulation.

Understanding 2-up[q] Advantage. Equation (15) defines a leakage property of a small "unit" of TEDT, pictured in Figure 7 (left). Concretely, it captures that the secret s_1 cannot be recovered from the involved leakages. Note that s_1 is used in two subsequent TBC-calls (this is why we call this assumption "2-up"), from which both the "input" and the "output" leakages may contain information on s_1: this clarifies the presence of $\mathsf{L}^{in}(s_1, T; P_A), \mathsf{L}^{out}(s_1, T; k_2)$, $\mathsf{L}^{in}(s_1, T; P_B)$, and $\mathsf{L}^{out}(s_1, T; y_2)$ in Equation (16). On the other hand, s_1 itself is the output of the previous cipher-call, and this explains the presence of $\mathsf{L}^{out}(s_0, T; s_1)$, which may contain information on s_1 as well. The goal of repeating the measurements p times is to fit into the requirement of providing challenge decryption leakages several times in the muCCAmL2 security game. It should be noted that while this repetition may reduce the measurement noise and make an SPA attack easier, the corresponding advantage of (17) should still be smaller than that of a DPA.

Testers: Measuring in Practice. The concrete values of $\mathbf{Adv}^{\text{2-up}[q]}$ can be measured by running the following tester against the best known SCA adversary $\mathcal{A}$. This along with Theorem 2 allows deriving concrete limits on the capability of the implementation.

1: **Tester for UP $\mathbf{Adv}^{\text{2-up}[q]}$**
2: Let the challenging adversary $\mathcal{A}$ specify s_0.
3: Pick the secret $s_1 \xleftarrow{\$} \{0,1\}^n$ and compute $P_{pre} \leftarrow (\mathsf{E}^T_{s_0})^{-1}(s_1)$.
4: Repeat $s_1 \leftarrow \mathsf{E}^T_{s_0}(P_{pre})$, $s_2 \leftarrow \mathsf{E}^T_{s_1}(P_A)$, and $z \leftarrow \mathsf{E}^T_{s_1}(P_B)$ for p times.
5: Pass the leakages of Step (4) to $\mathcal{A}$. In our model, this means $[\mathsf{L}^{out}(s_0, T; s_1), \mathsf{L}^{in}(s_1, T; P_A)]^p$, $[\mathsf{L}^{out}(s_1, T; s_2), \mathsf{L}^{in}(s_1, T; P_B), \mathsf{L}^{out}(s_1, T; z)]^p$ are returned to $\mathcal{A}$.
6: Let the challenging adversary $\mathcal{A}$ output q guesses $k_1, \ldots, k_q$, the adversary $\mathcal{A}$ wins as long as $s_1 \in \{k_1, \ldots, k_q\}$.

6.3 Capturing the (In)Security of the XOR

As a last step before stating the muCCAmL2 bound, we have to measure the leakage resulting from XORing the random (looking) block stream with the message blocks in TEDT. To capture this concrete information we follow Pereira et al. [PSV15] and we define

$$\mathbf{Adv}^{\mathsf{LORL2}}(\mathcal{A}) \stackrel{\text{def}}{=} \Big| \Pr_{\mathsf{IC},z}\Big[c^0 \leftarrow z \oplus m^0 : \mathcal{A}^{\mathsf{IC}}(c^0, \mathsf{leak}_0) \Rightarrow 1\Big] - \Pr_{\mathsf{IC},z}\Big[c^1 \leftarrow z \oplus m^1 : \mathcal{A}^{\mathsf{IC}}(c^1, \mathsf{leak}_1) \Rightarrow 1\Big] \Big| \ , \quad (18)$$

where leak_b again depends on values T, s specified by $\mathcal{A}$:

$$\mathsf{leak}_b = \Big([\mathsf{L}^{out}(s, T; z)]^p, \mathsf{L}_{\oplus}(z, m^b), [\mathsf{L}_{\oplus}(z, c^b)]^{p-1}\Big). \quad (19)$$

In the abbreviation LORL2, the suffix L stands for *leaking*, and the suffix 2 indicates both *encryption and decryption leakages* are given. We also define

$$\mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\mathsf{IC}}, t) \stackrel{\text{def}}{=} \max_{\mathcal{A}} \Big\{ \mathbf{Adv}^{\mathsf{LORL2}}_{T}(\mathcal{A}) \Big\}. \quad (20)$$

Understanding LORL2 Advantage. Equation (18) defines the information that an adversary might extract from the "basic" message manipulation made in TEDT, which involves XORing as pictured in Figure 7 (right). Concretely, the sensitive point is the key stream block z. This block is the output of a TBC-call, hence the presence of $[\mathsf{L}^{out}(s,T;z)]^p$ (repeated p times as clarified before). Then, the block z is used to mask the message block, and thus the leakage $\mathsf{L}_{\oplus}(z,m^b)$ comes. Finally, the presence of $[\mathsf{L}_{\oplus}(z,c^b)]^{p-1}$, the leakage from the decryption direction, still stems from the challenge decryption leakage requirements of muCCAmL2.

Like in [HLWW16, PSV15], if a single XOR of the message leaks a single bit, then no muCCAmL2 security would spring up. Thus, on the one hand it is legitimate to focus on protecting this part of leaking implementations. But on the other hand, we cannot claim that $\mathbf{Adv}^{\mathsf{LORL2}}(p,q_{\mathsf{IC}},t)$ is negligible. So our goal here is to faithfully reduce the muCCAmL2 to simple and precise pieces that are more easy to protect as isolated components. This type of methodology is not new in the theory community. For example, it is typically assumed that the PRP advantage of AES is concrete 0.01 rather than "negligible" [Tes11]. Yet, the more critical nature of physical leakages also make us deviate from these results. In some sense, we argue that the advantage *degrades in an inevitable rate* during the encryption, rather than to argue for some level of *security amplification*. So in this sense, TEDT is a *security-preserving domain extender* for a "single-block" encryption operation. In practice, the value $\mathbf{Adv}^{\mathsf{LORL2}}$ can be similarly measured by a tester and it is easier to study (and to reduce with relevant protections) than to study entire modes.

1: **Tester for LORL2 $\mathbf{Adv}^{\mathsf{LORL2}}$**
2: Let the challenging adversary $\mathcal{A}$ give s, T, and (m^0,m^1).
3: Pick the secrets $y \xleftarrow{\$} \{0,1\}^n$, $b \xleftarrow{\$} \{0,1\}$, and compute $P_{pre} \leftarrow (\mathsf{E}_s^T)^{-1}(y)$.
4: Compute $c \leftarrow y \oplus m^b$, and repeat $y \leftarrow \mathsf{E}_s^T(P_{pre})$ for p times while $m^b \leftarrow y \oplus c$ for $p-1$ times.
5: Pass c and the leakages of Step (4) to $\mathcal{A}$. In our model, this means $(c, [\mathsf{L}^{out}(s,T;y)]^p, \mathsf{L}_{\oplus}(y,m^b), [\mathsf{L}_{\oplus}(y,c)]^{p-1})$ are returned to $\mathcal{A}$.
6: Let the challenging adversary $\mathcal{A}$ output the guess b'.

6.4 muCCAmL2 Analysis of TEDT

We define the leakage function $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ of TEDT as:

- $\mathsf{L}_{\mathsf{enc}}$, the leakages generated during the encryption:
  - $\mathsf{L}^{in}(k,t;x)$ & $\mathsf{L}^{out}(k,t;y)$ generated by internal calls to $\mathsf{E}(k,t;x) \to y$ (excluding KDF- and TGF-calls which are again modeled as leak-free),
  - $\mathsf{L}_{\oplus}(a,b)$ generated by the internal actions $a \oplus b$,
  - all the intermediate values involved in the computations of the hash functions (i.e., hash functions are non-protected, and leak everything).
- $\mathsf{L}_{\mathsf{dec}}$, the above that are generated during the decryption.

We denote $\mathbf{q} = (q_m, q_e, q_d, p-1, q_{\mathsf{IC}})$ and we define

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT},\mathsf{L}}(u,\mathbf{q},t,\sigma) \stackrel{\text{def}}{=\!=} \max\left\{\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{TEDT},\mathsf{L},u}\right\},$$

where the maximum is taken over all $(\mathbf{q},t)$-bounded adversaries against u users that have at most σ blocks in all their (challenge & non-challenge) queries including AD.

Theorem 2. *In the ideal TBC model, with the* TEDT *leakage functions* $\mathsf{L} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$ *defined above, if the leakage functions* L^{in}, L^{out}, $\mathsf{L}_{\oplus}$ *satisfy the assumptions specified by*

Equation (15) and Equation (18), then the following holds:

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT}}(u,\mathbf{q},t,\sigma) \leq \frac{1}{n!}\cdot\left(\frac{4u}{2^n}\right)^n + \frac{2u^2}{2^{2n}} + \frac{4(n^2+9)q^*}{2^n} + \sigma\cdot\mathbf{Adv}^{\mathsf{LORL2}}(p,q^*,t^*) + 2\sigma\cdot\mathbf{Adv}^{\mathsf{2\text{-}up}[q^*]}(p,q^*,t^*). \quad (21)$$

where $\mathbf{Adv}^{\mathsf{2\text{-}up}[q^*]}$ *and* $\mathbf{Adv}^{\mathsf{LORL2}}$ *are defined in Eq. (17) and Eq. (20) respectively,* $q^* = 2q_{\mathsf{IC}}+4\sigma+6(q_e+q_d+q_m)$*,* $t^* = O(t+p\sigma t_l)$*, and* t_l *is the total time to evaluate* L^{in} *and* L^{out}*.*

Interpreting the bound. We focus on the last two terms since the others have been analyzed before. On the one hand, the term $\sigma\cdot\mathbf{Adv}^{\mathsf{LORL2}}(p,q^*,t^*)$ corresponds to the reduction to the "minimal" message manipulation. On the other hand, the term $2\sigma\cdot\mathbf{Adv}^{\mathsf{2\text{-}up}[q^*]}(p,q^*,t^*)$ captures the hardness of side-channel key recovery, and it is roughly of some birthday type, namely

$$O\left(\sigma\cdot\frac{q_{\mathsf{IC}}+\sigma+t}{c\cdot 2^n}\right) = O\left(\frac{(q_{\mathsf{IC}}+\sigma+t)\sigma}{c\cdot 2^n}\right),$$

for some parameter c that depends on the concrete conditions. Yet, it is nowadays a common assumption that with such a small data complexity (only 3 relevant leakage traces), the value of c should not be significant [MOP07, Pie09].

The term $\sigma\cdot\mathbf{Adv}^{\mathsf{2\text{-}up}}$ illustrates the security loss of "hybrid factor" mentioned in the introduction. A standard model-based proof for the black-box security of TEDT would also suffer from a similar term $\sigma\cdot\mathbf{Adv}^{\mathsf{TBC}}$: this is unavoidable for hybrid-based proofs, see [Men17]. However, for the black-box setting the term $\sigma\cdot\mathbf{Adv}^{\mathsf{TBC}}$ is not tight, and is merely an artifact of the technique. By contrast, here with leakage the term $\sigma\cdot\mathbf{Adv}^{\mathsf{2\text{-}up}}$ is *tight*. This is easy to see: this term captures the collision between the σ keying actions, and such a collision allows the adversary to obtain more than 2 leakage traces about a single key, while our Equation (16) only assumed security with 2 traces. In all, our assumption Equation (16), though a bit conservative, tightly results in a birthday-type bound.

6.5 Proof of Theorem 2

Let's see some intuition first. To prove muCCAmL2 security, we need to show:

(i) Messages encrypted by fresh nonce values have confidentiality, and

(ii) Leakages given by the decryption oracle would not affect the above confidentiality.

Since TEDT has been proved muCIML2 (Theorem 1), all non-trivial decryption queries will be invalid, and then the Encrypt-then-MAC design ensures that every such invalid decryption query only leaks a "useless" pseudorandom value, i.e. $(\mathsf{IC}_K^{*\|1})^{-1}(Z)$ for some Z. By these, the above (ii), harmlessness of decryption leakages, is clear. For the above (i), note that essentially, for each fresh nonce N, TEDT derives a new initial key $k_0 = \mathsf{E}_K^T(N)$ (for some K,T) and takes it as the key of a TBC-based stream cipher $\mathsf{TStream}_{k_0}[\mathsf{IC}]$ (which we formally describe in Fig. 8). This k_0 is somewhat independent from all the other information gained by the adversary. This enables a standard hybrid argument that reduces the left-or-right confidentiality of challenge messages to the indistinguishability of the leaking encryption of two messages $\mathsf{LTStream}_{k_0}[\mathsf{IC}](M^0)$ and $\mathsf{LTStream}_{k_0}[\mathsf{IC}](M^1)$ under a random initial key k_0. Formally, it requires to derive an upper bound for

$$\left|\Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](M^0))\Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](M^1))\Rightarrow 1]\right|$$

for any adversary $\mathcal{A}^{\mathsf{IC}}$ with bounded power. This indistinguishability resembles the classical *eavesdropper security* enhanced with leakages. To establish this "leakage eavesdropper security", we rely on an ideal stream cipher IStream for "relay": IStream encrypts a single message with a true random key stream and offers leakages. Now:

- The "leakage eavesdropper security" of $\mathsf{IStream}$ is easily reduced to $\mathbf{Adv}^{\mathsf{LORL2}}$;
- The outputs of $\mathsf{TStream}_{k_0}[\mathsf{IC}]$ and $\mathsf{IStream}$ can be proved indistinguishable, using the non-invertible assumption on the leakage functions.

The above presented a "top-down" style overview. For the proof, we proceed with a "bottom-up" style flow. In detail, we first formally define and study $\mathsf{TStream}_{k_0}[\mathsf{IC}]$ and $\mathsf{IStream}$ in subsection 6.5.1, then bound the leakage eavesdropper security of $\mathsf{IStream}$ in subsection 6.5.2, and then the leakage eavesdropper security of $\mathsf{TStream}_{k_0}[\mathsf{IC}]$ in 6.5.3, and finally derive the $\mathsf{muCCAmL2}$ bound in subsection 6.5.4.

6.5.1 TEDT versus the ideal stream cipher

Description of $\mathsf{TStream}_{k_0}[\mathsf{IC}](T, N, M)$, $M = m_1 \| \dots \| m_\ell$:

- Initializes an empty list leak for the leakage;
- For $i = 1, \dots, \ell$, computes $k_i \leftarrow \mathsf{IC}^T_{k_{i-1}}(P_i(N))$, $y_i \leftarrow \mathsf{IC}^T_{k_{i-1}}(Q_{i-1}(N))$, and $c_i \leftarrow y_i \oplus m_i$, and adds the leakage traces $[\mathsf{L}^{in}(k_{i-1}, T; P_i(N)), \mathsf{L}^{out}(k_{i-1}, T; k_i)]^p$, $[\mathsf{L}^{in}(k_{i-1}, T; Q_{i-1}(N)), \mathsf{L}^{out}(k_{i-1}, T; y_i)]^p$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathsf{L}_\oplus(y_i, c_i)]^{p-1}$ to the list leak.
- Returns $(c_1, \dots, c_\ell)$.

We define $\mathsf{LTStream}_{k_0}[\mathsf{IC}](T, N, M) = (\mathsf{TStream}_{k_0}[\mathsf{IC}](T, N, M), \mathsf{leak})$ for the list leak standing at the end of the above process.

Description of $\mathsf{IStream}(T, N, M)$, $M = m_1 \| \dots \| m_\ell$:

- Initializes an empty list leak for the leakage;
- For $i = 1, \dots, \ell$, samples $k_i \xleftarrow{\$} \{0,1\}^n$ and $y_i \xleftarrow{\$} \{0,1\}^n$ such that $k_i \neq y_i$, sets $c_i \leftarrow y_i \oplus m_i$, and adds the leakages $[\mathsf{L}^{in}(k_{i-1}, T; P_i(N)), \mathsf{L}^{out}(k_{i-1}, T; k_i)$, $\mathsf{L}^{in}(k_{i-1}, T; Q_{i-1}(N)), \mathsf{L}^{out}(k_{i-1}, T; y_i)]^p$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathsf{L}_\oplus(y_i, c_i)]^{p-1}$ to the list leak.
- Returns $(c_1, \dots, c_\ell)$.

We define $\mathsf{LIStream}(T, N, M) = (\mathsf{IStream}(T, N, M), \mathsf{leak})$ for the list leak standing at the end of the above process.

Figure 8: The real stream cipher $\mathsf{TStream}_{k_0}[\mathsf{IC}]$ and its ideal counterpart $\mathsf{IStream}$.

Formally, the two stream ciphers as well as their leakages are described in Fig. 8. For their indistinguishability, we have the following lemma.

Lemma 6. *For every ℓ-block message M, every T, N, and every (q_{IC}, t)-bounded distinguisher $\mathcal{D}^{\mathsf{IC}}$, it holds*

$$\begin{aligned}&|\Pr[\mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T, N, M)) \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{IC}}(\mathsf{LIStream}(T, N, M)) \Rightarrow 1]| \\ \leq &\ell \cdot \mathbf{Adv}^{2\text{-}\mathsf{up}[q_{\mathsf{IC}}]}\big(p, q_{\mathsf{IC}} + 2\ell, t^*\big),\end{aligned}$$

where $t^* = O(t + \ell \cdot p \cdot t_l)$, t_l is the total time needed for evaluating L^{in} and L^{out}.

There is no term capturing $\mathbf{Adv}^{\mathsf{PRP}}_{\mathsf{IC}}$ since it has been captured by the term $\mathbf{Adv}^{2\text{-}\mathsf{up}[q_{\mathsf{IC}}]}$: if $\mathcal{A}$ predicates the secret then it clearly breaks the PRP security of IC.

Proof. Consider the execution of $\mathcal{D}^{\mathsf{IC}}$ upon the inputs $\mathsf{LTStream}_{k_0}[\mathsf{IC}](T, N, M)$, $M = m_1 \| \dots \| m_\ell$. We define a bad event $\mathsf{BadQuery}$, which occurs when any of the internal keys $k_0, k_1, \dots, k_{\ell-1}$ appears in the key field of an IC query made by $\mathcal{D}^{\mathsf{IC}}$. This event, once happens, would cause the key stream blocks lose randomness. We remark that in [YSPY10,

Appendix A], an event with the same meaning termed Query_a was defined, and was later proved negligible. So here we just need to adapt Yu et al.'s argument to our setting. In detail, given an adversary $\mathcal{D}^{\mathsf{IC}}$, we construct an adversary $\mathcal{A}^{\mathsf{IC}}$ such that

$$\mathbf{Adv}^{\text{2-up}[q_{\mathsf{IC}}]}(\mathcal{A}) \leq \Pr\left[\mathsf{BadQuery} \text{ in } \mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T,N,M))\right]. \tag{22}$$

Concretely, $\mathcal{A}^{\mathsf{IC}}$ runs an instance of $\mathcal{D}$, and keeps record of $\mathcal{D}$'s queries to IC in a set τ_{IC} (as defined in previous proofs). $\mathcal{A}^{\mathsf{IC}}$ simulates the following process against $\mathcal{D}$:

(1) $\mathcal{A}^{\mathsf{IC}}$ randomly guesses an index $i \xleftarrow{\$} [0, \ell-1]$, uniformly samples an initial key k_0, and initializes an empty list leak;

(2) for $j = 1, \ldots, i-1$, $\mathcal{A}^{\mathsf{IC}}$ queries IC to obtain $k_j \leftarrow \mathsf{IC}^T_{k_{j-1}}(P_j(N))$ and $y_j \leftarrow \mathsf{IC}^T_{k_{j-1}}(Q_{j-1}(N))$, and computes $c_j \leftarrow y_j \oplus m_j$. $\mathcal{A}^{\mathsf{IC}}$ then adds the leakage traces $[\mathsf{L}^{in}(k_{j-1}, T; P_j(N)), \mathsf{L}^{out}(k_{j-1}, T; k_j)]^p$, $[\mathsf{L}^{in}(k_{j-1}, T; Q_{j-1}(N)), \mathsf{L}^{out}(k_{j-1}, T; y_j)]^p$, $\mathsf{L}_{\oplus}(y_j, m_j)$, and $[\mathsf{L}_{\oplus}(y_j, c_j)]^{p-1}$ to leak;

(3) $\mathcal{A}^{\mathsf{IC}}$ queries $y_i \leftarrow \mathsf{IC}^T_{k_{i-1}}(Q_{i-1}(N))$ and computes $c_i \leftarrow y_i \oplus m_i$. $\mathcal{A}^{\pi}$ then submits $T, P_A = P_{i+1}(N), P_B = Q_i(N)$, and $s_0 = k_{i-1}$ to its 2-$\mathsf{up}[q]$ challenger, and (according to our convention) this results in the outputs (s_2, z, leak), where

$$\mathsf{leak} = \left(\left[\mathsf{L}^{out}(k_{i-1}, T; s_1), \mathsf{L}^{in}(s_1, T; P_A), \mathsf{L}^{out}(s_1, T; s_2), \mathsf{L}^{in}(s_1, T; P_B), \mathsf{L}^{out}(s_1, T; z)\right]^p\right)$$

for the challenge secret $s_1 \in \{0,1\}^n$. $\mathcal{A}^{\pi}$ adds $[\mathsf{L}^{in}(k_{i-1}, T; P_i(N)), \mathsf{L}^{out}(k_{i-1}, T; s_1)]^p$, $[\mathsf{L}^{in}(k_{i-1}, T; Q_{i-1}(N)), \mathsf{L}^{out}(k_{i-1}, T; y_i)]^p$, $\mathsf{L}_{\oplus}(y_i, m_i)$, and $[\mathsf{L}_{\oplus}(y_i, c_i)]^{p-1}$ to leak as the leakage of the i-th iteration.

(4) $\mathcal{A}^{\mathsf{IC}}$ sets $k_{i+1} \leftarrow s_2$ and $y_{i+1} \leftarrow z$, and computes $c_{i+1} \leftarrow y_{i+1} \oplus m_{i+1}$. It (conceptually) takes the challenge s_1 as the key k_i and adds $[\mathsf{L}^{in}(s_1, T; P_{i+1}(N)), \mathsf{L}^{out}(s_1, T; k_{i+1})]^p$, $[\mathsf{L}^{in}(s_1, T; Q_i(N)), \mathsf{L}^{out}(s_1, T; z)]^p$, $\mathsf{L}_{\oplus}(y_i, m_i)$, and $[\mathsf{L}_{\oplus}(y_i, c_i)]^{p-1}$ to leak as the leakage of the $i+1$-th iteration.

(5) Then $\mathcal{A}^{\mathsf{IC}}$ starts from k_{i+1} to emulate the remaining actions of $\mathsf{LTStream}_{k_0}[\mathsf{IC}]$ encrypting the tail $m_{i+1} \| \ldots \| m_\ell$ to obtain $c_{i+1} \| \ldots \| c_\ell$. Eventually, $\mathcal{A}^{\mathsf{IC}}$ serves the ciphertext $c_1 \| \ldots \| c_\ell$ as well as the leakage list leak to $\mathcal{D}$, and outputs the set $\text{Guesses} = \{k : (k, t, x, y) \in \tau_{\mathsf{IC}} \text{ for some } t, x, y\}$.

The strategy of $\mathcal{A}^{\mathsf{IC}}$ is quite obvious: if $\mathcal{D}$ triggers the event $\mathsf{BadQuery}$ then the key k being queried must be in τ_{IC}. Therefore, $\mathcal{A}^{\mathsf{IC}}$ makes a uniform guess on the position of the first key on which such a query is made; guessing the first queried key ensuring that that key will only be correlated to one thing: the corresponding leakages (and not any previous call on IC). This guess will be correct with probability $1/\ell$. Then, $\mathcal{A}^{\mathsf{IC}}$ emulates the encryption process of $\mathsf{TStream}_{k_0}[\mathsf{IC}]$ and provides the leakages to $\mathcal{D}$, except for the i index, for which the leakages and IC output are replaced by those obtained from a challenger for the seed-preserving property. If the guess on the index i is correct, all the inputs sent to $\mathcal{D}$ are distributed exactly as those produced by $\mathsf{TStream}_{k_0}[\mathsf{IC}](T, N, M)$. Therefore, when $\mathcal{D}$ halts, if $\mathcal{D}$ made a query on s_1, then simply outputting τ_{IC} would break the game. So we have $\Pr[s_1 \in \text{Guesses} \mid \mathsf{BadQuery} \text{ in } \mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T, N, M))] = \frac{1}{\ell}$.

Now, we observe that

$$\Pr\left[s_1 \in \text{Guesses} \mid \mathsf{BadQuery} \text{ in } \mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T,N,M))\right] \leq \frac{\Pr[s_1 \in \text{Guesses}]}{\Pr\left[\mathsf{BadQuery} \text{ in } \mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T,N,M))\right]}.$$

And it can be seen $\mathcal{A}$ is $(q_{\mathsf{IC}} + 2\ell, t^*)$-bounded for $t^* = O(t + \ell \cdot p \cdot t_l)$: the factor p before t_l stems from the fact that the leakage functions are evaluated p times in total. By this,

$$\begin{aligned}\Pr\left[\mathsf{BadQuery} \text{ in } \mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T,N,M))\right] &\leq \ell \cdot \Pr[s_1 \in \text{Guesses}] \\ &\leq \ell \cdot \mathbf{Adv}^{\mathsf{2\text{-}up}[q_{\mathsf{IC}}]}(\mathcal{A}) \quad \text{(Eq. 15)} \\ &\leq \ell \cdot \mathbf{Adv}^{\mathsf{2\text{-}up}[q_{\mathsf{IC}}]}(p, q_{\mathsf{IC}} + 2\ell, t^*). \quad \text{(Eq. 17)}\end{aligned}$$

During the real execution $\mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T,N,M))$, as long as the event $\mathsf{BadQuery}$ never happens, all the keys and key stream blocks are fresh random values independent from τ_{IC} the transcript of IC queries of $\mathcal{D}$, and have the same distribution as those in the ideal execution $\mathcal{D}^{\mathsf{IC}}(\mathsf{LIStream}(T,N,M))$. Therefore,

$$\begin{aligned}&\left|\Pr[\mathcal{D}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T,N,M)) \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{IC}}(\mathsf{LIStream}(T,N,M)) \Rightarrow 1]\right| \\ &\leq \ell \cdot \mathbf{Adv}^{\mathsf{2\text{-}up}[q_{\mathsf{IC}}]}(p, q_{\mathsf{IC}} + 2\ell, t^*)\end{aligned}$$

as claimed. □

6.5.2 Leakage eavesdropper security of the ideal stream cipher

We then show how to reduce the leakage eavesdropper advantage of $\mathsf{IStream}$ encrypting an ℓ-block message to the defined term $\mathbf{Adv}^{\mathsf{LORL2}}$.

Lemma 7. *For every pair of ℓ-block messages M^0 and M^1 and (q_{IC}, t)-bounded adversary $\mathcal{A}^{\mathsf{IC}}$, it holds*

$$\begin{aligned}&\left|\Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{LIStream}(T,N,M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{LIStream}(T,N,M^1)) \Rightarrow 1]\right| \\ &\leq \ell \cdot \mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\mathsf{IC}}, t^*) + \frac{\ell}{2^n},\end{aligned}$$

where t^* is as defined in Lemma 6.

Proof. Let $M^0 = m_1^0 \| \ldots \| m_\ell^0$ and $M^1 = m_1^1 \| \ldots \| m_\ell^1$. We start by building a sequence of $\ell + 1$ messages $M_{h,0}, \ldots, M_{h,\ell}$ starting from M^0 and modifying its blocks one by one until obtaining M^1. That is, $M_{h,i} := m_1^1 \| \ldots \| m_i^1 \| m_{i+1}^0 \| \ldots \| m_\ell^0$. For any i, assuming a (q_{IC}, t)-bounded adversary $\mathcal{A}^{\mathsf{IC}}$ against $\mathsf{IStream}(T,N,M_{h,i-1})$ and $\mathsf{IStream}(T,N,M_{h,i})$, we build a (q_{IC}, t^*)-bounded adversary $\mathcal{A}_2^{\mathsf{IC}}$ against the distribution defined in Eq. (18). In detail, $\mathcal{A}_2^{\mathsf{IC}}$ proceeds in four steps:

(1) $\mathcal{A}_2^{\mathsf{IC}}$ uniformly samples k_0 and initializes an empty list leak;

(2) for $j = 1, \ldots, i-1$, $\mathcal{A}_2^{\mathsf{IC}}$ uniformly samples random values k_j, y_j such that $k_j \neq y_j$, computes $c_j \leftarrow y_j \oplus m_j^1$, and adds the traces $[\mathsf{L}^{in}(k_{j-1}, T; P_j), \mathsf{L}^{out}(k_{j-1}, T; k_j)]^p$, $[\mathsf{L}^{in}(k_{j-1}, T; Q_{j-1}), \mathsf{L}^{out}(k_{j-1}, T; y_j)]^p$, $\mathsf{L}_{\oplus}(y_j, m_j^1)$, and $[\mathsf{L}_{\oplus}(y_j, c_j)]^{p-1}$ to leak;

(3) $\mathcal{A}_2^{\mathsf{IC}}$ samples k_i, and submits $s = k_{i-1}, T$ to its $\mathsf{LORL2}$ challenger. According to our convention, $\mathcal{A}_2^{\mathsf{IC}}$ will receive the outputs (c^b, leak_b), where the leakage list $\mathsf{leak}_b = \left([\mathsf{L}^{out}(s, T; z)]^p, \mathsf{L}_{\oplus}(z, m^b), [\mathsf{L}_{\oplus}(z, c^b)]^{p-1}\right)$. $\mathcal{A}_2^{\mathsf{IC}}$ sets $y_i = z$, and adds the leakages $[\mathsf{L}^{in}(k_{i-1}, T; P_i), \mathsf{L}^{out}(k_{i-1}, T; k_i)]^p$, $[\mathsf{L}^{in}(k_{i-1}, T; Q_{i-1}), \mathsf{L}^{out}(k_{i-1}, T; y_i)]^p$, $\mathsf{L}_{\oplus}(y_i, m^b)$, and $[\mathsf{L}_{\oplus}(y_i, c^b)]^{p-1}$ to leak;

(4) Then $\mathcal{A}_2^{\mathsf{IC}}$ starts from k_i to emulate the remaining actions of $\mathsf{IStream}$ encrypting the tail $m_{i+1}^0 \| \ldots \| m_\ell^0$ to obtain $c_{i+1} \| \ldots \| c_\ell$. Eventually, $\mathcal{A}_2^{\mathsf{IC}}$ serves the ciphertext $c_1 \| \ldots \| c_{i-1} \| c^b \| c_{i+1} \| \ldots \| c_\ell$ as well as all the generated simulated leakages to $\mathcal{A}^{\mathsf{IC}}$, and outputs whatever $\mathcal{A}^{\mathsf{IC}}$ outputs.

It can be seen that, as long as $k_i \neq z$, depending on whether the input tuple received by $\mathcal{A}_2^{\mathsf{IC}}$ captures the $\mathsf{LORL2}$ challenger encrypting m_i^0 or m_i^1, the inputs to $\mathcal{A}^{\mathsf{IC}}$ capture $\mathsf{IStream}$ encrypting $M_{h,i-1}$ or $M_{h,i}$. Note that $\Pr[k_i = z] = 1/2^n$. Moreover, $\mathcal{A}_2^{\mathsf{IC}}$ is $(q_{\mathsf{IC}}, O(t + \ell \cdot p \cdot t_l))$-bounded if $\mathcal{A}^{\mathsf{IC}}$ is (q_{IC}, t)-bounded, which means

$$\begin{aligned}&\left|\Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{IStream}(T, N, M_{h,i-1})) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{IStream}(T, N, M_{h,i})) \Rightarrow 1]\right| \\ \leq &\mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\mathsf{IC}}, O(t + \ell \cdot p \cdot t_l)) + \frac{1}{2^n}\end{aligned}$$

by Eq. (20). Taking a union bound over the ℓ hybrids yields the main claim. □

6.5.3 Leakage eavesdropper security of the stream cipher TStream[π]

For simplicity, we define

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}, t, \ell) \\ &:= \max_{\mathcal{A}} \left\{ \left| \Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T, N, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{IC}}(\mathsf{LTStream}_{k_0}[\mathsf{IC}](T, N, M^1)) \Rightarrow 1] \right| \right\},\end{aligned}$$

where the abbreviation $\mathsf{eavl2}$ stands for *eavesdropper security with encryption and decryption leakages*, and the maximal is taken over all adversaries making q_{IC} queries to IC and running in time t. Gathering Lemmas 6 and 7, we could obtain an upper bound on $\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}, t, \ell)$.

Lemma 8. *For every pair of ℓ-block messages M^0 and M^1, (q_{IC}, t)-bounded adversary $\mathcal{A}^{\mathsf{IC}}$ and t^* is as defined in Lemma 6, it holds*

$$\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}, t, \ell) \leq \frac{\ell}{2^n} + \ell \cdot \mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\mathsf{IC}}, t^*) + 2\ell \cdot \mathbf{Adv}^{2\text{-up}[q_{\mathsf{IC}}]}(p, q_{\mathsf{IC}} + 2\ell, t^*).$$

Proof. By the previous lemmas, we have

$$\begin{aligned}&\left|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{TStream}_{k_0}(T, N, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{TStream}_{k_0}(T, N, M^1)) \Rightarrow 1]\right| \\ &\quad \leq \underbrace{\left|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IStream}_{k_0}(T, N, M^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IStream}_{k_0}(T, N, M^1)) \Rightarrow 1]\right|}_{\leq \ell \cdot \mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\mathsf{IC}}, O(t+\ell \cdot p \cdot t_l)) + \frac{\ell}{2^n} \text{ (by Lemma 7)}} \\ &\qquad + \underbrace{\sum_{b=0,1} \left|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{TStream}_{k_0}(T, N, M^b)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IStream}_{k_0}(T, N, M^b)) \Rightarrow 1]\right|}_{\leq 2\ell \cdot \mathbf{Adv}^{2\text{-up}[q_{\mathsf{IC}}]}(p, q_{\mathsf{IC}}+2\ell, O(t+\ell \cdot p \cdot t_l)) \text{ (by Lemma 6)}}\end{aligned}$$

Hence the claim. □

6.5.4 Completing the muCCAmL2 Proof

The core step is a bit similar the standard hybrid-based, eavesdropper-to-CPA security reduction: to prove that encrypting $M_1^0, M_2^0, \ldots$ is close to encrypting $M_1^1, M_2^1, \ldots$, we will replace encrypting M_i^0 by M_i^1 one-by-one, and bound the gap introduced by each such replacement using Lemma 8. What makes the situation more complicated is the presence of the decryption queries (since we are considering CCA rather than CPA). Decryption queries may be trivial or non-trivial: recall from subsection 5.4.2 that a decryption query $\mathsf{LDec}_{\mathbf{K},\mathbf{T}}(i, N, A, C)$ is *trivial* if the encryption $\mathsf{LEnc}_{\mathbf{K},\mathbf{T}}(i, N, A, M) \to C$ happens before. The leakages of trivial decryption queries may serve new information, thus requiring some treatments. Formally, we start by defining G_0 as the game capturing the interaction between $\mathcal{A}$ and $(\mathsf{LEnc}_{\mathbf{K},\mathbf{T}}, \mathsf{LEnc}^0_{\mathbf{K},\mathbf{T}}, \mathsf{L}_{\mathsf{decch}}, \mathsf{LDec}_{\mathbf{K},\mathbf{T}}, \mathsf{IC})$ (cf. subsection 2.2), and G_0^* as the game capturing the interaction between $\mathcal{A}$ and $(\mathsf{LEnc}_{\mathbf{K},\mathbf{T}}, \mathsf{LEnc}^1_{\mathbf{K},\mathbf{T}}, \mathsf{L}_{\mathsf{decch}}, \mathsf{LDec}_{\mathbf{K},\mathbf{T}}, \mathsf{IC})$. The subsequent proof consists of several small steps presented in the following paragraphs.

Idealizing TEDT. We then define two intermediate games G_1 and G_1^*: G_1, resp. G_1^*, is obtained from G_0, resp. G_0^*, by replacing all the KDF- and TGF-calls by calls to TRPFamily (as done in section 5.3). By Eq. (11) (although the formalisms are different), with $q_{\mathsf{IC}}^* = 4\sigma + 6(q_m + q_e + q_d) + 2q_{\mathsf{IC}}$ (which is tweaked from Eq. (7)), we have

$$\left|\Pr[\mathsf{G}_1 \Rightarrow 1] - \Pr[\mathsf{G}_0 \Rightarrow 1]\right| \leq \frac{(n^2+8)q_{\mathsf{IC}}^*}{2^n} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n \tag{23}$$

$$\left|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_0^* \Rightarrow 1]\right| \leq \frac{(n^2+8)q_{\mathsf{IC}}^*}{2^n} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n. \tag{24}$$

Excluding non-trivial decryption queries. We then define two new intermediate games G_2 and G_2^*: they differ from G_1 and G_1^* in that:

- They return $\perp$ (as well as the decryption leakages) for all non-trivial decryption queries
- They abort when $\exists i \neq j : K_i \| T_i = K_j \| T_j$.

The gaps have been bounded by Eq. (14) in section 5.4 (with $\Pr[\exists i \neq j : K_i \| T_i = K_j \| T_j]$ included). Therefore,

$$\left|\Pr[\mathsf{G}_2 \Rightarrow 1] - \Pr[\mathsf{G}_1 \Rightarrow 1]\right| \leq \frac{u^2}{2^{2n}} + \frac{(n^2+9)q_{\mathsf{IC}}^{***}}{2^n} \tag{25}$$

$$\left|\Pr[\mathsf{G}_2^* \Rightarrow 1] - \Pr[\mathsf{G}_1^* \Rightarrow 1]\right| \leq \frac{u^2}{2^{2n}} + \frac{(n^2+9)q_{\mathsf{IC}}^{***}}{2^n}. \tag{26}$$

The main hybrid-based argument. For this, 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_2 can be transited to G_2^*. For convenience, we define $\mathsf{G}_{3,0} = \mathsf{G}_2$, and define a sequence of games

$$\mathsf{G}_{3,1}, \mathsf{G}_{3,2}, \ldots, \mathsf{G}_{3,q_e},$$

such that in the j-th system $\mathsf{G}_{3,j}$, 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 $M_{j+1}^1, \ldots, M_{q_e}^1$. It can be seen that $\mathsf{G}_{3,q_e} = \mathsf{G}_2^*$.

We then show that for $j = 1, \ldots, q_e$, $\mathsf{G}_{3,j-1}$ and $\mathsf{G}_{3,j}$ are indistinguishable for $\mathcal{A}^{\mathsf{IC}}$. For this, from $\mathcal{A}^{\mathsf{IC}}$ we build an adversary $\mathcal{A}_2^{\mathsf{IC}}$, such that $|\Pr[\mathsf{G}_{3,j-1} \Rightarrow 1] - \Pr[\mathsf{G}_{3,j} \Rightarrow 1]|$ is related to $\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}^*, O(t), |M_j^0|/n)$. In detail, initially, $\mathcal{A}_2^{\mathsf{IC}}$ samples u distinct keys $K_1 \| T_1, \ldots, K_u \| T_u$ for subsequent simulations. It also keeps a pair of tables $(\textsc{ETable}, \textsc{ETable}^{-1})$, which have entries of the form $\textsc{ETable}(K, T, X) = Y$ and $\textsc{ETable}^{-1}(K, T, Y) = X$, to simulate the primitive TRPFamily via lazy sampling (note that TRPFamily is essentially an ideal cipher IC_2 independent from IC). It then runs $\mathcal{A}$. Upon $\mathcal{A}$'s action, $\mathcal{A}_2^{\mathsf{IC}}$ reacts as follows:

- Upon a query to IC: $\mathcal{A}_2$ simply relays the query to IC and the answer to $\mathcal{A}$.
- Upon a (non-challenge) encryption query (i^*, N^*, A^*, M^*) from $\mathcal{A}$,
  - if $(K_{i^*}, T_{i^*}, C_0(N^*)) \notin \textsc{ETable}$, $\mathcal{A}_2^{\mathsf{IC}}$ samples an initial key k_0 such that $(K_{i^*}, T_{i^*}, k_0) \notin \textsc{ETable}^{-1}$, defines the two entries $\textsc{ETable}(K_{i^*}, T_{i^*}, C_0(N^*)) \leftarrow k_0$ and $\textsc{ETable}^{-1}(K_{i^*}, T_{i^*}, k_0) \leftarrow C_0(N^*)$, and then emulates the encryption process $\mathsf{LTStream}_{k_0}[\mathsf{IC}](T_{i^*}, N^*, M^*)$ to get the ciphertext $\mathbf{c}^*$ and leakages. $\mathcal{A}_2^{\mathsf{IC}}$ then computes $V^* \| W^* \leftarrow \mathsf{H}[\mathsf{IC}](\mathsf{pad}(A^*, N^*, \mathbf{c}^*, T_{i^*}))$ and $Z^* \leftarrow$

$\mathrm{ETABLE}(K_{i^*}, W^*\|1, V^*)$ (if $(K_{i^*}, W^*\|1, V^*) \notin \mathrm{ETABLE}$ then $\mathcal{A}_2^{\mathsf{IC}}$ samples Z^* and defines $\mathrm{ETABLE}(K_{i^*}, W^*\|1, V^*) \leftarrow Z^*$ as before). For this entire process $\mathcal{A}_2^{\mathsf{IC}}$ has to make $4\ell^* + 6$ queries to IC with $\ell^* = |M^*|/n$ and costs $O(p \cdot \ell^* \cdot t_l)$ time to evaluate the leakage functions. Finally, $\mathcal{A}_2^{\mathsf{IC}}$ returns the ciphertext $\mathbf{c}\|Z$ and the leakages to $\mathcal{A}$;

- if $(K_{i^*}, T_{i^*}, C_0(N^*)) \in \mathrm{ETABLE}$, $\mathcal{A}_2^{\mathsf{IC}}$ simply emulates $\mathsf{LTStream}_{k_0}(T_{i^*}, N^*, M^*)$ with $k_0 = \mathrm{ETABLE}(K_{i^*}, T_{i^*}, C_0(N^*))$, calls $V^*\|W^* \leftarrow \mathsf{H}(\mathsf{pad}(A^*, N^*, \mathbf{c}^*, T_{i^*}))$ and computes the tag $Z^* \leftarrow \mathrm{ETABLE}(K_{i^*}, W^*\|1, V^*)$ on the obtained $\mathbf{c}^*$, and returns $\mathbf{c}^*\|Z^*$ and the leakages to $\mathcal{A}$. The cost is similar to the above case.

- Upon a trivial decryption query (i^*, N^*, A^*, C^*) from $\mathcal{A}$, $\mathcal{A}_2^{\mathsf{IC}}$ parses $C^* = \mathbf{c}^*\|Z^*$, computes $k_0 = \mathrm{ETABLE}(K_{i^*}, T_{i^*}, C_0(N^*))$, and simply emulates the decryption of $\mathsf{LTStream}_{k_0}[\mathsf{IC}]$ on $(T_{i^*}, N^*, \mathbf{c}^*)$, and relays the outputs to $\mathcal{A}$. The cost is similar to the encryption case.

- Upon a non-trivial decryption query (i^*, N^*, A^*, C^*) from $\mathcal{A}$, $\mathcal{A}_2^{\mathsf{IC}}$ parses $C^* = \mathbf{c}^*\|Z^*$, and computes $V^*\|W^* \leftarrow \mathsf{H}[\mathsf{IC}](\mathsf{pad}(A^*, N^*, \mathbf{c}^*, T_{i^*}))$. Then,
  - if $(K_{i^*}, W^*\|1, Z^*) \notin \mathrm{ETABLE}$, $\mathcal{A}_2^{\mathsf{IC}}$ samples V^{**} such that $(K_{i^*}, W^*\|1, V^*) \notin \mathrm{ETABLE}$, and defines the two table entries $\mathrm{ETABLE}(K_{i^*}, W^*\|1, V^{**}) \leftarrow Z^*$, $\mathrm{ETABLE}^{-1}(K_{i^*}, W^*\|1, Z^*) \leftarrow V^{**}$;
  - if $(K_{i^*}, W^*\|1, Z^*) \in \mathrm{ETABLE}$, $\mathcal{A}_2^{\mathsf{IC}}$ just sets $V^{**} \leftarrow \mathrm{ETABLE}^{-1}(K_{i^*}, W^*\|1, Z^*)$.

  Now $\mathcal{A}_2^{\mathsf{IC}}$ returns $(\perp, V^{**})$ to $\mathcal{A}$ (regardless of $V^{**} = V^*$ or not).

- Upon $\mathcal{A}$ submitting the ℓ-th challenge tuple $(i_\ell, N_\ell, A_\ell, M_\ell^0, M_\ell^1)$, it necessarily holds $(K_{i_\ell}, T_{i_\ell}, C_0(N_\ell)) \notin \mathrm{ETABLE}$, since:
  - Previous encryption queries to the i_ℓ-th user cannot define $\mathrm{ETABLE}(K_{i_\ell}, T_{i_\ell}, C_0(N_\ell))$ due to the challenge nonce-respecting restriction on $\mathcal{A}$;
  - Previous encryption queries to the $i_{\ell'}$-th users cannot define $\mathrm{ETABLE}(K_{i_\ell}, T_{i_\ell}, C_0(N_\ell))$ since $K_{i_\ell}\|T_{i_\ell} \neq K_{i_{\ell'}}, T_{i_{\ell'}}$.

  Therefore, depending on ℓ, $\mathcal{A}_2^{\mathsf{IC}}$ reacts as follows:
  - When $\ell < j$, it encrypts M_ℓ^0 and returns. In detail, $\mathcal{A}_2^{\mathsf{IC}}$ samples $k_0^{(\ell)}$, sets $\mathrm{ETABLE}(K_{i_\ell}, T_{i_\ell}, C_0(N_\ell)) \leftarrow k_0^{(\ell)}$ and $\mathrm{ETABLE}^{-1}(K_{i_\ell}, T_{i_\ell}, k_0^{(\ell)}) \leftarrow C_0(N_\ell)$, and then emulates $\mathsf{LTStream}_{k_0^{(\ell)}}[\mathsf{IC}](T_{i_\ell}, N_\ell, M_\ell^0) \to \mathbf{c}_\ell$, performs the tag generation accordingly to produce Z_ℓ and returns $(\mathbf{c}_\ell, Z_\ell)$ and the leakages to $\mathcal{A}^{\mathsf{IC}}$. The cost is similar to the non-challenge encryption queries.
  - When $\ell = j$, it relays M_ℓ^0 and M_ℓ^1 to its eavesdropper challenger to obtain $\mathbf{c}_\ell^b$ and leakages leak_{enc} and $[\mathsf{leak}_{dec}]^{p-1}$, and then performs the tag generation accordingly to produce Z_ℓ and returns $C_{ch}^b = (\mathbf{c}_\ell^b, Z_\ell)$ to $\mathcal{A}$. Note that this means the relation $\mathrm{ETABLE}(K_{i_\ell}, T_{i_\ell}, C_0(N_\ell)) = k_0^{ch}$ is implicitly fixed, where k_0^{ch} is the secret key generated inside the eavesdropper challenger;
  - When $\ell > j$, it simply encrypts M_ℓ^1 and returns. The details are similar to the described case $\ell < j$.

- Upon $\mathcal{A}$ making the λ-th query to $\mathsf{L}_{\mathsf{decch}}(\ell)$ $(1 \leq \lambda \leq p-1)$,
  - When $\ell \neq j$, $\mathcal{A}_2^{\mathsf{IC}}$ performs the corresponding decryption and returns the obtained leakages to $\mathcal{A}$;
  - When $\ell = j$, $\mathcal{A}_2^{\mathsf{IC}}$ simply returns the λ-th trace in the vector $[\mathsf{leak}_{dec}]^{p-1}$ as the answer.

It can be seen that, depending on whether $b = 0$ or 1, the whole process is the same as either $\mathsf{G}_{3,j-1}$ or $\mathsf{G}_{3,j}$. By the remarks before, besides running $\mathcal{A}$, $\mathcal{A}_2^{\mathsf{IC}}$ samples at most $2(q_m + q_e + q_d)$ random values (to emulate $\mathsf{TRPFamily}$) and internally processes $q_m + q_e + q_d - 1$ queries (except for the query encrypted by the challenger). Therefore, $\mathcal{A}_2^{\mathsf{IC}}$ makes $q^* = 4\sigma + 6(q_m + q_e + q_d)$ additional queries to IC, and evaluates the leakage functions for $2p\sigma$ times, resulting in $O(p\sigma t_l)$ additional running time. By these, and since $q_{\mathsf{IC}} + q^* \leq q_{\mathsf{IC}}^*$, we have

$$\left|\Pr[\mathsf{G}_{3,i} \Rightarrow 1] - \Pr[\mathsf{G}_{3,i-1} \Rightarrow 1]\right| \leq \mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}^*, O(t + p\sigma t_l), \ell_i),$$

where ℓ_i is the number of blocks in the ith challenge message. This means

$$\begin{aligned}\left|\Pr[\mathsf{G}_2^* \Rightarrow 1] - \Pr[\mathsf{G}_2 \Rightarrow 1]\right| &\leq \sum_{i=1}^{q_e} \left|\Pr[\mathsf{G}_{3,i} \Rightarrow 1] - \Pr[\mathsf{G}_{3,i-1} \Rightarrow 1]\right| \\ &\leq \sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}^*, O(t + p\sigma t_l), \ell_i). \end{aligned} \tag{27}$$

Summing over Eqs. (23), (24), (25), (26), and (27), we eventually reach

$$\begin{aligned}&\left|\Pr[\mathsf{G}_0 \Rightarrow 1] - \Pr[\mathsf{G}_0^* \Rightarrow 1]\right| \\ \leq &\frac{1}{n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{2u^2}{2^{2n}} + \frac{(4n^2 + 34)q_{\mathsf{IC}}^*}{2^n} + \sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p, q_{\mathsf{IC}}^*, O(t + p\sigma t_l), \ell_i), \end{aligned} \tag{28}$$

with the last term bounded by Lemma 8 as

$$\frac{\sigma}{2^n} + \sigma \cdot \mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\mathsf{IC}}^*, O(t + p\sigma t_l)) + 2\sigma \cdot \mathbf{Adv}^{\text{2-up}[q_{\mathsf{IC}}^*]}(p, q_{\mathsf{IC}}^*, O(t + p\sigma t_l)).$$

Using $\frac{(4n^2+34)q_{\mathsf{IC}}^*}{2^n} + \frac{\sigma}{2^n} \leq \frac{4(n^2+9)q_{\mathsf{IC}}^*}{2^n}$, we reach Eq. (21).

7 Performance Evaluation and Related Work

TEDT is designed in order to reach high security guarantees in the presence of leakage. The previous sections gave comprehensive theoretical analyzes of our security claims and physical assumptions. In this section, we finally discuss the cost of these guarantees.

In a first part, we provide performance estimations highlighting the interest of such a leakage-resistant mode of operation enabling leveled implementations, compared to a standard mode where all the blocks require to be uniformly protected against leakage.

In a second part, we provide a discussion of possible competitors. Our main conclusion is that published solutions illustrate an interesting tradeoff between mode-level and implementation-level side-channel resistance. In this respect, a fair comparison would require an evaluation of the performance overheads needed to satisfy the various assumptions used in these modes. We believe such an evaluation and a rigorous discussion of the best way to achieve given levels of confidentiality and integrity in the presence of leakage is among the most important open problems in this field of research.

7.1 Leveled versus Uniform Software Implementations

As a first step in our discussion, we compare two extreme situations with respect to leakage-resistance. First, TEDT reaches the top of the hierarchy in the definitional framework of Guo et al. [GPPS19a] (ignoring black box misuse-resistance) and is therefore expected to minimize the need of implementation-level countermeasures. For this purpose, TEDT

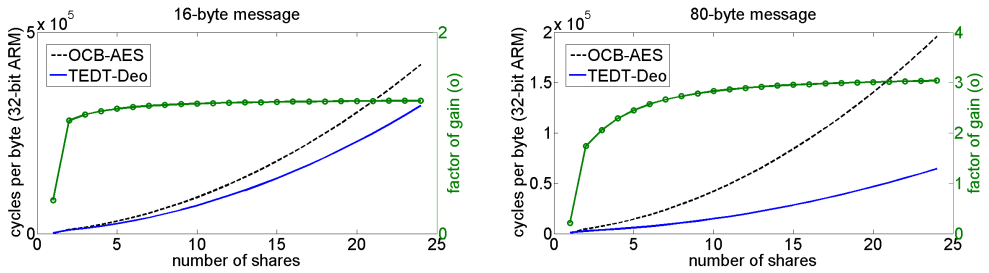

Figure 9: Performance estimations (I): uniformly masked implementation of OCB based on the AES versus leveled implementation of TEDT based on Deoxys-BC-256.

requires 4 TBC calls per message block and a decryption-based tag verification. (The first requirement is due to the need of a two-pass mode in order to ensure CCAmL2, where the encryption pass embeds a re-keying process that requires 2 TBC calls per message block and the hashing, based on the Hirose construction, needs another two calls; the second requirement is necessary to reach CIML2 in the unbounded leakage model). Next, OCB-AES provides no guarantees against leakage, but has an amortized cost of one TBC call per message block and is fully parallelizable (which is not possible with re-keying) [Rog04a] The rationale behind this choice of modes is to avoid a comparative discussion of different leakage assumptions that is challenging given the current (lack of) understanding of the respective implementation cost needed to fulfill these assumptions.

We consider masking as implementation-level countermeasure, and base our estimations on the state-of-the-art solutions proposed by Goudarzi and Rivain in [GR17]. We specifically exploit Table 3 in this reference, which provides cycle counts in function of the number of shares in the masking scheme. We selected Deoxys-BC-256 as instance of TBC to use in TEDT, which was recently elected as a winner of the CAESAR competition [JNPS]. This choice is mostly motivated by the AES-based structure of this cipher, which enables exploiting the same masking countermeasure. In particular, compared to the AES, Deoxys-BC-256 implies performance overheads of a factor 1.4 to 1.6 for unprotected implementations, and has a total of 224 S-boxes compared to 200 for the AES (i.e., 10×16 for the rounds and 10×4 for the key rounds). By counting the fraction of time spent on S-boxes (also provided in the work by Goudarzi and Rivain), this allows us to estimate the respective cycle counts for Deoxys-BC-256 and the AES, for various number of shares (with a ratio 1.5 for one share, converging to $\frac{224}{200}$ as the number of shares increases).

For illustration, we then compare implementations of OCB uniformly protected thanks to masking, and leveled implementations of TEDT. The first ones use $\ell + 2$ calls to a masked AES implementation, while the second ones uses 2 calls to masked Deoxys-BC-256 implementations and 4ℓ calls to the unprotected Deoxys-BC-256 implementation. The results of these estimations can be found in Figures 9 and 10 for messages of various sizes. The most interesting conclusion is that starting from 2 shares (i.e., the minimum amount of masking), TEDT compares favorably to OCB, independent of the message size. This is easily observed thanks to the "factor of gain" on the right Y axis of the figures. It is explained by the large overheads paid when moving from an unprotected implementation (that can run in a few thousands of cycles) to a masked one (e.g., a 2-share masked AES takes $> 50,000$ cycles in [GR17]). The second important conclusion is that the factor of gain approximately converges towards $\frac{\ell+2}{2}$ as the number of shares increases. Quite naturally, the gains get larger and the full convergence requires more shares as the message sizes increase. Concretely, factors of gain larger than 10 can already be observed for medium size messages (e.g., $\ell = 100$ blocks) and 4 to 8 shares. Note that

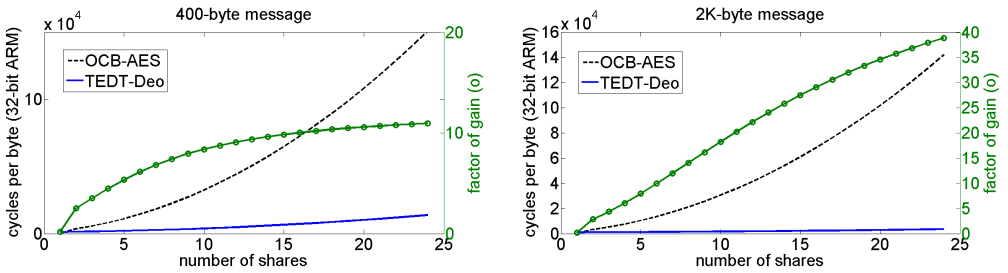

Figure 10: Performance estimations (II): uniformly masked implementation of OCB based on the AES versus leveled implementation of TEDT based on Deoxys-BC-256.

the factor of gain is actually slightly lower than $\frac{\ell+2}{2}$, which can be explained by the slightly worse performances (and larger number of S-boxes) of Deoxys-BC-256 compared to the AES. In this respect, it is worth mentioning that combining TEDT with an underlying cipher specifically optimized for efficient masked implementation, such as the LS-designs in [GLSV15], would actually allow even better results. Note also that these estimations ignore the cost needed to protect the whitening values Δ's in OCB.

We insist that these estimations are only used to give trends and show that mode-level side-channel resistance gains interest as physical security becomes a concern (exact figures require algorithm-specific optimizations that are out of scope here). We also mention that we expect similar (energy) gains in hardware. The only difference is that the energy overheads of the masking countermeasure may then be due both to an increase of the cycle count (as in software) and to a increased area [GMK17]. Also, hardware implementations provide additional opportunities to implement the weakly protected implementation at minimum energy cost (e.g., thanks to low-latency designs [KDH+12]).

7.2 Competing Modes of Operation & Discussion

We conclude this work with a brief overview of other approaches to leakage-resistant (or resilient) authenticated encryption and emerging challenges for their fair comparison.

Starting with seed results, and as already mentioned in Section 3.1, Berti et al. [BKP+18, BPPS17] and Guo et al. [GPPS19a] introduced several security definitions and modes of operation for leakage-resistant AEADs. While their security definitions are the starting point of the TEDT design, we extend their work in several directions. (i) TEDT is designed to offer strong mu security, which is ignored in previous modes, and which, as mentioned before, is highly important for practical use. (ii) TEDT is fully specified, while their proposals left several questions open, including the way to implement a hash function. (iii) TEDT is analyzed in several security models, including the ideal cipher model based on very mild assumptions on leakages (one-wayness) with tight security bounds, while such a tight security analysis is not available for other modes.[17]

Following with competitors, the most relevant and conceptually closest proposal is ISAP, which is an elegant two-passes sponge-based AEAD originally aiming at mode-level DPA resistance [DEM+17]. Subsequently, two works independently proposed methodologies for formally analyzing the leakage security of sponge/duplex designs [GPPS19b, DM19]. While being based on different constructions and primitives, ISAP uses the Encrypt-then-MAC pattern that is similar to TEDT and the analysis in [GPPS19b] suggests that its leveled implementations achieve similar security guarantees as TEDT under different assumptions.

[17] While tight security proofs under idealized assumptions must admittedly always be interpreted with care, it remains that they are desirable targets in general, and practically-relevant in many cases.

Namely, its strongly protected components depend on a rate-1 sponge-based re-keying function (rather than a masked TBC) and its tag verification is not inverse-based so has to be protected against DPA to avoid forgeries.[18] Analyzing ISAP and TEDT reveals why comparing leakage-resistant modes of operation is challenging and requires more research. For example, it depends on (i) whether the sponge-based re-keying of ISAP gives better or worse concrete security guarantees than the masked TBC of TEDT for a given cost/performance target, which additionally depends on whether encryption or decryption leakages are considered, and (ii) whether a DPA-resistant tag verification is preferable or not to an inverse-based tag verification which can leak in full.[19]

Yet, one concrete advantage of ISAP over TEDT is that it anyway enjoys simplified passes: two TBC calls are replaced by a single permutation calls. Several other modes take advantage of the appealing structure of sponges for leakage-resistance (or resilience). For example, Guo et al. proposed a sponge-based variant of TEDT that they denote as TEDTSponge and (as key contribution) a single-pass design TETSponge that only guarantees CCAmL1 (and CIML2) security under similar assumptions as TEDT. It serves as a basis for the Spook candidate to the NIST lightweight cryptography standardization effort (https://www.spook.dev).[20] The Ascon cipher, both elected as a winner of the CAESAR competition and candidate to the NIST lightweight cryptography standardization effort (https://ascon.iaik.tugraz.at) provides similar guarantees (CCAmL1 and CIML2) under different assumptions: its leveled implementations use masking as TETSponge but its tag verification is not inverse-based [GPPS19b]. In general, the sponge duplex construction (without complementary ingredients) also provides some (weaker) security guarantees (formally, CPA security and ciphertext integrity with leakage in encryption only) at low(er) cost [GPPS19b, DM19], which means that implementation-level countermeasures are then needed if stronger security targets must be reached. So overall, comparing the security that these proposals in a fair manner requires understanding the implementation overheads needed to fulfill their respective physical assumptions. We believe the exhibition of this mode-level vs. implementation-level side-channel security tradeoff is an important conclusion of this work and opens interesting avenues for further invetigations.

We finally mention the work of Barwell et al. [BMOS17] who propose a mode of operation and a choice of PRF that satisfy a complementary definition of leakage-resilient and misuse-resistant AEAD (rather than leakage-resistant and misuse-resilient for TEDT). Their security definition is different from ours, notably in the fact that they do not offer to the adversary any leakage that would come from queries in which the game challenger would encrypt either a real or a random message. Such a definition could classify as secure an implementation that leaks plaintexts in full but, on the other hand, makes it feasible to require misuse-resistance instead of misuse-resilience. Their mode of operation is also incompatible with leveled implementations: each block-cipher call needs to be equally well protected, and they propose using, for that purpose, a PRF that is based on pairings. As such, the (e.g., energy) efficiency of a uniformly protected implementation of their mode would be considerably lower than a leveled implementation of TEDT as we propose.

Acknowledgments. Chun Guo was partly supported by the Program of Qilu Young Scholars of Shandong University. Thomas Peters is a postdoctoral researcher and François-Xavier Standaert is a senior research associate of the Belgian Fund for Scientific Research (F.R.S.-FNRS). This work has been funded in parts by the European Union through the

[18] Its confidentiality relies on oracle-free leakages but since permutation-based and contrary to TEDT, it is unknown whether its leakage-resistance can be proven without such an idealized assumption.

[19] As for black box security guarantees, ISAP has not been designed for mu security either, and is actually quite sensitive to mu attacks on its authentication part. It is unknown if our techniques could be used in ISAP to avoid mu security degradation without increasing the secret key-size.

[20] A similar single-pass design can also be derived from TEDT with a similar limitation to CCAmL1, which we illustrate in supplementary material, Appendix E for completeness.

ERC project SWORD (724725), the INNOVIRIS projects SCAUT and C-Cure, and the European Union and Walloon Region FEDER USERMedia project 501907-379156.

References

[AB00] Michel Abdalla and Mihir Bellare. Increasing the lifetime of a key: a comparative analysis of the security of re-keying techniques. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 546–559. Springer, Heidelberg, December 2000.

[ABB+] Jean-Philippe Aumasson, Steve Babbage, Daniel J. Bernstein, Carlos Cid, Joan Daemen, Orr Dunkelman, Kris Gaj, Shay Gueron, Pascal Junod, Adam Langley, David McGrew, Kenny Paterson, Bart Preneel, Christian Rechberger, Vincent Rijmen, Matt Robshaw, Palash Sarkar, Patrick Schaumont, Adi Shamir, and Ingrid Verbauwhede. CHAE: Challenges in Authenticated Encryption. ECRYPT-CSA D1.1, Revision 1.05, 1 March 2017. https://chae.cr.yp.to/whitepaper.html.

[ADL17] Tomer Ashur, Orr Dunkelman, and Atul Luykx. Boosting authenticated encryption robustness with minimal modifications. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 3–33. Springer, Heidelberg, August 2017.

[BBLT18] Subhadeep Banik, Andrey Bogdanov, Atul Luykx, and Elmar Tischhauser. SUNDAE: Small universal deterministic authenticated encryption for the internet of things. *IACR Trans. Symm. Cryptol.*, 2018(3):1–35, 2018.

[BDH+15] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. CAESAR Submission: Keyak v2, 2015.

[BDPV12] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. Duplexing the sponge: Single-pass authenticated encryption and other applications. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 320–337. Springer, Heidelberg, August 2012.

[BGRV15] Josep Balasch, Benedikt Gierlichs, Oscar Reparaz, and Ingrid Verbauwhede. DPA, bitslicing and masking at 1 GHz. In Tim Güneysu and Helena Handschuh, editors, *CHES 2015*, volume 9293 of *LNCS*, pages 599–619. Springer, Heidelberg, September 2015.

[BGSD10] Shivam Bhasin, Sylvain Guilley, Laurent Sauvage, and Jean-Luc Danger. Unrolling cryptographic circuits: A simple countermeasure against side-channel attacks. In Josef Pieprzyk, editor, *CT-RSA 2010*, volume 5985 of *LNCS*, pages 195–207. Springer, Heidelberg, March 2010.

[BHT18] Priyanka Bose, Viet Tung Hoang, and Stefano Tessaro. Revisiting AES-GCM-SIV: Multi-user security, faster key derivation, and better bounds. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 468–499. Springer, Heidelberg, April / May 2018.

[BKP+18] Francesco Berti, François Koeune, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Ciphertext integrity with misuse and leakage: Definition and efficient constructions with symmetric primitives. In Jong Kim, Gail-Joon Ahn, Seungjoo Kim, Yongdae Kim, Javier López, and Taesoo Kim, editors, *ASIACCS 18*, pages 37–50. ACM Press, April 2018.

[BMOS17] Guy Barwell, Daniel P. Martin, Elisabeth Oswald, and Martijn Stam. Authenticated encryption in the face of protocol and side channel leakage. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 693–723. Springer, Heidelberg, December 2017.

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

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

[BT16] Mihir Bellare and Björn Tackmann. The multi-user security of authenticated encryption: AES-GCM in TLS 1.3. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 247–276. Springer, Heidelberg, August 2016.

[BY03] Mihir Bellare and Bennet S. Yee. Forward-security in private-key cryptography. In Marc Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 1–18. Springer, Heidelberg, April 2003.

[CDMP05] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård revisited: How to construct a hash function. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 430–448. Springer, Heidelberg, August 2005.

[CDNY18] Avik Chakraborti, Nilanjan Datta, Mridul Nandi, and Kan Yasuda. Beetle family of lightweight and secure authenticated encryption ciphers. *IACR TCHES*, 2018(2):218–241, 2018. https://tches.iacr.org/index.php/TCHES/article/view/881.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.

[CIMN17] Avik Chakraborti, Tetsu Iwata, Kazuhiko Minematsu, and Mridul Nandi. Blockcipher-based authenticated encryption: How small can we go? In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 277–298. Springer, Heidelberg, September 2017.

[CJRR99] Suresh Chari, Charanjit S. Jutla, Josyula R. Rao, and Pankaj Rohatgi. Towards sound approaches to counteract power-analysis attacks. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 398–412. Springer, Heidelberg, August 1999.

[CS14] Shan Chen and John P. Steinberger. Tight security bounds for key-alternating ciphers. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 327–350. Springer, Heidelberg, May 2014.

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

[DFS19] Alexandre Duc, Sebastian Faust, and François-Xavier Standaert. Making masking security proofs concrete (or how to evaluate the security of any leaking device), extended version. *J. Cryptology*, 32(4):1263–1297, 2019.

[DK18] Daniel Dinu and Ilya Kizhvatov. EM analysis in the IoT context: Lessons learned from an attack on thread. *IACR TCHES*, 2018(1):73–97, 2018. https://tches.iacr.org/index.php/TCHES/article/view/833.

[DM19] Christoph Dobraunig and Bart Mennink. Leakage Resilience of the Duplex Construction. Cryptology ePrint Archive, Report 2019/225, 2019.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *49th FOCS*, pages 293–302. IEEE Computer Society Press, October 2008.

[EKM+08] Thomas Eisenbarth, Timo Kasper, Amir Moradi, Christof Paar, Mahmoud Salmasizadeh, and Mohammad T. Manzuri Shalmani. On the power of power analysis in the real world: A complete break of the keeloqcode hopping scheme. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 203–220. Springer, Heidelberg, August 2008.

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

[GL17] Shay Gueron and Yehuda Lindell. Better bounds for block cipher modes of operation via nonce-based key derivation. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 17*, pages 1019–1036. ACM Press, October / November 2017.

[GLSV15] Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, and Kerem Varici. LS-designs: Bitslice encryption for efficient masked software implementations. In Carlos Cid and Christian Rechberger, editors, *FSE 2014*, volume 8540 of *LNCS*, pages 18–37. Springer, Heidelberg, March 2015.

[GMK17] Hannes Groß, Stefan Mangard, and Thomas Korak. An efficient side-channel protected AES implementation with arbitrary protection order. In Helena Handschuh, editor, *CT-RSA 2017*, volume 10159 of *LNCS*, pages 95–112. Springer, Heidelberg, February 2017.

[GPPS19a] Chun Guo, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Authenticated encryption with nonce misuse and physical leakage: Definitions, separation results and first construction - (extended abstract). In *LATINCRYPT*, volume 11774 of *Lecture Notes in Computer Science*, pages 150–172. Springer, 2019.

[GPPS19b] Chun Guo, Olivier Pereira, Thomas Peters, and François-Xavier Standaert. Towards Low-Energy Leakage-Resistant Authenticated Encryption from the Duplex Sponge Construction. *IACR Cryptology ePrint Archive*, 2019:133, 2019.

[GPT15] Daniel Genkin, Itamar Pipman, and Eran Tromer. Get your hands off my laptop: physical side-channel key-extraction attacks on pcs - extended version. *J. Cryptographic Engineering*, 5(2):95–112, 2015.

[GR17] Dahmun Goudarzi and Matthieu Rivain. How fast can higher-order masking be in software? In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 567–597. Springer, Heidelberg, April / May 2017.

[GST17] Daniel Genkin, Adi Shamir, and Eran Tromer. Acoustic cryptanalysis. *Journal of Cryptology*, 30(2):392–443, April 2017.

[Hir06] Shoichi Hirose. Some plausible constructions of double-block-length hash functions. In Matthew J. B. Robshaw, editor, *FSE 2006*, volume 4047 of *LNCS*, pages 210–225. Springer, Heidelberg, March 2006.

[HLWW16] Carmit Hazay, Adriana López-Alt, Hoeteck Wee, and Daniel Wichs. Leakage-resilient cryptography from minimal assumptions. *Journal of Cryptology*, 29(3):514–551, July 2016.

[IMPS17] Tetsu Iwata, Kazuhiko Minematsu, Thomas Peyrin, and Yannick Seurin. ZMAC: A fast tweakable block cipher mode for highly secure message authentication. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 34–65. Springer, Heidelberg, August 2017.

[JNPS] Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. Deoxys v1.41. *Submitted to CAESAR, October 2016*.

[Jut08] Charanjit S. Jutla. Encryption modes with almost free message integrity. *Journal of Cryptology*, 21(4):547–578, October 2008.

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

[KR01] Joe Kilian and Phillip Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *Journal of Cryptology*, 14(1):17–35, January 2001.

[KR07] Lars R. Knudsen and Vincent Rijmen. Known-key distinguishers for some block ciphers. In Kaoru Kurosawa, editor, *ASIACRYPT 2007*, volume 4833 of *LNCS*, pages 315–324. Springer, Heidelberg, December 2007.

[KY01] Jonathan Katz and Moti Yung. Unforgeable encryption and chosen ciphertext secure modes of operation. In Bruce Schneier, editor, *FSE 2000*, volume 1978 of *LNCS*, pages 284–299. Springer, Heidelberg, April 2001.

[LRW11] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. *Journal of Cryptology*, 24(3):588–613, July 2011.

[MBKP11] Amir Moradi, Alessandro Barenghi, Timo Kasper, and Christof Paar. On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from Xilinx Virtex-II FPGAs. In Yan Chen, George Danezis, and Vitaly Shmatikov, editors, *ACM CCS 11*, pages 111–124. ACM Press, October 2011.

[Men13] Bart Mennink. Indifferentiability of double length compression functions. In Martijn Stam, editor, *14th IMA International Conference on Cryptography and Coding*, volume 8308 of *LNCS*, pages 232–251. Springer, Heidelberg, December 2013.

[Men17] Bart Mennink. Insuperability of the standard versus ideal model gap for tweakable blockcipher security. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 708–732. Springer, Heidelberg, August 2017.

[Min14] Kazuhiko Minematsu. Parallelizable rate-1 authenticated encryption from pseudorandom functions. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 275–292. Springer, Heidelberg, May 2014.

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

[MOSW15] Daniel P. Martin, Elisabeth Oswald, Martijn Stam, and Marcin Wójcik. A leakage resilient MAC. In Jens Groth, editor, *15th IMA International Conference on Cryptography and Coding*, volume 9496 of *LNCS*, pages 295–310. Springer, Heidelberg, December 2015.

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

[MSGR10] Marcel Medwed, François-Xavier Standaert, Johann Großschädl, and Francesco Regazzoni. Fresh re-keying: Security against side-channel and fault attacks for low-cost devices. In Daniel J. Bernstein and Tanja Lange, editors, *AFRICACRYPT 10*, volume 6055 of *LNCS*, pages 279–296. Springer, Heidelberg, May 2010.

[MV04] David A. McGrew and John Viega. The security and performance of the Galois/counter mode (GCM) of operation. In Anne Canteaut and Kapalee Viswanathan, editors, *INDOCRYPT 2004*, volume 3348 of *LNCS*, pages 343–355. Springer, Heidelberg, December 2004.

[NIS19] NIST. Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process, 2019. https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/final-lwc-submission-requirements-august2018.pdf.

[OC15] Colin O'Flynn and Zhizhang (David) Chen. Side channel power analysis of an AES-256 bootloader. In *IEEE 28th Canadian Conference on Electrical and Computer Engineering, CCECE 2015, Halifax, NS, Canada, May 3-6, 2015*, pages 750–755. IEEE, 2015.

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

[PS15] Thomas Peyrin and Yannick Seurin. Counter-in-Tweak: Authenticated Encryption Modes for Tweakable Block Ciphers. Version 20160524:153228. Cryptology ePrint Archive, Report 2015/1049, 2015.

[PSV15] Olivier Pereira, François-Xavier Standaert, and Srinivas Vivek. Leakage-resilient authentication and encryption from symmetric cryptographic primitives. In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15*, pages 96–108. ACM Press, October 2015.

[RBBK01] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In *ACM CCS 01*, pages 196–205. ACM Press, November 2001.

[Rog02] Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM CCS 02*, pages 98–107. ACM Press, November 2002.

[Rog04a] Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, 2004.

[Rog04b] Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In Pil Joong Lee, editor, *ASIACRYPT 2004*, volume 3329 of *LNCS*, pages 16–31. Springer, Heidelberg, December 2004.

[Rog04c] Phillip Rogaway. Nonce-based symmetric encryption. In Bimal K. Roy and Willi Meier, editors, *FSE 2004*, volume 3017 of *LNCS*, pages 348–359. Springer, Heidelberg, February 2004.

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

[SBK+17] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. The first collision for full SHA-1. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 570–596. Springer, Heidelberg, August 2017.

[SPY13] François-Xavier Standaert, Olivier Pereira, and Yu Yu. Leakage-resilient symmetric cryptography under empirically verifiable assumptions. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 335–352. Springer, Heidelberg, August 2013.

[ST16] Thomas Shrimpton and R. Seth Terashima. Salvaging weak security bounds for blockcipher-based constructions. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 429–454. Springer, Heidelberg, December 2016.

[Sta19] François-Xavier Standaert. Towards and open approach to secure cryptographic implementations (invited talk), EUROCRYPT 2019. pages xv, https://www.youtube.com/watch?v=KdhrsuJT1sE, 2019.

[Tes11] Stefano Tessaro. Security amplification for the cascade of arbitrarily weak PRPs: Tight bounds via the interactive hardcore lemma. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 37–54. Springer, Heidelberg, March 2011.

[Tes15] Stefano Tessaro. Optimally secure block ciphers from ideal primitives. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 437–462. Springer, Heidelberg, November / December 2015.

[VMKS12] Nicolas Veyrat-Charvillon, Marcel Medwed, Stéphanie Kerckhof, and François-Xavier Standaert. Shuffling against side-channel attacks: A comprehensive study with cautionary note. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 740–757. Springer, Heidelberg, December 2012.

[WHF03] Doug Whiting, Russell Housley, and Niels Ferguson. Counter with CBC-MAC (CCM). *RFC*, 3610:1–26, 2003.

[YSPY10] Yu Yu, François-Xavier Standaert, Olivier Pereira, and Moti Yung. Practical Leakage-Resilient Pseudorandom Generators. In *ACM CCS 2010*, pages 141–151, 2010.

[ZYSQ13] Yuanyuan Zhou, Yu Yu, François-Xavier Standaert, and Jean-Jacques Quisquater. On the need of physical security for small embedded devices: A case study with COMP128-1 implementations in SIM cards. In Ahmad-Reza Sadeghi, editor, *FC 2013*, volume 7859 of *LNCS*, pages 230–238. Springer, Heidelberg, April 2013.

Supplementary Material

Common belief arguing about the infeasibility to achieve nonce-misuse resistance with (non-restricted) leakage is given in Appendix A. Appendix B shows that XEX is not an ideal TBC. In the misuse-resilience black-box setting (i.e. without leakage), TEDT competes with standards and even outperforms them in the multi-user setting. Unfortunately, the details and the proofs of its optimal security must have been deferred to Appendix D due to space constraints. Eventually, Appendix C shows that even in the standard model TEDT keeps its leakage confidentiality. This final analysis simply gives as more confidence in the rationale of our mode.

A Full Nonce Robustness (Misuse-Resistance) is Hard to Achieve Facing Leakage

(Very informal) The hardness of realizing misuse-resistance in the leakage setting was argued from two different ideas due to Berti et al. [BKP+18] and Guo et al. [GPPS19a]. Below we elaborate in detail. We ignore the associated data A since it's irrelevant here. Concentrating on the re-keying designs, if nonce N is fixed in CPA attacks, then the initial state of the re-keying is fixed, and likely a same ephemeral key will appear in each encryption process. This allows the adversary to observe long-term information about this ephemeral key and recover it via SPA [BKP+18]. Such a "state-fixing" attack is always possible unless the AEAD scheme first derives a digest $D \leftarrow f(N, M)$ that depends on both N and M. But in the latter case, the scheme has to rely on: (i) a variable input length PRF f_K, which again faces the problem of "state-fixing" attacks, or (ii) a keyless hash function f (like that used in SIVAT [BMOS17]), which leaks easy-to-compare information about the message M and ruins confidentially (we stress that SIVAT was designed for applications *without challenge leakage*, so our observation **does not contradict** the security claim in [BMOS17]). The above "state-fixing" attack won't be effective on the leakage-resilient PRF/PRP-based DAE or MR AE schemes (such as those proposed by Barwell [BMOS17]): it's effective on re-keying designs just because the latter are expected to use PRF/PRPs that are weak w.r.t. leakage-resilience. But for these DAE schemes, another possibility for distinguishing occurs. Concretely, virtually all (D)AE perform actions on the message block-by-block. For example, in the SIV(-like) composition, message is absorbed block-by-block by a universal hash function (SCT [PS15] and ZAE [IMPS17] are two concrete examples); in sponge, message is absorbed block-by-block by a iterative process (e.g., SUNDAE [BBLT18]). In all, the prefix of the leakage $\mathsf{L}_{\mathsf{enc}}(m_1 \| m_2 \| \ldots \| m_\ell)$ only depends on m_1 (or m_1 and m_2, when Feistel networks are used). Therefore, using $\mathsf{L}_{\mathsf{enc}}(m_1 \| \ldots \| m_\ell)$ as a template, for $M^0 = m_1 \| \ldots$ and $M^1 = m'_1 \| \ldots$, $m'_1 \neq m_1$, it's easy to distinguish $\mathsf{L}_{\mathsf{enc}}(M^0)$ from $\mathsf{L}_{\mathsf{enc}}(M^1)$ and break the LOR game. In all, while there isn't any formal analysis, "full" robustness against nonce-misuse in the presence of challenge query leakage seems impossible. We further stress that: (1) The (informal) attacks described here **never contradict** the security claims in the mentioned papers [PS15, IMPS17, BBLT18, BMOS17]: the former three only claimed security *without leakage*, while the last [BMOS17] is only tailored to the setting *without challenge query leakage*; (2) The impossibility of "full" robustness against nonce-misuse with challenge query leakage isn't an artifact of the security model. Instead, it's *the reality*. It shows if challenge query leakage cannot be avoided, then one has to manage the nonce more carefully.

B XEX is not an Ideal TBC

We first recall the definition of XEX, which was introduced by Rogaway [Rog04b] as an instance of a construction of [LRW11]. XEX uses $\Delta = \mathsf{x}_1^{\alpha_1} \cdots \mathsf{x}_\ell^{\alpha_\ell} E_k(T)$ to mask the input and output of a classical block cipher E, where $\mathsf{x}_1, \ldots, \mathsf{x}_\ell \in \mathrm{GF}(2^n)$ are some pre-defined generators. Formally,

$$\mathrm{XEX}_k((\alpha_1, \ldots, \alpha_\ell, T), m) = E_k(m \oplus \Delta) \oplus \Delta.$$

If the generators and the tweak space are defined such that the $\mathsf{x}_1^{\alpha_1} \cdots \mathsf{x}_\ell^{\alpha_\ell}$ are unique and unequal to 1 for all tweaks, XEX achieves birthday bound security [Rog04b]. For example, practical applications with $n = 128$ often take $\ell \leq 3$ and $(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3) = (2, 3, 7)$, and an allowed tweak space would be $[1, 2^{n/2}] \times [0, 10] \times [0, 10] \times \{0, 1\}^n$.

Note that the security of XEX crucially relies on the secretness of the key k: otherwise, regardless of the choice of $\mathsf{x}_1, \ldots, \mathsf{x}_\ell$, the adversary could find $\alpha_1, \ldots, \alpha_\ell, T, \alpha'_1, \ldots, \alpha'_\ell, T'$ such that $\mathsf{x}_1^{\alpha_1} \cdots \mathsf{x}_\ell^{\alpha_\ell} E_k(T) = \mathsf{x}_1^{\alpha'_1} \cdots \mathsf{x}_\ell^{\alpha'_\ell} E_k(T')$ via querying E_k^{-1}, which then satisfy $\mathrm{XEX}_k((\alpha_1, \ldots, \alpha_\ell, T), m) = \mathrm{XEX}_k((\alpha'_1, \ldots, \alpha'_\ell, T'), m)$ for any m. By this, XEX is not secure in the known-key setting [KR07] and thus not ideal.

C An Analysis in the Standard Model

Compared to subsection 6.5, the analysis is even closer to the FEMALE proof in [GPPS19a]. To ease comparison, in this subsection we adopt the notations of [GPPS19a].

Assumptions. In this section, we model the TBC E as a *strong tweakable pseudorandom permutation*, defined as follows.

Definition 1 (Strong Tweakable Pseudorandom Permutation)**.** A function $\mathsf{E} : \mathcal{K} \times \mathcal{TW} \times \mathcal{M} \to \mathcal{M}$ is a $(q, t, \varepsilon_{\mathsf{E}})$-strong tweakable pseudorandom permutation (STPRP) for a security parameter n if, for all (q, t)-bounded adversaries $\mathcal{A}$, we have

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

where $\mathcal{TP}$ denotes the set of all tweakable permutations on $\mathcal{M}$ and with tweak space $\mathcal{TW}$ so that for any tweakable permutation P, and for any tweak tw, $P^{tw} = P(tw, \cdot)$ and $P^{tw,-1} = P^{-1}(tw, \cdot)$ are the inverse of each other.

Further, we assume the leakages are "recyclably simulatable", which was first introduced in [GPPS19a] and is defined below (adapted to TPRP setting) based on (p, q)-rsim-game in Table 2. We abbreviate it as (p, q)-recyclable-simulatability. This (p, q)-recyclable-simulatability assumption is an extension of the q-simulatability notion [SPY13], by allowing each of the q leakages in q-simulatability to be repeated p times.

Table 2: The (p, q)-rsim-game for TBC.

Game (p, q)-$\mathsf{rsim}(\mathcal{A}, \mathsf{E}, \mathsf{L}, \mathcal{S}, b)$.		
The challenger selects two random keys $k, k^* \xleftarrow{\$} \mathcal{K}$. The output of the game is a bit b' computed by $\mathcal{A}^{\mathsf{L}}$ based on the challenger responses to a total of at most q adversarial queries of the following type, each repeated at most p times:		
Query	Response if $b = 0$	Response if $b = 1$
$\mathsf{TEnc}(T, x)$	$\mathsf{E}_k^T(x), \mathsf{L}(k, T, x)$	$\mathsf{E}_k^T(x), \mathcal{S}^{\mathsf{L}}(k^*, T, x, \mathsf{E}_k(T, x))$
and one query of the following type, repeated at most p times:		
Query	Response if $b = 0$	Response if $b = 1$
$\mathsf{Gen}(k_{pre}, T_{pre}, x)$	$\mathcal{S}^{\mathsf{L}}(k_{pre}, T_{pre}, x, k)$	$\mathcal{S}^{\mathsf{L}}(k_{pre}, T_{pre}, x, k^*)$

Definition 2 ((p,q)-recyclable-simulatability of leakages)**.** Let E be a TPRP with L as its leakage function. Then the leakages of E are said to have $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,q)\text{-}rsim})$ (p,q)-recyclable-simulatability, if there exists a $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded simulator $\mathcal{S}^{\mathsf{L}}$ such that, for every (q_l, t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$ (making at most q_l queries to L and running in time t), we have

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

Throughout the remaining, we would simply call such leakages *R-simulatable*. It isn't hard to see (p,q)-recyclable-simulatability captures very similar SPA security setting as Eq. (15), the non-invertible leakage assumption. Please see [GPPS19a] for additional discussion on this assumption.

To the muCCAmL2 **Advantage.** We define

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT}}(\mathbf{q},t,\sigma)\stackrel{\text{def}}{=\!=}\max\left\{\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT}}(\mathcal{A})\right\},$$

where $\mathbf{q}=(q_m,q_e,q_d,p-1,q_l)$, and the maximal is taken over all $(\mathbf{q},t)$-bounded adversaries: that make q_m queries to the non-challenge encryption oracle, q_d queries to the decryption oracle, $p-1$ queries to $\mathsf{L}_{\mathsf{decch}}$, q_e queries to the challenge encryption oracle, and q_l queries to L, run in time t, and have at most σ blocks in its queries. For the formal analysis, we define the leakage function $\mathsf{L}=(\mathsf{L}_{\mathsf{enc}},\mathsf{L}_{\mathsf{dec}})$ of TEDT as:

- $\mathsf{L}_{\mathsf{enc}}$ consists of the follows that are generated during the encryption:
  - the leakages $\mathsf{L}_{\mathsf{E}}(s,T,x)$ generated by all the internal calls to $\mathsf{E}_s(T,x)$, and
  - the leakages $\mathsf{L}_{\oplus}(a,b)$ generated by all the internal actions $a\oplus b$, and
  - all the intermediate values involved in the computations of the hash functions.
- $\mathsf{L}_{\mathsf{dec}}$ consists of the above that are generated during the decryption.

Our security reduction is made against (i) the simulatability of the leaking blocks, (ii) the security of the encryption of one single block with a fresh key. In detail, following Pereira et al.'s approach [PSV15], we consider a Leaking Real Single-block Encryption scheme LRSE defined in Fig. 11 as the basic unit of TEDT, and as the leakage confidentiality assumption for TEDT in the standard model. Since for each generated key k_{ch} LRSE will be used to encrypt a single message m composed of a single block, we assume that given a security parameter n, LRSE is $(p,q_l,t,\varepsilon_{s\text{-}block})$ secure in the following sense: for any (q_l,t)-bounded eavesdropper adversary $\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}$ choosing $T,p_A,p_B,m^0,m^1\in\{0,1\}^n$ with $p_A\neq p_B$, it holds

$$\left|\Pr[\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}(\mathsf{LRSEnc}^{+}_{k_{ch}}(m^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}(\mathsf{LRSEnc}^{+}_{k_{ch}}(m^1))\Rightarrow 1]\right|\leq\varepsilon_{s\text{-}block},\tag{29}$$

where $\mathsf{LRSEnc}^{+}_{k_{ch}}(m^b)=(\mathsf{LRSEnc}_{k_{ch}}(m^b),[\mathsf{L}_{\mathsf{rsdec}}(k_{ch},c^b)]^{p-1},k_{pre})$ for the pair of outputs $(c^b,k_{up})=\mathsf{RSEnc}_{k_{ch}}(m^b)$. The reason why the adversary also gets the auxiliary outputs k_{pre} and k_{up} is for composability purpose which will be apparent in the proof. Similar to Eq. (15), $\varepsilon_{s\text{-}block}$ may not be negligible.

Based on the assumption (29), the muCCAmL2 advantage bound could be established below.

Theorem 3. *Let* $\mathsf{E}:\{0,1\}^n\times\{0,1\}^n\times\{0,1\}^n\to\{0,1\}^n$ *be a* $(2q_e+2q_d+2q_m,t',\varepsilon_{\mathsf{E}^*})$*-STPRP. Assume that* E *has two implementations: a strongly protected implementation* E^* *underlying* KDF *and* TGF *is leak free, and a plain implementation* E *have leakage function* L_{E} *that is* $(q_{\mathcal{S}},t_{\mathcal{S}},q_l,t,\varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$*-R-simulatable. Then the* TEDT *implementation with leakage function* $\mathsf{L}=(\mathsf{L}_{\mathsf{enc}},\mathsf{L}_{\mathsf{dec}})$ *defined before has*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT}}(\mathbf{q},t,\sigma)\leq&2(q_e+q_d+q_m)\varepsilon_{\mathsf{E}}+\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT}}(q_e+q_d+q_m,t')\\&+4\sigma(\varepsilon_{\mathsf{E}}+\varepsilon_{(p,2)\text{-}rsim})+\sigma\cdot\varepsilon_{s\text{-}block},\end{aligned}\tag{30}$$

Description of LRSE scheme: (tool for the proof and for capturing the confidentiality advantage)

$\mathsf{RSGen}(1^n)$ picks $k_{ch} \xleftarrow{\$} \{0,1\}^n$, $\mathcal{M}, \mathcal{C} = \{0,1\}^n$ $(T, p_A, p_B \in \{0,1\}^n,\ p_A \neq p_B)$

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

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

The leakage $\mathsf{L}_{\mathsf{RSE}} = (\mathsf{L}_{\mathsf{rsenc}}, \mathsf{L}_{\mathsf{rsdec}}, k_{pre})$ resulting from the LRSE implementation is defined as

- $\mathsf{L}_{\mathsf{rsenc}}(k_{ch}, m) = (\mathsf{L}_{\mathsf{E}}(k_{ch}, T, p_A), \mathsf{L}_{\mathsf{E}}(k_{ch}, T, p_B), \mathsf{L}_{\oplus}(y_{ch}, m), \mathcal{S}^{\mathsf{L}}(k_{pre}, T, p_A, k_{ch}))$,
- $\mathsf{L}_{\mathsf{rsdec}}(k_{ch}, c) = (\mathsf{L}_{\mathsf{E}}(k_{ch}, T, p_A), \mathsf{L}_{\mathsf{E}}(k_{ch}, T, p_B), \mathsf{L}_{\oplus}(y_{ch}, c), \mathcal{S}^{\mathsf{L}}(k_{pre}, T, p_A, k_{ch}))$

for a fixed random $k_{pre} \xleftarrow{\$} \{0,1\}^n$. As usual we denote $\mathsf{LRSEnc}_{k_{ch}}(m) = (\mathsf{RSEnc}_{k_{ch}}(m), \mathsf{L}_{\mathsf{rsenc}}(k_{ch}, T, p_A, p_B, m))$.

Figure 11: Basic unit: the single-block encryption scheme LRSE.

where σ is the number of blocks in the challenge messages and $\varepsilon_{s\text{-}block}$ is as defined in Eq. (29). Here $t' = t + (q_e + q_d + q_m)(t_{\$} + t_{1-pass})$, t_{1-pass} is the maximum running time of TEDT upon a single (encryption or decryption) query, and $t_{\$}$ is the time needed for randomly sampling a value from $\{0,1\}^n$.

As proved in section 6, the term $\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT}}(q_e + q_d + q_m, t_{\mathcal{A}^{\mathsf{L}'}})$ is likely close to optimal. The other terms are of some birthday type and are comparable to Theorem 2. In some sense, this result can also be seen as *domain extension of the single-block encryption* LRSE. It's worth noting that by setting $\varepsilon_{(p,2)\text{-}rsim} = 0$ and $\varepsilon_{s\text{-}block} = 0$ we obtain a $\mathsf{muCCAm\$}$ bound in the standard model:

$$\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathsf{TEDT}}(q_m, q_e, q_d, t, \sigma) \leq 2(q_e + q_d + q_m + 2\sigma)\varepsilon_{\mathsf{E}} + \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT}}(q_e + q_d + q_m, t').$$

This concrete bound is weak. It doesn't reflect the influence of key collisions. Also it suffers from the (unreal) "hybrid security loss" (as discussed subsequently to Theorem 2). On the positive side, the above two bounds show that when the TBC isn't ideal, TEDT still enjoys (a weaker) provable security.

The proof flow for Theorem 3 is very similar to Section 6.5. We first prove that based on the pseudorandomness of E and the simulatability of the leakage, the "real-leaking world" $(\mathsf{E}^T_k(p), \mathsf{L}(k, T, p))$ is indistinguishable from the "ideal-simulating" world $(\$, \mathcal{S}^{\mathsf{L}}(k, T, p, \$))$. This is actually a lemma of Guo et al. [GPPS19a] adapted to our TPRP setting.

Lemma 9 (Indistinguishability of Real-Leak and Ideal-Simulation)**.** *Let $\mathsf{E} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\mathsf{E}})$-TPRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every $k_{pre}, T, p_A, p_B, z \in \{0,1\}^n$ and every $(q_l - q^*, t - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$, the following holds:*

$$\begin{aligned}&\big|\Pr[k_{ch} \xleftarrow{\$} \{0,1\}^n : \mathcal{D}^{\mathsf{L}}(\mathsf{E}^T_{k_{ch}}(p_A), \mathsf{E}^T_{k_{ch}}(p_B), [\mathsf{L}_{\mathsf{E}}(k_{ch}, T, p_A), \mathsf{L}_{\mathsf{E}}(k_{ch}, T, p_B), \mathcal{S}^{\mathsf{L}}(k_{pre}, T, z, k_{ch})]^p) \Rightarrow 1]\\ &- \Pr[k_{ch}, c_A, c_B \xleftarrow{\$} \{0,1\}^n, c_A \neq c_B :\\ &\qquad \mathcal{D}^{\mathsf{L}}(c_A, c_B, [\mathcal{S}^{\mathsf{L}}(k_{ch}, T, p_A, c_A), \mathcal{S}^{\mathsf{L}}(k_{ch}, T, p_B, c_B), \mathcal{S}^{\mathsf{L}}(k_{pre}, T, z, k_{ch})]^p) \Rightarrow 1]\big| \leq \varepsilon_{\mathsf{E}} + \varepsilon_{(p,2)\text{-}rsim}.\end{aligned}$$

Here $q^* = 3p \cdot q_{\mathcal{S}}$, while $t^* = Max\{t_r, t_{sim}\}$, in which t_r is equal to $3p \cdot t_{\mathcal{S}}$ augmented with the time needed to make 2 oracle queries to the TPRP challenger and select a uniformly

Description of LISE: (tool for the proof)

$\mathbf{ISGen}(\mathbf{1}^{n})$ picks $k_{ch} \xleftarrow{\$} \{0,1\}^n$, $\mathcal{M}, \mathcal{C} = \{0,1\}^n$ $(T, p_A, p_B \in \{0,1\}^n,\ p_A \neq p_B)$

$\mathbf{ISEnc}_{k_{ch}}(m)$ returns (k_{up}, c), where $c = y_{ch} \oplus m$, and $k_{up}, y_{ch} \xleftarrow{\$} \{0,1\}^n$, $k_{up} \neq y_{ch}$.

$\mathbf{ISDec}_{k_{ch}}(c)$ proceeds in the natural way.

The leakage $\mathsf{L}_{\mathsf{ISE}} = (\mathsf{L}_{\mathsf{isenc}}, \mathsf{L}_{\mathsf{isdec}}, k_{pre})$ resulting from the LISE implementation is defined as $\mathsf{L}_{\mathsf{isenc}}(k_{ch}, m) = (\mathcal{S}^{\mathsf{L}}(k_{ch}, T, p_A, k_{up}), \mathcal{S}^{\mathsf{L}}(k_{ch}, T, p_B, y_{ch}), \mathsf{L}_{\oplus}(y_{ch}, m), \mathcal{S}^{\mathsf{L}}(k_{pre}, T, p_A, k_{ch}))$, $\mathsf{L}_{\mathsf{isdec}}(k_{ch}, c) = (\mathcal{S}^{\mathsf{L}}(k_{ch}, T, p_A, k_{up}), \mathcal{S}^{\mathsf{L}}(k_{ch}, T, p_B, y_{ch}), \mathsf{L}_{\oplus}(y_{ch}, c), \mathcal{S}^{\mathsf{L}}(k_{pre}, T, p_A, k_{ch}))$ for a fixed random $k_{pre} \xleftarrow{\$} \{0,1\}^n$.

Figure 12: The ideal single-block encryption scheme ISEnc.

random key in $\{0,1\}^n$, and t_{sim} is the time needed to relay the content of $2p$ TEnc and p Gen queries from and to a $(p,2)$-rsim challenger.

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

We then define the tweakable variant of the "single-block scheme" in [GPPS19a]: roughly, all the intermediate values are replaced by random and all the leakages are replaced by simulation. The resulted algorithm is in Fig. 12. We also define $\mathsf{LISEnc}^{+}_{k_{ch}}(m) = (\mathsf{LISEnc}_{k_{ch}}(m), [\mathsf{L}_{\mathsf{isdec}}(k_{ch}, c)]^{p-1}, k_{pre})$ for $(c, k_{up}) = \mathsf{ISEnc}_{k_{ch}}(m)$.

Lemma 10 (Indistinguishability of ISEnc and RSEnc)**.** *Let $\mathsf{E} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\mathsf{E}})$-TPRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every $T, p_A, p_B \in \{0,1\}^n$, $p_A \neq p_B$, and every $(q_l - q^*, t - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$, the following holds:*

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

Here $q^* = 3p \cdot q_{\mathcal{S}} + p$, while $t^* = Max\{t_r, t_{sim}\}$, in which t_r is equal to $3p \cdot t_{\mathcal{S}} + 2t_{\oplus}$ augmented with the time needed to make 2 oracle queries to the TPRP challenger and select a uniformly random key in $\{0,1\}^n$, $t_{\oplus}$ is the time needed to evaluate the $\oplus$ action on an n-bit input, and t_{sim} is the time needed to relay the content of four TEnc and two Gen queries from and to a $(p,2)$-rsim challenger.

Proof. Follows the same line as Lemma 9. □

We then define (L)$\mathsf{TStream}$ and (L)$\mathsf{IStream}$ in the standard model in Fig. 13 and prove their indistinguishability.

Lemma 11 (Indistinguishability of $\mathsf{LTStream}$ and $\mathsf{LIStream}$)**.** *Let $\mathsf{E} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\mathsf{E}})$-TPRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}}, t_{\mathcal{S}}, q_l, t, \varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}}, t_{\mathcal{S}})$-bounded leakage simulator. Then, for every ℓ-block message m, every T, N, p_A, p_B, and every $(q_l - 2q_r - q^*, t - 2t_r - t^*)$-bounded distinguisher $\mathcal{D}^{\mathsf{L}}$, the following holds:*

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

Here $q_r = \ell(2q_{\mathcal{S}} + 3)$, q^* and t^* are as defined in Lemma 10, and $t_r = 2\ell(t_{\mathcal{S}} + t_{\$} + t_{\mathsf{E}}) + \ell \cdot t_{\oplus}$, where t_{E} is the time needed for evaluating E once, $t_{\$}$ is the time needed for randomly sampling a value from $\{0,1\}^n$, and $t_{\oplus}$ is the time needed for evaluating $\oplus$ once.

Description of $\mathsf{TStream}$:

- Gen picks $k_0 \xleftarrow{\$} \{0,1\}^n$
- $\mathsf{TStream}_{k_0}(T, N, m_1, \ldots, m_\ell)$ proceeds in two steps:
  (1) Initializes an empty list leak for the leakage;
  (2) for $i = 1, \ldots, \ell$, computes $k_i \leftarrow \mathsf{E}_{k_{i-1}}(T, C_i(N))$, $y_i \leftarrow \mathsf{E}_{k_{i-1}}(T, D_{i-1}(N))$, and $c_i \leftarrow y_i \oplus m_i$, and adds $[\mathsf{L}_\mathsf{E}(k_{i-1}, T, C_i(N)), \mathsf{L}_\mathsf{E}(k_{i-1}, T, D_{i-1}(N))]^p$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathsf{L}_\oplus(y_i, c_i)]^{p-1}$ to the list leak.

  $\mathsf{TStream}_{k_0}(T, N, m_1, \ldots, m_\ell)$ eventually returns $(c_1, \ldots, c_\ell)$. And we define $\mathsf{LTStream}_{k_0}(T, N, m) = (\mathsf{TStream}_{k_0}(T, N, m), \mathsf{leak})$ for the list leak standing at the end of the above process.

Description of $\mathsf{IStream}$:

- $\mathsf{IStream}_{k_0}(T, N, m_1, \ldots, m_\ell)$ proceeds in two steps:
  (1) Initializes an empty list leak for the leakage;
  (2) for $i = 1, \ldots, \ell$, samples $k_i \xleftarrow{\$} \{0,1\}^n$ and $y_i \xleftarrow{\$} \{0,1\}^n$ such that $k_i \neq y_i$, sets $c_i \leftarrow y_i \oplus m_i$, and adds $[\mathcal{S}^\mathsf{L}(k_{i-1}, T, C_i(N), k_i), \mathcal{S}^\mathsf{L}(k_{i-1}, T, D_{i-1}(N), y_i)]^p$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathsf{L}_\oplus(y_i, c_i)]^{p-1}$ to the list leak.

  $\mathsf{IStream}_{k_0}(T, N, m_1, \ldots, m_\ell)$ eventually returns $(c_1, \ldots, c_\ell)$. And we define $\mathsf{LIStream}_{k_0}(m) = (\mathsf{IESM}_{k_0}(m), \mathsf{leak})$ for the list leak standing at the end of the above process.

Figure 13: The $\mathsf{TStream}$ and $\mathsf{IStream}$ scheme.

Proof. We define G_0 as the security game in which $\mathcal{A}^\mathsf{L}$ receives $\mathsf{TStream}_{k_0}(T, N, m)$ as the input, and G_ℓ as the game in which $\mathcal{A}^\mathsf{L}$ receives $\mathsf{IStream}_{k_0}(T, N, m)$ as the input. We show that G_0 could be transited to G_ℓ via a sequence of games $\mathsf{G}_1, \mathsf{G}_2, \ldots, \mathsf{G}_{\ell-1}$. In detail, for i from 1 to ℓ, we consider the game G_{i-1}: we replace the two intermediate values $\mathsf{E}^T_{k_{i-1}}(C_i(N))$ and $\mathsf{E}^T_{k_{i-1}}(D_{i-1}(N))$ by two distinct random values k_i and y_i, and replace the leakages $[\mathsf{L}_\mathsf{E}(k_{i-1}, T, C_i(N)), \mathsf{L}_\mathsf{E}(k_{i-1}, T, D_{i-1}(N))]^p$, $\mathsf{L}_\oplus(\mathsf{E}^T_{k_{i-1}}(D_{i-1}(N)), m_i)$, and $[\mathsf{L}_\oplus(\mathsf{E}^T_{k_{i-1}}(D_{i-1}(N)), c_i)]^{p-1}$ by the simulated $[\mathcal{S}^\mathsf{L}(k_{i-1}, T, C_i(N), k_i), \mathcal{S}^\mathsf{L}(k_{i-1}, T, D_{i-1}(N), y_i)]^p$, $\mathsf{L}_\oplus(y_i, m_i)$, and $[\mathsf{L}_\oplus(y_i, c_i)]^{p-1}$. This yields the game G_i.

We next derive an upper bound for $|\Pr[(\mathcal{D}^\mathsf{L})^{\mathsf{G}_i} \Rightarrow 1] - \Pr[(\mathcal{D}^\mathsf{L})^{\mathsf{G}_{i-1}} \Rightarrow 1]|$. For this, we assume a $(q_l - p \cdot q_r - q^*, t - p \cdot t_r - t^*)$-bounded distinguisher $\mathcal{D}^\mathsf{L}$ against G_i and G_{i-1}, and we build a distinguisher $\mathcal{D}^{\mathsf{L}'}$ against the real-leaking and the ideal-simulation-world. Assume $\mathcal{D}^{\mathsf{L}'}$ receives $(c_A, c_B, [\mathsf{leak}_1, \mathsf{leak}_2, \mathcal{S}^\mathsf{L}(k_{pre}, T, C_{i-1}(N), k_{ch})]^p, k_{pre})$ as inputs, with $c_A \neq c_B$. $\mathcal{D}^{\mathsf{L}'}$ proceeds in six steps:

(1) $\mathcal{D}^{\mathsf{L}'}$ first uniformly samples k_0;

(2) For $j = 1, \ldots, i-3$, $\mathcal{D}^{\mathsf{L}'}$ uniformly samples 2 random values $k_j \neq y_j$, simulates the traces $[\mathcal{S}^\mathsf{L}(k_{j-1}, T, C_j(N), k_j), \mathcal{S}^\mathsf{L}(k_{j-1}, T, D_{j-1}(N), y_j)]^p$, computes $c_j \leftarrow y_j \oplus m_j$ and $m_j \leftarrow y_j \oplus c_j$ and obtains the traces $\mathsf{L}_\oplus(y_j, m_j)$ and $[\mathsf{L}_\oplus(y_j, c_j)]^{p-1}$;

(3) $\mathcal{D}^{\mathsf{L}'}$ then sets $y_{i-2} \leftarrow k_{pre}$, samples $y_{i-2} \neq k_{pre}$, obtains $[\mathcal{S}^\mathsf{L}(k_{i-3}, T, D_{i-3}(N), y_{i-2})]^p$, computes $c_{i-2} \leftarrow y_{i-2} \oplus m_{i-2}$ and $m_{i-2} \leftarrow y_{i-2} \oplus c_{i-2}$ and obtains the traces $\mathsf{L}_\oplus(y_{i-2}, m_{i-2})$ and $[\mathsf{L}_\oplus(y_{i-2}, c_{i-2})]^{p-1}$. These along with $[\mathcal{S}^\mathsf{L}(k_{pre}, T, C_{i-1}(N), k_{ch})]^p$ are used as the leakages of the $i-2$ th iteration;

(4) Then it uniformly samples y_{i-1}, computes $c_{i-1} \leftarrow y_{i-1} \oplus m_{i-1}$, $m_{i-1} \leftarrow y_{i-1} \oplus c_{i-1}$, and uses $[\mathcal{S}^\mathsf{L}(k_{i-2}, T, C_{i-1}(N), k_{ch}), \mathcal{S}^\mathsf{L}(k_{i-2}, T, D_{i-2}(N), y_{i-1})]^p$, $\mathsf{L}_\oplus(y_{i-1}, m_{i-1})$, $[\mathsf{L}_\oplus(y_{i-1}, c_{i-1})]^{p-1}$ as the traces of the $i-1$ th iteration in the first pass;

(5) Sets $k_i \leftarrow c_A$ and $y_i \leftarrow c_B$, computes $c_i \leftarrow y_i \oplus m_i$, and uses $[\mathsf{leak}_1, \mathsf{leak}_2]^p$, $\mathsf{L}_\oplus(y_i, m_i)$, $[\mathsf{L}_\oplus(y_i, c_i)]^{p-1}$ as the corresponding leakages;

(6) Takes k_i as the starting point and emulates the remaining part of the execution of $\mathsf{TStream}$ encryption. Eventually, $\mathcal{D}^{\mathsf{L}'}$ serves the obtained ciphertext $c_1\|\ldots\|c_\ell$ as well as the leakage traces to $\mathcal{D}^{\mathsf{L}}$, and outputs whatever $\mathcal{D}^{\mathsf{L}}$ outputs.

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

We further show that to perform the additional operations, $\mathcal{D}^{\mathsf{L}'}$ makes at most $p \cdot s_r$ additional queries to L and spend $p \cdot t_r$ additional time. To this end, we note that the encryption process of $\mathsf{TStream}$ involves $2\ell - 1$ calls to E and ℓ xor operations. Moreover, to emulate the "hybrid" encryption process once, $\mathcal{D}^{\mathsf{L}'}$ needs at most $(2\ell-1)(q_{\mathcal{S}}+1)+\ell = q_r$ queries to L and $(2\ell-1)(t_{\mathsf{E}}+t_{\$}+t_{\mathcal{S}})+\ell\cdot t_{\oplus} = t_r$ running time. To obtain the required decryption leakage traces, $\mathcal{D}^{\mathsf{L}'}$ has to additionally perform the "hybrid" *decryption* process for $p-1$ times, which contributes to $(p-1)q_r$ more queries and $(p-1)t_r$ more time. Therefore, as claimed, $\mathcal{D}^{\mathsf{L}'}$ makes at most $p \cdot q_r$ additional queries to L and spends $p \cdot t_r$ additional time for the additional operations. By the above and Lemma 9, we have

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

Therefore, the ℓ transitions yield

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

in total. □

Then is the standard model version of Lemma 7. Unlike Lemma 7, there is no query to the ideal TBC here, and thus the notation q_{IC} disappears in $\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p,t,\ell)$.

Lemma 12 (1-Block Advantage to ℓ-Block)**.** *For every pair of ℓ-block messages m^0 and m^1 and (q_l,t)-bounded adversary $\mathcal{A}^{\mathsf{L}}$, there exists a $(q_l+2q_r, t+2t_r)$-bounded adversary $\mathcal{A}^{\mathsf{L}'}$ such that*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p,t,\ell) :=& \left|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IStream}_{k_0}(T,N,m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IStream}_{k_0}(T,N,m^1)) \Rightarrow 1]\right| \\ \leq& \sum_{i=1}^{\ell} \left|\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^{+}_{k_{i-1}}(m_i^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^{+}_{k_{i-1}}(m_i^1)) \Rightarrow 1]\right|,\end{aligned}$$

where $k_0,\ldots,k_{\ell-1}$ are chosen uniformly at random, and m_i^0 and m_i^1 are the i-th block of m^0 and m^1 respectively. Here $q_r = \ell(2q_{\mathcal{S}}+1)$ and $t_r = \ell(2t_{\mathcal{S}}+2t_{\$}+t_{\oplus})$, where t_{E}, $t_{\$}$, and $t_{\oplus}$ are as assumed in Lemma 11.

Proof. The proof follows the same line as Lemma 7. □

Gathering Lemmas 10, 11, and 12, and following the same line as Lemma 8 we obtain

Lemma 13. *Let $\mathsf{E}:\{0,1\}^n\times\{0,1\}^n\times\{0,1\}^n\to\{0,1\}^n$ be a $(2,t,\varepsilon_{\mathsf{E}})$-TPRP, whose implementation has a leakage function L_{E} having $(q_{\mathcal{S}},t_{\mathcal{S}},q_l,t,\varepsilon_{(p,2)\text{-}rsim})$ $(p,2)$-R-simulatable leakages, and let $\mathcal{S}^{\mathsf{L}}$ be an appropriate $(q_{\mathcal{S}},t_{\mathcal{S}})$-bounded leakage simulator. Then, for every pair of ℓ-block messages m^0 and m^1 and $(q_l-2q_r-q^*, t-2t_r-t^*)$-bounded adversary $\mathcal{A}^{\mathsf{L}}$, it holds*

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

where q_r, t_r are as defined in Lemma 11, and q^*, t^* are as defined in Lemma 10.

Completing the muCCAmL2 Proof. We start by defining G_0 as the game $\mathsf{PrivK}^{\mathsf{muCCAmL2},0}_{\mathcal{A}^{\mathsf{L}},\mathsf{TEDT}}$, and G^*_0 as the game $\mathsf{PrivK}^{\mathsf{muCCAmL2},1}_{\mathcal{A}^{\mathsf{L}},\mathsf{TEDT}}$. From G_0 we obtain G_1 by replacing the u instances of $\mathsf{E}^*_{K_1},\ldots,\mathsf{E}^*_{K_u}$ invoked during the execution by u independent tweakable random permutations $\widetilde{P}_1,\ldots,\widetilde{P}_u$. For each such K_i it's a quite standard trick to build a $(2q_e+2q_d+2q_m,t_{\mathcal{B}})$-bounded adversary $\mathcal{B}_{STPRP}$ against the STPRP security of E^*, with $t_{\mathcal{B}} \le t+(q_e+q_d+q_m)t_{1-pass} \le t'$. The number of such K_i involved in the execution is $\le q_e+q_d+q_m$, thus

$$|\Pr[\mathsf{G}_1 \Rightarrow 1]-\Pr[\mathsf{G}_0 \Rightarrow 1]| \le (q_e+q_d+q_m)\varepsilon_{\mathsf{E}^*}$$

follows from the assumption that E^* is a $(2q_e+2q_d+2q_m,t',\varepsilon_{\mathsf{E}^*})$-secure STPRP. Similarly, we replace $\mathsf{E}^*_{K_1},\ldots,\mathsf{E}^*_{K_u}$ by $\widetilde{P}_1,\ldots,\widetilde{P}_u$ to turn G^*_0 into G^*_1, which also introduces a gap of $(q_e+q_d+q_m)\varepsilon_{\mathsf{E}^*}$.

Then, following the proof of Eq. (27), the gap between G_1 and G^*_1 can be bounded:

$$\begin{aligned}&\left|\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_1}\Rightarrow 1]-\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}^*_1}\Rightarrow 1]\right| \\ \le&\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT}}(q_e+q_d+q_m,t')+\frac{q_e+q_m-1}{2^n}+\underbrace{\sum_{i=1}^{q_m}\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{TStream}}(p,t,\ell)}_{\le 4\sigma(\varepsilon_{\mathsf{E}}+\varepsilon_{(2,2)\text{-}rsim})+\sigma\cdot\varepsilon_{s\text{-}block}\ \text{(by Lemma 13)}},\end{aligned}$$

with ℓ_i the number of blocks in the ith challenge message. This plus the above gap $2(q_e+q_d+q_m)\varepsilon_{\mathsf{E}^*}$ yield the final claim.

D Black-box CCA Security of TEDT

In this section, we prove asymptotically optimal CCA security (without leakage) bounds for TEDT. We first present the security model in subsection D.1. The result and proof are later given in subsection D.2.

D.1 CCA Security with Misuse-Resilience

Ashur et al. [ADL17] 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 *misuse-resistance* [RS06] we will not refer to *misuse-resilience* with its initials but as CCAm$ since it 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. Formally, given a nonce-based authenticated encryption $\mathsf{AEAD}=(\mathsf{Enc},\mathsf{Dec})$, the *multi-user chosen ciphertext misuse resilience advantage* of an adversary $\mathcal{A}$ against AEAD with u users is

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

where the probability is taken over the u user keys $\mathbf{K}=(K_1,\ldots,K_u)$, with $K_i \leftarrow \mathcal{K}$, over $\mathcal{A}$'s random tape and the ideal TBC 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; $\mathsf{Dec}(i,N,A,C)$ outputs $\mathsf{Dec}_{K_i}(N,A,C)$; $\perp(i,N,A,C)$ outputs $\perp$; for each user i: (i) nonce N cannot be used both in query to $O_1(i,N,*,*)$ and $O_2(i,N,*,*)$; (ii) $O_2(i,*,*,*)$ is nonce-respecting; (iii) if C is returned by $O_1(i,N,A,M)$ or $O_2(i,N,A,M)$ query $O_3(i,N,A,C)$ is forbidden; (iv) a nonce used twice with O_1 cannot be used for an

O_3 query. The *extended* muCCAm$ advantage muCCAm$* is defined as the muCCAm$ one where the last restriction *(iv)* is waived from the definition. We stress that all these conditions were already introduced in [ADL17], mainly to enable standard to fall into misuse-resilient AEAD.

D.2 muCCAm$ Security of TEDT

To simplify the notations, we define

$$\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathsf{TEDT}}(u,\mathbf{q},\sigma) \stackrel{\text{def}}{=\!=} \max\left\{\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathsf{TEDT},\mathcal{A},u}\right\},$$

where $\mathbf{q}=(q_m,q_e,q_d,q_{\mathsf{IC}})$, and the maximum is taken over all $(\mathbf{q},t)$-bounded adversaries against u users that have at most σ blocks in all their queried plaintext (both challenge and non-challenge) and ciphertext including AD.

Theorem 4. *When the u public-keys $T_1,\ldots,T_u$ are uniformly distributed, $n\geq 6$, and $2\sigma+3(q_e+q_d)+q_{\mathsf{IC}}\leq 2^n/8$, then the following holds in the ideal TBC model:*

$$\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathsf{TEDT}}(u,\mathbf{q},\sigma) \leq \frac{2u^2}{2^{2n}}+\frac{1}{n!}\cdot\left(\frac{4u}{2^n}\right)^n+\frac{(7n^2+26)(4\sigma+6(q_e+q_d)+2q_{\mathsf{IC}})}{2^n}.$$

Partly thanks to the GCM-like counters the muCCAm$* bound is asymptotically optimal $O\Big(\frac{u^2}{2^{2n}}+\frac{n^2\sigma+n^2q_{\tilde{\mathsf{IC}}}}{2^n}\Big)$. Moreover, the terms $\frac{2u^2}{2^{2n}}$ and $\frac{1}{n!}\cdot(4u/2^n)^n$ disappear if the u public-keys are distinct. While the bound seems not to depend on q_m, the non-challenge queries actually affect σ which, in turn, affects the bound. The bound of GCM-SIV with KDF is at best $\frac{\sigma\ell_{max}+q_{\tilde{\mathsf{IC}}}}{2^n}$, which we surpass when the maximal query length ℓ_{max} exceeds n^2: see Appendix D.3 for the details.

In the remaining of this subsection we prove Theorem 4. We rely on a balls-in-bin lemma from [PS15] presented as follows.

Lemma 14 (Balls-in-Bin)**.** *Consider throwing a ball into a bin that is chosen independently uniformly at random from $2^n\geq 8$ bins. Then the probability that, after throwing σ balls with $8\leq\sigma\leq 2^n$, any bin contains $2\log_2\sigma$ balls or more, is less than $\frac{1}{2^n}$.*

Proof. See [PS15, Appendix A]. That proof covered more general cases which we don't rely on. □

Then, note that in the misuse resilience setting, schemes which achieve both CPA confidentiality and authenticity also achieve CCA confidentiality [ADL17]:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathcal{A},\mathsf{AEAD},u} &= \Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Dec}_{\mathbf{K},\mathbf{T}},\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1\big]-\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\$,\perp,\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1\big]\Big|\\ &\leq \underbrace{\Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Dec}_{\mathbf{K},\mathbf{T}},\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1\big]-\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\perp,\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1\big]\Big|}_{\mathbf{Adv}^{\mathsf{mu\text{-}INT\text{-}CTXT}}_{\mathcal{A},\mathsf{AEAD},u}:\ \mathsf{mu\ INT\text{-}CTXT}\text{ advantage of }\mathcal{A}\text{ on AEAD}}\\ &\quad+\underbrace{\Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\perp,\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1\big]-\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\$,\perp,\mathsf{IC},\mathsf{IC}^{-1}}\Rightarrow 1\big]\Big|}_{\text{defined as }\mathbf{Adv}^{\mathsf{muCPAm\$^*}}_{\mathcal{A},\mathsf{AEAD},u}}. \quad (31)\end{aligned}$$

Clearly, $\mathbf{Adv}^{\mathsf{mu\text{-}INT\text{-}CTXT}}_{\mathcal{A},\mathsf{TEDT},u}\leq\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{A},\mathsf{TEDT},u}$. Therefore, we focus on the CPA advantage $\mathbf{Adv}^{\mathsf{muCPAm\$^*}}_{\mathcal{A},\mathsf{TEDT},u}$—switching to the CPA setting greatly simply the setting as well as the notations. Again we employ the H-coefficient technique, and present the two steps in two subsequent subsections.

D.2.1 Bad Transcripts

Following section 5.3, during the interaction, we also reveal some underlying IC queries to $\mathcal{D}$ and include them in the transcript. In detail,

- First, we reveal all the IC queries underlying the non-challenge encryption queries (i.e., queries to the first encryption oracle);
- Second, for the challenge encryption queries, we emulate the corresponding hash evaluations, and reveal all the induced IC queries.

We merge these queries with the adversarial queries to obtain a set τ^*_{IC}. Thus we have $\left|\tau^*_{\mathsf{IC}}\right| \leq q^*_{\mathsf{IC}}$ which is as defined by Eq. (7). Note that this upper bound is a bit coarse, but it's enough for the remaining argument.

We also organize the hash query transcript τ^*_{H}. Besides, the set

$$\tau_e = \left((u^{(1)}, N^{(1)}, A^{(1)}, M^{(1)}, C^{(1)}), \ldots, (u^{(q_e)}, N^{(q_e)}, A^{(q_e)}, M^{(q_e)}, C^{(q_e)})\right)$$

summarizes the queries to the challenge (second) encryption oracle. Recall that we've switched to the CPA setting, so these are "enough": transcripts are defined as

$$\tau = (\tau^*_{\mathsf{H}}, \tau_e, \tau^*_{\mathsf{IC}}, \mathbf{T}, \mathbf{K}).$$

For an encryption query $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)})$, we denote

$$M^{(i)} = m_1^{(i)} \| \ldots \| m_{\ell_i}^{(i)}$$

and

$$C^{(i)} = \mathbf{c}^{(i)} \| Z^{(i)} = c_1^{(i)} \| \ldots \| c_{\ell_i}^{(i)} \| Z^{(i)},$$

i.e., $m_j^{(i)}$, resp. $c_j^{(i)}$, denotes the j th n-bit block of $M^{(i)}$, resp. $C^{(i)}$. Wlog assume that $|m_j^{(i)}| = n$ for any block. And we define an auxiliary quantity

$$\mu_Y := \max_{y \in \{0,1\}^n} \left|\{(i,j) : m_j^{(i)} \oplus c_j^{(i)} = y\}\right|.$$

Note that in the ideal world, all the blocks in $C^{(1)}, \ldots, C^{(q_e)}$ are uniformly distributed in $\{0,1\}^n$. Therefore,

$$\Pr[T_{id} = \tau] = \Pr(\mathbf{K}, \mathbf{T}) \cdot \Pr[\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}] \cdot \left(\frac{1}{2^n}\right)^{q_e + \sum_{i=1}^{q_e} \ell_i}. \tag{32}$$

With the above, a transcript τ is bad if one of the following is fulfilled:

- (B-1) there exists two user indices i, j such that $K_i \| T_i = K_j \| T_j$;
- (B-2) $\mu_T \geq n$, or $\mu_W \geq n$;
- (B-3) there exists an encryption query (u, N, A, M, C), $C = \mathbf{c} \| Z$, such that $(K_u, T_u, x, y) \in \tau^*_{\mathsf{IC}}$ or $(K_u, W \| 1, x, y) \in \tau^*_{\mathsf{IC}}$ for some x, y, where W comes from its corresponding hash record $(\mathsf{pad}(A, N, \mathbf{c}, T_u), V \| W) \in \tau^*_{\mathsf{H}}$;
- (B-4) $\mu_Y \geq 2 \log_2 \sigma$;
- (B-5) there exists distinct queries $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)})$ and $(u^{(j)}, N^{(j)}, A^{(j)}, M^{(j)}, C^{(j)})$ with the corresponding hash records $(U^{(i)}, V^{(i)} \| W^{(i)})$ and $(U^{(j)}, V^{(j)} \| W^{(j)})$ in τ^*_{H} satisfying $(U^{(i)} = \mathsf{pad}(A^{(i)}, N^{(i)}, \mathbf{c}^{(i)}, T_{u^{(i)}})$, $U^{(j)} = \mathsf{pad}(A^{(j)}, N^{(j)}, \mathbf{c}^{(j)}, T_{u^{(j)}}))$:

– *hash collision*: $V^{(i)}\|W^{(i)} = V^{(j)}\|W^{(j)}$, or
– *contradiction*: $W^{(i)} = W^{(j)}$ and $Z^{(i)} = Z^{(j)}$.

The first three conditions have been analyzed in subsection 5.3.2. We recycle the results as follows: when $\mathbf{T}$ is uniform, we have

$$\Pr[\text{(B-1)}] \le \frac{u^2}{2^{2n}}, \qquad \Pr[\text{(B-3)} \mid \text{(B-2)}] \le \frac{n^2 q_{\mathsf{IC}}^*}{2^n},$$
$$\Pr[\text{(B-2)} \vee \textit{hash collision}] \le \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{8q_{\mathsf{IC}}^*}{2^n},$$

where q_{IC}^* is defined by Eq. (7).

For (B-4), in the ideal world $c_j^{(i)}$ and thus $m_j^{(i)} \oplus c_j^{(i)}$ is uniformly and independent from anything else. In addition $\left|\{(i,j)\}\right| \le \sigma_2 \le \sigma \ll 2^n$ with σ_2 denoting the number of blocks in queries to the challenge encryption oracle. So Lemma 14 yields

$$\Pr[\text{(B-4)}] = \Pr[\mu_Y \ge 2\log_2 \sigma] \le \frac{1}{2^n}.$$

For (B-5), the collision event has been included in the above bounds. On the other hand, for any two indices i, j, it can be seen from the proof in Section 5.2 that $\Pr[W^{(i)} = W^{(j)}] \le \frac{2}{2^n - q_{\mathsf{IC}}^*}$. On the other hand, the tags $Z^{(i)}$ and $Z^{(j)}$ are uniform in the ideal world, and thus $\Pr[Z^{(i)} = Z^{(j)}] = \frac{1}{2^n}$. Since we have $\le q_e^2/2$ such pairs of indices (i,j), it holds

$$\Pr[\textit{contradiction in } \text{(B-5)}] \le \frac{q_e^2}{2^n(2^n - q_{\mathsf{IC}}^*)} \le \frac{2q_e^2}{2^{2n}} \le \frac{q_{\mathsf{IC}}^*}{2^n}.$$

In all,

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \le \frac{u^2}{2^{2n}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(n^2+9)q_{\mathsf{IC}}^* + 1}{2^n}. \tag{33}$$

D.2.2 Ratio of Probabilities of Good Transcripts

For a good transcript τ, by ¬(B-3), for any $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)}) \in \tau_e$ the initial session key $k_0^{(i)} = \mathsf{IC}_{K_{u^{(i)}}}^{T_{u^{(i)}}}(P_0(N^{(i)}))$ is uniform. With this observation, we define the first predicate $\mathsf{BadKD}(\mathsf{IC})$ to capture the "badness" of this key. Formally, $\mathsf{BadKD}(\mathsf{IC})$ is fulfilled if one of the following conditions is fulfilled:

- (C-1) *"none-freshness" of the key*: there exists $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)}) \in \tau_e$ such that the key $k_0^{(i)} = \mathsf{IC}_{K_{u^{(i)}}}^{T_{u^{(i)}}}(P_0(N^{(i)}))$ satisfies $(k_0^{(i)}, T_{u^{(i)}}, P_1(N^{(i)}), y) \in \tau_{\mathsf{IC}}^*$, or $(k_0^{(i)}, T_{u^{(i)}}, Q_0(N^{(i)}), y) \in \tau_{\mathsf{IC}}^*$ for some y, or $(k_0^{(i)}, T_{u^{(i)}}, x, y_1^{(i)}) \in \tau_{\mathsf{IC}}^*$ for some x;
- (C-2) *nonce-reuse across different users*: there exists two queries $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)})$ and $(u^{(j)}, N^{(j)}, A^{(j)}, M^{(j)}, C^{(j)})$ in τ_e such that $T_{u^{(i)}} = T_{u^{(j)}} = T$, $N^{(i)} = N^{(j)} = N$, and the two keys $k_0^{(i)} = \mathsf{IC}_{K_{u^{(i)}}}^{T}(P_0(N^{(i)}))$ and $k_0^{(j)} = \mathsf{IC}_{K_{u^{(j)}}}^{T}(P_0(N^{(j)}))$ are identical.

We remark that the event of nonce-reusing for a single user, i.e., there exists two queries of the form $(u, N, A, M, C), (u, N, A', M', C')$ in τ_e, is also bad for the subsequent probability calculation. But this is forbidden by the $\mathsf{muCCAm\$}$ security definition.

For a specific public-key T, a nonce N, and an index j, we define an auxiliary set of keys

$$\tau_{\mathsf{IC}}^*[T, j, N] := \Big\{k : (k, T, P_{j+1}(N), y) \in \tau_{\mathsf{IC}}^* \text{ or } (k, T, Q_j(N), y) \in \tau_{\mathsf{IC}}^* \text{ for some } y\Big\}.$$

In addition, for T and a key stream block $y \in \{0,1\}^n$, define

$$\tau^*_{\mathsf{IC}}[T,y]^{-1} := \{k : (k,T,x,y) \in \tau^*_{\mathsf{IC}} \text{ for some } x\}.$$

Conditioned on $\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$ and the values of

$$k_0^{(1)} = \mathsf{IC}_{K_{u^{(1)}}}^{T_{u^{(1)}}}(N^{(1)}), \ldots, k_0^{(i-1)} = \mathsf{IC}_{K_{u^{(i-1)}}}^{T_{u^{(i-1)}}}(N^{(i-1)}),$$

the key $k_0^{(i)} = \mathsf{IC}_{K_{u^{(i)}}}^{T_{u^{(i-1)}}}(N^{(i)})$ is uniform in $\geq 2^n - q^*_{\mathsf{IC}} - q_e$ possibilities: the first half is due to ¬(B-3), while the second half is due to the non-repeating property of the triple (K,T,N) (for the same user N can't be repeated, while for different users i,j we've $K_i\|T_i \neq K_j\|T_j$ by ¬(B-1)). Therefore,

$$\Pr[\text{(C-1)}] \leq \sum_{i=1}^{q_e} \frac{\left|\tau^*_{\mathsf{IC}}[T_{u_i},0,N^{(i)}]\right|}{2^n - q^*_{\mathsf{IC}} - q_e} + \sum_{i=1}^{q_e} \frac{\left|\tau^*_{\mathsf{IC}}[T_{u_i},y_1^{(i)}]^{-1}\right|}{2^n - q^*_{\mathsf{IC}} - q_e}.$$

For (C-2), the number of choice for $(u^{(i)},N^{(i)},A^{(i)},M^{(i)},C^{(i)})$ is $\leq q_e$. For each such choice, the number of choice for the $(u^{(j)},N^{(j)},A^{(j)},M^{(j)},C^{(j)})$ is $\leq \mu_T$ due to the restriction $T_{u^{(i)}} = T_{u^{(j)}}$. Thus

$$\Pr[\text{(C-2)}] \leq \frac{\mu_T q_e}{2^n - q^*_{\mathsf{IC}} - q_e} \leq \frac{2\mu_T q_e}{2^n}.$$

Thus when $q^*_{\mathsf{IC}} + q_e \leq 2^n/2$, we have

$$\Pr_{\mathsf{IC}}[\mathsf{BadKD}(\mathsf{IC}) \mid \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}] \leq \frac{2\mu_T q_e}{2^n} + \sum_{i=1}^{q_e} \frac{2\left|\tau^*_{\mathsf{IC}}[T_{u^{(i)}},0,N^{(i)}]\right|}{2^n} + \sum_{i=1}^{q_e} \frac{2\left|\tau^*_{\mathsf{IC}}[T_{u^{(i)}},y_1^{(i)}]^{-1}\right|}{2^n}. \tag{34}$$

We then analyze the q_e encryption queries in turn, and define a sequence of bad predicates

$$\begin{aligned} &\mathsf{BadE}_1^{(1)}, \mathsf{BadE}_2^{(1)}, \ldots, \mathsf{BadE}_{\ell_1-1}^{(1)}, \\ &\ldots \\ &\mathsf{BadE}_1^{(q_e)}, \mathsf{BadE}_2^{(q_e)}, \ldots, \mathsf{BadE}_{\ell_{q_e}-1}^{(q_e)}. \end{aligned} \tag{35}$$

As will be seen, each predicate concerns with the encryption of a specific plaintext block. Formally, for $1 \leq i \leq q_e$, consider the i-th query $(u^{(i)},N^{(i)},A^{(i)},M^{(i)},C^{(i)})$, and for $1 \leq j \leq \ell_i - 1$, let

$$k_0^{(i)} = \mathsf{IC}_{K_{u^{(i)}}}^{T_{u^{(i)}}}(P_0(N^{(i)})), k_1^{(i)} = \mathsf{IC}_{k_0^{(i)}}^{T_{u^{(i)}}}(P_1(N^{(i)})), \ldots, k_j^{(i)} = \mathsf{IC}_{k_{j-1}^{(1)}}^{T_{u^{(i)}}}(P_j(N^{(i)}))$$

be the derived intermediate values. Then $\mathsf{BadE}_j^{(i)}(\mathsf{IC})$ is fulfilled, if at least one of the following conditions is fulfilled:

- (C-ij1): $k_j^{(i)} \in \tau^*_{\mathsf{IC}}[T_{u^{(i)}},j,N^{(i)}]$, or
- (C-ij2): there exists $s < i$ such that for the s-th encryption query $(u^{(s)},N^{(s)},A^{(s)},M^{(s)},C^{(s)})$, it holds:
  - $T_{u^{(s)}} = T_{u^{(i)}}$ and $N^{(s)} = N^{(i)}$, and
  - $k_j^{(i)}$ equals the intermediate value $k_j^{(s)}$ derived correspondingly.

- (C-$ij3$): $k_j^{(i)} \in \tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, y_{j+1}^{(i)}\big]^{-1}$, or
- (C-$ij4$): there exists two indices (s,t) such that *either* $s<i$, *or* $s=i$ *and* $t<j$, and:
  - $y_{t+1}^{(s)} = y_{j+1}^{(i)}$, and
  - $k_j^{(i)}$ equals the intermediate value $k_t^{(s)}$ derived correspondingly.
- (C-$ij5$): $k_j^{(i)} = y_j^{(i)} = m_j^{(i)} \oplus c_j^{(i)}$.

It isn't hard to see conditioned on $\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$ and $\neg\mathsf{BadKD}(\mathsf{IC})$ and $\neg\mathsf{BadE}_{j-1}^{(i)}(\mathsf{IC}) \wedge \ldots \wedge \neg\mathsf{BadE}_1^{(1)}(\mathsf{IC})$, the value $k_j^{(i)}$ is uniform in $\geq 2^n - q^*_{\mathsf{IC}} - q_e$ possibilities. Therefore,

$$\Pr[(\text{C-}ij1) \vee (\text{C-}ij3)] \leq \frac{\left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, j, N^{(i)}\big]\right| + \left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, y_{j+1}^{(i)}\big]^{-1}\right|}{2^n - q^*_{\mathsf{IC}} - q_e}.$$

For (C-$ij2$), due to the restriction $T_{u^{(s)}} = T_{u^{(i)}}$ and the design of TEDT, it can be seen the number of such index s is at most $\mu_T - 1$. Similarly, for (C-$ij4$), the restriction $y_{t+1}^{(s)} = y_{j+1}^{(i)}$ indicates the number of such pairs of indices (s,t) is at most $\mu_Y - 1$. Therefore,

$$\Pr[(\text{C-}ij2) \vee (\text{C-}ij4) \vee (\text{C-}ij5)] \leq \frac{\mu_T - 1 + \mu_Y - 1 + 1}{2^n - q^*_{\mathsf{IC}} - q_e}.$$

Thus when $q^*_{\mathsf{IC}} + q_e \leq 2^n/2$ we have

$$\begin{aligned}&\Pr[\mathsf{BadE}_j^{(i)}(\mathsf{IC}) \mid \neg\mathsf{BadE}_{j-1}^{(i)}(\mathsf{IC}) \wedge \ldots \wedge \neg\mathsf{BadE}_1^{(1)}(\mathsf{IC}) \wedge \neg\mathsf{BadKD}(\mathsf{IC}) \wedge \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}]\\ \leq& 2\frac{\left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, j, N^{(i)}\big]\right| + \left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, y_{j+1}^{(i)}\big]^{-1}\right| + \mu_T + \mu_Y}{2^n}.\end{aligned}$$

For $1 \leq i \leq q_e$ and $1 \leq j \leq \ell_i$, conditioned on $\neg\mathsf{BadE}_j^{(i)}(\mathsf{IC}) \wedge \neg\mathsf{BadE}_{j-1}^{(i)}(\mathsf{IC}) \wedge \ldots \wedge \neg\mathsf{BadE}_1^{(1)}(\mathsf{IC}) \wedge \neg\mathsf{BadKD}(\mathsf{IC}) \wedge \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$, it can be seen the value $y^\dagger = \mathsf{IC}_{k_{j-1}^{(i)}}^{T_{u^{(i)}}}(Q_{j-1}(N^{(i)}))$ is uniform in $\geq 2^n - q^*_{\mathsf{IC}} - q_e$ possibilities, and *these possibilities include* $y_j^{(i)}$. Therefore,

$$\Pr[y^\dagger = y_j^{(i)}] \geq \frac{1}{2^n}. \tag{36}$$

The probabilities of the predicates cumulate to

$$\begin{aligned}&\Pr[\underbrace{\mathsf{BadE}_{\ell_{q_e}-1}^{(q_e)}(\mathsf{IC}) \vee \ldots \vee \mathsf{BadE}_1^{(1)}(\mathsf{IC}) \vee \neg\mathsf{BadKD}(\mathsf{IC})}_{=\mathsf{Bad}(\mathsf{IC})} \mid \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}]\\ \leq& \sum_{i=1}^{q_e}\sum_{j=1}^{\ell_i} \frac{2\left(\left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, j, N^{(i)}\big]\right| + \left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, y_{j+1}^{(i)}\big]^{-1}\right| + \mu_T + \mu_Y\right)}{2^n}.\end{aligned}$$

This plus Eq. (34) yields

$$\Pr[\mathsf{Bad}(\mathsf{IC}) \mid \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}] \leq \sum_{i=1}^{q_e}\sum_{j=0}^{\ell_i-1} \frac{2\left(\left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, j, N^{(i)}\big]\right| + \left|\tau^*_{\mathsf{IC}}\big[T_{u^{(i)}}, y_{j+1}^{(i)}\big]^{-1}\right| + \mu_T + \mu_Y\right)}{2^n}.$$

It's easy to see $\sum_{i=1}^{q_e}\sum_{j=0}^{\ell_i-1}(\mu_T+\mu_Y) \le (\mu_T+\mu_Y)\sigma$. On the other hand, for the summation $\sum_{i=1}^{q_e}\sum_{j=0}^{\ell_i-1}\left|\tau^*_{\mathsf{IC}}\left[T_{u^{(i)}}, j, N^{(i)}\right]\right|$, we reorganize it according to different T values. In this vein, we have

$$\sum_{i=1}^{q_e}\sum_{j=0}^{\ell_i-1}\left|\tau^*_{\mathsf{IC}}\left[T_{u^{(i)}}, j, N^{(i)}\right]\right| \le \mu_T \cdot \sum_{\substack{T\in\{0,1\}^n, N\in\mathcal{N},\\ j\in\{0,\ldots,\ell_i-1\}}}\left|\tau^*_{\mathsf{IC}}\left[T, j, N\right]\right| \le \mu_T|\tau^*_{\mathsf{IC}}| \le \mu_T q^*_{\mathsf{IC}}.$$

Similarly,

$$\sum_{i=1}^{q_e}\sum_{j=1}^{\ell_i}\left|\tau^*_{\mathsf{IC}}\left[T_{u^{(i)}}, y^{(i)}_j\right]^{-1}\right| \le \mu_Y q^*_{\mathsf{IC}}.$$

Gathering the above and Eq. (36) yields

$$\begin{aligned}
&\Pr\left[\mathsf{TEDT}[\mathsf{IC}].\mathsf{Enc}_{\mathbf{K},\mathbf{T}}(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}) = \mathbf{c}^{(i)}, \text{ for all } i \in \{1,\ldots,q_e\} \mid \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}\right] \\
&\quad \ge \Pr\left[\neg\mathsf{Bad}(\mathsf{IC}) \mid \mathsf{IC} \vdash \tau^*_{\mathsf{IC}}\right]\left(\frac{1}{2^n}\right)^{\sum_{i=1}^{q_e}\ell_i} \\
&\quad \ge \left(1 - \frac{2\mu_T\sigma + 2\mu_Y\sigma + 2\mu_T q^*_{\mathsf{IC}} + 2\mu_Y q^*_{\mathsf{IC}}}{2^n}\right)\left(\frac{1}{2^n}\right)^{\sum_{i=1}^{q_e}\ell_i}. \qquad (37)
\end{aligned}$$

It remains to analyze the produced tags. Let the hash query record corresponding to $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)})$ be $(U^{(i)}, V^{(i)}\|W^{(i)})$. Therefore, the event that the q_e tags equal $Z^{(1)},\ldots,Z^{(q_e)}$ is equivalent to q_e equalities as follows:

$$\mathsf{IC}^{W^{(1)}\|1}_{K_{u^{(1)}}}(V^{(1)}) = Z^{(1)}, \ldots, \mathsf{IC}^{W^{(q_e)}\|1}_{K_{u^{(q_e)}}}(V^{(q_e)}) = Z^{(q_e)}.$$

Consider the first equality. The entry $\mathsf{IC}^{W^{(1)}\|1}_{K_{u^{(1)}}}(V^{(1)})$ may have been rendered non-random due to the condition $\mathsf{TEDT}[\mathsf{IC}].\mathsf{Enc}_{\mathbf{K},\mathbf{T}}(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}) = \mathbf{c}^{(i)}$ for all $i \in \{1,\ldots,q_e\}$ or due to $\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$. However, the former condition only affects entries with the tweak $T\|0$, while the latter would not affect $\mathsf{IC}^{W^{(1)}\|1}_{K_{u^{(1)}}}(V^{(1)})$ due to ¬(B-3). Therefore, $\Pr[\mathsf{IC}^{W^{(1)}\|1}_{K_{u^{(1)}}}(V^{(1)}) = Z^{(1)}] = \frac{1}{2^n}$.

In a similar vein, for any index $i \in \{1,\ldots,q_e\}$, under the conditions that $\mathsf{IC} \vdash \tau^*_{\mathsf{IC}}$ and "$\mathsf{TEDT}[\mathsf{IC}].\mathsf{Enc}_{\mathbf{K},\mathbf{T}}(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}) = \mathbf{c}^{(i)}$ for all $i \in \{1,\ldots,q_e\}$", the output $\mathsf{IC}^{W^{(i)}\|1}_{K_{u^{(i)}}}(V^{(i)})$ remains uniform. We need to additionally consider the condition "$\mathsf{IC}^{W^{(j)}\|1}_{K_{u^{(j)}}}(V^{(j)}) = Z^{(j)}$ for $j = 1,\ldots,i-1$". By ¬(B-5), $V^{(j)}\|W^{(j)} \ne V^{(i)}\|W^{(i)}$ and $Z^{(j)}\|W^{(j)} \ne Z^{(i)}\|W^{(i)}$ for any $j < i$. By this, $\Pr[\mathsf{IC}^{W^{(i)}\|1}_{K_{u^{(i)}}}(V^{(i)}) = Z^{(i)}] \ge \frac{1}{2^n}$, and thus

$$\Pr[\mathsf{IC}^{W^{(i)}\|1}_{K_{u^{(i)}}}(V^{(i)}) = Z^{(i)} \text{ for } i = 1,\ldots,q_e] \ge \frac{1}{2^{q_e n}}. \qquad (38)$$

Gathering Eq. (32), (37), and (38), and with $\sigma \le 2^n/48 \Rightarrow \log_2\sigma \le n$, we have

$$\begin{aligned}
\frac{\Pr[T_{re} = \tau]}{\Pr[T_{id} = \tau]} &\ge \left(1 - \frac{2\mu_T\sigma + 2\mu_Y\sigma + 2\mu_T q^*_{\mathsf{IC}} + 2\mu_Y q^*_{\mathsf{IC}}}{2^n}\right)\left(\frac{1}{2^n}\right)^{q_e+\sum_{i=1}^{q_e}\ell_i} \Big/ \left(\frac{1}{2^n}\right)^{q_e+\sum_{i=1}^{q_e}\ell_i} \\
&\ge 1 - \frac{2n(\sigma + q^*_{\mathsf{IC}}) + 4n(\sigma + q^*_{\mathsf{IC}})}{2^n}, \qquad (\mu_T \le n, \mu_Y \le 2\log_2\sigma \le 2n) \\
&\ge 1 - \frac{6n(\frac{q^*_{\mathsf{IC}}}{4} + q^*_{\mathsf{IC}})}{2^n} \ge 1 - \frac{8nq^*_{\mathsf{IC}}}{2^n}, \qquad (\sigma \le \frac{q^*_{\mathsf{IC}}}{4})
\end{aligned}$$

This plus Eq. (33) yield

$$\mathbf{Adv}^{\mathsf{muCPAm\$}^*}_{\mathcal{D},\mathsf{TEDT},u} \leq \frac{u^2}{2^{2n}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(n^2+8n+9)q^*_{\mathsf{IC}}+1}{2^n}.$$

By Eq. (31), this plus the bound in Eq. (1) yields

$$\begin{aligned}
&\frac{u^2}{2^{2n}} + \frac{(2n^2+17)(4\sigma+6(q_e+q_d)+2q_{\mathsf{IC}})}{2^n} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n \\
&+ \frac{u^2}{2^{2n}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(n^2+8n+9)(4\sigma+6(q_e+q_d)+2q_{\mathsf{IC}})+1}{2^n} \\
=&\frac{2u^2}{2^{2n}} + \frac{1}{n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(3n^2+8n+26)(4\sigma+6(q_e+q_d)+2q_{\mathsf{IC}})}{2^n} \\
\leq&\frac{2u^2}{2^{2n}} + \frac{1}{n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(7n^2+26)(4\sigma+6(q_e+q_d)+2q_{\mathsf{IC}})}{2^n} \quad (n \geq 2).
\end{aligned}$$

D.3 Detailed Comparison with GCM-SIV with KDF

GCM-SIV with nonce-based KDF was recently proposed by Gueron and Lindell [GL17] as a very elegant and efficient BBB secure variant of GCM-SIV. Its security bounds are further improved by Bose et al. [BHT18] in the ideal cipher model.

In summary, AES-128-GCM-SIV with KDF uses $n = 128$ key bits, and its advantages are:

(i) Misuse-resistance (in black-box setting)

(ii) Asymptotically optimal mu security when nonce is random

(iii) Much less calls to the block cipher

For $n = 128$ TEDT uses 255 key bits but only 128 bits are *secret*. Its advantages are:

(i) Asymptotically optimal mu security for fresh nonce (misuse-resilience), say allowing arbitrary nonce reuse across different users, and slightly better bounds than AES-128-GCM-SIV (see below)

(ii) Provable side-channel security.

Concretely, in the mu setting, if nonce is arbitrarily reused across many users, then security of AES-128-GCM-SIV is capped at $\frac{u^2}{2^{128}}$. While Bose et al. proved that using random nonce could forbid arbitrarily reuse and achieve mu BBB, this method turns ineffective when the nonces in use are the same in each session (e.g., TLS 1.3 uses the sequence number to compute the nonces). Also according to Rogaway random nonce may be more prone to misuse. In comparison, in TEDT we employ additional 127-bit public randomness to achieve mu BBB. Though don't support full nonce misuse, TEDT does support arbitrarily reusing the same nonce across different users.

In terms of bounds, with 128-bit secret keys, the mu bound of AES-128-GCM-SIV is

$$\frac{\sigma\ell_{max} + d(q_{\mathsf{IC}}+\sigma)}{2^{128}},$$

where ℓ_{max} and d are upper bounds, respectively, the number of blocks encrypted *per user-nonce pair*, and of the number of users that re-use a particular nonce value. Setting $d = 1$ recovers its su bound. By this, TEDT probably achieves slightly better bounds $\frac{2^{14}\sigma+2^{14}q_{\mathsf{IC}}}{2^{128}}$ even in the su setting (revisiting the calculations in Eq. (9) shows that the factor n^2 could be improved to n in the su setting, so its su bound could be better $\frac{128\sigma+128q_{\mathsf{IC}}}{2^{128}}$). As mentioned, we believe this shows the benefits of protocol-level leakage-resilient designs.

E A single-pass TBC Mode TET

The single-pass mode TET is defined in Figure 14 and illustrated in Figure 15. It achieves muCIML2 with nearly optimal bounds as TEDT. Yet, since it has only one pass, the consensus is that the confidentially is unavoidably altered given decryption leakages. Consequently, TET only ensures muCCAmL1 security, with bounds comparable to Theorem 2.

algorithm $\mathsf{TET}[\mathsf{E}].\mathsf{Enc}_K(N, A, M)$

$\ell \leftarrow \lceil |M|/n \rceil$
$\nu \leftarrow \lceil |A|/n \rceil$
parse M as $M[1]\| \ldots \|M[\ell]$, with $|M[1]| = \ldots = |M[\ell-1]| = n$ and $1 \leq |M[\ell]| \leq n$
$M[\ell] \leftarrow M[\ell]\|0^*$
parse A as $A[1]\| \ldots \|A[\nu]$, with $|A[1]| = \ldots = |A[\nu-1]| = n$ and $1 \leq |A[\nu]| \leq n$
$A[\nu] \leftarrow A[\nu]\|0^*$
$T \leftarrow PK\|0$
$S \leftarrow \mathsf{E}_K^T(N\|0^*)$
$a \leftarrow \mathsf{E}_S^{PK\|0}(N\|0^*)$
$b \leftarrow \mathsf{E}_S^{PK\|1}(N\|0^*)$
$h_1 \leftarrow \mathsf{ls}_{n-1}(a \oplus b)$
$k_1 \leftarrow b \oplus (N\|0^*)$
for $i = 1$ **to** ℓ **do**
    $C[i] \leftarrow \mathsf{E}_{k_i}^{h_i\|0}(M[i])$
    $c_i^* \leftarrow \mathsf{E}_{k_i}^{h_i\|0}(M[i])$
    $h_{i+1} \leftarrow \mathsf{ls}_{n-1}(C[i] \oplus c_i^*)$
    $k_{i+1} \leftarrow c_i^* \oplus M[i]$
for $i = 1$ **to** ν **do**
    $j \leftarrow i + \ell$
    $d_i \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(A[i])$
    $d_i^* \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(A[i])$
    $h_{j+1} \leftarrow \mathsf{ls}_{n-1}(d_j \oplus d_j^*)$
    $k_{j+1} \leftarrow d_j^* \oplus A[i]$
$j \leftarrow \ell + \nu + 1$
$tail \leftarrow [|M|]_{n/2}\|[|A|]_{n/2}$
$d \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(tail)$
$d^* \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(tail)$
$V \leftarrow \mathsf{ls}_{n-1}(d \oplus d^*)$
$U \leftarrow d^* \oplus tail$
$Z \leftarrow \mathsf{E}_K^{V\|1}(U)$
$C \leftarrow C[1]\| \ldots \|C[\ell^M]\|Z$
return C

algorithm $\mathsf{TET}[\mathsf{E}].\mathsf{Dec}_K(N, A, C)$

$\ell \leftarrow \lceil \frac{|C|-n}{n} \rceil$
$\nu \leftarrow \lceil |A|/n \rceil$
parse C as $C[1]\| \ldots \|C[\ell]\|Z$, with $|C[1]| = \ldots = |C[\ell-1]| = |Z| = n$ and $1 \leq |C[\ell]| \leq n$
parse A as $A[1]\| \ldots \|A[\nu]$, with $|A[1]| = \ldots = |A[\nu-1]| = n$ and $1 \leq |A[\nu]| \leq n$
$A[\nu] \leftarrow A[\nu]\|0^*$
$T \leftarrow PK\|0$
$S \leftarrow \mathsf{E}_K^T(N\|0^*)$
$a \leftarrow \mathsf{E}_S^{PK\|0}(N\|0^*)$
$b \leftarrow \mathsf{E}_S^{PK\|1}(N\|0^*)$
$h_1 \leftarrow \mathsf{ls}_{n-1}(a \oplus b)$
$k_1 \leftarrow b \oplus (N\|0^*)$
for $i = 1$ **to** ℓ **do**
    $M[i] \leftarrow (\mathsf{E}_{k_i}^{h_i\|0})^{-1}(C[i])$
    $c_i^* \leftarrow \mathsf{E}_{k_i}^{h_i\|0}(M[i])$
    $h_{i+1} \leftarrow \mathsf{ls}_{n-1}(C[i] \oplus c_i^*)$
    $k_{i+1} \leftarrow c_i^* \oplus M[i]$
for $i = 1$ **to** ν **do**
    $j \leftarrow i + \ell$
    $d_i \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(A[i])$
    $d_i^* \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(A[i])$
    $h_{j+1} \leftarrow \mathsf{ls}_{n-1}(d_j \oplus d_j^*)$
    $k_{j+1} \leftarrow d_j^* \oplus A[i]$
$j \leftarrow \ell + \nu + 1$
$tail \leftarrow [|M|]_{n/2}\|[|A|]_{n/2}$
$d \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(tail)$
$d^* \leftarrow \mathsf{E}_{k_j}^{h_j\|0}(tail)$
$V \leftarrow \mathsf{ls}_{n-1}(d \oplus d^*)$
$U \leftarrow d^* \oplus tail$
$U^* \leftarrow (\mathsf{E}_K^{V\|1})^{-1}(Z)$
if $U^* \neq U$ **then return** $\perp$
$M \leftarrow M[1]\| \ldots \|M[\ell]$
return M

Figure 14: Definition of the TET mode, using a TBC E.

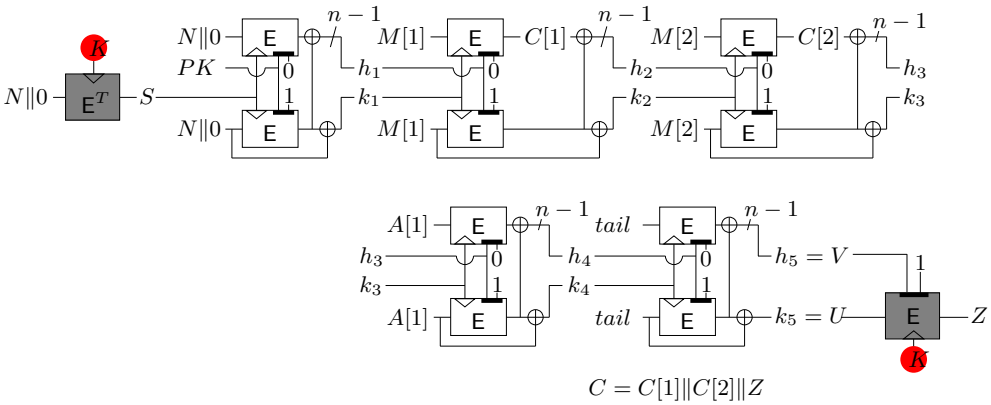

Figure 15: Illustrating the TET AEAD. The two dark blocks are "leak-free" TBC-calls. The other TBC-calls are leaking. For each square, the input to the triangle denotes the key input. The "tail" block is $tail = [|M|]_{n/2}\|[|A|]_{n/2}$, and $T = PK\|0$.