TEDT, a Leakage-Resilient AEAD mode for High (Physical) Security Applications

Francesco Berti, Chun Guo, Olivier Pereira, Thomas Peters, François-Xavier Standaert
ICTEAM Institute, Crypto Group, Université catholique de Louvain, Louvain-la-Neuve, Belgium.

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 asymptotically optimal security 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 offers KDM security in the multi-user setting, that is, its security is maintained even if key-dependent messages are encrypted. (iv) It offers full leakage-resilience, 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. (v) 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 & incremental associated data efficiently. Concretely, TEDT encourages leveled implementations, in which two TBCs are implemented: 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 considerably more energy efficient implementations compared to traditional AEAD schemes, whose side-channel security requires to uniformly protect every (T)BC execution.

Index Terms

Authenticated encryption, re-keying, tweakable block cipher, beyond-birthday bound, multi-user security, side-channel security, key-dependent messages security, leveled implementations, low energy implementations.

I. 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 [1], [2], 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 [3, chapter 1.1] – see also [4]–[10], for example.

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) [11]. 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 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 [12], CCM [13], or GCM [14] 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 DPAs apply.

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 [15], 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), 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 [16], [17]. The overhead for the full AEAD mode is then of the same order, since all message blocks need to be processed by one such strongly protected block cipher.

Another approach consists in designing leakage-resilient modes [18]–[21]. 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 [22], [23] 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 [18], [19]. 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-resilient 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. A leveled implementation will rely, on the one hand, on the limited use of highly protected or, in effect, leak-free components. In practice, these would use strong state-of-the-art protections, like high-order masking. 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 [24], [25], and plain unprotected implementations could even be sufficient in case of hardware devices (with a good level of parallelism) [26].

The expected benefits of this leakage-resilient 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 an Rivain at Eurocrypt 2017 [16], 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 factor 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-resilient modes bring clear requirement 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. And, 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 of operation.

Apart from its leakage-resilience, TEDT is also highly competitive in the black-box setting as detailed next:

- TEDT offers optimal multi-user (mu) security and even surpasses, in this respect, the recently GCM-SIV with nonce-based key-derivation [27], [28]. The gain is obtained from the TEDT re-keying process introduced for leakage-resilience, and shows another benefit of investigating protections against side-channel attacks at the mode of operation level (i.e., side-channel protections can improve other security properties). Note that such a Beyond Birthday Bound (BBB) security was highlighted in [27] and [3, chapter 1.2], while mu security is crucial for defending against mass surveillance [29].
- TEDT offers mu nonce misuse-resilience in the sense of Ashur et al. [30], and this property is preserved in the presence of leakages, under the definitions in [31]. 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 [32], 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; but is also believed impossible in many leakage settings (see Appendix A).
- TEDT offers mu key-dependent message (KDM) security. This property is granted thanks to the KDF function that is integrated in TEDT as part of its rekeying procedure for leakage resilience. (The use of such a KDF was identified by Bellare and Keelveedhi as a useful ingredient to obtain KDM security [33]).

Our approach for improving mu security may be of independent interest. While it is known that increasing the length of secret keys and randomizing nonces can help [29], we show how to improve mu security from a public key (used in addition to the usual secret key).

TEDT **analysis.** Our security proofs model the TBC as an *ideal TBC*. Our motivation for appealing to this ideal model partly inherits from previous works [28], [29]: adversarial queries to the ideal TBC give a clear and rigorous way to measure the offline computation. In fact, non-degrading mu security results typically rely on the ideal model, e.g., [34] and some of them, e.g., XGCM [29] and its underlying FX-key length extension [35], 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 (and we do this in appendix). 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 [36], [37] (this constitutes another equally important reason for relying on ideal model analysis).

In terms of leakages, the ideal cipher model prompted us to make very simple assumptions. Indeed, in such a model, leakages about a key are guaranteed to be useless as long as they do not lead to a full key-recovery. Our strongly protected components are then modeled as leak-free, that is, hiding their key and, in the case of confidentiality, also mildly hiding their output, which

follows [38]. Our weakly protected components are assumed to leak their state in full as far as authentication is concerned, which follows [19], and to offer hard-to-invert leakages for confidentiality, which follows [23] and appears theoretically minimal & practically measurable (which we believe is essential for modes in use). We also provide analyzes in the standard model in appendix: they require to make stronger assumptions of leakage simulatability [39] regarding the weakly protected components, and the analyzes lead to comparable bounds.

II. Related Works

ISAP is an elegant sponge-based AEAD aiming at design-level DPA security [21]. While being based on different constructions and primitives, ISAP uses the Encrypt-then-MAC pattern that is common to most proposals for leakage-resilience. The understanding of side-channel protections for sponge-based constructions is however scarce, and the lack of available tools made the authors of ISAP postpone any rigorous analysis of their mode for future work. The use of a sponge and the definition of an authentication tag as a truncation of the final sponge state can be expected to raise some specific difficulties: indeed it forces computing correct tags every time a ciphertext is sent for decryption, and the leakages of this computation may eventually lead to simple forgeries. ISAP has not been designed for mu security either, and is actually quite sensitive to mu attacks on its authentication part. Is is unknown how much key size increases would be required to address this difficulty, of if the techniques used in our paper could be used in ISAP.

Barwell et al. [18] recently proposed security definitions, a mode of operation, and a choice of a PRF, for a leakage-resilient AEAD mode. Their security definitions are 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.

Berti et al. [19], [20] and Guo et al. [31] also introduced several security definitions and modes of operation for leakage resilient 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, and we offer a performance evaluation, 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.

III. 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. 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.

A. 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 X. We call such TBCs (κ,t,n)-TBC. Similarly, we denote (κ,n)-blockcipher 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 $\widetilde{\mathsf{IC}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a TBC sampled uniformly at random from all (n,n,n)-TBCs. In this case, $\widetilde{\mathsf{IC}}_K^T$ is a random independent permutation of $\{0,1\}^n$ for each $(K,T) \in \{0,1\}^n \times \{0,1\}^n$ even if the key K is *public*. Throughout the remaining, we simply use the notation $\widetilde{\mathsf{E}}$ 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 scheme 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 $\mathcal{N}$, 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.

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.

B. Security Definitions in the Multi-User Setting

We extend important existing security definitions to the multi-user setting: they include the (black-box) notion of nonce-misuse resilience due to Ashur et al. [30] as well as two leakage-resilience notions due to Berti et al. [20] and Guo et al. [31], for integrity and confidentiality. While single-user (su-) notion does not always imply its multi-user (mu-) counterpart in some leakage settings [31], we show in Appendix B that our mu-extensions reduce to the su-cases.

1) Misuse-Resilience: Ashur et al. [30] 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* [32] we will not refer to *misuse-resilience* with its initials but as $\mathsf{CCAm\$}$ 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

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAm\$}}_{\mathsf{AEAD},\mathcal{A},u} \;:=\; &\Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K}},\mathsf{Enc}_{\mathbf{K}},\mathsf{Dec}_{\mathbf{K}},\widetilde{\mathsf{IC}}} \Rightarrow 1\big]\\ &-\; \Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K}},\$,\perp,\widetilde{\mathsf{IC}}} \Rightarrow 1\big]\Big|,\end{aligned}$$

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 $\widetilde{\mathsf{IC}}$[1] and where $\mathsf{Enc}_{\mathbf{K}}(i, N, A, M)$: if $1 \le i \le u$, outputs $\mathsf{Enc}_{K_i}(N, A, M)$; $\$(i, N, A, M)$ outputs and associates a fresh random ciphertext $C \xleftarrow{\$} \mathcal{C}_{|M|}$ to fresh input, and the associated C otherwise; $\mathsf{Dec}(i, N, A, C)$ outputs $\mathsf{Dec}_{K_i}(N, A, C)$ if (i, N, A, C) is not an oracle answer to an encryption query (i, N, A, M) for some M, and $\perp$ otherwise; $\perp(i, N, A, C)$ outputs $\perp$; for each user i: (i) nonce N cannot be used both in query to $O_1(i, N, *, *)$ and $O_2(i, N, *, *)$; (ii) $O_2(i, *, *, *)$ is nonce-respecting; (iii) if C is returned by $O_1(i, N, A, M)$ or $O_2(i, N, A, M)$ query $O_3(i, N, A, C)$ is forbidden; (iv) a nonce used twice with O_1 cannot be used for an O_3 query. The *extended* $\mathsf{muCCAm\$}$ advantage $\mathsf{muCCAm\$}^*$ is defined as the $\mathsf{muCCAm\$}$ one where the last restriction *(iv)* is waived from the definition.

2) Leakage-Resilience: 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* [40], which naturally splits the level of confidence we might expect to achieve both notions. Here, we will focus on misuse-resilient AEAD with misuse-*resistant* integrity and misuse-*resilient* confidentiality (as mentioned earlier, misuse-*resistant* confidentiality does not seem possible: 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 [19], [20], 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

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

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 $\widetilde{\mathsf{IC}}$, 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 $\mathsf{L}_{\mathsf{enc}}(K_i, N, A, M)$;
- $\mathsf{LDec}_{\mathbf{K}}(\ldots)$: outputs $\big(\mathsf{Dec}_{K_i}(N, A, C), \mathsf{L}_{\mathsf{dec}}(K_i, N, A, C)\big)$;
- $\mathsf{LDec}^{\perp}_{\mathbf{K}}(\ldots)$: computes $\mathsf{leak_d} \leftarrow \mathsf{L}_{\mathsf{dec}}(K_i, N, A, C)$ 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 $(\perp, \mathsf{leak_d})$.

[1] Since we target security in the ideal cipher model, we follow Bellare and Tackmann [29] and highlight $\widetilde{\mathsf{IC}}$ in the definition.

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

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

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

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

Security against chosen-ciphertext attacks with misuse-*resilience* and leakage [31], denoted CCAmL2, is a confidentiality guarantee of authenticated encryption in the leaking setting. Below we define its 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

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

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

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

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

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, was intended to capture the informativeness of valid decryption leakage, and its motivation stems from the harmness of decryption leakage in some applications such as secure bootloading [41]. It was proved [31] that the combination CIML2 + CCAmL2 is not implied by other combinations of any natural relaxation of these both notions. To provide a high leakage-resilience guarantee AEAD should satisfy both CIML2 and CCAmL2 (while in this paper we consider mu extensions of both).

It is tempting to ask why leakage CCA is not defined in the "real-or-random" form — in particular, requiring the real world with leakage to be indistinguishable from the ideal world with a *leakage simulator* (as the notion *honest-but-curious indifferentiability* [42]). The reason appears that the simulation-based "real-or-random" definition does not catch the desired properties of encryption schemes: leakages containing the entire challenge messages can be simulated, but completely ruin message confidentiality.

IV. Background and Design Consideration

We next describe how we modify the AEAD mode AEDT in [31] which is itself an enhancement of the Encrypt, Digest and Tag (EDT) mode [20] allowing to handle associated data. All these modes rely on re-keying, so we start by discussing the benefits and limitations of this technique before presenting our practical and technical improvements.

A. Overview of the Starting Point AEDT

While composing leakage-resilient building blocks does not necessarily leads to a leakage resilient scheme, AEDT is a successful Encrypt-then-MAC (EtM) composition of a re-keying encryption due to [38] and a plain keyless Hash-then-SPRP authentication schemes. Next, we pinpoint the core ideas behind this CCAmL2 and CIML2 secure AEAD even if few considerations have been made in terms of practicality. A picture is given in appendix C (Figure 8).

Re-keying against Side-Channel Key-Recovery. Re-keying renders DPA infeasible 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 AEDT, upon each encryption, a new session key k_0 is computed via a nonce-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 safely kept. Therefore, as long as the nonce is fresh, the resulting encryption process roots at 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 by a new ephemeral key $k_1 \leftarrow \mathsf{E}_{k_0}(0)$. This process is repeated until the number of key stream blocks matches that of message blocks. In all, upon a message of ℓ blocks, ℓ keys are produced during the encryption, and E is rekeyed 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 typically 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 (when targeting side-channel security – see Section VIII).

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 [38]. To remedy this situation, in AEDT each message block is involved in *only a single and easy to protect XOR operation*. This is arguably the minimum that cannot be avoided.

EtM & Hash-then-MAC for Side-Channel Integrity. As efficiency remains the prior 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 [43], OCB [12], TAE [44], OTR [45], COFB [46], and sponge-based proposals [47], [48]. However, integrated designs typically employ the Decrypt-then-Verify style decryption, which (as observed by Barwell et al. [18]) would leak unverified plaintext and alter the side-channel security. In contrast, as independently observed in [18], [20], [21], the plain Encrypt-then-MAC (EtM) paradigm grants resilience 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, though less elegant, EtM seems the most suitable classical solution in order to mitiagte 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 $T = \mathsf{TGF}_K(\mathsf{H}(U))$ ($U = N\|A\|C$ in AEDT 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 extensive analyzed [19], [21], [49]. Interestingly, 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 (A)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 AEDT. For AEDT, the security of both encryption and authentication are tightly birthday *even in the black-box setting*. While the security of the encryption part could be overcome by using counters (i.e., by using GCM idea) instead of two fixed constants, the hash-then-SPRP authentication is more problematic: once a collision is found, a forgery immediately follows. Since the hash digest is of only n bits with typically $n = 128$, forgery is possible within 2^{64} offline computations, which is a serious (real) threat.

In addition, unlike classical modes such as GCM-SIV [28], the multi-user security of AEDT cannot be boosted by simply increasing the key length of the blockcipher. This constitutes another limitation for its practical use. In fact, it is not even clear how to use a $(2n, n)$-blockcipher in a rekeying encryption mode while keeping the rate $1/2$.

B. Our New Ideas

Besides replacing the two constants in AEDT by GCM-style counters to achieve BBB encryption (as already mentioned), TEDT uses three main new ideas that we describe next.

Hash-then-TBC for Efficient BBB Authentication. After many failed trials, it appears that the simplest and most efficient approach to BBB secure leakage-resilient 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 $Z = \mathsf{TGF}_K(W, V) = (\widetilde{\mathsf{E}}_K^W)(V)$. To resist verification/ decryption leakage, we utilize the inverse, i.e., upon (I, Z), "If $(\widetilde{\mathsf{E}}_K^W)^{-1}(Z) = V$ for $V\|W = \mathsf{H}(I)$ then accept Else reject".

If one insists on using classical (n, n)-blockciphers, then it seems two calls are necessary for the BBB (leakage-resilient) authentication. Both of them as well as the additional internal wires have to be well protected, which could be problematic.

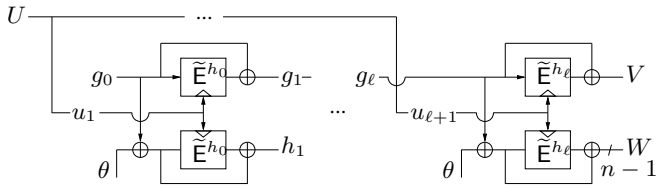

Figure 2: Hash function H upon a message $U = u_1 \| \dots \| u_{\ell+1}$ of $\ell + 1$ blocks, with IV $g_0 \| h_0$. θ is a domain separation constant used by the $\mathsf{Hir}[\widetilde{\mathsf{E}}]$ compression function.

We believe a single protected TBC-call would be more efficient/secure than two protected classical block cipher calls. As will be seen, the use of (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 $\widetilde{\mathsf{E}}$, as in Figure 2. For this purpose, we (have to) view $\widetilde{\mathsf{E}}$ as an ideal TBC $\widetilde{\mathsf{IC}}$. This model has no difference with an ideal $(2n, n)$-blockcipher. 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 $\mathsf{Hir}[\widetilde{\mathsf{E}}]$ [50], and then use $\mathsf{Hir}[\widetilde{\mathsf{E}}]$ in the strengthened Merkle-Damgård to build $\mathsf{H}[\widetilde{\mathsf{E}}]$. The formal description is given later in Fig V. In this vein, the resulted mode is purely TBC-based which may reduce implementation costs. Yet, we note that DBL hash does *not* trivially result in security: see the end of this section.

Public Randomness to Remedy mu **Security Degradation.** Roughly, mu security degradation stems from two aspects: first, collision between user keys; second, multiple user keys 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 AEDT 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 $\mu \ll u$ (can be achieved by ensuring distinct PK values, or picking PK at random), for issue (ii) the additional public randomness reduces the probability to $\frac{\mu}{2^n} \ll \frac{u}{2^n}$.[2] 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. In order to fill in the technical part, we mainly have to overcome the two following problems:

(1) Proving that the use of public randomness does avoid mu degradation 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).[3]

V. Specification for TEDT

We now define the TEDT mode of operation.

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

As usual, we expect that the secret key K is picked *uniformly* (from now on we eschew the notation SK used in section IV-B). On the other hand, PK could be either *uniformly picked* or *ensured distinct* for each session. See section IV-B: the motivation of PK is to avoid mu security degradation. To be conservative we recommend changing PK along with K during key updating. TEDT accepts $\frac{3n}{4}$-bit nonce and results in n-bit stretch. From a nonce N its 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

[2]This clarification is slightly oversimplified. As will be seen, we'll also rely on some non-standard collision properties of H.

[3]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 [51] nor $\mathsf{Hir}[\widetilde{\mathsf{IC}}]$ [52] is indifferentiable.

Table I
TEDT PARAMETERS.

	General n	$n = 128$
Key size	$2n - 1$: n secret, $n - 1$ public	255 bits: 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

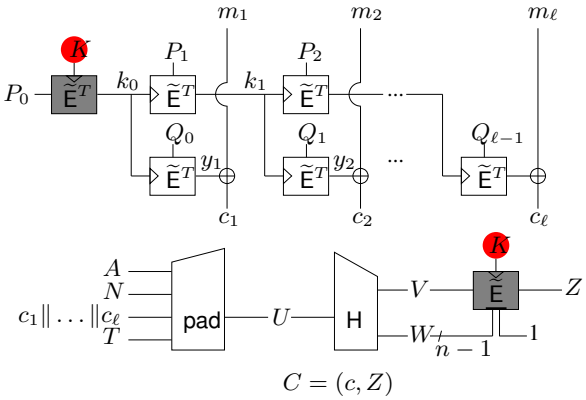

Figure 3: The TEDT AEAD. The dark blocks are KDF_K and TGF_K: for side-channel security they need heavy protection to be "leak-free". The other TBC-calls are leaking. For each square, the input to the triangle denotes the key input. The tweak $T = PK\|0$ is the public-key PK padded with 0.

$i \in [0, \ldots, 2^{\frac{n}{4}-1} - 1]$. As will be seen, i corresponds to the message block index, and the first sequence $P_i(N)$ will be used for re-keying, while $Q_i(N)$ will be used to generate key stream blocks: see Figure 3. By this, a single message cannot exceed $n \cdot 2^{\frac{n}{4}-1}$ bits.

In Table I, we list the TEDT parameters for the general n-bit case, and for the primary use case $n = 128$.

Hash and Padding. Our TBC-based hash function $\mathsf{H}[\widetilde{\mathsf{E}}]$ is a Merkle-Damgård iteration of the Hirose DBL compression function $\mathsf{Hir}[\widetilde{\mathsf{E}}]$. The domain separation constant in $\mathsf{Hir}[\widetilde{\mathsf{E}}]$ 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 reduce some constant factors in the security proof. Formally, $\mathsf{H}[\widetilde{\mathsf{E}}]$ is described in Figure V. For our purpose, the hash input is a 4-tuple $(A, N, \overrightarrow{c}, T)$, with A first to handle static and some type of incremental associated data. A suffix-free padding $\mathsf{pad}(A, N, \overrightarrow{c}, T)$ is needed for the Merkle-Damgård iteration. This padding must prevent the attacks trying to confuse the A and $\overrightarrow{c}$ fields of variable length. For this, we define $\mathsf{pad}(A, N, \overrightarrow{c}, T) := A\|N\|\overrightarrow{c}\|T\|0^*\|[|A|]_{n/2}\|[|\overrightarrow{c}|]_{n/2}$. This padding upper-bounds the length of A by $2^{n/2} - 1$, as shown in Table I. 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 [38] and the Hash-then-TBC tag generation, as discussed in section IV-B. Both KDF (in the encryption) and TGF (in the authentication) are instatiated 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 halve W and use $W\|1$ for the tweak of TGF_K. The whole process is described by the algorithm $\mathsf{TEDT}[\widetilde{\mathsf{E}}].\mathsf{Enc}_{K,PK}(N, A, M)$ in Figure V. We separate KDF and TGF from the other TBC-calls (though algorithmically equal) for conceptual convenience: both highlighting the heavily protected calls and simplifying languages. Since $T = PK\|0$, in the remaining 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 $\mathsf{TEDT}[\widetilde{\mathsf{E}}].\mathsf{Dec}_{K,PK}(N, A, C)$ in Figure V.

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 [53], [54], 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 [54, section 6.1].

algorithm $\mathsf{TEDT}[\widetilde{\mathsf{E}}].\mathsf{Enc}_{K,PK}(N, A, M)$
1. $\ell \leftarrow \lceil |M|/n \rceil$
2. parse M as $m_1 \| \ldots \| m_\ell$, with $|m_1| = \ldots = |m_{\ell-1}| = n$ and $1 \leq |m_\ell| \leq n$
3. $T \leftarrow PK \| 0$
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. $\quad y_i \leftarrow \widetilde{\mathsf{E}}^T_{k_{i-1}}(Q_{i-1}(N))$
// $Q_i(N) = N \| [i]_{\frac{n}{4}-1} \| 1$
8. $\quad c_i \leftarrow y_i \oplus m_i$
9. $\quad k_i \leftarrow \widetilde{\mathsf{E}}^T_{k_{i-1}}(P_i(N))$
// line 8 omitted for i=ℓ
10. $\overrightarrow{c} \leftarrow c_1 \| \ldots \| c_\ell$
11. $U \leftarrow \mathsf{pad}(A, N, \overrightarrow{c}, T)$
12. $V \| W \leftarrow \mathsf{H}[\widetilde{\mathsf{E}}](U)$
13. $Z \leftarrow \mathsf{TGF}(K, W, V)$, $C \leftarrow \overrightarrow{c} \| Z$
14. **return** C

algorithm $\mathsf{TEDT}[\widetilde{\mathsf{E}}].\mathsf{Dec}_{K,PK}(N, A, C)$
1. $\ell \leftarrow \lceil |C|/n \rceil - 1$
2. parse C as $\overrightarrow{c} \| Z$, with $\overrightarrow{c} = c_1 \| \ldots \| c_\ell$, $|c_1| = \ldots = |c_{\ell-1}| = |Z| = n$, and $1 \leq |c_\ell| \leq n$
3. $T \leftarrow PK \| 0$
4. $U \leftarrow \mathsf{pad}(A, N, \overrightarrow{c}, T)$
5. $V \| W \leftarrow \mathsf{H}[\widetilde{\mathsf{E}}](U)$
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. $\quad y_i \leftarrow \widetilde{\mathsf{E}}^T_{k_{i-1}}(Q_{i-1}(N))$
12. $\quad m_i \leftarrow y_i \oplus c_i$
13. $\quad k_i \leftarrow \widetilde{\mathsf{E}}^T_{k_{i-1}}(P_i(N))$
// line 12 omitted for i=ℓ
14. $M \leftarrow m_1 \| \ldots \| m_\ell$
15. **return** M

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

algorithm $\mathsf{pad}(A, N, \overrightarrow{c}, T)$
1. $u \leftarrow A \| N \| \overrightarrow{c} \| T$
2. $\ell \leftarrow |u|$
3. $\Delta \leftarrow \lceil \frac{\ell}{n} \rceil \cdot n - \ell$
4. **return** $u \| [|A|]_{n/2} \| [|\overrightarrow{c}|]_{n/2}$

algorithm $\mathsf{KDF}(K, T, I)$
1. **return** $\widetilde{\mathsf{E}}^T_K(I)$

algorithm $\mathsf{TGF}(K, W, V)$
1. **return** $\widetilde{\mathsf{E}}^{W\|1}_K(V)$

algorithm $\mathsf{TGF}^{-1}(K, W, Z)$
1. **return** $(\widetilde{\mathsf{E}}^{W\|1}_K)^{-1}(Z)$

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

Figure 4: Definition of the TEDT mode, using a TBC $\widetilde{\mathsf{E}}$.

VI. Leakage Security of TEDT

We now establish the leakage-resilient integrity and confidentiality of TEDT. Before proving muCIML2 and muCCAmL2, we specify the leakage function pair $\mathsf{L}_{\mathsf{TEDT}} = (\mathsf{L}_{\mathsf{enc}}, \mathsf{L}_{\mathsf{dec}})$. Our analysis relies on the ideal cipher model.

A. Modeling Leakage Functions

We model the leakage as *probabilistic* efficient functions manipulating and/or computing (partially) secret values. In TEDT, each computation of $\widetilde{\mathsf{E}}$ (resp., $\oplus$) comes with some additional (internal) information given by $\mathsf{L}_{\widetilde{\mathsf{E}}}$ (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 $\widetilde{\mathsf{E}}$. Indeed, while KDF and TGF both use $\widetilde{\mathsf{E}}$, the implementation of these algorithms might offer different levels of protection compared to all the other calls to $\widetilde{\mathsf{E}}$ in TEDT, included those of the hash function in Figure 2.

Moreover, we may also split the leakage trace resulting from the computation of any function $\mathsf{F} \in \{\widetilde{\mathsf{E}}, \mathsf{KDF}, \mathsf{TGF}, \ldots\}$ between its *input* and *output* parts: if $\mathsf{F}_A(X) \rightarrow Y$, $\mathsf{L}_{\mathsf{F}}(A, X) := (\mathsf{L}^{in}_{\mathsf{F}}(A; X), \mathsf{L}^{out}_{\mathsf{F}}(A; Y))$ with semicolon. This distinction comes in handy when we have to assume different level of protection for the input (e.g., N is a public input of KDF, with $A = (K, T)$) and the output (e.g., k_0 is a protected output of KDF). While a bit theoretical at first, this distinction 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 VI-C).

Finally, we insist one more time on the probabilistic feature of the leakage functions, even for $\mathsf{L}^{in}_{\mathsf{F}}$ and $\mathsf{L}^{out}_{\mathsf{F}}$ and any F (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 $[\mathsf{L}_{\mathsf{F}}]^p$ for the vector of p leakage traces of F. Because of the plentiful possible uses of $\widetilde{\mathsf{E}}$, we will next denote its input-output leakage function pair as (L^{in},L^{out}) for simplicity.

Oracle-free leakage function. An artifact of modeling leakages as probabilistic functions is that $\mathsf{L}_{\widetilde{\mathsf{E}}}$ might contain "future" calls to $\widetilde{\mathsf{E}}$ [22]. 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" [23]. For instance, if the leakage resulting from $\widetilde{\mathsf{E}}(k_0, T, P_1(N)) \rightarrow k_1$ in TEDT might also already contain $y_2 = \widetilde{\mathsf{E}}(k_1, T, Q_1(N))$, the value y_2 cannot remain unpredictable in the next block. Preventing $\mathsf{L}_{\widetilde{\mathsf{E}}}(k_0, *, *)$ from calling $\widetilde{\mathsf{E}}(k_1, *, *)$ 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,[4] there are only two reliable existing proof approaches which are (i) the leakage simulatability assumption in the standard model [39], and (ii) the non-invertible leakage assumption in the ideal model [23]. 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 $\mathsf{muCCAmL2}$) based on the former approach in Appendix D.

With this in mind, and to prevent future computation attacks in the ideal model, we assume oracle-free leakage functions [23]: they cannot make any call to $\widetilde{\mathsf{IC}}$, which is natural for an implementation not to evaluate computations that are unrelated to its current state — e.g., $\widetilde{\mathsf{E}}(k_0, *, *)$ in our above example. Therefore, we will say that the leakage function associated to F is oracle-free, if $\tau(\mathsf{L}_{\mathsf{F}}^{in}) = \tau(\mathsf{L}_{\mathsf{F}}^{out}) = \emptyset$, where $\tau(\mathsf{L}_{\mathsf{F}}^{*})$ is the transcript of queries and answers made by $\mathsf{L}_{\mathsf{F}}^{*}$ to $\widetilde{\mathsf{IC}}$ when $\mathsf{L}_{\mathsf{F}}^{*}$ is evaluated on its inputs.

As discussed in [23], 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.

B. muCIML2 *of* TEDT

We now prove the $\mathsf{muCIML2}$ security of TEDT in the "unbounded leakage" setting [19], [20]. Formally, we assume that all the intermediate values completely leak except the master key K of KDF_K and TGF_K which remains secret. This means that $\mathsf{L}^{in}(K, T; X) = \{K, T, X\}$ and $\mathsf{L}^{out}(K, T; Y) = \{K, T, Y\}$ as well as $\mathsf{L}_{\oplus}(Y, M) = \{Y, M, Y \oplus M\}$. We denote this family of leakage functions by L^*. Since we are analyzing TEDT in the ideal TBC model, we can prove information theoretic security, and we only need to limit the number of adversarial queries.

Therefore, we denote $\overrightarrow{q} = (q_e, q_d, q_{\widetilde{\mathsf{IC}}})$, and using the definition given in Section III-B2, we denote

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

where the maximum is taken over all $(\overrightarrow{q}, t)$-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, $n \geq 6$, $2\sigma + 3(q_e + q_d) + q_{\widetilde{\mathsf{IC}}} \leq 2^n/8$, and leakage L^* is "unbounded" as above. Then*

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

in the ideal TBC model. The proof is available in Appendix E.

As long as we carefully protect the key of KDF_K and TGF_K, the $\mathsf{muCIML2}$ bound is asymptotically optimal $O\left(\frac{u^2}{2^{2n}} + \frac{n^2\sigma + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}\right)$. Concretely, when $n = 128$, the integrity security is up to $u \approx 2^{126}$ users, $\sigma \approx 2^{114}$ blocks, and $q_{\widetilde{\mathsf{IC}}} \approx 2^{114}$ offline computation. If the u public-keys are ensured to be *distinct*, then we could drop the terms $\frac{u^2}{2^{2n}}$ and $\frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n$, but the bound does not substantially improve.

Intuition of the proof. All the internal values of the TBC-calls, except K, are given to the adversary $\mathcal{A}$. Conceptually, we could "wrap up" these leaked TBC-queries into the offline computation power of $\mathcal{A}$. This simplifies the situation and the interaction between $\mathcal{A}$ and a hash-then-TBC authentication scheme, where the offline computation power of $\mathcal{A}$ balloons up to $O(\sigma + q_{\widetilde{\mathsf{IC}}})$. Then, we have two steps:

First, we replace $\mathsf{KDF}_{K_1}, \mathsf{TGF}_{K_1}, \ldots, \mathsf{KDF}_{K_u}, \mathsf{TGF}_{K_u}$ 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* $K_i \neq K_j$ (they are not always independent and thus the term $\frac{u^2}{2^n}$ does not emerge), and prove the indistinguishability of this transition using the H-coeffcients technique [57]. We need to show the offline computation does not introduce the undesired term $O\left(\frac{u(\sigma + q_{\widetilde{\mathsf{IC}}})}{2^n}\right)$. We rely on the fact that these $O(\sigma + q_{\widetilde{\mathsf{IC}}})$ offline queries are "separated" by the tweak input for this purpose: the offline queries with the tweak $W_i\|1$ are only helpful for "breaking" (either recovering the key or distinguishing) the TGF calls with the tweak $W_i\|1$. Therefore, as long as a single value $W_i\|1$ is not used by too many encryption and decryption queries (say, n^2 queries), each offline query simultaneously "targets" at most n^2 TGF calls.

[4]Using the standard model and PPT leakage functions, existing provably secure re-keying schemes typically require a more complicated alternating structure [22], [55], or public randomness [23], [56] in order to overcome the "future computation" attack.

It suffices to prove that the probability to obtain n^2 semi-collisions on the outputs of H is small enough: by a careful analysis, we show this probability is $O\Big(\frac{(\sigma+q_{\widetilde{\mathsf{IC}}})^n}{2^{n(n-1)}}\Big) = O\Big(\frac{\sigma+q_{\widetilde{\mathsf{IC}}}}{2^n}\Big)$. By these arguments, the term $\frac{O(u(\sigma+q_{\widetilde{\mathsf{IC}}}))}{2^n}$ does not emerge in this step (while there is a term $\frac{n^2\sigma+n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}$ emerging instead).

At this point, the adversary faces an interaction with information that is independent of the secret keys $\mathbf{K}$. We then argue the unforgeability with two crucial goals: (i) prove that user-key collision does not help breaking integrity; (ii) prove that the forging probability is $\frac{\sigma^2}{2^{2n}}$ instead of $\frac{\sigma^2}{2^n}$.

For the goal (i), note that if the public-keys T_i are not part of the input of the hash, then forgery is obvious using user key collision (following the idea in section IV-B). But after T_i, T_j are appended, since the probability to have $K_i\|T_i = K_j\|T_j$ is reduced to $\frac{u^2}{2^n}$, we have $K_i = K_j \Rightarrow T_i \neq T_j \Rightarrow \mathsf{TGF}_{K_i}(\mathsf{H}(\mathsf{pad}(A, N, \overrightarrow{c}, T_i))) \neq \mathsf{TGF}_{K_i}(\mathsf{H}(\mathsf{pad}(A, N, \overrightarrow{c}, T_j)))$ since the two hash digests are different, rendering the key collision forgery infeasible.

For the goal (ii), since the hash digest is $2n$-bit, the collision probability is reduced to $\frac{q_{\widetilde{\mathsf{IC}}}^2}{2^{2n}}$ which blocks the hash-collision attack (more precisely, $O\Big(\frac{(\sigma+q_{\widetilde{\mathsf{IC}}})^2}{2^{2n}}\Big)$, with the above remark on offline computation power). This is slightly oversimplified, and we invite the reader to read Appendix E for more details. Also, (not surprisingly) the proof crucially relies on the feature that decryption only makes backward calls TGF^{-1}: otherwise there indeed exists attacks, as discussed in section IV-A. See appendix E-D for more discussion.

C. Non-Invertible Leakage Assumption

We now turn to the leakage-resilient confidentiality of TEDT. Section VI-A already highlighted the need of additional leakage assumptions — a standard model proof is deferred to Appendix D. To capture the harmlessness of the leakages, we here follow [23] and require that they preserve the secrecy of the ephemeral TBC key in the following sense: the probability that an adversary recovers the ephemeral key before it is being refreshed should be small.

Yu et al. required the adversary $\mathcal{A}$ to *precisely output the secret* [23, 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. [23] (which should be an artifact of the proof technique). More formally, we define

$$\mathbf{Adv}^{\text{2-up}[q]}_{T,P_A,P_B}(\mathcal{A}) := \Pr_{\widetilde{\mathsf{IC}},s_1}\Big[s_2 \leftarrow \widetilde{\mathsf{IC}}^T_{s_1}(P_A), z \leftarrow \widetilde{\mathsf{IC}}^T_{s_1}(P_B), \\ Guesses \leftarrow \mathcal{A}^{\widetilde{\mathsf{IC}}}(s_2, z, \mathsf{leak}) : s_1 \in Guesses\Big], \quad (2)$$

where T, P_A, P_B are public constants such that $P_A \neq P_B$, $|Guesses| = q$, and $\mathcal{A}$'s input leak is a list of leakages depending on a value s_0 specified by $\mathcal{A}$:[5]

$$\mathsf{leak} = \Big(\big[\mathsf{L}^{out}(s_0, T; s_1), \mathsf{L}^{in}(s_1, T; P_A), \\ \mathsf{L}^{out}(s_1, T; s_2), \mathsf{L}^{in}(s_1, T; P_B), \mathsf{L}^{out}(s_1, T; z)\big]^p\Big). \quad (3)$$

And we further define

$$\mathbf{Adv}^{\text{2-up}[q]}(p, q_{\widetilde{\mathsf{IC}}}, t) := \max\Big\{\mathbf{Adv}^{\text{2-up}[q]}_{T,P_A,P_B}(\mathcal{A})\Big\}, \quad (4)$$

where the maximum is taken over all valid T, P_A, P_B and with all adversaries that repeat their measurements p times, makes $q_{\widetilde{\mathsf{IC}}}$ $\widetilde{\mathsf{IC}}$-queries, and runs in time t.

Understanding 2-up[q] Advantage. Equation (2) defines a leakage property of a small "unit" of TEDT, pictured in Figure 5 (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 (3). 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 $\mathsf{muCCAmL2}$ 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 (4) should still be much 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 challenging SCA adversary $\mathcal{A}$. This along with Theorem 2 allows deriving concrete limits on the capability of the implementation (like in [27]).

[5]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 (6).

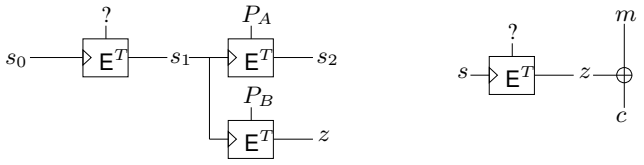

Figure 5: (Left) Illustration on the 2-up assumption; (Right) The "basic" message manipulating operation.

1: **Tester for UP** $\mathbf{Adv}^{\mathsf{2\text{-}up}}[q]$
2: Let the challenging adversary $\mathcal{A}$ give s_0
3: Pick the secret: $s_1 \xleftarrow{\$} \{0,1\}^n$
4: $s_2 \leftarrow \widetilde{\mathsf{E}}^T_{s_1}(P_A)$, $z \leftarrow \widetilde{\mathsf{E}}^T_{s_1}(P_B)$
5: **return** $[\mathsf{L}^{out}(s_0, T; s_1), \mathsf{L}^{in}(s_1, T; P_A), \mathsf{L}^{out}(s_1, T; s_2)]^p$,
6: and $[\mathsf{L}^{in}(s_1, T; P_B), \mathsf{L}^{out}(s_1, T; z)]^p$
7: Let the challenging adversary $\mathcal{A}$ output q guesses $k_1, \ldots, k_q$, the adversary $\mathcal{A}$ wins as long as $s_1 \in \{k_1, \ldots, k_q\}$

D. Capturing the (In)Security of the XOR

As a last step before moving properly into establishing the muCCAmL2 bound, we have to measure the leakage resulting from XORing the (supposedly) random block stream with the message blocks in TEDT. To capture this concrete information we follow Pereira et al. [38] and we define

$$\mathbf{Adv}^{\mathsf{LORL2}}_{T}(\mathcal{A}) := \Big|\Pr_{\widetilde{\mathsf{IC}},z}\Big[c^0 \leftarrow z \oplus m^0 : \mathcal{A}^{\widetilde{\mathsf{IC}}}(c^0, \mathsf{leak}_0) \Rightarrow 1\Big] - \Pr_{\widetilde{\mathsf{IC}},z}\Big[c^1 \leftarrow z \oplus m^1 : \mathcal{A}^{\widetilde{\mathsf{IC}}}(c^1, \mathsf{leak}_1) \Rightarrow 1\Big]\Big| \quad , \quad (5)$$

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

$$\mathsf{leak}_b = \Big([\mathsf{L}^{out}(s, T; z)]^p, \mathsf{L}_{\oplus}(z, m^b), [\mathsf{L}_{\oplus}(z, c^b)]^{p-1}\Big). \quad (6)$$

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

$$\mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\widetilde{\mathsf{IC}}}, t) := \max_{T,\mathcal{A}} \Big\{\mathbf{Adv}^{\mathsf{LORL2}}_{T}(\mathcal{A})\Big\}. \quad (7)$$

Understanding LORL2 Advantage. Equation (5) defines the information an adversary might extract from the "basic" message manipulation made in TEDT, which involved XORing as pictured in Figure 5 (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 [38], [58], 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 $\mathbf{Adv}^{\mathsf{LORL2}}(p, q_{\widetilde{\mathsf{IC}}}, t)$ is negligible. So our goal here is to faithfully reduce the muCCAmL2 to simple and precise pieces that are more easy to protect as isolate 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" [59]. 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 a inevitable rate* during the encryption, rather than to argue that the advantage *turns better* by designs. So in this sense, TEDT is a *security-preserving domain extender* for a "single-block" encryption operation. On the practical side, the value $\mathbf{Adv}^{\mathsf{LORL2}}$ can be similarly measured by a tester, see Appendix F, and it is easier to study the relevant protection and advantage than to study those of the entire modes.

E. 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 $\widetilde{\mathsf{E}}(k, t; x) \to y$ (excluding KDF- and TGF-calls),[6]

[6]We remark that "$\mathsf{L}^{out}_{\mathsf{KDF}}(k, t; x)$", the "output leakage" of KDF, does not need to completely hide x. It can leak information about x as long as: (i) the leakage is comparable to L^{out}, and (ii) it is independent of k. But for simplicity we did not formalize this observation.

- $\mathsf{L}_{\oplus}(a, b)$ 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 $\overrightarrow{q} = (q_m, q_e, q_d, p-1, q_{\widetilde{\mathsf{IC}}})$ and we define

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT},\mathsf{L}}(u, \overrightarrow{q}, t, \sigma) := \max\Big\{\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathcal{A},\mathsf{TEDT},\mathsf{L},u}\Big\},$$

where the maximum is taken over all $(\overrightarrow{q}, t)$-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 as above, if the leakage functions* L^{in}, L^{out}, $\mathsf{L}_{\oplus}$ *satisfy the assumptions specified by Equation (5) and Equation (2), then the following holds:*

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT}}(u, \overrightarrow{q}, t, \sigma) \le \\ &\frac{1}{n!}\cdot\Big(\frac{4u}{2^n}\Big)^n + \frac{2u^2}{2^{2n}} + \frac{(3n^2+26)(4\sigma + 6(q_m+q_e+q_d)+2q_{\widetilde{\mathsf{IC}}})}{2^n} \\ &+ \sigma\cdot\mathbf{Adv}^{\mathsf{LORL2}}(p, q^*, t^*) + 2\sigma\cdot\mathbf{Adv}^{\mathsf{2\text{-}up}[q^*]}(p, q^*, t^*),\end{aligned} \tag{8}$$

where $\mathbf{Adv}^{\mathsf{LORL2}}$ *and* $\mathbf{Adv}^{\mathsf{2\text{-}up}[q_{\widetilde{\mathsf{IC}}}+q^*]}$ *are defined in Eq. (7) and Eq. (4) respectively,* $q^* = q_{\widetilde{\mathsf{IC}}} + 4\sigma + 6(q_e+q_d+q_m)$, $t^* = O(t + p\sigma t_l)$, *and* t_l *is the total time for evaluating* L^{in} *and* L^{out}.

The proof follows the approach of [31] which is built upon the well established method of [38]. See Appendix G.

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

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

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 [55].

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 $\sigma\cdot\mathbf{Adv}^{TBC}$: this is unavoidable for hybrid-based proofs, see [36], [37]. However, for the black-box setting the term $\sigma\cdot\mathbf{Adv}^{TBC}$ 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, which is clearly beyond Equation (3), our assumption of security with 2 traces. In all, our assumption Equation (3), though a bit conservative, tightly results in a birthday-type bound.

VII. Black-box Security Analysis of TEDT

In this section, we no more consider leakages. We first prove the *extended* mu CCA security muCCAm\$* for TEDT in the ideal TBC model. Formally, we define

$$\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathsf{TEDT}}(u, \overrightarrow{q}, \sigma) := \max\Big\{\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathsf{TEDT},\mathcal{A},u}\Big\},$$

where $\overrightarrow{q} = (q_m, q_e, q_d, q_{\widetilde{\mathsf{IC}}})$, and the maximum is taken over all $(\overrightarrow{q}, t)$-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 3. *When the* u *public-keys* $T_1, \ldots, T_u$ *are uniformly distributed,* $n \ge 6$, *and* $2\sigma + 3(q_e+q_d) + q_{\widetilde{\mathsf{IC}}} \le 2^n/8$, *then the following holds in the ideal TBC model:*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAm\$^*}}_{\mathsf{TEDT}}(u, \overrightarrow{q}, \sigma) &\le \frac{2u^2}{2^{2n}} + \frac{1}{n!}\cdot\Big(\frac{4u}{2^n}\Big)^n \\ &+ \frac{(7n^2+26)(4\sigma+6(q_e+q_d)+2q_{\widetilde{\mathsf{IC}}})}{2^n}.\end{aligned}$$

The complete proof is available in Appendix H. Partly thanks to the GCM-like counters we have this $O\Big(\frac{u^2}{2^{2n}} + \frac{n^2\sigma + n^2 q_{\widetilde{\mathsf{IC}}}}{2^n}\Big)$ bound similar to Theorem 1. Again, the terms $\frac{2u^2}{2^{2n}}$ and $\frac{1}{n!}\cdot\Big(\frac{4u}{2^n}\Big)^n$ disappear if the u public-keys are distinct. While it seems that the bound is not affected by the q_m non-challenge queries, these queries affect σ which, in turn, affects the bound. The bound of GCM-SIV with KDF is at best $\frac{\sigma\ell_{max}+q_{\widetilde{\mathsf{IC}}}}{2^n}$, which we surpass when the maximal query length ℓ_{max} exceeds n^2: see Appendix H-C for the details.

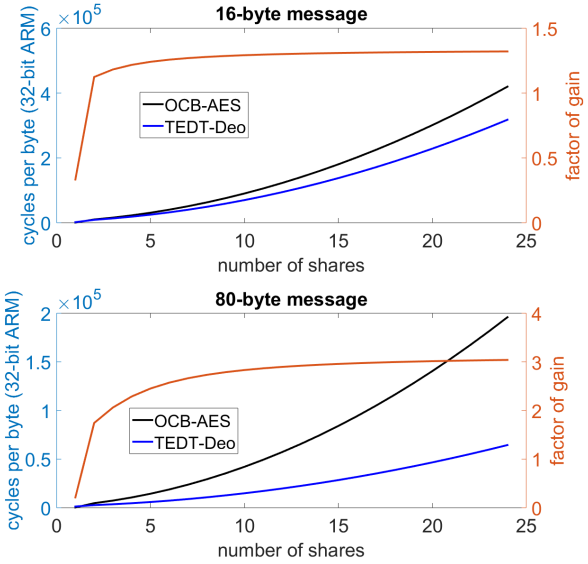

Figure 6: Performance evaluation (I): uniformly masked implementation of OCB based on the AES vs. leveled implementation of TEDT based on Deoxys-BC-256.

KDM Security. Finally, in the ideal TBC model, TEDT achieves birthday $2^{(3n/4)/2} = 2^{3n/8}$ security against key-dependent-message attacks (KDM) when the nonce is random and no restriction is on the message deriving functions, an asymptotically optimal $2^n/n^2$ KDM security when the nonce is respecting and message deriving functions are oracle-free. A formal discussion is given in Appendix I.

VIII. Performance Evaluation of TEDT

In order to put forward the relevance of leveled implementations as a solution to reach high-security against side-channel attacks, we conclude this paper with an evaluation of the performances that can be reached by TEDT in software, based on the masked AES implementations proposed by Goudarzi and Rivain in [16]. We specifically consider Table 3 in this reference, which provides cycle counts in function of the number of shares in the masking scheme. We then selected Deoxys-BC-256 as instance of TBC to use in TEDT, which was recently elected as a finalist of the CAESAR competition [60]. This choice is mostly motivated by the AES-based structure of this cipher, which enables exploiting the same masking countermeasure. In particular, we note that 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 compared 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 evaluations can be found in Figures 6 and 7 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 [16]). The second important conclusion is that the factor of gain approximately converges towards $\frac{l+2}{2}$ 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. We finally note that the factor of gain is actually slightly lower than $\frac{l+2}{2}$, 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 [61] would actually allow an even better trend.

We note finally that we expect similar (energy) gains to be observed in hardware. The only difference is that the energy overheads of the masking countermeasure in this case may be both due to an increase of the cycle count (as in software) and to a increased area [17]. Also, hardware implementations provide additional opportunities to implement the weakly protected implementation at very minimum energy cost (e.g., thanks to low-latency designs [62]).

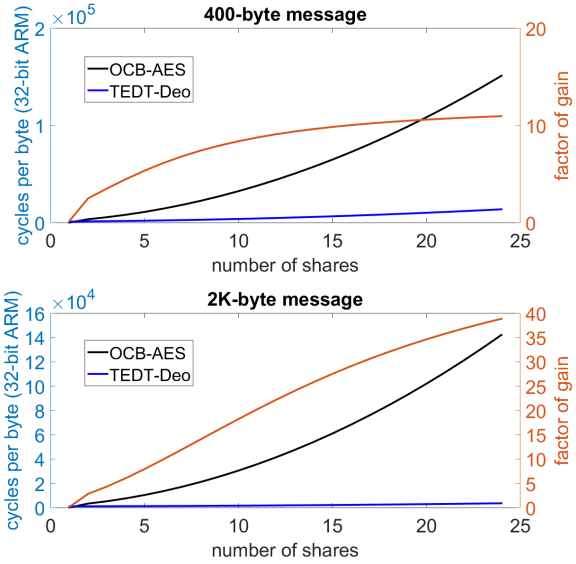

Figure 7: Performance evaluation (II): uniformly masked implementation of OCB based on the AES vs. leveled implementation of TEDT based on Deoxys-BC-256.

Acknowledgments. Thomas Peters and François-Xavier Standaert are respectively postdoctoral researcher and senior associate researcher of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in parts by the ERC project Consolidator Grant 724725 (acronym SWORD) and by the European Union and Walloon Region FEDER USERMedia project 501907-379156.

References

[1] J. Katz and M. Yung, "Unforgeable Encryption and Chosen Ciphertext Secure Modes of Operation," in *FSE*, pp. 284–299.
[2] M. Bellare and C. Namprempre, "Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm," *J. Cryptology*, vol. 21, no. 4, pp. 469–491, 2008.
[3] J.-P. Aumasson, S. Babbage, D. J. Bernstein, C. Cid, J. Daemen, O. Dunkelman, K. Gaj, S. Gueron, P. Junod, A. Langley, D. McGrew, K. Paterson, B. Preneel, C. Rechberger, V. Rijmen, M. Robshaw, P. Sarkar, P. Schaumont, A. Shamir, and I. Verbauwhede, "CHAE: Challenges in Authenticated Encryption," ECRYPT-CSA D1.1, Revision 1.05, 1 March 2017, https://chae.cr.yp.to/whitepaper.html.
[4] T. Eisenbarth, T. Kasper, A. Moradi, C. Paar, M. Salmasizadeh, and M. T. M. Shalmani, "On the Power of Power Analysis in the Real World: A Complete Break of the KeeLoqCode Hopping Scheme," in *CRYPTO 2008*, pp. 203–220.
[5] A. Moradi, A. Barenghi, T. Kasper, and C. Paar, "On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from xilinx virtex-ii fpgas," in *ACM CCS 2011*, pp. 111–124.
[6] Y. Zhou, Y. Yu, F. Standaert, and J. Quisquater, "On the Need of Physical Security for Small Embedded Devices: A Case Study with COMP128-1 Implementations in SIM Cards," in *Financial Crypto 2013*, pp. 230–238.
[7] J. Balasch, B. Gierlichs, O. Reparaz, and I. Verbauwhede, "DPA, Bitslicing and Masking at 1 GHz," in *CHES 2015*, pp. 599–619.
[8] D. Genkin, I. Pipman, and E. Tromer, "Get your hands off my laptop: physical side-channel key-extraction attacks on pcs - extended version," *J. Cryptographic Engineering*, vol. 5, no. 2, pp. 95–112, 2015.
[9] D. Genkin, A. Shamir, and E. Tromer, "Acoustic cryptanalysis," *J. Cryptology*, vol. 30, no. 2, pp. 392–443, 2017.
[10] D. Dinu and I. Kizhvatov, "EM analysis in the iot context: Lessons learned from an attack on thread," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2018, no. 1, pp. 73–97.
[11] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer, 2007.
[12] P. Rogaway, M. Bellare, J. Black, and T. Krovetz, "OCB: a block-cipher mode of operation for efficient authenticated encryption," in *ACM CCS '01*, pp. 196–205.
[13] D. Whiting, R. Housley, and N. Ferguson, "Counter with CBC-MAC (CCM)," *RFC*, vol. 3610, pp. 1–26, 2003. [Online]. Available: https://doi.org/10.17487/RFC3610
[14] D. A. McGrew and J. Viega, "The Security and Performance of the Galois/Counter Mode (GCM) of Operation," in *INDOCRYPT 2004*, pp. 343–355.
[15] S. Chari, C. S. Jutla, J. R. Rao, and P. Rohatgi, "Towards sound approaches to counteract power-analysis attacks," in *CRYPTO 1999*, pp. 398–412.
[16] D. Goudarzi and M. Rivain, "How Fast Can Higher-Order Masking Be in Software?" in *EUROCRYPT 2017, Part I*, pp. 567–597.
[17] H. Groß, S. Mangard, and T. Korak, "An Efficient Side-Channel Protected AES Implementation with Arbitrary Protection Order," in *CT-RSA 2017*, pp. 95–112.
[18] G. Barwell, D. P. Martin, E. Oswald, and M. Stam, "Authenticated Encryption in the Face of Protocol and Side Channel Leakage," in *ASIACRYPT 2017, Part I*, pp. 693–723.
[19] F. Berti, F. Koeune, O. Pereira, T. Peters, and F. Standaert, "Ciphertext Integrity with Misuse and Leakage: Definition and Efficient Constructions with Symmetric Primitives," pp. 37–50.
[20] F. Berti, O. Pereira, T. Peters, and F. Standaert, "On Leakage-Resilient Authenticated Encryption with Decryption Leakages," *IACR Trans. Symmetric Cryptol.*, vol. 2017, no. 3, pp. 271–293, 2017.
[21] C. Dobraunig, M. Eichlseder, S. Mangard, F. Mendel, and T. Unterluggauer, "ISAP - Towards Side-Channel Secure Authenticated Encryption," *IACR Trans. Symmetric Cryptol.*, vol. 2017, no. 1, pp. 80–105, 2017.
[22] S. Dziembowski and K. Pietrzak, "Leakage-Resilient Cryptography," in *FOCS 2008*, pp. 293–302.
[23] Y. Yu, F. Standaert, O. Pereira, and M. Yung, "Practical Leakage-Resilient Pseudorandom Generators," in *ACM CCS 2010*, pp. 141–151.

[24] M. Medwed, F. Standaert, J. Großschädl, and F. Regazzoni, "Fresh re-keying: Security against side-channel and fault attacks for low-cost devices," in *AFRICACRYPT 2010*, pp. 279–296.
[25] N. Veyrat-Charvillon, M. Medwed, S. Kerckhof, and F. Standaert, "Shuffling against Side-Channel Attacks: A Comprehensive Study with Cautionary Note," in *ASIACRYPT 2012*, pp. 740–757.
[26] S. Bhasin, S. Guilley, L. Sauvage, and J. Danger, "Unrolling cryptographic circuits: A simple countermeasure against side-channel attacks," in *CT-RSA 2010*, pp. 195–207.
[27] S. Gueron and Y. Lindell, "Better Bounds for Block Cipher Modes of Operation via Nonce-Based Key Derivation," in *ACM CCS 2017*, pp. 1019–1036.
[28] P. Bose, V. T. Hoang, and S. Tessaro, "Revisiting AES-GCM-SIV: Multi-user Security, Faster Key Derivation, and Better Bounds," in *EUROCRYPT 2018, Part I*, pp. 468–499.
[29] M. Bellare and B. Tackmann, "The Multi-user Security of Authenticated Encryption: AES-GCM in TLS 1.3," in *CRYPTO 2016, Part I*, pp. 247–276.
[30] T. Ashur, O. Dunkelman, and A. Luykx, "Boosting Authenticated Encryption Robustness with Minimal Modifications," in *CRYPTO 2017, Part III*, pp. 3–33.
[31] C. Guo, O. Pereira, T. Peters, and F.-X. Standaert, "Leakage-Resilient Authenticated Encryption with Misuse in the Leveled Leakage Setting: Definitions, Separation Results, and Constructions," Cryptology ePrint Archive, Report 2018/484, 2018.
[32] P. Rogaway and T. Shrimpton, "Deterministic Authenticated-Encryption: A Provable-Security Treatment of the Key-Wrap Problem," in *EUROCRYPT 2006*, pp. 373–390.
[33] M. Bellare and S. Keelveedhi, "Authenticated and Misuse-Resistant Encryption of Key-Dependent Data," in *CRYPTO 2011*, pp. 610–629.
[34] S. Tessaro, "Optimally Secure Block Ciphers from Ideal Primitives," in *ASIACRYPT 2015, Part II*, pp. 437–462.
[35] J. Kilian and P. Rogaway, "How to Protect DES Against Exhaustive Key Search (an Analysis of DESX)," *J. Cryptology*, vol. 14, no. 1, pp. 17–35, 2001.
[36] T. Shrimpton and R. S. Terashima, "Salvaging Weak Security Bounds for Blockcipher-Based Constructions," in *ASIACRYPT 2016, Part I*, pp. 429–454.
[37] B. Mennink, "Insuperability of the Standard Versus Ideal Model Gap for Tweakable Blockcipher Security," in *CRYPTO 2017, Part II*, pp. 708–732.
[38] O. Pereira, F. Standaert, and S. Vivek, "Leakage-Resilient Authentication and Encryption from Symmetric Cryptographic Primitives," in *ACM CCS 2015*, pp. 96–108.
[39] F. Standaert, O. Pereira, and Y. Yu, "Leakage-Resilient Symmetric Cryptography under Empirically Verifiable Assumptions," in *CRYPTO 2013*, 2013, pp. 335–352.
[40] S. Micali and L. Reyzin, "Physically Observable Cryptography (Extended Abstract)," in *TCC 2004*, pp. 278–296.
[41] C. O'Flynn and Z. D. Chen, "Side channel power analysis of an AES-256 bootloader," in *CCECE 2015*, pp. 750–755.
[42] Y. Dodis and P. Puniya, "On the Relation Between the Ideal Cipher and the Random Oracle Models," in *TCC 2006*, pp. 184–206.
[43] C. S. Jutla, "Encryption Modes with Almost Free Message Integrity," *J. Cryptology*, vol. 21, no. 4, pp. 547–578, 2008.
[44] M. Liskov, R. L. Rivest, and D. A. Wagner, "Tweakable Block Ciphers," *J. Cryptology*, vol. 24, no. 3, pp. 588–613, 2011.
[45] K. Minematsu, "Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions," in *EUROCRYPT 2014*, pp. 275–292.
[46] A. Chakraborti, T. Iwata, K. Minematsu, and M. Nandi, "Blockcipher-Based Authenticated Encryption: How Small Can We Go?" in *CHES 2017*, pp. 277–298.
[47] A. Chakraborti, N. Datta, M. Nandi, and K. Yasuda, "Beetle Family of Lightweight and Secure Authenticated Encryption Ciphers," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, vol. 2018, no. 2, pp. 218–241.
[48] S. Banik, A. Bogdanov, A. Luykx, and E. Tischhauser, "SUNDAE: Small Universal Deterministic Authenticated Encryption for the Internet of Things," *IACR Trans. Symmetric Cryptol.*, vol. 2018, no. 3, pp. 1–35, Sep.
[49] D. P. Martin, E. Oswald, M. Stam, and M. Wójcik, "A Leakage Resilient MAC," in *IMACC 2015*, pp. 295–310.
[50] S. Hirose, "Some Plausible Constructions of Double-Block-Length Hash Functions," in *FSE 2006*, pp. 210–225.
[51] J. Coron, Y. Dodis, C. Malinaud, and P. Puniya, "Merkle-Damgård Revisited: How to Construct a Hash Function," in *CRYPTO 2005*, pp. 430–448.
[52] B. Mennink, "Indifferentiability of Double Length Compression Functions," in *IMACC2013*, pp. 232–251.
[53] T. Peyrin and Y. Seurin, "Counter-in-Tweak: Authenticated Encryption Modes for Tweakable Block Ciphers. Version 20160524:153228," Cryptology ePrint Archive, Report 2015/1049, 2015.
[54] T. Iwata, K. Minematsu, T. Peyrin, and Y. Seurin, "ZMAC: A Fast Tweakable Block Cipher Mode for Highly Secure Message Authentication," in *CRYPTO 2017, Part III*, pp. 34–65.
[55] K. Pietrzak, "A Leakage-Resilient Mode of Operation," in *EUROCRYPT 2009*, pp. 462–482.
[56] S. Faust, K. Pietrzak, and J. Schipper, "Practical Leakage-Resilient Symmetric Cryptography," in *CHES 2012*, pp. 213–232.
[57] S. Chen and J. P. Steinberger, "Tight Security Bounds for Key-Alternating Ciphers," in *EUROCRYPT 2014*, pp. 327–350.
[58] C. Hazay, A. López-Alt, H. Wee, and D. Wichs, "Leakage-Resilient Cryptography from Minimal Assumptions," *J. Cryptology*, vol. 29, no. 3, pp. 514–551, 2016.
[59] S. Tessaro, "Security Amplification for the Cascade of Arbitrarily Weak PRPs: Tight Bounds via the Interactive Hardcore Lemma," in *TCC 2011*, pp. 37–54.
[60] J. Jean, I. Nikolic, T. Peyrin, and Y. Seurin, "Deoxys v1.41," *Submitted to CAESAR, October 2016*.
[61] V. Grosso, G. Leurent, F. Standaert, and K. Varici, "Ls-designs: Bitslice encryption for efficient masked software implementations," in *FSE 2014*, pp. 18–37.
[62] S. Kerckhof, F. Durvaux, C. Hocquet, D. Bol, and F. Standaert, "Towards Green Cryptography: A Comparison of Lightweight Ciphers from the Energy Viewpoint," in *CHES 2012*, pp. 390–407.
[63] T. Jager, M. Stam, R. Stanley-Oakes, and B. Warinschi, "Multi-key Authenticated Encryption with Corruptions: Reductions Are Lossy," in *TCC 2017, Part I*, pp. 409–441.
[64] P. Farshim, L. Khati, and D. Vergnaud, "Security of Even-Mansour Ciphers under Key-Dependent Messages," *IACR Trans. Symmetric Cryptol.*, vol. 2017, no. 2, pp. 84–104.
[65] J. Black, P. Rogaway, and T. Shrimpton, "Encryption-Scheme Security in the Presence of Key-Dependent Messages," in *SAC 2002*, pp. 62–75.
[66] T. Holenstein, R. Künzler, and S. Tessaro, "The Equivalence of the Random Oracle Model and the Ideal Cipher Model, Revisited," in *STOC 2011*, pp. 89–98.

APPENDIX 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. [19] and Guo et al. [31]. 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 [19]. 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 [18]), 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 [18]). 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 [18]): 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 [53] and ZAE [54] are two concrete examples); in sponge, message is absorbed block-by-block by a iterative process (e.g., SUNDAE [48]). In all, the prefix of the leakage $\mathsf{L}_{\mathsf{enc}}(m_1\|m_2\|\dots\|m_\ell)$ only depends on m_1 (or m_1 and m_2, when Feistel networks are used). Therefore, using $\mathsf{L}_{\mathsf{enc}}(m_1\|\dots\|m_\ell)$ as a template, for $M^0 = m_1\|\dots$ and $M^1 = m_1'\|\dots$, $m_1' \neq m_1$, 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 [18], [48], [53], [54]: the former three only claimed security *without leakage*, while the last [18] 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.

APPENDIX B
CCAm$ ⇒ muCCAm$, CIML2 ⇒ muCIML2, CCAmL2 ⇒ muCCAmL2

Here we sketch the proof ideas for the three equivalence relations. For CCAm$ ⇒ muCCAm$, we consider the real muCCAm$ setting, replace the real user oracles by the ideal oracles one-by-one, and this eventually reaches the ideal muCCAm$ setting. Each such replacement introduces a security loss of $\mathbf{Adv}^{\mathsf{CCAm\$}}$, thus the conclusion $\mathbf{Adv}^{\mathsf{muCCAm\$}} \leq u \cdot \mathbf{Adv}^{\mathsf{CCAm\$}}$. The proof ideas for the other are basically the same. Let $\mathsf{X} \in \{\mathsf{CIML2}, \mathsf{CCAmL2}\}$. We show that if a scheme isn't muX secure then a X-adversary $\mathcal{A}_{\mathsf{X}}$ can be built as well. Basically, $\mathcal{A}_{\mathsf{X}}$ picks additional $u-1$ keys according to the relevant key distribution and simulates their interfaces. These plus its own challenge oracle form u simulated users. $\mathcal{A}_{\mathsf{X}}$ runs these simulated users against the muX-adversary $\mathcal{A}_{\mathsf{muX}}$ (which we assumed existing). With probability $\geq \frac{1}{u}$ ($> \frac{1}{u}$ in the CCAmL2 setting when the multiple challenge queries are made to different users), $\mathcal{A}_{\mathsf{muX}}$ collapses the user simulated by the challenge oracle of $\mathcal{A}_{\mathsf{X}}$, and this results in $\mathcal{A}_{\mathsf{X}}$ succeeding. Thus the conclusion $\mathbf{Adv}^{\mathsf{muX}} \leq u \cdot \mathbf{Adv}^{\mathsf{X}}$. This "simulation" approach requires the assumption that given the keys, the *leaking oracle* can be simulated. This is typically fulfilled in theory: most theoretical models for leakage, including PPT leakage functions [22], [23], [55] and simulatable leakage [39] (in the latter setting a public "leakage oracle" is available, which models the possibility of offline training in real side-channel attacks) allow efficiently simulating (given the key, of course), and this also seems the case in reality. On the downside, Jager et al. showed that the factor u is unavoidable for generic reductions [63]. So generic reductions typically result in quantitatively weaker bounds, and it has been a well-known trend that symmetric community seeks for *quantitatively the same* bounds via direct approach.

APPENDIX C
FIGURE OF AEDT

APPENDIX D
AN ANALYSIS IN THE STANDARD MODEL

The analysis is very similar to that of AEDT in [31]. To ease comparison, in this subsection we adopt the notations of [31].

Assumptions. In this section, we model the TBC $\widetilde{\mathsf{E}}$ as a *strong tweakable pseudorandom permutation*, defined as follows.

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

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

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.

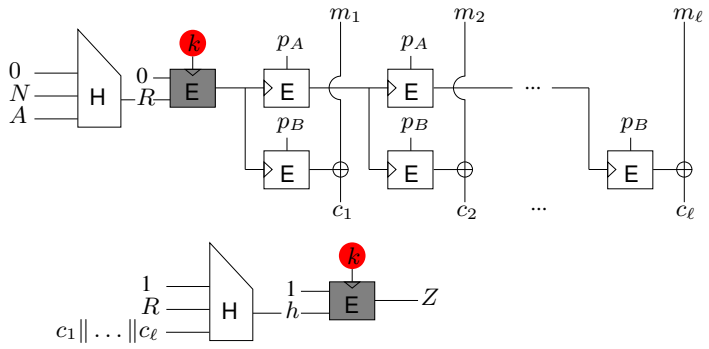

Figure 8: The AEAD mode AEDT. The two dark blocks are "leak-free" KDF_K and TGF_K instantiated with a single blockcipher-call. p_A and p_B could be any two distinct constants, while in section IV-A we consider the simplest case $p_A = 0$ and $p_B = 1$. Note that the notation E in this figure denotes a *classical blockcipher*: AEDT is a blockcipher-based AEAD mode.

And we assume the leakages are "recyclably simulatable". We define

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

where $\overrightarrow{q} = (q_m, q_e, q_d, p-1, q_l)$, and the maximal is taken over all $(\overrightarrow{q}, t)$-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. In the remaining, we first present the leakage assumption and then our results.

Recyclable Leakage Simulatability is defined below (adapted to TPRP setting) based on (p,q)-rsim-game in Table II. We abbreviate it as (p,q)-recyclable-simulatability. This (p,q)-recyclable-simulatability assumption is an extension of the q-simulatability notion [39], by allowing each of the q leakages in q-simulatability to be repeated p times.

Game (p,q)-$\mathsf{rsim}(\mathcal{A}, \widetilde{\mathsf{E}}, \mathsf{L}, \mathcal{S}, b)$.		
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 $b=1$
$\mathsf{TEnc}(T,x)$	$\widetilde{\mathsf{E}}^T_k(x), \mathsf{L}(k,T,x)$	$\widetilde{\mathsf{E}}^T_k(x), \mathcal{S}^{\mathsf{L}}(k^*,T,x,\widetilde{\mathsf{E}}_k(T,x))$
and one query of the following type, repeated at most p times:		
Query	Response if $b=0$	Response if $b=1$
$\mathsf{Gen}(k_{pre}, T_{pre}, x)$	$\mathcal{S}^{\mathsf{L}}(k_{pre}, T_{pre}, x, k)$	$\mathcal{S}^{\mathsf{L}}(k_{pre}, T_{pre}, x, k^*)$

Table II
THE (p,q)-RSIM-GAME FOR TBC.

Definition 2 ((p,q)-recyclable-simulatability of leakages)**.** *Let $\widetilde{\mathsf{E}}$ be a TPRP with L as its leakage function. Then the leakages of $\widetilde{\mathsf{E}}$ 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*

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

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. (2), the non-invertible leakage assumption. Please see [31] for additional discussion on this assumption.

To the muCCAmL2 **Advantage.** We formally 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 $\mathsf{L}_{\widetilde{\mathsf{E}}}(s,T,x)$ generated by all the internal calls to $\widetilde{\mathsf{E}}_s(T,x)$, 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 [38], we consider a Leaking Real Single-block Encryption scheme LRSE defined in Fig. 9 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

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

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

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

$\mathbf{RSDec}_{k_{ch}}(c)$ 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

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

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 9: Basic unit: the single-block encryption scheme LRSE.

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

$$\Big|\Pr[\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}(\mathsf{LRSEnc}^{+}_{k_{ch}}(m^0)) \Rightarrow 1] - \Pr[\mathcal{A}^{\mathsf{L}_{\mathsf{RSE}}}(\mathsf{LRSEnc}^{+}_{k_{ch}}(m^1)) \Rightarrow 1]\Big| \leq \varepsilon_{s\text{-}block}, \tag{9}$$

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. (2), $\varepsilon_{s\text{-}block}$ may not be negligible.

Based on the assumption (9), the muCCAmL2 advantage bound could be established below.

Theorem 4. *Let $\widetilde{\mathsf{E}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2q_e + 2q_d + 2q_m, t', \varepsilon_{\widetilde{\mathsf{E}}^*})$-STPRP. Assume that $\widetilde{\mathsf{E}}$ has two implementations: a strongly protected implementation $\widetilde{\mathsf{E}}^*$ underlying KDF and TGF is leak free, and a plain implementation $\widetilde{\mathsf{E}}$ have leakage function $\mathsf{L}_{\widetilde{\mathsf{E}}}$ 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*

$$\mathbf{Adv}^{\mathsf{muCCAmL2}}_{\mathsf{TEDT}}(\overrightarrow{q}, t, \sigma) \leq 2(q_e + q_d + q_m)\varepsilon_{\widetilde{\mathsf{E}}} + \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT}}(q_e + q_d + q_m, t') + 4\sigma(\varepsilon_{\widetilde{\mathsf{E}}} + \varepsilon_{(p,2)\text{-}rsim}) + \sigma \cdot \varepsilon_{s\text{-}block}, \tag{10}$$

where σ is the number of blocks in the challenge messages and $\varepsilon_{s\text{-}block}$ is as defined in Eq. (9). 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 VI, 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 muCCAm$ bound in the standard model:

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

This concrete bound is weak. It doesn't reveal the influence of key collisions. Also it suffers from the (unreal) "hybrid security loss" (as discussed subsequently to Theorem 2), which, as we stressed several times and as discussed in [36], [37], constitutes one of our motivations to mainly rely on ideal model analysis. On the positive side, the above two bounds show that when the TBC isn't ideal, TEDT still enjoys (a weaker) provable security.

Proof of Theorem 4 is almost the same as the proof for Theorem 5 & 10 in [31]; also the flow is very similar to Appendix G. We first prove that based on the pseudorandomness of $\widetilde{\mathsf{E}}$ and the simulatability of the leakage, the "real-leaking world" $(\widetilde{\mathsf{E}}^T_k(p), \mathsf{L}(k, T, p))$ is indistinguishable from the "ideal-simulating" world $(\$, \mathcal{S}^{\mathsf{L}}(k, T, p, \$))$. This is actually a lemma of Guo et al. [31] adapted to our TPRP setting.

Lemma 1 (Indistinguishability of Real-Leak and Ideal-Simulation)**.** *Let $\widetilde{\mathsf{E}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\widetilde{\mathsf{E}}})$-TPRP, whose implementation has a leakage function $\mathsf{L}_{\widetilde{\mathsf{E}}}$ 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:*

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

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 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.

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 10: The ideal single-block encryption scheme ISEnc.

Description of RESM:
- Gen picks $k_0 \xleftarrow{\$} \{0,1\}^n$
- $\mathsf{RESM}_{k_0}(T, N, m_1, \ldots, m_\ell)$ proceeds in two steps:

(1) Initializes an empty list leak for the leakage;
(2) for $i = 1, \ldots, \ell$, computes $k_i \leftarrow \widetilde{\mathsf{E}}_{k_{i-1}}(T, C_i(N))$, $y_i \leftarrow \widetilde{\mathsf{E}}_{k_{i-1}}(T, D_{i-1}(N))$, and $c_i \leftarrow y_i \oplus m_i$, and adds $[\mathsf{L}_{\widetilde{\mathsf{E}}}(k_{i-1}, T, C_i(N)), \mathsf{L}_{\widetilde{\mathsf{E}}}(k_{i-1}, T, D_{i-1}(N))]^p$, $\mathsf{L}_{\oplus}(y_i, m_i)$, and $[\mathsf{L}_{\oplus}(y_i, c_i)]^{p-1}$ to the list leak.

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

Description of IESM:
- $\mathsf{IESM}_{k_0}(T, N, m_1, \ldots, m_\ell)$ 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.

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

Figure 11: The RESM and IESM scheme.

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 $\widetilde{\mathsf{E}}_{k_{ch}}(T, p_A)$ and $\widetilde{\mathsf{E}}_{k_{ch}}(T, p_B)$ by two distinct random values to obtain the target inputs, relying on the assumption that $\widetilde{\mathsf{E}}$ is a TPRP. □

We then define the tweakable variant of the "single-block scheme" in [31]: roughly, all the intermediate values are replaced by random and all the leakages are replaced by simulation. The resulted algorithm is defined in Fig. 10. 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 2 (Indistinguishability of ISEnc and RSEnc)**.** *Let $\widetilde{\mathsf{E}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\widetilde{\mathsf{E}}})$-TPRP, whose implementation has a leakage function $\mathsf{L}_{\widetilde{\mathsf{E}}}$ 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:*

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

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 1. □

We then define (L)RESM and (L)IESM in the standard model in Fig. 11 and prove their indistinguishability.

Lemma 3 (Indistinguishability of LRESM and LIESM)**.** *Let $\widetilde{\mathsf{E}} : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a $(2, t, \varepsilon_{\widetilde{\mathsf{E}}})$-TPRP, whose implementation has a leakage function $\mathsf{L}_{\widetilde{\mathsf{E}}}$ 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:*

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

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

Proof. We define G_0 as the security game in which $\mathcal{A}^{\mathsf{L}}$ receives $\mathsf{RESM}_{k_0}(T,N,m)$ as the input, and G_ℓ as the game in which $\mathcal{A}^{\mathsf{L}}$ receives $\mathsf{IESM}_{k_0}(T,N,m)$ 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 $\widetilde{\mathsf{E}}^T_{k_{i-1}}(C_i(N))$ and $\widetilde{\mathsf{E}}^T_{k_{i-1}}(D_{i-1}(N))$ by two distinct random values k_i and y_i, and replace the leakages $[\mathsf{L}_{\widetilde{\mathsf{E}}}(k_{i-1},T,C_i(N)),\mathsf{L}_{\widetilde{\mathsf{E}}}(k_{i-1},T,D_{i-1}(N))]^p$, $\mathsf{L}_\oplus(\widetilde{\mathsf{E}}^T_{k_{i-1}}(D_{i-1}(N)),m_i)$, and $[\mathsf{L}_\oplus(\widetilde{\mathsf{E}}^T_{k_{i-1}}(D_{i-1}(N)),c_i)]^{p-1}$ by the simulated leakage traces $[\mathcal{S}^{\mathsf{L}}(k_{i-1},T,C_i(N),k_i),\mathcal{S}^{\mathsf{L}}(k_{i-1},T,D_{i-1}(N),y_i)]^p$, $\mathsf{L}_\oplus(y_i,m_i)$, 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 $[\mathcal{S}^{\mathsf{L}}(k_{j-1},T,C_j(N),k_j),\mathcal{S}^{\mathsf{L}}(k_{j-1},T,D_{j-1}(1$ 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 $[\mathcal{S}^{\mathsf{L}}(k_{i-2},T,C_{i-1}(N),k_{ch}),\mathcal{S}^{\mathsf{L}}(k_{i-2},T,$ $\mathsf{L}_\oplus(y_{i-1},m_{i-1})$, $[\mathsf{L}_\oplus(y_{i-1},c_{i-1})]^{p-1}$ 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 RESM 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 RESM involves $2\ell-1$ calls to $\widetilde{\mathsf{E}}$ 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 $(2\ell-1)(t_{\widetilde{\mathsf{E}}}+t_{\$}+t_{\mathcal{S}})+\ell\cdot t_\oplus=t_r$ 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 1, we have

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

Therefore, the ℓ transitions yield

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

in total. □

Then is the standard model version of Lemma 12.

Lemma 4 (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*

$$\begin{aligned}&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IESM}_{k_0}(T,N,m^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IESM}_{k_0}(T,N,m^1))\Rightarrow 1]|\\ \leq&\sum_{i=1}^{\ell}|\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{k_{i-1}}(m_i^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\mathsf{L}'}(\mathsf{LISEnc}^+_{k_{i-1}}(m_i^1))\Rightarrow 1]|,\end{aligned}$$

where $k_0,\ldots,k_{\ell-1}$ are chosen uniformly at random, 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 $t_{\widetilde{\mathsf{E}}}$, $t_{\$}$, and $t_\oplus$ are as assumed in Lemma 3.

Proof. The proof follows the same line as Lemma 12. □

Gathering Lemmas 2, 3, and 4, and following the same line as Lemma 13 we obtain

Lemma 5. *Let $\widetilde{\mathsf{E}}:\{0,1\}^n\times\{0,1\}^n\times\{0,1\}^n\to\{0,1\}^n$ be a $(2,t,\varepsilon_{\widetilde{\mathsf{E}}})$-TPRP, whose implementation has a leakage function $\mathsf{L}_{\widetilde{\mathsf{E}}}$ 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*

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

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

1) 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 $\widetilde{\mathsf{E}}^*_{K_1},\ldots,\widetilde{\mathsf{E}}^*_{K_u}$ 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 $\widetilde{\mathsf{E}}^*$, 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

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

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

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

$$\begin{aligned}&\left|\Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}_1} \Rightarrow 1] - \Pr[(\mathcal{A}^{\mathsf{L}})^{\mathsf{G}^*_1} \Rightarrow 1]\right| \\ \le &\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathsf{TEDT}}(q_e+q_d+q_m, t') + \frac{q_e+q_m-1}{2^n} + \underbrace{\sum_{i=1}^{q_m} \varepsilon_{\mathsf{FEMALE}\text{-}eav}(\ell_i)}_{\le 4\sigma(\varepsilon_{\widetilde{\mathsf{E}}}+\varepsilon_{(2,2)\text{-}rsim})+\sigma\cdot\varepsilon_{s\text{-}block} \text{ (by Lemma 5)}},\end{aligned}$$

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

APPENDIX E
PROOF OF THEOREM 1

The proof proceeds in two steps:

(i) Below in section E-B, we transit the scheme $\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}}$ to its idealized version, via replacing calls to KDF and TGF by several independent tweakable random permutations $\widetilde{\pi}_1,\ldots,\widetilde{\pi}_u$. These permutations cannot be queried by $\mathcal{A}$. We show the real and idealized schemes are indistinguishable. The actual goal of this step is to argue that during the interaction, the information gained by the adversary is indeed independent of the secret keys $\mathbf{K}$.

(ii) Then in section E-C, we prove unforgability for the idealized scheme to complete the muCIML2 proof.

The proof would rely on some properties of the Hirose compression function and the pad scheme presented in section E-A below (before the main analysis). Finally, as mentioned, the TBC-call in line 6 in Dec (Fig. V) has to be backward to resist decryption leakage. To serve more insights, at the end in section E-D we discuss where our proof approach fails for the design without the backward TBC-call.

A. Properties of Hirose Compression Function and pad

Denote by $\mathsf{Hir}[\widetilde{\mathsf{IC}}]$ the Hirose compression function based on the ideal TBC $\widetilde{\mathsf{IC}}$. Note that any adversary $\mathcal{A}$ against $\mathsf{Hir}[\widetilde{\mathsf{IC}}]$ 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 $\widetilde{\mathsf{IC}}_K(T,X)$, $\mathcal{A}'$ makes a query $\widetilde{\mathsf{IC}}_K(T, X\oplus\theta) \to Y'$ right after relaying $\widetilde{\mathsf{IC}}_K(T,X) \to Y$, and
- each time $\mathcal{A}$ makes a backward query $\widetilde{\mathsf{IC}}^{-1}_K(T,Y)$, $\mathcal{A}'$ makes a query $\widetilde{\mathsf{IC}}_K(T, X\oplus\theta) \to Y'$ right after relaying $\widetilde{\mathsf{IC}}^{-1}_K(T,Y) \to X$.

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

Lemma 6. *For any $\mathcal{A}$ making q pairs of matching queries to $\widetilde{\mathsf{IC}}$ with $1 \le q \le 2^n/4$, it holds*

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

Proof. The collision resistance of $\mathsf{Hir}[\widetilde{\mathsf{IC}}]$ does not necessarily imply the collision resistance of $\mathsf{chop}\circ\mathsf{Hir}[\widetilde{\mathsf{IC}}]$. 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 $\widetilde{\mathsf{IC}}$ queries in different orders. Then the collision indicates $\mathsf{chop}(g\oplus y) = \mathsf{chop}(g\oplus\theta\oplus z)$. Wlog 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,

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

Therefore, 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

$$g \oplus y = g' \oplus y' \wedge \mathsf{chop}(g \oplus \theta \oplus z) = \mathsf{chop}(g' \oplus \theta \oplus z') \tag{11}$$

or

$$g \oplus y = g' \oplus \theta \oplus z' \wedge \mathsf{chop}(g \oplus \theta \oplus z) = \mathsf{chop}(g' \oplus y'). \tag{12}$$

Similarly to Case 1, in any subcase, we have

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

so that $\Pr[\text{Eq. (11)}] \leq \frac{2}{(2^n-2q)^2}$. A similar reasoning shows $\Pr[\text{Eq. (12)}] \leq \frac{2}{(2^n-2q)^2}$. Therefore, when $q \leq 2^n/4$ we have

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

In summary,

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

as claimed. □

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

Lemma 7. *Consider the Davies-Meyer function* $\mathsf{DM}[\widetilde{\mathsf{IC}}](K\|T,h) = \widetilde{\mathsf{IC}}^T_K(h) \oplus h$ *built upon the ideal TBC* $\widetilde{\mathsf{IC}}$*. Then, for any adversary* $\mathcal{A}$ *making at most* $q \leq 2^n/2$ *queries to* $\widetilde{\mathsf{IC}}$ *and any integer* λ*, it holds*

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

Proof. Consider any λ $\widetilde{\mathsf{IC}}$ 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 $q \leq 2^n/2$, we have

(i) $\Pr[\mathsf{chop}(z_2) = \mathsf{chop}(z_1)] \leq \frac{2}{2^n-q} \leq \frac{4}{2^n}$
(ii) ...
(iii) $\Pr[\mathsf{chop}(z_\lambda) = \mathsf{chop}(z_1)] \leq \frac{2}{2^n-q} \leq \frac{4}{2^n}$

Thus in total we have

$$\Pr[\lambda \text{ collisions after chopping}] \leq \binom{q}{\lambda} \cdot \left(\frac{4}{2^n}\right)^{\lambda-1} \leq \frac{(4q)^\lambda}{\lambda! 2^{(\lambda-1)n}}$$

as claimed. □

Last, we claim the pad scheme is effective.

Lemma 8. *It holds* $\mathit{pad}(A,N,\overrightarrow{c},T) \neq \mathit{pad}(A',N',\overrightarrow{c}',T')$ *for any two distinct tuples* $(A,N,\overrightarrow{c},T)$ *and* $(A',N',\overrightarrow{c}',T')$*.*

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

B. Idealizing TEDT

To formally define the idealized scheme, we define an ideal primitive $\mathsf{TRPFamily}$ in Fig. 12. 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 $\widetilde{\pi}_1,\ldots,\widetilde{\pi}_u$.[7] The definition and use of $\mathsf{TRPFamily}$ are similar to Gueron and Lindell [27]. 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 $\mathsf{TEDT}[\widetilde{\mathsf{IC}},\mathsf{TRPFamily}]_{\mathbf{T}}$.

[7]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 $\frac{u^2}{2^n}$.

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

1) Preparation for the Indistinguishability: Notations.: Formally, we are to derive an upper bound for

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\widetilde{\mathsf{IC}},\mathsf{TRPFamily}]_{\mathbf{T}},\mathsf{L}^*,u}.$$

for any $\overrightarrow{q}$-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-coefficients technique [57]. We first define the transcripts of the distinguisher when interacting with the real/ideal scheme, and recall the technique.

In detail, we summarize the adversarial queries to the ideal TBC $\widetilde{\mathsf{IC}}$ in a set

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

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

Note that by our assumption, all the $\widetilde{\mathsf{IC}}$ queries (except for KDF and TGF) made by TEDT are leaked to $\mathcal{D}$. These queries also result in records of the form (k_i, t_i, x_i, y_i). To make a distinction, we denote by $\tau^*_{\widetilde{\mathsf{IC}}}$ the union of these records and the adversarial query transcript $\tau_{\widetilde{\mathsf{IC}}}$. It isn't hard to see:

- upon an encryption query $\mathsf{Enc}(i, N, A, M)$, TEDT makes at most $2 \cdot 2 \cdot \lceil \frac{|M|}{n} \rceil + 2 \cdot \lceil \frac{|A|}{n} \rceil \le 4(\lceil \frac{|M|}{n} \rceil + \lceil \frac{|A|}{n} \rceil)$ $\widetilde{\mathsf{IC}}$ 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 $\widetilde{\mathsf{IC}}$ calls is at most $4(\lceil \frac{|M|}{n} \rceil + \lceil \frac{|A|}{n} \rceil) + 6$;
- upon a decryption query $\mathsf{Dec}(i, N, A, C)$, the number of internal $\widetilde{\mathsf{IC}}$ calls is at most $4(\lceil \frac{|M|}{n} \rceil + \lceil \frac{|A|}{n} \rceil) + 6$, with $C = \overrightarrow{c} \| Z$.

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

$$q^*_{\widetilde{\mathsf{IC}}} := \left|\tau^*_{\widetilde{\mathsf{IC}}}\right| \le 4\sigma + 6(q_e + q_d) + q_{\widetilde{\mathsf{IC}}}. \tag{13}$$

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:

- interacting with the real AEAD scheme $\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}}$, the i-th query is either $\mathsf{KDF}/\mathsf{TGF}(K_i, t_i, x_i) \to y_i$ or $\mathsf{KDF}^{-1}/\mathsf{TGF}^{-1}(K_i, t_i, y_i)$ x_i; and,
- interacting with the idealized scheme $\mathsf{TEDT}[\widetilde{\mathsf{IC}}, \mathsf{TRPFamily}]_{\mathbf{T}}$, 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 12).

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, and this eventually yields the above $\tau_{\widetilde{\pi}}$.

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 $\mathsf{H}[\widetilde{\mathsf{IC}}]$. Note that by the specification of TEDT, none of the U_i can be empty. As we assumed all the underlying $\widetilde{\mathsf{IC}}$ queries have been leaked and included in $\tau^*_{\widetilde{\mathsf{IC}}}$, this list contains redundant information: it can be recovered from $\tau^*_{\widetilde{\mathsf{IC}}}$. But it simplifies the language, e.g., Eq. (14).

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 12).

This is wlog 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 $(\tau^*_{\mathsf{H}}, \tau^*_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\pi}})$ and obtain what we call the *transcript*

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

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 $\widetilde{\mathsf{IC}}$ are leaked, the transcripts $\tau^*_{\widetilde{\mathsf{IC}}}$ and $\tau_{\widetilde{\pi}}$ allow completely recover the encryption/decryption queries and answers. Therefore, there is no need for their presence in this section. (τ_e and τ_d will indeed be used in Sections VII and I.)

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

- in the real world, we say $\widetilde{\mathsf{IC}}$ extends $\tau_{\widetilde{\pi}}$, denoted $\widetilde{\mathsf{IC}} \vdash \tau_{\widetilde{\pi}}$, if $\widetilde{\mathsf{IC}}^t_K(x) = y$ 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 $\tau = (\tau^*_{\mathsf{H}}, \tau^*_{\widetilde{\mathsf{IC}}}, \tau_{\widetilde{\pi}}, \mathbf{T}, \mathbf{K})$, the interaction of $\mathcal{D}$ with oracles $(\mathsf{TEDT}[\widetilde{\mathsf{IC}}, \mathsf{TRPFamily}]_{\mathbf{T}}, \widetilde{\mathsf{IC}})$ produces τ if and only if $\widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}$ and $\mathsf{TRPFamily} \vdash \tau_{\widetilde{\pi}}$.

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

Lemma 9. *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 [57].

2) Completing the Transition: By 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

$$\begin{aligned} \mu_T &:= \max_{t \in \{0,1\}^n} \big|\{i \in \{1, \ldots, u\} : T_i = t\}\big|, \\ \mu_W &:= \max_{w \in \{0,1\}^{n-1}} \Big|\big\{(U, V\|W) \in \tau^*_{\mathsf{H}} : W = w\big\}\Big|. \end{aligned} \tag{14}$$

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

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

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

$$\Pr[\mu_T \geq n] \leq \binom{u}{n} \cdot \frac{1}{2^{(n-1)(n-1)}} \leq \frac{2^{n-1} \cdot u^n}{n! 2^{(n-1)n}} \leq \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n. \tag{15}$$

Whereas if $T_1, \ldots, T_u$ are distinct, then trivially $\mu_T = 1$ 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 10. *Consider the interaction between an* muCIML2 $\overrightarrow{q}$*-adversary $\mathcal{A}$ and the scheme* $\mathsf{TEDT}[\widetilde{\mathsf{IC}}, \mathit{TRPFamily}]_{\mathbf{T}}$*, and $\mathcal{A}$ has at most σ blocks in all its queried plaintext, ciphertext, and associated data. Define*

$$q^{**}_{\widetilde{\mathsf{IC}}} := 4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}}. \tag{16}$$

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

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

(ii) 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 $\mathsf{TEDT}[\widetilde{\mathsf{IC}}, \mathsf{TRPFamily}]_{\mathbf{T}}$. Following Hirose [50], we normalize the game: for each $\widetilde{\mathsf{IC}}$ query either made by $\mathcal{A}$ or made by the TEDT mode, we assume

the system makes its Hirose matching query immediately (see section E-A). Denote G_2 the resulted normalized game. We now count the number of $\widetilde{\mathsf{IC}}$ queries in G_2:

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

Therefore, in G_2, the number of $\widetilde{\mathsf{IC}}$ queries doesn't exceed $q^{**}_{\widetilde{\mathsf{IC}}} = 4\sigma+6(q_e+q_d)+2q_{\widetilde{\mathsf{IC}}}$ as in Eq. (16), and the upper bound on the number of matching $\widetilde{\mathsf{IC}}$ query pairs is $\frac{q^{**}_{\widetilde{\mathsf{IC}}}}{2}$.

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:
  - $(u, h, g, g \oplus g^*), (u, h, g \oplus \theta, g \oplus \theta \oplus h^*), (u, h, g, g \oplus g^{**}), (u, h, g \oplus \theta, g \oplus \theta \oplus h^{**}) \in \tau^*_{\widetilde{\mathsf{IC}}}$ for some g^*, h^*, g^{**}, h^{**}, and
  - $\mathsf{chop}(g^*\|h^*) = \mathsf{chop}(g^{**}\|h^{**})$.
- (C-2) Hitting initial-vector: $\exists u, g, h : \mathsf{Hir}[\tau^*_{\widetilde{\mathsf{IC}}}](u\|g\|h) = [0]^{2n}$. Formally, there exists a $3n$-bit value $u\|g\|h$ such that $(u, h, g, g), (u, h, g \oplus \theta, g \oplus \theta) \in \tau^*_{\widetilde{\mathsf{IC}}}$;
- (C-3) n-collision: there exists n records $(u_1, h_1, g_1, z_1), \ldots, (u_n, h_n, g_n, z_n) \in \tau^*_{\widetilde{\mathsf{IC}}}$ such that $\mathsf{chop}(g_1 \oplus z_1) = \ldots = \mathsf{chop}(g_n \oplus z_n)$.

For (C-1), from Lemma 6 we have (when $q^{**}_{\widetilde{\mathsf{IC}}}/2 \leq 2^n/4$, i.e., $q^{**}_{\widetilde{\mathsf{IC}}} \leq 2^n/2$)

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

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 $\Pr[y = x \wedge y' = x \oplus \theta] \leq \frac{1}{(2^n - q^{**}_{\widetilde{\mathsf{IC}}})^2} \leq \frac{4}{2^{2n}}$ when $q^{**}_{\widetilde{\mathsf{IC}}} \leq 2^n/2$, and thus

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

For (C-3), an application of Lemma 7 (also requires $q^{**}_{\widetilde{\mathsf{IC}}} \leq 2^n/2$) yields $\Pr[\text{(C-3)}] \leq \frac{(4q^{**}_{\widetilde{\mathsf{IC}}})^n}{n!2^{n(n-1)}}$. Using $n! \geq \left(\frac{n}{e}\right)^n \geq 2^n$ (since $n \geq 6 > 2e$), we reach

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

A union bound thus results in

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

Then, conditioned on $\neg$(C-1) $\wedge$ $\neg$(C-2) $\wedge$ $\neg$(C-3), we show the two claims hold. We first consider claim (i). 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., $U = header\|u\|tail$, $U^* = header^*\|u^*\|tail$, $|u| = |u^*| = n$, $u \neq u^*$, and $|header|, |header^*|$, and $|tail|$ are multiples of n. Then we distinguish two cases:

Case 1: either $header\|u$ or $header^*\|u^*$ is empty. Wlog assume $header\|u$ is empty. Since U isn't empty, this means $tail$ isn't empty. Then we have $\mathsf{H}(header^*\|u^*) \neq [0]_{2n}$ 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 $V\|W \neq V^*\|W^*$.

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

We then show claim (ii). 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

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

contradicting ¬(C-3). These complete the proof. □

Lemma 10 indicates

$$\Pr[\mu_W \geq n] \leq \frac{8q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{17}$$

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 from 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" $\mathbf{K}^\dagger = (K^\dagger_1, \ldots, K^\dagger_u)$, 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 $\mathcal{K}^\dagger$ as

$$\mathcal{K}^\circ := \left\{\mathbf{K}^\circ = (K^\circ_1, \ldots, K^\circ_u) : \forall (i, j), K^\dagger_i = K^\dagger_j \text{ if and only if } K^\circ_i = K^\circ_j\right\}.$$

Then it isn't hard to see 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

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

Then it's easy to see $\sum_{t \in \{0,1\}^n} |\tau^*_{\widetilde{\mathsf{IC}}}[t]| = q^*_{\widetilde{\mathsf{IC}}}$; and by the above, we have

$$\begin{aligned}
\Pr[\text{(B-2)}] \leq & \sum_{i=1}^{u} \sum_{t \in \{0,1\}^n : (K,t,*,*) \in \tau_{\widetilde{\pi}}} \Pr\left[K_i \in \tau_{\widetilde{\mathsf{IC}}}[t]\right] \\
= & \sum_{i=1}^{u} \sum_{t \in \{0,1\}^n : (K,t,*,*) \in \tau_{\widetilde{\pi}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t]\right|}{2^n} \\
= & \sum_{i=1}^{u} \sum_{t \in \{0,1\}^{n-1} : (K,t\|0,*,*) \in \tau_{\widetilde{\pi}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t\|0]\right|}{2^n} + \sum_{i=1}^{u} \sum_{W \in \{0,1\}^{n-1} : (K,W\|1,*,*) \in \tau_{\widetilde{\pi}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[W\|1]\right|}{2^n}.
\end{aligned}$$

By the construction of TEDT, queries of the form $(K, t\|0, x, y)$ in $\tau_{\widetilde{\pi}}$ are nececessarily due to KDF calls, for which $K = K_i$ and $t = PK_i$ for the user index i. Since $\mu_T \leq n$, we have

$$\sum_{i=1}^{u} \sum_{t \in \{0,1\}^{n-1} : (K,t\|0,*,*) \in \tau_{\widetilde{\pi}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t\|0]\right|}{2^n} \leq n \cdot \sum_{t \in \{0,1\}^{n-1}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t\|0]\right|}{2^n}.$$

On the other hand, queries of the form $(K, W\|0, x, y)$ in $\tau_{\widetilde{\pi}}$ are due to TGF calls. For any such query $(K, W\|0, x, y)$, there necessarily exists at least one H query $(U, V\|W) \in \tau^*_{\mathsf{H}}$. By this, conditioned on ¬(B-1), we have

$$\begin{aligned}
\sum_{i=1}^{u} \sum_{W \in \{0,1\}^{n-1} : (K,W\|1,*,*) \in \tau_{\widetilde{\pi}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[W\|1]\right|}{2^n} \leq & \sum_{i=1}^{u} \sum_{W : (\mathsf{pad}(A,N,\overrightarrow{c},T_i),*\|W) \in \tau^*_{\mathsf{H}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[W\|1]\right|}{2^n} \\
\leq & \mu_T \cdot \sum_{T \in \{0,1\}^n} \sum_{W : (\mathsf{pad}(A,N,\overrightarrow{c},T),*\|W) \in \tau^*_{\mathsf{H}}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[W\|1]\right|}{2^n} \\
\leq & \mu_T \cdot \mu_W \cdot \sum_{W \in \{0,1\}^{n-1}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[W\|1]\right|}{2^n} \\
\leq & n^2 \cdot \sum_{W \in \{0,1\}^{n-1}} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[W\|1]\right|}{2^n}.
\end{aligned} \tag{18}$$

In all, summing over the two terms, we reach

$$\Pr[\text{(B-2)} \mid \neg\text{(B-1)}] \leq n^2 \cdot \sum_{t \in \{0,1\}^n} \frac{\left|\tau_{\widetilde{\mathsf{IC}}}[t]\right|}{2^n} \leq \frac{n^2 q^*_{\widetilde{\mathsf{IC}}}}{2^n} \leq \frac{n^2 q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}. \tag{19}$$

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

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

With this notation, it's clear that

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

For the real world distribution, we have

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

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

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

Therefore, for any good transcript τ we have

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

and thus

$$\begin{aligned}&\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}},\mathsf{L}^*,u} - \mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\widetilde{\mathsf{IC}},\mathsf{TRPFamily}]_{\mathbf{T}},\mathsf{L}^*,u} \\ \le&\Pr[T_{id} \in \mathcal{T}_{bad}] \le \Pr[\text{(B-2)} \mid \neg\text{(B-1)}] + \Pr[\text{(B-1)}] \\ \le&\frac{(n^2+8)q^{**}_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{1}{2n!} \cdot \Big(\frac{4u}{2^n}\Big)^n. \quad \text{(from Eqs. (15), (17), and (19))} \end{aligned} \tag{20}$$

C. Unforgeability of the Idealized TEDT

Denote by G_1 the game that captures the interaction between the muCIML2 adversary $\mathcal{A}$ and $\mathsf{TEDT}[\widetilde{\mathsf{IC}}, \mathsf{TRPFamily}]_{\mathbf{T}}$. 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 $\mathcal{A}$ is unable to forge in G_1 as long as none of these conditions is fulfilled.

1) Bad Conditions for Unforgeability.: We identify the following conditions during an execution of G_2:

- (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 10 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*: $\mu_W \le \lambda$.

For (C-1), if $\mathbf{T}$ is distinct then $\Pr[\text{(C-1)}] = 0$, and if $\mathbf{T}$ uniform then $\Pr[\text{(C-1)}] \le \binom{u}{2} \cdot \frac{1}{2^{2n-1}} = \frac{u^2}{2^{2n}}$. The other bound $\Pr[\text{(C-2)}] \le \frac{8q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}$ immediately follows from Lemma 10, where $q^{**}_{\widetilde{\mathsf{IC}}}$ is defined in Eq. (16). For simplicity let $\mathsf{Bad} = \text{(C-1)} \vee \text{(C-2)}$, then

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

2) Unforgeability unless Bad.: Conditioned on $\neg\mathsf{Bad}$, we argue all non-trivial decryption queries result in $\perp$ except with a bounded probability. Let $C = \overrightarrow{c}\|Z$. If this query doesn't give rise to $\perp$, then right after this query is processed, there exists a hash record $(\mathsf{pad}(A,N,\overrightarrow{c},T_i), V\|W)$ and a $\mathsf{TRPFamily}$ query $(i, W^*\|1, V^*, Z)$ 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:

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

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

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

- if $j = i$ and $(N, A, \overrightarrow{C}) = (N', A', \overrightarrow{C}')$, then since we forbid trivial decryption queries, the tag produced by $\mathsf{LEnc}_{\mathbf{K},\mathbf{T}}(j, N', A', M')$ cannot be Z, and hence cannot create the query $(i, W\|1, V^*, Z)$;
- if $j = i$ while $(N, A, \overrightarrow{C}) \neq (N', A', \overrightarrow{C}')$, then by Lemma 8 we have $\mathsf{pad}(A, N, \overrightarrow{C}, T_i) \neq \mathsf{pad}(A', N', \overrightarrow{C}', T_i)$, which further implies a hash collision and contradicts ¬(C-2);
- if $j \neq i$ and $K_j \neq K_i$, then it isn't possible;
- finally, if $j \neq i$ yet $K_j = K_i$, then we have $T_j \neq T_i$ by ¬(C-1), which by Lemma 8 further implies $\mathsf{pad}(A, N, \overrightarrow{C}, T_i) \neq \mathsf{pad}(A', N', \overrightarrow{C}', T_j)$ and contradicts ¬(C-2).

In all, $(i, W\|1, V^*, Z)$ has to be backward. During the execution of G_2, 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 $(i, W\|1, V^*, Z)$, we have

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

Therefore,

$$\Pr[\text{Case 1} \mid \neg\mathsf{Bad}] \leq \frac{2nq_d}{2^n}. \tag{21}$$

b) Case 2: $(i, W^*\|1, V^*, Z)$ 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 $\geq 2^n - q^{**}_{\widetilde{\mathsf{IC}}}$ possibilities. So

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

As argued before,

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

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

$$\Pr[\text{Case 2}] \leq \frac{8q^{**}_{\widetilde{\mathsf{IC}}}(q_d + q_e)}{2^{2n}}. \tag{22}$$

Note that these arguments are significantly simplified by the normalization of the game: without the normalization, $(i, W^*\|1, V^*, Z)$ may be created between the two "matching" $\widetilde{\mathsf{IC}}$ queries, giving rise to many additional cases. In summary, gathering (21) and (22) yields

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

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

$$\mathbf{Adv}^{\mathsf{muCIML2}}_{\mathcal{D},\mathsf{TEDT}[\widetilde{\mathsf{IC}},\mathsf{TRPFamily}]_{\mathbf{T}},\mathsf{L}^*,u} \leq \frac{u^2}{2^{2n}} + \frac{8q^{**}_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2nq_d}{2^n} + \frac{8q^{**}_{\widetilde{\mathsf{IC}}}(q_d + q_e)}{2^{2n}} \leq \frac{u^2}{2^{2n}} + \frac{(n^2+9)q^{**}_{\widetilde{\mathsf{IC}}}}{2^n} \tag{23}$$

since $q^{**}_{\widetilde{\mathsf{IC}}} \leq 4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}}$, $\frac{8q^{**}_{\widetilde{\mathsf{IC}}}(q_d+q_e)}{2^{2n}} \leq \frac{8(q^{**}_{\widetilde{\mathsf{IC}}})^2}{6\cdot 2^{2n}} \leq \frac{q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}$, and $\frac{2nq_d}{2^n} \leq \frac{n^2 q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}$. Eq. (23) further plus Eq. (20) yield Eq. (1):

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

D. The Necessity of Inverting

Upon each decryption query $\mathsf{Dec}_{K,PK}(N, A, \overrightarrow{C}\|Z)$, after deriving the digest $V\|W = \mathsf{H}(\mathsf{pad}(A, N, \overrightarrow{C}, PK\|0))$, if the subsequent checking is "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 E-C, the query $(i, W\|1, V^*, Z)$ may *not* be backward (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 E-C. In all, to achieve integrity against decryption/verification leakage, the inverse of the underlying primitives seem necessary.

[8]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.

APPENDIX F
TESTER FOR LORL2 ADVANTAGE

As stressed many times, depending on the context, the concrete value of $\mathbf{Adv}^{\mathsf{LORL2}}$ may not be negligible.

Tester for LORL2 $\mathbf{Adv}^{\mathsf{LORL2}}$
Let the challenging adversary $\mathcal{A}$ give s, T, and (m^0, m^1)
Pick the secret: $y \xleftarrow{\$} \{0,1\}^n$, $b \xleftarrow{\$} \{0,1\}$
$c \leftarrow y \oplus m^b$, $x \leftarrow \widetilde{\mathsf{E}}_s^T(y)$
return $(c, [\mathsf{L}^{out}(s, T; y)]^p, \mathsf{L}_{\oplus}(y, m^b), [\mathsf{L}_{\oplus}(y, c)]^{p-1})$
Let the challenging adversary $\mathcal{A}$ output the guess b'

APPENDIX G
PROOF OF THEOREM 2

We proceed in three steps in three subsubsections. First, we define the process of encrypting a single message of ℓ blocks. We in particular define both the real and the ideal encryption processes: the real process queries $\widetilde{\mathsf{IC}}$ for encrypting and resembles TEDT, while the ideal process just samples many random values for encrypting.

We show that the two processes are indistinguishable. This step mainly requires us to bound certain "bad events" in the real and ideal executions, for which the non-invertible assumption on the leakage functions is helpful.

While the 1st step shows the ciphertext of TEDT is "sufficiently random", it says nothing about the message confidentiality since the leakages may leak M. Therefore, we focus on the idealized process $(\$, \mathsf{L}_{ideal}(M))$, and show how to relate its eavesdropper advantage to the term defined by Eq. (5): it resembles using the minimal message processing operation to independently encrypt $|M|/n$ blocks. The eavesdropper advantage of $(\mathsf{TEDT}(M), \mathsf{L}_{\mathsf{TEDT}(M)})$ can be derived via the following chain:

$$\begin{aligned}
&\text{(leaking) eavesdropper advantage of the minimal operation } \mathbf{Adv}^{\mathsf{LORL2}}\\
\Rightarrow&\text{(leaking) eavesdropper advantage of the ideal } (\$, \mathsf{L}_{ideal}(M))\\
\Rightarrow&\text{(leaking) eavesdropper advantage of encrypting a single message using TEDT}\\
\Rightarrow&\mathsf{muCCAmL2} \text{ advantage of TEDT.}
\end{aligned}$$

As the 3rd step, based on the eavesdropper advantage of $(\mathsf{TEDT}(M), \mathsf{L}_{\mathsf{TEDT}(M)})$, we establish the muCCAmL2 advantage. This step relies on the following features of TEDT:

(i) Every invalid decryption query only leaks a "useless" pseudorandom value, i.e. $\widetilde{\mathsf{IC}}_k^{*\|1}(Z)$ for some Z;
(ii) For each challenge encryption query, since the nonce is used only once during the experiment, the process starts from a ephemeral key k_0 that is different from any other ephemeral key of the other encryption queries. By this, encryption of this challenge is quite independent from the other encryption queries, and we can view the entire experiment as an eavesdropper adversary against $(\mathsf{TEDT}(M), \mathsf{L}_{\mathsf{TEDT}(M)})$ with a lot of offline computations (i.e. all the other encryptions are turned into offline computations).

Below we expose in detail.

1) The Ideal Single-Message Encryption Process.: Formally, they are defined by the following pseudocode.

Description of $\mathsf{RESM}[\widetilde{\mathsf{IC}}]$:

- Gen picks $k_0 \xleftarrow{\$} \{0,1\}^n$
- $\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m_1, \ldots, m_\ell)$ proceeds in two steps:

(1) Initializes an empty list leak for the leakage;
(2) for $i = 1, \ldots, \ell$, computes $k_i \leftarrow \widetilde{\mathsf{IC}}_{k_{i-1}}^T(P_i(N))$, $y_i \leftarrow \widetilde{\mathsf{IC}}_{k_{i-1}}^T(Q_{i-1}(N))$, 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.

$\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m_1, \ldots, m_\ell)$ eventually returns $(c_1, \ldots, c_\ell)$.
We define $\mathsf{LRESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m) = (\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m), \mathsf{leak})$ for the list leak standing at the end of the above process.

Description of IESM (an ideal process independent from $\widetilde{\mathsf{IC}}$):

- $\mathsf{IESM}_{k_0}(T, N, m_1, \ldots, m_\ell)$ 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 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.

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

The real and ideal single-message encryption processes (with leakages) are indistinguishable.

Lemma 11. *For every ℓ-block message M, every T, N, and every $(q_{\widetilde{\mathsf{IC}}}, t)$-bounded distinguisher $\mathcal{D}^{\widetilde{\mathsf{IC}}}$, it holds*

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

where t_l is the total time needed for evaluating L^{in} and L^{out}.

One may wonder why the term capturing $\mathbf{Adv}^{PRP}_{\widetilde{\mathsf{IC}}}$ is missed. For this, we remark that this term has been captured by the term $\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}$: if $\mathcal{A}$ predicates the secret then it clearly breaks the PRP security of $\widetilde{\mathsf{IC}}$.

Proof. Consider the execution of $\mathcal{D}^{\widetilde{\mathsf{IC}}}$ upon the inputs $(M, \mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, M))$. We define a bad event $\mathsf{BadQuery}$, which occurs when any of the internal keys $k_0, k_1, \ldots, k_{\ell-1}$ appears in the key field of an $\widetilde{\mathsf{IC}}$ query made by $\mathcal{D}^{\widetilde{\mathsf{IC}}}$. This event, once happens, would cause the key stream blocks lose randomness. We remark that in [23, 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 $\mathcal{D}^{\widetilde{\mathsf{IC}}}$, we construct an adversary $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ such that

$$\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(\mathcal{A}) \leq \Pr[\mathsf{BadQuery} \text{ in } \mathcal{D}^{\widetilde{\mathsf{IC}}}(\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m_1, \ldots, m_\ell))]. \tag{24}$$

Concretely, upon inputs (s_2, z, leak) with

$$\mathsf{leak} = \big(\mathsf{L}^{out}(s_0, T; s_1), \mathsf{L}^{in}(s_1, T; P_2), \mathsf{L}^{out}(s_1, T; s_2), \mathsf{L}^{in}(s_1, T; Q_2), \mathsf{L}^{out}(s_1, T; z)\big),$$

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

(1) $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ randomly guesses an index $i \xleftarrow{\$} [0, \ell - 1]$, uniformly samples an initial key k_0, and initializes an empty list leak;
(2) for $j = 1, \ldots, i-1$, $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ queries $\widetilde{\mathsf{IC}}$ to obtain $k_j \leftarrow \widetilde{\mathsf{IC}}^T_{k_{j-1}}(P_j(N))$ and $y_j \leftarrow \widetilde{\mathsf{IC}}^T_{k_{j-1}}(Q_{j-1}(N))$, and computes $c_j \leftarrow y_j \oplus m_j$. $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ 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) $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ queries $y_i \leftarrow \widetilde{\mathsf{IC}}^T_{k_{i-1}}(Q_{i-1}(N))$ and computes $c_i \leftarrow y_i \oplus m_i$. It takes k_{i-1} as the value s_0 mentioned before Eq. (2) and 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) $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ 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 takes the challenge secret s_1 as the key k_i and adds $[\mathsf{L}^{in}(s, T; P_{i+1}(N)), \mathsf{L}^{out}(s, T; k_{i+1})]^p$, $[\mathsf{L}^{in}(s, T; Q_i(N)), \mathsf{L}^{out}(s, T; z)]^p$, $\mathsf{L}_{\oplus}(y_i, m_i)$, and $[\mathsf{L}_{\oplus}(y_i, c_i)]^{p-1}$ to leak as the leakage of the $i+1$ th iteration (note that these leakages are inputs of $\mathcal{A}^{\widetilde{\mathsf{IC}}}$).
(5) Then $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ starts from k_{i+1} to emulate the remaining actions of $\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}]$ encrypting the tail $m_{i+1} \| \ldots \| m_\ell$ to obtain $c_{i+1} \| \ldots \| c_\ell$.
(6) Eventually, $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ serves the ciphertext $c_1 \| \ldots \| c_\ell$ as well as the leakage list leak to $\mathcal{D}$, and outputs the set $Guesses = \{k : (k, t, x, y) \in \tau_{\widetilde{\mathsf{IC}}} \text{ for some } t, x, y\}$.

The strategy of $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ is quite obvious: if $\mathcal{D}$ triggers the event $\mathsf{BadQuery}$ then the key k being queried must be in $\tau_{\widetilde{\mathsf{IC}}}$. Therefore, $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ 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 $\widetilde{\mathsf{IC}}$). This guess will be correct with probability $1/\ell$. Then, $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ emulates the encryption process of RESM_{k_0} and provides the leakages to $\mathcal{D}$, except for the i index, for which the leakages and $\widetilde{\mathsf{IC}}$ 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 $\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m_1, \ldots, m_\ell)$. Therefore, when $\mathcal{D}$ halts, if $\mathcal{D}$ made a query on s_1, then simply outputting $\tau_{\widetilde{\mathsf{IC}}}$ would break the game. So we have $\Pr[s_1 \in Guesses \mid \mathsf{BadQuery} \text{ in } \mathcal{D}^{\widetilde{\mathsf{IC}}}(M, \mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, M))] = \frac{1}{\ell}$.

Now, we observe that

$$\begin{aligned}&\Pr[s_1 \in Guesses \mid \mathsf{BadQuery} \text{ in } \mathcal{D}^{\widetilde{\mathsf{IC}}}(M, \mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m))]\\ \leq&\frac{\Pr[s_1 \in Guesses]}{\Pr[\mathsf{BadQuery} \text{ in } \mathcal{D}^{\widetilde{\mathsf{IC}}}(M, \mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T, N, m))]}.\end{aligned}$$

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

$$\begin{aligned}\Pr[\mathsf{BadQuery}\text{ in }\mathcal{D}^{\widetilde{\mathsf{IC}}}(M,\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T,N,m))] &\le \ell\cdot Pr[s_1\in Guesses]\\ &\le \ell\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(\mathcal{A}) \quad \text{(Eq. 2)}\\ &\le \ell\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(p,q_{\widetilde{\mathsf{IC}}}+2\ell,O(t+\ell\cdot p\cdot t_l)). \quad \text{(Eq. 4)}\end{aligned}$$

It can be seen as long as the event $\mathsf{BadQuery}$ never happens during the real execution $\mathcal{D}^{\widetilde{\mathsf{IC}}}(M,\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T,N,m))$, in the two executions all the keys and key stream blocks are fresh random values independent from $\tau_{\widetilde{\mathsf{IC}}}$ the transcript of $\widetilde{\mathsf{IC}}$ queries of $\mathcal{D}$, and have the same distribution. Therefore,

$$\begin{aligned}&|\Pr[\mathcal{D}^{\widetilde{\mathsf{IC}}}(M,\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T,N,M))\Rightarrow 1]-\Pr[\mathcal{D}^{\widetilde{\mathsf{IC}}}(M,\mathsf{IESM}_{k_0}[\widetilde{\mathsf{IC}}](T,N,M))\Rightarrow 1]|\\ \le&\ell\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(p,q_{\widetilde{\mathsf{IC}}}+2\ell,O(t+\ell\cdot p\cdot t_l))\end{aligned}$$

as claimed. □

2) From 1-Block to ℓ-Block Advantage.: We then show the eavesdropper advantage of $\mathsf{IESM}[\widetilde{\mathsf{IC}}]$ encrypting an ℓ-block message is related to the defined term $\mathbf{Adv}^{\mathsf{LORL2}}$.

Lemma 12. *For every pair of ℓ-block messages M^0 and M^1 and $(q_{\widetilde{\mathsf{IC}}},t)$-bounded adversary $\mathcal{A}^{\widetilde{\mathsf{IC}}}$, it holds*

$$\begin{aligned}&|\Pr[\mathcal{A}^{\widetilde{\mathsf{IC}}}(\mathsf{IESM}_{k_0}(T,N,M^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\widetilde{\mathsf{IC}}}(\mathsf{IESM}_{k_0}(T,N,M^1))\Rightarrow 1]|\\ \le&\ell\cdot\mathbf{Adv}^{\mathit{LORL2}}(p,q_{\widetilde{\mathsf{IC}}},O(t+\ell\cdot p\cdot t_l))+\frac{\ell}{2^n},\end{aligned}$$

where t_l is as defined in Lemma 11.

Proof. Let $M^0=m_1^0\|\dots\|m_\ell^0$ and $M^1=m_1^1\|\dots\|m_\ell^1$. We start by building a sequence of $\ell+1$ messages $M_{h,0},\dots,M_{h,\ell}$ starting from M^0 and modifying its blocks one by one until obtaining M^1. That is, $M_{h,i}:=m_1^1\|\dots\|m_i^1\|m_{i+1}^0\|\dots\|m_\ell^0$. For any i, assuming a $(q_{\widetilde{\mathsf{IC}}},t)$-bounded adversary $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ against $\mathsf{IESM}_{k_0}(T,N,M_{h,i-1})$ and $\mathsf{IESM}_{k_0}(T,N,M_{h,i})$, we build a $(q_{\widetilde{\mathsf{IC}}},t+t_r)$-bounded adversary $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ against the distribution defined in Eq. (5). In detail, upon inputs (c^b,leak_b) with $\mathsf{leak}_b=\big([\mathsf{L}^{out}(s,T;z)]^p,\mathsf{L}_\oplus(z,m^b),[\mathsf{L}_\oplus(z,c^b)]^{p-1}\big)$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ proceeds in four steps:

(1) $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ uniformly samples k_0 and initializes an empty list leak;
(2) for $j=1,\dots,i-1$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ 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) $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ samples k_i, takes k_{i-1} as the s value mentioned before Eq. (5) & the challenge key stream block z as y_i, and adds the traces $[\mathsf{L}^{in}(k_{i-1},T;P_i),\mathsf{L}^{out}(k_{i-1},T;k_i)]^p$, $[\mathsf{L}^{in}(k_{i-1},T;Q_{i-1}),\mathsf{L}^{out}(k_{i-1},T;z)]^p$, $\mathsf{L}_\oplus(y,m^b)$, and $[\mathsf{L}_\oplus(y,c^b)]^{p-1}$ to leak;
(4) Then $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ starts from k_i to emulate the remaining actions of IESM encrypting the tail $m_{i+1}^0\|\dots\|m_\ell^0$ to obtain $c_{i+1}\|\dots\|c_\ell$. Eventually, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ serves the ciphertext $c_1\|\dots\|c_{i-1}\|c^b\|c_{i+1}\|\dots\|c_\ell$ as well as all the generated simulated leakages to $\mathcal{A}^{\widetilde{\mathsf{IC}}}$, and outputs whatever $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ outputs.

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

$$\begin{aligned}&|\Pr[\mathcal{A}^{\widetilde{\mathsf{IC}}}(\mathsf{IESM}_{k_0}(T,N,M_{h,i-1}))\Rightarrow 1]-\Pr[\mathcal{A}^{\widetilde{\mathsf{IC}}}(\mathsf{IESM}_{k_0}(T,N,M_{h,i}))\Rightarrow 1]|\\ \le&\mathbf{Adv}^{\mathsf{LORL2}}(p,q_{\widetilde{\mathsf{IC}}},O(t+\ell\cdot p\cdot t_l))+\frac{1}{2^n}\end{aligned}$$

by Eq. (7). This along with a simple summation implies the main claim. □

Gathering Lemmas 11 and 12, we obtain upper bounds on the eavesdropper advantage of RESM (which is also the eavesdropper advantage of TEDT) stated in Lemma 13.

Lemma 13. *For every pair of ℓ-block messages M^0 and M^1 and $(q_{\widetilde{\mathsf{IC}}},t)$-bounded adversary $\mathcal{A}^{\widetilde{\mathsf{IC}}}$, it holds*

$$\begin{aligned}\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{RESM}}(p,q_{\widetilde{\mathsf{IC}}},t,\ell)=&|\Pr[\mathcal{A}^{\widetilde{\mathsf{IC}}}(\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T,N,M^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\widetilde{\mathsf{IC}}}(\mathsf{RESM}_{k_0}[\widetilde{\mathsf{IC}}](T,N,M^1))\Rightarrow 1]|\\ \le&\frac{\ell}{2^n}+\ell\cdot\mathbf{Adv}^{\mathit{LORL2}}(p,q_{\widetilde{\mathsf{IC}}},O(t+\ell\cdot p\cdot t_l))+2\ell\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(p,q_{\widetilde{\mathsf{IC}}}+2\ell,O(t+\ell\cdot p\cdot t_l)),\end{aligned}$$

where t_l is as defined in Lemma 11.

Proof.

$$\begin{aligned}
&|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{RESM}_{k_0}(T,N,M^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{RESM}_{k_0}(T,N,M^1))\Rightarrow 1]|\\
\le&\underbrace{|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IESM}_{k_0}(T,N,M^0))\Rightarrow 1]-\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IESM}_{k_0}(T,N,M^1))\Rightarrow 1]|}_{\le \ell\cdot\mathbf{Adv}^{\mathsf{LORL2}}(p,q_{\widetilde{\mathsf{IC}}},O(t+\ell\cdot p\cdot t_l))+\frac{\ell}{2^n}\text{ (by Lemma 12)}}\\
&+\underbrace{\sum_{b=0,1}|\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{RESM}_{k_0}(T,N,M^b))\Rightarrow 1]-\Pr[\mathcal{A}^{\mathsf{L}}(\mathsf{IESM}_{k_0}(T,N,M^b))\Rightarrow 1]|}_{\le 2\ell\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(p,q_{\widetilde{\mathsf{IC}}}+2\ell,O(t+\ell\cdot p\cdot t_l))\text{ (by Lemma 11)}}\\
\le&\ell\cdot\mathbf{Adv}^{\mathsf{LORL2}}(p,q_{\widetilde{\mathsf{IC}}},O(t+\ell\cdot p\cdot t_l))+\frac{\ell}{2^n}+2\ell\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}]}(p,q_{\widetilde{\mathsf{IC}}}+2\ell,O(t+\ell\cdot p\cdot t_l)).
\end{aligned}$$

The claim thus follows. □

3) Completing the muCCAmL2 *Proof.:* Theorem 2 could then be derived from Lemma 13. Recall from page 28 that a decryption query $\mathsf{Dec}_{\mathbf{K},\mathbf{T}}(i,N,A,C)$ is *trivial* if the action $\mathsf{Enc}_{\mathbf{K},\mathbf{T}}(i,N,A,M)\to C$ happens before. The leakages of trivial decryption queries may serve new information, thus requiring explicit treatments.

Then we step into the proof. We start by defining G_0 as the game capturing the interaction between $\mathcal{A}$ and $(\mathsf{LEnc}_{\mathbf{K},\mathbf{T}},\mathsf{LEnc}^0_{\mathbf{K},\mathbf{T}},\mathsf{LDec}_{\mathbf{K},\mathbf{T}},\widetilde{\mathsf{IC}})$ and G^*_0 as the game capturing the interaction between $\mathcal{A}$ and $(\mathsf{LEnc}_{\mathbf{K},\mathbf{T}},\mathsf{LEnc}^1_{\mathbf{K},\mathbf{T}},\mathsf{LDec}_{\mathbf{K},\mathbf{T}},\widetilde{\mathsf{IC}})$.

We then define two 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 E-B). By Eq. (20) (although the formalisms are different), with $q^{***}_{\widetilde{\mathsf{IC}}}=4\sigma+6(q_m+q_e+q_d)+2q_{\widetilde{\mathsf{IC}}}$ (which is tweaked from Eq. (16)), we have

$$\left|\Pr[\mathsf{G}_1\Rightarrow 1]-\Pr[\mathsf{G}_0\Rightarrow 1]\right|\le\frac{(n^2+8)q^{***}_{\widetilde{\mathsf{IC}}}}{2^n}+\frac{1}{2n!}\cdot\left(\frac{4u}{2^n}\right)^n \tag{25}$$

and

$$\left|\Pr[\mathsf{G}^*_1\Rightarrow 1]-\Pr[\mathsf{G}^*_0\Rightarrow 1]\right|\le\frac{(n^2+8)q^{***}_{\widetilde{\mathsf{IC}}}}{2^n}+\frac{1}{2n!}\cdot\left(\frac{4u}{2^n}\right)^n, \tag{26}$$

We then prove

$$\begin{aligned}
\left|\Pr[\mathsf{G}_1\Rightarrow 1]-\Pr[\mathsf{G}^*_1\Rightarrow 1]\right|\le&\frac{2u^2}{2^{2n}}+\frac{(n^2+9)q^{***}_{\widetilde{\mathsf{IC}}}}{2^n}\\
&+\underbrace{\textstyle\sum_{i=1}^{q_m}\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{RESM}}(p,q_{\widetilde{\mathsf{IC}}}+q^*,O(t+p\sigma t_l),\ell_i)}_{\le\frac{\sigma}{2^n}+\sigma\cdot\mathbf{Adv}^{\mathsf{LORL2}}(p,q_{\widetilde{\mathsf{IC}}}+q^*,O(t+p\sigma t_l))+2\sigma\cdot\mathbf{Adv}^{\text{2-up}[q_{\widetilde{\mathsf{IC}}}+q^*]}(p,q_{\widetilde{\mathsf{IC}}}+q^*,O(t+p\sigma t_l))\text{ (Lemma 13)}},
\end{aligned} \tag{27}$$

where ℓ_i is the number of blocks in the ith challenge message, and $q^*=4\sigma+6(q_m+q_e+q_d)$ and t_l defined in Lemma 11. This plus the gaps in Eq. (25) and Eq. (26) yield the claim (note that $\sigma\le q^{***}_{\widetilde{\mathsf{IC}}}$). To this end, we denote the q_e challenge tuples by (the suffix c stands for "challenge")

$$(uc_1,Nc_1,Ac_1,Mc^0_1,Mc^1_1),\ldots,(uc_{q_e},Nc_{q_e},Ac_{q_e},Mc^0_{q_e},Mc^1_{q_e}).$$

Then, we use q_e hops to replace $Mc^0_1,\ldots,Mc^0_{q_e}$ by $Mc^1_1,\ldots,Mc^1_{q_e}$ in turn, to show that G_1 can be transited to G^*_1. For convenience, we define $\mathsf{G}_{2,0}=\mathsf{G}_1$, and define a sequence of games

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

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

We then show that for $i=1,\ldots,q_e$, $\mathsf{G}_{2,i-1}$ and $\mathsf{G}_{2,i}$ are indistinguishable for $\mathcal{A}^{\widetilde{\mathsf{IC}}}$. For this, from $\mathcal{A}^{\widetilde{\mathsf{IC}}}$ we build an adversary $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$, such that $|\Pr[\mathsf{G}_{2,i-1}\Rightarrow 1]-\Pr[\mathsf{G}_{2,i}\Rightarrow 1]|$ is related to $\mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{RESM}}(\mathcal{A}_2^{\widetilde{\mathsf{IC}}})$. In detail, initially, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ samples two key vectors $\mathbf{K}$ and $\mathbf{T}=(T_1,\ldots,T_u)$ for subsequent simulations. It aborts if there exists two indices $i\ne j$ yet $K_i\|T_i=K_j\|T_j$: this event is denoted BadKeyColl. It also keeps a pair of tables $(ICTable,ICTable^{-1})$ to simulate the primitive TRPFamily via lazy sampling (note that TRPFamily is essentially an ideal cipher $\widetilde{\mathsf{IC}}_2$ independent from $\widetilde{\mathsf{IC}}$). Assume that entries in the tables are of the form $ICTable(K,T,X)=Y$ and $ICTable^{-1}(K,T,Y)=X$. It then runs $\mathcal{A}$, reacting as follows:

- Upon a query to $\widetilde{\mathsf{IC}}$: simply relays.
- Upon a (non-challenge) encryption query (u_i,N_i,A_i,M_i) from $\mathcal{A}$,
  - if $(K_{u_i},T_{u_i},C_0(N_i))\notin ICTable$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ samples an initial key $k_0^{(i)}$ such that $(K_{u_i},T_{u_i},k_0^{(i)})\notin ICTable^{-1}$, defines $ICTable(K_{u_i},T_{u_i},C_0(N_i))\leftarrow k_0^{(i)}$ and $ICTable^{-1}(K_{u_i},T_{u_i},k_0^{(i)})\leftarrow C_0(N_i)$, and then runs the encryption process $\mathsf{RESM}_{k_0^{(i)}}[\widetilde{\mathsf{IC}}](T_{u_i},N_i,M_i)$ to get the ciphertext $\overrightarrow{c}_i$ and leakages. $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ then computes $V_i\|W_i\leftarrow\mathsf{H}[\widetilde{\mathsf{IC}}](\mathsf{pad}(A_i,N_i,\overrightarrow{c}_i,T_{u_i}))$

and $Z_i \leftarrow ICTable(K_{u_i}, W_i\|1, V_i)$ (if $(K_{u_i}, W_i\|1, V_i) \notin ICTable$ then $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ defines $ICTable(K_{u_i}, W_i\|1, V_i)$ as a newly sampled value). For this entire process $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ has to make $4\ell_i + 6$ queries to $\widetilde{\mathsf{IC}}$ with $\ell_i = |M_i|/n$ and costs $2t_{\$} + 2p\ell_i t_l$ time to evaluating the leakage functions. Finally, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ returns the outputs $\overrightarrow{c}_i\|Z_i$ and the leakages to $\mathcal{A}$;
  - if $(K_{u_i}, T_{u_i}, C_0(N_i)) \in ICTable$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ simply runs $\mathsf{RESM}_{k_0}(T_{u_i}, N_i, M_i)$ with $k_0 = ICTable(K_{u_i}, T_{u_i}, C_0(N_i))$, calls $V_i\|W_i \leftarrow \mathsf{H}(\mathsf{pad}(A_i, N_i, c_i, T_{u_i}))$ and computes the tag $Z_i \leftarrow ICTable(K_{u_i}, W_i\|1, V_i)$ on the obtained $\overrightarrow{c}_i$, and returns $\overrightarrow{c}_i\|Z_i$ and the leakages to $\mathcal{A}$. The cost is similar to the above case.
- Upon a trivial decryption query (u_j, N_j, A_j, C_j) from $\mathcal{A}$ (cf. the beginning of this subsection for "trivial"), $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ parses $C_j = \overrightarrow{c}_j\|Z_j$ and simply runs the decryption $\mathsf{RESM}[\widetilde{\mathsf{IC}}].\mathsf{Dec}_{k_0^j}(T_{u_j}, N_j, \overrightarrow{c}_j)$ for $k_0^j = ICTable(K_{u_j}, T_{u_j}, C_0(N_j))$, and relays the outputs to $\mathcal{A}$. The cost is similar to the encryption case.
- Upon a non-trivial decryption query (u_j, N_j, A_j, C_j) from $\mathcal{A}$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ parses $C_j = \overrightarrow{c}_j\|Z_j$, and computes $V_j\|W_j \leftarrow \mathsf{H}[\widetilde{\mathsf{IC}}](\mathsf{pad}(A_j, N_j, \overrightarrow{c}_j, T_{u_j}))$. Then,
  - if $(K_{u_j}, W_j\|1, Z_j) \notin ICTable$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ samples V_j^* such that $(K_{u_j}, W_j\|1, V_j^*) \notin ICTable$, and sets $ICTable(K_{u_j}, W_j\|1, V_j^*) \leftarrow Z_j$, $ICTable^{-1}(K_{u_j}, W_j\|1, Z_j) \leftarrow V_j^*$;
  - if $(K_{u_j}, W_j\|1, Z_j) \in ICTable$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ just sets $V_j^* \leftarrow ICTable^{-1}(K_{u_j}, W_j\|1, Z_j)$.

  Now $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ aborts if $V_j = V_j^*$ (this type of abortion is defined as BadDec), and returns $(\perp, V_j^*)$ to $\mathcal{A}$ otherwise.
- Upon $\mathcal{A}$ submitting the j-th challenge tuple $(uc_j, Nc_j, Ac_j, Mc_j^0, Mc_j^1)$, conditioned on $\neg\mathsf{BadKeyColl}$, it necessarily holds $(K_{uc_j}, T_{uc_j}, C_0(Nc_j)) \notin ICTable$ by the challenge nonce-respecting restriction on $\mathcal{A}$ on a single user. Therefore, depending on j, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ reacts as follows:
  - When $j < i$, it encrypts Mc_j^0 and returns. In detail, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ samples $kc_0^{(j)}$, sets $ICTable(K_{uc_j}, T_{uc_j}, C_0(Nc_j)) \leftarrow kc_0^{(j)}$ and $ICTable^{-1}(K_{uc_j}, T_{uc_j}, kc_0^{(j)}) \leftarrow C_0(Nc_j)$, and then runs $\mathsf{RESM}_{kc_0^{(j)}}[\widetilde{\mathsf{IC}}](Mc_j^0) \rightarrow cc_j$, performs the tag generation accordingly to produce Zc_j and returns (cc_j, Zc_j) and the leakages to $\mathcal{A}^{\widetilde{\mathsf{IC}}}$. The cost is similar to the non-challenge encryption queries.
  - When $j = i$, it relays Mc_j^0 and Mc_j^1 to its eavesdropper challenger to obtain cc_j^b and leakages leak_{enc} and $[\mathsf{leak}_{dec}]^{p-1}$, and then performs the tag generation accordingly to produce Zc_j and returns $C_{ch}^b = (cc_j^b, Zc_j)$ to $\mathcal{A}$. Note that this means the relation $ICTable(K_{uc_i}, T_{uc_i}, C_0(Nc_i)) = k_0^{ch}$ is implicitly fixed, where k_0^{ch} is the secret key generated inside the eavesdropper challenger;
  - When $j > i$, it simply encrypts Mc_j^1 and returns. The details are similar to the described case $j < i$.
- Upon $\mathcal{A}$ making the λ-th query to $\mathsf{L}_{\mathsf{decch}}(j)$ $(1 \le \lambda \le p-1)$,
  - When $j \neq i$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ performs the corresponding decryption and returns the obtained leakages to $\mathcal{A}$;
  - When $j = i$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ simply returns the λ-th trace in the vector $[\mathsf{leak}_{dec}]^{p-1}$ as the answer.

It can be seen that as long as $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ never aborts, the whole process is the same as either $\mathsf{G}_{2,i-1}$ or $\mathsf{G}_{2,i}$ depending on whether $b = 0$ or 1. We have $\Pr[\mathcal{A}_2^{\widetilde{\mathsf{IC}}} \text{ aborts}] = \Pr[\mathsf{BadKeyColl} \vee \mathsf{BadDec}]$, and clearly $\Pr[\mathsf{BadKeyColl}] \le \frac{u^2}{2^{2n}}$. By the remarks before, besides running $\mathcal{A}$, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ 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, $\mathcal{A}_2^{\widetilde{\mathsf{IC}}}$ makes $q^* = 4\sigma + 6(q_m + q_e + q_d)$ additional queries to $\widetilde{\mathsf{IC}}$, and evaluates the leakage functions for $2p\sigma$ times, resulting in $O(p\sigma t_l)$ additional running time. By these and the definitions and Eq. (23) in section E-C, it can be seen

$$\Pr[\mathsf{BadDec}] \le \frac{u^2}{2^{2n}} + \frac{(n^2+9)q_{\widetilde{\mathsf{IC}}}^{***}}{2^n}. \tag{28}$$

By all the above, we have

$$\begin{aligned}\Pr[\mathsf{G}_{2,i} \Rightarrow 1] - \Pr[\mathsf{G}_{2,i-1} \Rightarrow 1] \le{}& \Pr[\mathsf{G}_{2,i} \Rightarrow 1 \wedge \mathcal{A}_2^{\widetilde{\mathsf{IC}}} \text{ aborts}] - \Pr[\mathsf{G}_{2,i-1} \Rightarrow 1 \wedge \mathcal{A}_2^{\widetilde{\mathsf{IC}}} \text{ aborts}] \\ &+ \mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{RESM}}(p, q_{\widetilde{\mathsf{IC}}} + q^*, O(t + p\sigma t_l), \ell_i).\end{aligned}$$

This means

$$\begin{aligned}\big|\Pr[\mathsf{G}_1^* \Rightarrow 1] - \Pr[\mathsf{G}_1 \Rightarrow 1]\big| &\le \Pr[\mathsf{G}_{2,q_e} \Rightarrow 1] - \Pr[\mathsf{G}_{2,0} \Rightarrow 1] \\ &\le \sum_{i=1}^{q_e} \Big(\Pr[\mathsf{G}_{2,i} \Rightarrow 1] - \Pr[\mathsf{G}_{2,i-1} \Rightarrow 1]\Big) \\ &\le \frac{2u^2}{2^{2n}} + \frac{(n^2+9)q_{\widetilde{\mathsf{IC}}}^{***}}{2^n} + \textstyle\sum_{i=1}^{q_e} \mathbf{Adv}^{\mathsf{eavl2}}_{\mathsf{RESM}}(p, q_{\widetilde{\mathsf{IC}}} + q^*, O(t + p\sigma t_l), \ell_i)\end{aligned}$$

which is the claim in Eq. (27).

APPENDIX H
PROOF OF THEOREM 3

We'll rely on a balls-in-bin lemma from [53] 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 [53, 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 [30]:

$$\begin{aligned}\mathbf{Adv}^{\mathsf{muCCAm\$}^*}_{\mathcal{A},\mathsf{AEAD},u} &= \Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Dec}_{\mathbf{K},\mathbf{T}},\widetilde{\mathsf{IC}}} \Rightarrow 1\big] - \Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\$,\perp,\widetilde{\mathsf{IC}}} \Rightarrow 1\big]\Big| \\ &\leq \underbrace{\Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Dec}_{\mathbf{K},\mathbf{T}},\widetilde{\mathsf{IC}}} \Rightarrow 1\big] - \Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\perp,\widetilde{\mathsf{IC}}} \Rightarrow 1\big]\Big|}_{\mathbf{Adv}^{\mathsf{mu\text{-}INT\text{-}CTXT}}_{\mathcal{A},\mathsf{AEAD},u}\text{: mu INT-CTXT advantage of } \mathcal{A} \text{ on AEAD}} \\ &\quad + \underbrace{\Big|\Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\perp,\widetilde{\mathsf{IC}}} \Rightarrow 1\big] - \Pr\big[\mathcal{A}^{\mathsf{Enc}_{\mathbf{K},\mathbf{T}},\$,\perp,\widetilde{\mathsf{IC}}} \Rightarrow 1\big]\Big|}_{\text{defined as } \mathbf{Adv}^{\mathsf{muCPAm\$}^*}_{\mathcal{A},\mathsf{AEAD},u}}. \end{aligned} \tag{29}$$

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 $\mathbf{Adv}^{\mathsf{muCPAm\$}^*}_{\mathcal{A},\mathsf{TEDT},u}$. Again we employ the H-coefficients technique, and present the two steps in two subsequent subsections.

A. Bad Transcripts

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

- First, we reveal all the $\widetilde{\mathsf{IC}}$ 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 $\widetilde{\mathsf{IC}}$ queries.

We merge these queries with the adversarial queries to obtain a set $\tau^*_{\widetilde{\mathsf{IC}}}$. Thus we have $\big|\tau^*_{\widetilde{\mathsf{IC}}}\big| \leq q^*_{\widetilde{\mathsf{IC}}}$ which is as defined by Eq. (13). 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

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

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

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

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

$$M^{(i)} = m^{(i)}_1 \| \ldots \| m^{(i)}_{\ell_i}$$

and

$$C^{(i)} = \overrightarrow{c}^{(i)} \| Z^{(i)} = c^{(i)}_1 \| \ldots \| c^{(i)}_{\ell_i} \| Z^{(i)},$$

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

$$\mu_Y := \max_{y \in \{0,1\}^n} \big|\{(i,j) : m^{(i)}_j \oplus c^{(i)}_j = y\}\big|.$$

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

$$\Pr[T_{id} = \tau] = \Pr[\mathbf{K}, \mathbf{T}] \cdot \Pr[\widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}] \cdot \left(\frac{1}{2^n}\right)^{q_e + \sum_{i=1}^{q_e} \ell_i}. \tag{30}$$

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), $C = \overrightarrow{c}\|Z$, such that either $(K_u, T_u, x, y) \in \tau^*_{\widetilde{\mathsf{IC}}}$ or $(K_u, W\|1, x, y) \in \tau^*_{\widetilde{\mathsf{IC}}}$ for some x, y, where W comes from its corresponding hash record $(\mathsf{pad}(A, N, \overrightarrow{c}, T_u), V\|W) \in \tau^*_{\mathsf{H}}$;
- (B-4) $\mu_Y \geq 2\log_2 \sigma$;
- (B-5) there exists two distinct encryption 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 ($U^{(i)} = \mathsf{pad}(A^{(i)}, N^{(i)}, \overrightarrow{c}^{(i)}, T_{u^{(i)}})$, $U^{(j)} = \mathsf{pad}(A^{(j)}, N^{(j)}, \overrightarrow{c}^{(j)}, T_{u^{(j)}})$):
  - *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 Section VI-B. We recycle the results as follows: when $\mathbf{T}$ is uniform, we have

$$\Pr[\text{(B-1)}] \leq \frac{u^2}{2^{2n}},\ \Pr[\text{(B-2)} \vee \mathit{hash\ collision}] \leq \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{8q^{**}_{\widetilde{\mathsf{IC}}}}{2^n},\ \text{and}\ \Pr[\text{(B-3)} \mid \text{(B-2)}] \leq \frac{n^2 q^{**}_{\widetilde{\mathsf{IC}}}}{2^n},$$

where $q^{**}_{\widetilde{\mathsf{IC}}}$ is defined by Eq. (16).

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

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

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 appendix E-A that $\Pr[W^{(i)} = W^{(j)}] \leq \frac{2}{2^n - q^{**}_{\widetilde{\mathsf{IC}}}}$. 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 $\leq q_e^2/2$ such pairs of indices (i, j), it holds

$$\Pr[\mathit{contradiction\ in}\ \text{(B-5)}] \leq \frac{q_e^2}{2^n(2^n - q^{**}_{\widetilde{\mathsf{IC}}})} \leq \frac{2q_e^2}{2^{2n}} \leq \frac{q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}.$$

In all,

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \leq \frac{u^2}{2^{2n}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(n^2+9)q^{**}_{\widetilde{\mathsf{IC}}} + 1}{2^n}. \tag{31}$$

B. 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 $k^{(i)}_0 = \widetilde{\mathsf{IC}}^{T_{u^{(i)}}}_{K_{u^{(i)}}}(P_0(N^{(i)}))$ is uniform. With this observation, we define the first predicate $\mathsf{BadKD}(\widetilde{\mathsf{IC}})$ to capture the "badness" of this key. Formally, $\mathsf{BadKD}(\widetilde{\mathsf{IC}})$ 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 $k^{(i)}_0 = \widetilde{\mathsf{IC}}^{T_{u^{(i)}}}_{K_{u^{(i)}}}(P_0(N^{(i)}))$ satisfies $(k^{(i)}_0, T_{u^{(i)}}, P_1(N^{(i)}), y) \in \tau^*_{\widetilde{\mathsf{IC}}}$, or $(k^{(i)}_0, T_{u^{(i)}}, Q_0(N^{(i)}), y) \in \tau^*_{\widetilde{\mathsf{IC}}}$ for some y, or $(k^{(i)}_0, T_{u^{(i)}}, x, y^{(i)}_1) \in \tau^*_{\widetilde{\mathsf{IC}}}$ for some x;
- (C-2)*nonce-reusing across two different users*: there exists two encryption queries $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)})$ and $(u^{(j)}, N^{(j)}, A^{(j)}, M^{(j)}$ in τ_e such that $T_{u^{(i)}} = T_{u^{(j)}} = T$, $N^{(i)} = N^{(j)} = N$, and the two keys $k^{(i)}_0 = \widetilde{\mathsf{IC}}^{T}_{K_{u^{(i)}}}(P_0(N^{(i)}))$ and $k^{(j)}_0 = \widetilde{\mathsf{IC}}^{T}_{K_{u^{(j)}}}(P_0(N^{(j)}))$ 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 muCCAm$ security definition.

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

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

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

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

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

$$k^{(1)}_0 = \widetilde{\mathsf{IC}}^{T_{u^{(1)}}}_{K_{u^{(1)}}}(N^{(1)}), \ldots, k^{(i-1)}_0 = \widetilde{\mathsf{IC}}^{T_{u^{(i-1)}}}_{K_{u^{(i-1)}}}(N^{(i-1)}),$$

the key $k^{(i)}_0 = \widetilde{\mathsf{IC}}^{T_{u^{(i-1)}}}_{K_{u^{(i)}}}(N^{(i)})$ is uniform in $\geq 2^n - q^*_{\widetilde{\mathsf{IC}}} - q_e$ 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,

$$\Pr[\text{(C-1)}] \leq \sum_{i=1}^{q_e} \frac{\left|\tau^*_{\widetilde{\mathsf{IC}}}[T_{u_i}, 0, N^{(i)}]\right|}{2^n - q^*_{\widetilde{\mathsf{IC}}} - q_e} + \sum_{i=1}^{q_e} \frac{\left|\tau^*_{\widetilde{\mathsf{IC}}}[T_{u_i}, y^{(i)}_1]^{-1}\right|}{2^n - q^*_{\widetilde{\mathsf{IC}}} - q_e}.$$

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

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

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

$$\Pr_{\widetilde{\mathsf{IC}}}[\mathsf{BadKD}(\widetilde{\mathsf{IC}}) \mid \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}] \leq \frac{2\mu_T q_e}{2^n} + \sum_{i=1}^{q_e} \frac{2\left|\tau^*_{\widetilde{\mathsf{IC}}}\left[T_{u^{(i)}}, 0, N^{(i)}\right]\right|}{2^n} + \sum_{i=1}^{q_e} \frac{2\left|\tau^*_{\widetilde{\mathsf{IC}}}\left[T_{u^{(i)}}, y_1^{(i)}\right]^{-1}\right|}{2^n}. \tag{32}$$

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

$$\begin{aligned}&\mathsf{BadE}_1^{(1)}, \mathsf{BadE}_2^{(1)}, \ldots, \mathsf{BadE}_{\ell_1-1}^{(1)},\\ &\ldots\\ &\mathsf{BadE}_1^{(q_e)}, \mathsf{BadE}_2^{(q_e)}, \ldots, \mathsf{BadE}_{\ell_{q_e}-1}^{(q_e)}.\end{aligned} \tag{33}$$

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

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

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

- (C-ij1): $k_j^{(i)} \in \tau^*_{\widetilde{\mathsf{IC}}}\left[T_{u^{(i)}}, j, N^{(i)}\right]$, or
- (C-ij2): there exists an index $s < i$ 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): $k_j^{(i)} \in \tau^*_{\widetilde{\mathsf{IC}}}\left[T_{u^{(i)}}, y_{j+1}^{(i)}\right]^{-1}$, 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 $\widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}$ and $\neg\mathsf{BadKD}(\widetilde{\mathsf{IC}})$ and $\neg\mathsf{BadE}_{j-1}^{(i)}(\widetilde{\mathsf{IC}}) \wedge \ldots \wedge \neg\mathsf{BadE}_1^{(1)}(\widetilde{\mathsf{IC}})$, the value $k_j^{(i)}$ is uniform in $\geq 2^n - q^*_{\widetilde{\mathsf{IC}}} - q_e$ possibilities. Therefore,

$$\Pr[\text{(C-}ij\text{1)} \vee \text{(C-}ij\text{3)}] \leq \frac{\left|\tau^*_{\widetilde{\mathsf{IC}}}\left[T_{u^{(i)}}, j, N^{(i)}\right]\right| + \left|\tau^*_{\widetilde{\mathsf{IC}}}\left[T_{u^{(i)}}, y_{j+1}^{(i)}\right]^{-1}\right|}{2^n - q^*_{\widetilde{\mathsf{IC}}} - q_e}.$$

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,

$$\Pr[\text{(C-}ij\text{2)} \vee \text{(C-}ij\text{4)} \vee \text{(C-}ij\text{5)}] \leq \frac{\mu_T - 1 + \mu_Y - 1 + 1}{2^n - q^*_{\widetilde{\mathsf{IC}}} - q_e}.$$

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

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

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

$$\Pr[y^\dagger = y_j^{(i)}] \geq \frac{1}{2^n}. \tag{34}$$

The probabilities of the predicates cumulate to

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

This plus Eq. (32) yields

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

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

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

Similarly,

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

Gathering the above and Eq. (34) yields

$$\begin{aligned}
&\Pr[\mathsf{TEDT}[\widetilde{\mathsf{IC}}].\mathsf{Enc}_{\mathbf{K},\mathbf{T}}(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}) = \overrightarrow{C}^{(i)} \text{ for all } i \in \{1,\ldots,q_e\} \mid \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}] \\
\geq &\Pr[\neg\mathsf{Bad}(\widetilde{\mathsf{IC}}) \mid \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}]\left(\frac{1}{2^n}\right)^{\sum_{i=1}^{q_e}\ell_i} \\
\geq &\left(1 - \frac{2\mu_T\sigma + 2\mu_Y\sigma + 2\mu_T q^*_{\widetilde{\mathsf{IC}}} + 2\mu_Y q^*_{\widetilde{\mathsf{IC}}}}{2^n}\right)\left(\frac{1}{2^n}\right)^{\sum_{i=1}^{q_e}\ell_i}. \qquad (35)
\end{aligned}$$

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:

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

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

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

$$\Pr[\widetilde{\mathsf{IC}}^{W^{(i)}\|1}_{K_{u^{(i)}}}(V^{(i)}) = Z^{(i)} \text{ for } i = 1,\ldots,q_e] \geq \frac{1}{2^{q_e n}}. \qquad (36)$$

Gathering Eq. (30), (35), and (36), and with $\sigma \leq 2^n/48 \Rightarrow \log_2\sigma \leq n$, we have

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

This plus Eq. (31) yield

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

By Eq. (29), this plus the bound in Eq. (1) yields

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

C. Detailed Comparison with GCM-SIV with KDF

GCM-SIV with nonce-based KDF was recently proposed by Gueron and Lindell [27] as a very elegant and efficient BBB secure variant of GCM-SIV. Its security bounds are further improved by Bose et al. [28] 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 blockcipher

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

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

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 $\frac{2^{14}\sigma+2^{14}q_{\widetilde{\mathsf{IC}}}}{2^{128}}$ even in the su setting (revisiting the calculations in Eq. (18) shows that the factor n^2 could be improved to n in the su setting, so its su bound could be better $\frac{128\sigma+128q_{\widetilde{\mathsf{IC}}}}{2^{128}}$). As mentioned, we believe this shows the benefits of protocol-level leakage-resilient designs.

APPENDIX I
KEY-DEPENDENT MESSAGE SECURITY OF TEDT

In this section we analyze the (mu) KDM security of TEDT in the ideal TBC model. We first recall the setting, then present our results, and then the proofs.

A. The Setting

Bellare and Keelveedhi characterized the KDM security of AEAD where messages can be key-dependent [33]. They defined three KDM security settings: *random nonce*, *universal nonce* (i.e., nonce-respecting), and *nonce-misuse*. In this paper we do not target the nonce misuse case as it implies traditional misuse resistance AEAD — while misuse resilience (such as muCCAm$) offers a nice security/efficiency tradeoff for AEAD. However, adapting and reaching the definition to the *misuse resilient* scenario without necessarily implying traditional misuse resistance is an interesting scope for further research since it is non trivial to secure challenge ciphertexts if partial message recovery is feasible from the non-challenge ciphertexts which might use several times the same nonce, just because messages depend on the keys. See appendix I-E for more details.

Following [33], we make a distinction of the settings via the nonce type nt: $\mathsf{nt} = \mathsf{r}$ stands for random nonce, while $\mathsf{nt} = \mathsf{u}$ stands for universal nonce. Below, we rephrase the definition of [33, Fig. 2. (right)] which already considered multiple keys.

Initialization(u):
1. $\mathbf{K} = (K_1, \ldots, K_u) \xleftarrow{\$} (\mathcal{K})^u$, $\mathcal{S}_1, \ldots, \mathcal{S}_u \leftarrow \emptyset$

Encryption: $\mathsf{Enc}_{\mathbf{K}}(i, N, A, \phi_m)$
1. $M \leftarrow \phi_m(\mathbf{K})$
2. If $(\mathsf{nt} = \mathsf{r})$ then $N \xleftarrow{\$} \mathcal{N}$
3. If $(b = 1)$ then $C \leftarrow \mathsf{Enc}_{K_i}(N, A, M)$
4. Else $\ell \leftarrow \mathsf{ciphertextlength}(M)$, $C \xleftarrow{\$} \{0,1\}^\ell$
5. $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{(N, A, C)\}$
6. Return (N, C)

Decryption: $\mathsf{Dec}_{\mathbf{K}}(i, N, A, C)$
1. If $(N, A, C) \in \mathcal{S}_i$ then return $\perp$
2. If $(b = 1)$ then $M \leftarrow \mathsf{Dec}_{K_i}(N, A, C)$
3. Else $M \leftarrow \perp$
4. If $M = \perp$ then $valid \leftarrow 0$ else $valid \leftarrow 1$
5. Return $valid$

Figure 13: $\mathsf{KDAE}^{[\mathsf{nt}],b}_{\mathsf{AEAD}}(\mathcal{A}, u)$: the AEAD KDI security game, where $\mathsf{nt} \in \{\mathsf{r}, \mathsf{u}\}$.

We will denote it muKDI\$ to highlight both the mu setting and the real-or-random indistinguishability from \$—the term "KDI" stands for "key-dependent inputs", which are synonym to KDM but avoid the suffix M and reduce confusion (since we used suffix M to denote misuse resistance/resilience properties).

Definition 3 (muKDI\$ Advantage)**.** *Given a nonce-based authenticated encryption* $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$*, the multi-user key-dependent message security advantage of an adversary* $\mathcal{A}$ *against* AEAD *with* u *users is*

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

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

- $\mathsf{Enc}^{nt}_{\mathbf{K}}(i, N, A, \phi_m)$*: for* $1 \le i \le u$*, computes* $M \leftarrow \phi_m(\mathbf{K})$*, if* $\mathsf{nt} = \mathsf{u}$ *and* (i, N) *is fresh, outputs* $C = \mathsf{Enc}_{K_i}(N, A, M)$*; else if* $\mathsf{nt} = \mathsf{r}$ *picks random* $N \leftarrow \mathcal{N}$*, computes* $C = \mathsf{Enc}_{K_i}(N, A, M)$ *and outputs* (N, C)*;*
- $\mathsf{Dec}_{\mathbf{K}}(i, N, A, C)$*: for* $1 \le i \le u$*, if* C *is an answer to some encryption query* (i, N, A, ϕ_m)*, returns* $\perp$*; else outputs* $\mathsf{Dec}_{K_i}(N, A, C)$*;*
- $\$(i, N, A, \phi_m)$*: if* $1 \le i \le u$*, if* $\mathsf{nt} = \mathsf{u}$ *and* (i, N) *is fresh, outputs* $C \leftarrow \mathcal{C}_{|\phi_m(\mathbf{K})|}$*; else if* $\mathsf{nt} = \mathsf{r}$ *picks* $N \xleftarrow{\$} \mathcal{N}$ *and* $C \xleftarrow{\$} \mathcal{C}_{|\phi_m(\mathbf{K})|}$ *and outputs* (N, C)*;*
- $\perp(i, N, A, C)$*: outputs* $\perp$*.*

We also recall a more faithful game-based KDI definition [33] in Table 13. It's equivalent to muKDI\$ advantage.

Definition 4 (KDAE Advantage)**.** *Let* $\mathsf{AEAD} = (\mathsf{Enc}, \mathsf{Dec})$ *be a nonce-based authenticated encryption. The (multi-user) key-dependent input advantage of an adversary* $\mathcal{A}$ *against* AEAD *with nonce-type* $\mathsf{nt} \in \{\mathsf{r}, \mathsf{u}\}$ *(and* u *users) is*

$$\mathbf{Adv}^{\mathsf{KDAE}[nt]}_{\mathsf{AEAD},\mathcal{A},u} := \left| 2 \cdot \Pr\left[b \xleftarrow{\$} \{0,1\}, b' = \mathsf{KDAE}^{[nt],b}_{\mathsf{AEAD}}(\mathcal{A}, u) : b = b'\right] - 1 \right|,$$

where the security game $\mathsf{KDAE}^{[nt],b}_{\mathsf{AEAD}}(\mathcal{A}, u)$ *is defined in Table 13.*

In the nonce-respecting case, i.e. $\mathsf{nt} = \mathsf{u}$, we cannot expect to get a "sufficiently" small advantage without a restriction on the allowed key-dependent message deriving functions ϕ_m due to the impossibility results of [33]. Therefore, muKDI\$ advantage will be associated to a class Φ of KDM deriving functions. Given the number u of users/keys, we consider all the functions $\mathcal{K}^u \mapsto \mathcal{M}$ which are oracle-free as defined in [64]: a function ϕ_m is oracle-free, if for any input $\mathbf{K}$ we have $\tau(\phi_m(\mathbf{K})) = \emptyset$, where $\tau(\phi_m(\mathbf{K}))$ is the transcript of queries made by ϕ_m to $\widetilde{\mathsf{IC}}$ when ϕ_m is evaluated on the input $\mathbf{K}$.

It's tempting to ask how muKDI\$ security interacts with leakages. The situation becomes more complicated since the secret keys are frequently used for deriving inputs, and it needs careful discussion whether leakages produced by these actions must be taken into account and, if not, why. This is out of the scope of this paper.

B. muKDI\$ *Security of* TEDT

Formally, define $\overrightarrow{q} = (q_e, q_d, q_{\widetilde{\mathsf{IC}}})$, and

$$\mathbf{Adv}^{\mathsf{muKDI\$}[\mathsf{nt}]}_{\mathsf{TEDT}}(u, \overrightarrow{q}, \sigma) := \max\left\{\mathbf{Adv}^{\mathsf{muKDI\$}[\mathsf{nt}]}_{\mathsf{TEDT},\mathcal{A},u}\right\},$$

where the maximal is taken over all $\overrightarrow{q}$-bounded adversaries against u users that have at most σ blocks in all its queried plaintext and ciphertext including associated data. In addition, we follow Black et al. [65] and assume that the message deriving functions are of *fixed-length*: formally, for any $i \in \{1, \ldots, q_e\}$, $|\phi_i[\widetilde{\mathsf{IC}}](\mathbf{K})| = |\phi_i[\widetilde{\mathsf{IC}}'](\mathbf{K}')|$ for any two ideal TBCs $\widetilde{\mathsf{IC}}, \widetilde{\mathsf{IC}}'$ and any $\mathbf{K}, \mathbf{K}' \in (\mathcal{K})^u$. Note that without this assumption, the key may be leaked by the length of the derived message, which completely ruins security. In addition, to model the setting where keys are passwords (likely in the muKDI$ setting), we assume that a secret key K is picked according to the distribution $\mathcal{K}$, and its min-entropy is

$$\mathbf{H}_\infty(\mathcal{K}) = -\log_2\big(\max_{K^\dagger \in \mathcal{K}} \Pr[K \xleftarrow{\$} \{0,1\}^n : K = K^\dagger]\big).$$

Then in the random nonce case we have

Theorem 5. *When the u public-keys $T_1, \ldots, T_u$ are uniformly distributed and $4 \le 4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}} \le 2^n/4$, in the ideal TBC model it holds*

$$\mathbf{Adv}^{\mathsf{muKDI\$}[r]}_{\mathsf{TEDT}}(u, \overrightarrow{q}, \sigma) \le \frac{6q_e(4\sigma + 6(q_e+q_d) + 2q_{\widetilde{\mathsf{IC}}})}{|\mathcal{N}|} + \frac{2u(u + 4\sigma + 6(q_e+q_d) + 2q_{\widetilde{\mathsf{IC}}})}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{18(6\sigma + 4(q_e+q_d) + 2q_{\widetilde{\mathsf{IC}}})^2}{2^n}. \tag{37}$$

Proof. See appendix I-C. The proof idea is similar to [65], and we employ the randomness mapping argument [66] to make it a bit more rigorous. □

In the universal nonce case, we follow [33] and make the additional assumption that the adversary only chooses *oracle-free* message deriving functions (see the definition subsequent to Definition 3). With this restriction, we have:

Theorem 6. *When the u public-keys $T_1, \ldots, T_u$ are uniformly distributed, $n \ge 6$, and $2\sigma + 3(q_e + q_d) + q_{\widetilde{\mathsf{IC}}} \le 2^n/8$, for any adversary that only queries with oracle-free message deriving functions, in the ideal TBC model it holds*

$$\mathbf{Adv}^{\mathsf{muKDI\$}[u]}_{\mathsf{TEDT}}(u, \overrightarrow{q}, \sigma) \le \frac{u^2}{2^{n+\mathbf{H}_\infty(\mathcal{K})}} + \frac{1}{2n!} \cdot \Big(\frac{4u}{2^n}\Big)^n + \frac{(5n^2+9)q^{**}_{\widetilde{\mathsf{IC}}} + 2q_d + 1}{2^n}.$$

Proof. The bound is comparable to muCCAm$ (Theorem 3), and the proof is also similar: see appendix I-D. □

While the oracle-freeness assumption appears quite strong here, it seems crucial for universal nonce KDM security. Moreover, it still captures the simple polynomial message deriving functions.

C. Proof of Theorem 5

Wlog assume that both $\mathcal{D}$ and ϕ are deterministic and with unlimited computation power. In this vein, the nonce and the ideal TBC $\widetilde{\mathsf{IC}}$ constitute the only underlying randomness of the muKDI$ security game. Denote by G_1 the real muKDI$ security game, and by G_3 the ideal game. We introduce some "bad events" and helper variables into G_3, and define the obtained game as G_2. In detail,

(1) We keep all the query-response pairs of $\widetilde{\mathsf{IC}}$ in a set $ICSet$;
(2) We maintain a set $SecretKeys$ for the keys that are supposed to be unknown to $\mathcal{D}$. These keys include the u user keys as well as all the ephemeral keys appeared during the encryption pass.
(3) We distinguish between "public" and "private" $\widetilde{\mathsf{IC}}$ oracles. The former is captured by a public procedure $\widetilde{\mathsf{IC}}^t_k(\delta, v)$, while the latter by a private procedure $\mathsf{ICinner}^t_k(\delta, v)$. The former is offered to $\mathcal{D}$ as the $\widetilde{\mathsf{IC}}$ oracle, while the latter is used by $\overrightarrow{\phi}$ and the encryption pass. The functions of the ideal TBC is actually implemented by $\mathsf{ICinner}^t_k$, and $\widetilde{\mathsf{IC}}^t_k$ only relays the call to $\mathsf{ICinner}^t_k$. However, upon a query, $\widetilde{\mathsf{IC}}^t_k$ would check if the key $k \in SecretKeys$. This constitutes the mechanism to prevent $\mathcal{D}$ from "guessing" the secret keys.
(4) Most importantly, while the ciphertext still consists of uniform blocks, we internally "enforce" them as query-response pairs of $\widetilde{\mathsf{IC}}$. This design resembles the idea used in [65] and we follow [66] and name it *adaptation*.
(5) G_2 aborts upon some "bad events". These in particular include the cases the "enforcement" of query-response pairs of $\widetilde{\mathsf{IC}}$ cause inconsistency.

In addition, to ease the argument we'll rely on the normalization (as in section E-C), which is made formal by defining another game G'_2. In detail, G'_2 internally performs normalization in the procedure $\mathsf{ICinner}^t_k$: this is the only difference between G_2 and G'_2. We formally describe both games in the following paragraph of pseudocode, with the normalization statements (in G'_2 only) clearly marked.

Game G_2, G'_2
Variables:
  Sets $ICSet$, $HSet$, and $\mathcal{S}_1, \ldots, \mathcal{S}_u$, initially empty
  Set $SecretKeys$ initialized to $\{K_1, \ldots, K_u\}$
  Keys $\mathbf{T}$ and $\mathbf{K}$, initialized according to their distributions.
Public procedure $\mathsf{Enc}(i, A, \phi)$

1. $encnum \leftarrow qunm$
2. $M \leftarrow \phi[\mathsf{ICinner}](\mathbf{K})$
3. parse M as $m_1\| \ldots \|m_\ell$
4. $C \xleftarrow{\$} \{0,1\}^{(\ell+1)n}$
5. parse C as $c_1\| \ldots \|c_\ell\|Z$
6. $y_1\| \ldots \|y_\ell \leftarrow (m_1\| \ldots \|m_\ell) \oplus (c_1\| \ldots \|c_\ell)$ // the "imagined" key stream
7. $N \xleftarrow{\$} \mathcal{N}$ // sample the nonce and start emulating the encryption
8. **If** $\exists y$ such that $(K_i, T_i, P_0(N), y) \in ICSet$ **then abort** // a bad nonce
9. $k_0 \leftarrow \mathsf{ICinner}^{T_i}_{K_i}(+, P_0(N))$
10. Add k_0 to $SecretKeys$
11. **If** $\exists t, x, y$ such that $(k_0, t, x, y) \in ICSet$ **then abort** // sampled a bad k_0
12. **for** $i = 1$ **to** ℓ **do** // emulate the encryption
13. $\quad \mathsf{Adapt}(k_{i-1}, T, Q_{i-1}(N), y_i)$
14. $\quad k_i \leftarrow \mathsf{ICinner}^{T}_{k_{i-1}}(+, P_i(N))$ // omitted for $i = \ell$
15. $\quad$ Add k_i to $SecretKeys$ // omitted for $i = \ell$
16. $\quad$ **If** $\exists t, x, y$ such that $(k_i, t, x, y) \in ICSet$ **then abort** // bad next key k_i
17. $V\|W \leftarrow \mathsf{H}[\widetilde{\mathsf{IC}}](\mathsf{pad}(A, N, c_1\| \ldots \|c_\ell, T_i))$
18. $\mathsf{Adapt}(K_i, W\|1, V, Z)$
19. $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{(N, A, C)\}$
20. Return (N, C)

Public procedure $\mathsf{Dec}(i, N, A, C)$

1. If $(N, A, C) \in \mathcal{S}_i$ then return $\perp$
2. parse C as $c_1\| \ldots \|c_\ell\|Z$
3. **if** $\exists V, W$ such that $(K_i, W, V, Z) \in ICSet$,
   and (K_i, W, V, Z) was created in an $\mathsf{ICinner}$-call with $\delta = +$ **then abort**
4. $V\|W \leftarrow \mathsf{H}[\widetilde{\mathsf{IC}}](\mathsf{pad}(A, N, c_1\| \ldots \|c_\ell, T_i))$
5. $V^* \leftarrow \mathsf{ICinner}^{W\|1}_{K_i}(-, Z)$
6. **if** $V = V^*$ **then abort**
7. Return 0 // $valid$ always equals 0

Public procedure $\mathsf{H}[\widetilde{\mathsf{IC}}](U)$

1. It calls the public procedure $\widetilde{\mathsf{IC}}^t_k$ to evaluate. Details omitted.

Public procedure $\widetilde{\mathsf{IC}}^t_k(\delta, v)$

1. **If** $k \in SecretKeys$ **then abort** // the case $\mathcal{D}$ guesses a secret key
2. **return** $\mathsf{ICinner}^t_k(\delta, v)$

Private procedure $\mathsf{ICinner}^t_k(\delta, v)$

1. **If** $\delta = +$ **and** $\nexists w : (k, t, v, w) \in ICSet$ **then** // forward query
2. $\quad w \xleftarrow{\$} \{0,1\}^n$ such that $\nexists v' : (k, t, v', w) \in ICSet$
3. $\quad$ Add (k, t, v, w) to $ICSet$
4. **If** $\delta = -$ **and** $\nexists w : (k, t, w, v) \in ICSet$ **then** // forward query
5. $\quad w \xleftarrow{\$} \{0,1\}^n$ such that $\nexists v' : (k, t, w, v') \in ICSet$
6. $\quad$ **If** $k = K_i$ **and** $\exists U, t$ such that $(U, w\|t) \in HSet$ **then abort**
7. $\quad$ Add (k, t, w, v) to $ICSet$
8. **If** $\delta = +$ **and** $\nexists w : (k, t, v \oplus \theta, w) \in ICSet$ **then** $\mathsf{ICinner}^t_k(+, v \oplus \theta)$ [G'_2 only!]
9. **If** $\delta = -$ **and** $\nexists w : (k, t, w \oplus \theta, w) \in ICSet$ **then** $\mathsf{ICinner}^t_k(+, w \oplus \theta)$ [G'_2 only!]
10. **If** $\delta = +$ **then** Find w such that $(k, t, v, w) \in ICSet$ and return w
11. **Else** Find w such that $(k, t, w, v) \in ICSet$ and return w

Private procedure $\mathsf{Adapt}(k, t, x, y)$

1. **If** $\exists x', y'$ such that $(k, t, x, y') \in ICSet$ or $(k, t, x', y) \in ICSet$ **then abort**
2. Add (k, t, x, y) to $\tau_{\widetilde{\mathsf{IC}}}$

While G_2 includes many complicated actions, it can be seen it actually behaves the same as G_3 in the view of $\mathcal{D}$ unless abortion. Therefore,

$$\Pr[\mathcal{D}^{\mathsf{G}_3} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_2} \Rightarrow 1] \leq \Pr[\mathsf{G}_2 \text{ aborts}],$$

and below we could concentrate on bounding $\Pr[\mathcal{D}^{\mathsf{G}_2} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_1} \Rightarrow 1]$. To this end, we follow the proof idea of Black et al. [65], but use the randomness mapping technique [66] to make it more rigorous. The analysis consists of two steps:

(1) First, we bound $\Pr[\mathsf{G}_2 \text{ aborts}]$, the probability of abortion in G_2 executions;
(2) Second, we consider the sets resulted from a non-aborting G_2 execution, and fix the internal randomness of G_1 to extend these sets. It's easy to show the transcripts of $\mathcal{D}$ in such G_1 and G_2 executions are the same, so that $\mathcal{D}$ outputs the same. This cinches the indistinguishability result.

1) Abort Probability of G_2.: We analyze the abort conditions in turn.

a) Inside the Enc *Procedure.:* For line 8, since the nonce is randomly picked, the probability clearly does not exceed $\frac{q_e q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|}$, where $q^*_{\widetilde{\mathsf{IC}}}$ is defined by Eq. (13). It's also bad if the nonce collides with a previously picked nonce, but this case has been included in the count $q^*_{\widetilde{\mathsf{IC}}}$.

For lines 11 and 16, if abortion did not occur then all the keys are newly sampled and thus uniform in $\geq 2^n - q^*_{\widetilde{\mathsf{IC}}}$ possibilities. Therefore, for each key the abort probability is $\leq \frac{q^*_{\widetilde{\mathsf{IC}}}}{2^n - q^*_{\widetilde{\mathsf{IC}}}} \leq \frac{2q^*_{\widetilde{\mathsf{IC}}}}{2^n}$ assuming $q^*_{\widetilde{\mathsf{IC}}} \leq 2^n/2$. Since the number of such ephemeral keys is at most $q^*_{\widetilde{\mathsf{IC}}}$, the total probability is $\leq \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n}$. It isn't hard to see if these abortions never occur, then the calls to Adapt in line 13 never abort either.

We finally consider the call to Adapt in line 18. Since N is uniformly picked from $\mathcal{N}$, the derived $n-1$ bit half W can also be proved uniform in at least $|\mathcal{N}|$ possibilities. Therefore, for the call to $\mathsf{Adapt}(K_i, W\|1, V, Z)$, we have

- $\Pr[\exists Z' : (K_i, W\|1, V, Z') \in ICSet] \leq \frac{q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|}$, and
- $\Pr[\exists V' : (K_i, W\|1, V', Z) \in ICSet] \leq \frac{q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|}$. This probability can be further reduced using the uniformness of Z, but this cannot improve the overall bound.

Therefore, the probability of aborting at line 18 is $\leq \frac{2q_e q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|}$. We make a side remark that, while the ciphertext $c_1\| \ldots \|c_\ell$ contains more entropy than N when $\ell \geq 1$, it may also be an empty string (when the derived message is empty). Consequently, we cannot rely on the uniformness of $c_1\| \ldots \|c_\ell$.

Summing over the above, we obtain

$$\Pr[\mathsf{Enc} \text{ aborts}] \leq \frac{3q_e q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|} + \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n}. \tag{38}$$

b) Inside the $\widetilde{\mathsf{IC}}^t_k$ Procedure.: As mentioned, the abort condition captures the scenario that $\mathcal{D}$ succeeds in guessing a secret key. To obtain a better bound, we distinguish between the user-keys $\{K_1, \ldots, K_u\}$ and the internal ephemeral keys. And then we follow the argument of Black et al. [65]:

- First, note that if $\mathcal{D}$ was given no information about the user-keys then clearly $\Pr[k \in \{K_1, \ldots, K_u\}] = \frac{u}{2^{\mathbf{H}_\infty(\mathcal{K})}}$;
- Second, note that the view provided to $\mathcal{D}$ in G_2 is indeed independent of the keys:
  (1) every encryption query returns a random string whose length is independent of the keys (due to the fixed-length assumption on $\overrightarrow{\phi}$);
  (2) unless abortion occurred, every $\widetilde{\mathsf{IC}}^t_k$ query returns a random n-bit value independent of the keys (this value only does equal some previously known values);
  (3) unless abortion occurred, every Dec query simply returns $\perp$ or 0 independent of the keys (the return value can be fully decided by the sets $\mathcal{S}_1, \ldots, \mathcal{S}_u$).

Since the number of calls to $\widetilde{\mathsf{IC}}^t_k$ does not exceed $q^*_{\widetilde{\mathsf{IC}}}$, the probability of this type of abortion is $\leq \frac{u q^*_{\widetilde{\mathsf{IC}}}}{2^{\mathbf{H}_\infty(\mathcal{K})}}$.

On the other hand, an ephemeral key k^* is derived during the emulated encryption process. If abortion never occurred, then k^* was uniform in $\geq 2^n - q^*_{\widetilde{\mathsf{IC}}}$ possibilities. In a similar vein to the above, the view provided to $\mathcal{D}$ is indeed independent of these "secret keys". Therefore, the probability of this type of abortion is $\leq \frac{q^*_{\widetilde{\mathsf{IC}}} \cdot q^*_{\widetilde{\mathsf{IC}}}}{2^n - q^*_{\widetilde{\mathsf{IC}}}} \leq \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n}$ since the number of such "secret keys" is $\leq q^*_{\widetilde{\mathsf{IC}}}$, and thus

$$\Pr[\widetilde{\mathsf{IC}}^t_k \text{ aborts}] \leq \frac{u q^*_{\widetilde{\mathsf{IC}}}}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n}. \tag{39}$$

c) Inside the calls to $\mathsf{ICinner}^t_k$ with $\delta = -$.: This condition is quite clear:

$$\Pr[\mathsf{ICinner}^t_k \text{ aborts}] \leq \frac{q^*_{\widetilde{\mathsf{IC}}} \cdot q^*_{\widetilde{\mathsf{IC}}}}{2^n - q^*_{\widetilde{\mathsf{IC}}}} \leq \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n}. \tag{40}$$

d) Inside the Dec *Procedure.:* We first consider line 3, which captures the scenario that a right tag Z was created during the evaluation of $\overrightarrow{\phi}$, which—of course,—was unknown to $\mathcal{D}$, but $\mathcal{D}$ latter "wisely" succeeds in guessing Z. Since line 3 only considered the case Z resulting from a forward call to $\mathsf{ICinner}^t_k$, such a "target" Z is uniform in $\geq 2^n - q^*_{\widetilde{\mathsf{IC}}}$ values. Therefore, in a similar vein to the argument for Eq. (39), it can be shown the probability of this type of abortion is $\leq \frac{q^*_{\widetilde{\mathsf{IC}}} \cdot q^*_{\widetilde{\mathsf{IC}}}}{2^n - q^*_{\widetilde{\mathsf{IC}}}} \leq \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n}$.

For line 6, we rely on the obvious fact that

$$\Pr[\text{In } \mathsf{G}_2,\ \text{Dec aborts at line 6}] \le \Pr[\text{In } \mathsf{G}'_2,\ \text{Dec aborts at line 6}]. \tag{41}$$

We introduce two helper bad events in G'_2:

- (C-1) there exists two user indices i, j such that $K_i = K_j$;
- (C-2) hash collision: there exists two distinct records $(U, V\|W)$ and $(U', V'\|W')$ in $HSet$ such that $V\|W = V'\|W'$.

Via an analysis similar to section E-C, we reach

$$\Pr[\text{(C-1)} \vee \text{(C-2)}] \le \frac{u^2}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{4q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}, \tag{42}$$

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

Conditioned on ¬(C-1) and ¬(C-2), we argue none (non-trivial) decryption queries aborts at line 6 except negligible. Consider a decryption query $\mathsf{Dec}_{\mathbf{K}}(i, N, A, C)$, and let $C = \overrightarrow{c}\|Z$. If this query aborts at line 6, then at that time there exists a hash record $(\mathsf{pad}(A\|N\|\overrightarrow{c}\|T_i), V\|W)$ and an $\widetilde{\mathsf{IC}}$ query $(K_i, W^*\|1, V^*, Z)$ such that $V^* = V$ and $W^* = W$. This implies the existence of the following queries:

$$\big((u, h, g, g \oplus V), (u, h, g \oplus \theta, g \oplus \theta \oplus W\|b)\big), (K_i, W^*\|1, V^*, Z),$$

where u is the last block of $\mathsf{pad}(A\|N\|\overrightarrow{c}\|T_i)$, b is either 0 or 1, and $V^* = V$ and $W^* = W$. Note that it necessarily holds $u \notin SecretKeys$ since the previous calls to $\widetilde{\mathsf{IC}}^h_u$ did not abort. We distinguish two cases:

e) 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 since line 3 did not cause abortion, this query was not created due to calls made by $\overrightarrow{\phi}$. So it's due to an earlier encryption query $\mathsf{LEnc}_{\mathbf{K}}(j, N', A', M') \to \overrightarrow{c}'\|Z$, and during the hash-evaluation of this query, the last-block call also gives rise to the digest $V\|W$. Now,

- if $j = i$ and $(N, A, \overrightarrow{c}) = (N', A', \overrightarrow{c}')$, then the tag produced by $\mathsf{LEnc}_{\mathbf{K}}(j, N', A', M')$ cannot be Z, and thus cannot create the query $(K_i, W\|1, V^*, Z)$;
- if $j = i$ while $(N, A, \overrightarrow{c}) \neq (N', A', \overrightarrow{c}')$, then it contradicts ¬(C-2);
- if $j \neq i$ then $K_j \neq K_i$ by ¬(C-1), and thus impossible.

In all, $(K_i, W\|1, V^*, Z)$ has to be backward. Then $V = V^*$ would contradict the non-abortion of $\mathsf{ICinner}^t_k$-calls. Consequently, this case never causes abortion at line 6.

f) CASE 2*: $(K_i, W^*\|1, V^*, Z)$ is created Before the pair of IC queries.:* Then, following the same line as the muCIML2 analysis, for the query $(u, h, g, g \oplus V)$, it holds

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

Therefore,

$$\Pr[\text{Case 2} \mid \neg\text{(C-1)} \wedge \neg\text{(C-2)}] \le \frac{2(q^{**}_{\widetilde{\mathsf{IC}}})^2}{2^n}. \tag{43}$$

Since we assumed $q^{**}_{\widetilde{\mathsf{IC}}} \ge 4$, $\frac{4q^{**}_{\widetilde{\mathsf{IC}}}}{2^n} \le \frac{(q^{**}_{\widetilde{\mathsf{IC}}})^2}{2^n}$. Therefore, gathering Eq. (41), (42), and (43) yields

$$\Pr[\mathsf{G}_2 \text{ aborts at line 6 in Dec}] \le \frac{u^2}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{3(q^{**}_{\widetilde{\mathsf{IC}}})^2}{2^n}. \tag{44}$$

Note that here we cannot rely on the muCIML2 result, since $\widetilde{\mathsf{IC}}_{K_i}$ can be called by $\overrightarrow{\phi}$ which interrupts the analysis. In summary, Eq. (38), (39), (40), and (44) yield

$$\begin{aligned}\Pr[\mathsf{G}_2 \text{ aborts}] \le& \frac{3q_e q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|} + \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n} + \frac{uq^*_{\widetilde{\mathsf{IC}}}}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n} + \frac{2(q^*_{\widetilde{\mathsf{IC}}})^2}{2^n} + \frac{u^2}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{3(q^{**}_{\widetilde{\mathsf{IC}}})^2}{2^n} \\ \le& \frac{3q_e q^*_{\widetilde{\mathsf{IC}}}}{|\mathcal{N}|} + \frac{u(u + q^*_{\widetilde{\mathsf{IC}}})}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{9(q^{**}_{\widetilde{\mathsf{IC}}})^2}{2^n} \qquad (q^*_{\widetilde{\mathsf{IC}}} \le q^{**}_{\widetilde{\mathsf{IC}}}).\end{aligned} \tag{45}$$

2) The Randomness Mapping.: Consider a non-aborting G_2 execution. We collect the sets of variables standing at the end of this execution as

$$\tau = (ICSet, \mathbf{N}, \mathbf{K}, \mathbf{T}).$$

In this tuple, the entry $ICSet$, $\mathbf{K}$, and $\mathbf{T}$ are clear, while the vector $\mathbf{N} = (N_1, \ldots, N_{q_e})$ contains the q_e randomly picked nonce values. It isn't hard to see this tuple τ keeps all the randomness necessary and sufficient to recover the corresponding G_2 execution. This is clear for $\widetilde{\mathsf{IC}}$ and Dec queries. For Enc queries, the random ciphertext blocks have been kept in $ICSet$ while nonce kept in $\mathbf{N}$, so it's also clear.

We next make the randomness in G_1 explicit. Note that the randomness source $\widetilde{\mathsf{IC}}$ has been "explicit". We only need to add a tape of uniform values h, and assume that the nonce values are "downloaded" from this tape as $N_i \leftarrow h(i)$ for the i-th encryption query. And we denote by $h \vdash \mathbf{N}$ the event $\forall i : h(i) = N_i$. Then clearly, for a fixed "target" tuple $\tau = (ICSet, \mathbf{N}, \mathbf{K}^\dagger, \mathbf{T}^\dagger)$ we have

$$\Pr_{\widetilde{\mathsf{IC}},h,\mathbf{K},\mathbf{T}}[\widetilde{\mathsf{IC}} \vdash ICSet \wedge h \vdash \mathbf{N} \wedge \mathbf{K} = \mathbf{K}^\dagger \wedge \mathbf{T} = \mathbf{T}^\dagger] \geq \Pr[\widetilde{\mathsf{IC}} \vdash ICSet] \cdot \Pr(\mathbf{N}, \mathbf{K}^\dagger, \mathbf{T}^\dagger).$$

On the other hand, since in a G_2 execution every adapted entry has an n-bit value picked uniformly, it can be seen the probability that a G_2 execution produces the variables satisfies

$$\Pr[\mathsf{G}_2 \to (ICSet, \mathbf{N}, \mathbf{K}^\dagger, \mathbf{T}^\dagger)] = \Pr[\widetilde{\mathsf{IC}} \vdash ICSet^R] \cdot \left(\frac{1}{2^n}\right)^{|ICSet^A|} \cdot \Pr(\mathbf{N}, \mathbf{K}^\dagger, \mathbf{T}^\dagger),$$

where $ICSet^R$ and $ICSet^A$ are the sets of sampled and adapted entries in $ICSet$ respectively. Therefore,

$$\frac{\Pr_{\widetilde{\mathsf{IC}},h,\mathbf{K},\mathbf{T}}[\widetilde{\mathsf{IC}} \vdash ICSet \wedge h \vdash \mathbf{N} \wedge \mathbf{K} = \mathbf{K}^\dagger \wedge \mathbf{T} = \mathbf{T}^\dagger]}{\Pr[\mathsf{G}_2 \to (ICSet, \mathbf{N}, \mathbf{K}^\dagger, \mathbf{T}^\dagger)]} \geq \frac{\Pr[\widetilde{\mathsf{IC}} \vdash ICSet^A \mid \widetilde{\mathsf{IC}} \vdash ICSet^R]}{\left(\frac{1}{2^n}\right)^{|ICSet^A|}} \geq 1. \tag{46}$$

In addition, it can be seen for a G_2 execution that produces $(ICSet, \mathbf{N}, \mathbf{K}, \mathbf{T})$, if a G_1 execution is ran with $\mathbf{K}, \mathbf{T}$, and randomness $\widetilde{\mathsf{IC}} \vdash ICSet$ and $h \vdash \mathbf{N}$, then the transcripts of queries and answers of $\mathcal{D}$ obtained in these two executions are exactly the same, so that $\mathcal{D}$ outputs the same. This could be formally proved via a transcript centric argument like in [66]; to keep it cleaner we eschew this tedious analysis. Denote by $\mathcal{T}_{nb}$ the set of variables τ that can be produced by non-aborting G_2 executions. Then we have (note $q^*_{\widetilde{\mathsf{IC}}} \leq q^{**}_{\widetilde{\mathsf{IC}}} \leq 4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}}$)

$$\begin{aligned}
&\Pr\left[\mathcal{D}^{\mathsf{G}_2} \Rightarrow 1\right] - \Pr\left[\mathcal{D}^{\mathsf{G}_1} \Rightarrow 1\right] \\
\leq &\Pr[\mathsf{G}_2 \text{ aborts}] + \sum_{\tau \in \mathcal{T}_{nb}} \Pr[\mathsf{G}_2 \to \tau] \cdot \left(1 - \frac{\Pr[(\widetilde{\mathsf{IC}}, h, \mathbf{K}, \mathbf{T}) \vdash \tau]}{\Pr[\mathsf{G}_2 \to \tau]}\right) \\
\leq &\Pr[\mathsf{G}_2 \text{ aborts}] \leq \frac{3q_e(4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}})}{|\mathcal{N}|} + \frac{u(u + 4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}})}{2^{\mathbf{H}_\infty(\mathcal{K})}} + \frac{9(4\sigma + 6(q_e + q_d) + 2q_{\widetilde{\mathsf{IC}}})^2}{2^n}.
\end{aligned}$$

This and $\Pr[\mathcal{D}^{\mathsf{G}_3} \Rightarrow 1] - \Pr[\mathcal{D}^{\mathsf{G}_2} \Rightarrow 1] \leq \Pr[\mathsf{G}_2 \text{ aborts}]$ establish Eq. (37).

D. Proof of Theorem 6

The proof is much simpler than section I-C since $\overrightarrow{\phi}$ are oracle-free: we could simply apply H-coefficients technique and follow the proof idea in section VII.

1) Bad Transcripts.: In this setting, transcripts are defined as $\tau = (\tau^*_{\mathsf{H}}, \tau_e, \tau_d, \tau^*_{\widetilde{\mathsf{IC}}}, \mathbf{T}, \mathbf{K})$, where

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

and

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

summarize the queries to the encryption and decryption oracles respectively, and $\tau^*_{\widetilde{\mathsf{IC}}}$ contains the adversarial $\widetilde{\mathsf{IC}}$ queries & the $\widetilde{\mathsf{IC}}$ queries made by H (note that here there isn't any "non-challenge" encryption queries, so it's slightly different from section H-A). Note that τ is attainable only if τ can be produced in the ideal world, and this means $b^{(1)} = \ldots = b^{(q_e)} = 0$.

Now, a transcript τ is bad if one of the following conditions 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 $\mu_W \geq n$, or $\mu_Y \geq 2\log_2 \sigma$;
- (B-3) there exists a 4-tuple $(i, N, A, \overrightarrow{c})$ such that,
  - either $(i, N, A, \phi, \overrightarrow{c} \| Z) \in \tau_e$, or $(i, N, A, \overrightarrow{c} \| Z, b) \in \tau_d$; and
  - $(K_i, T_i, *, *) \in \tau^*_{\widetilde{\mathsf{IC}}}$, or $(K_i, W \| 1, *, *) \in \tau^*_{\widetilde{\mathsf{IC}}}$ for the corresponding hash record $(\mathsf{pad}(A, N, \overrightarrow{c}, T_i), V \| W) \in \tau^*_{\mathsf{H}}$.

- (B-4) *hash collision*: there exists two distinct hash query records $(U, V\|W)$ and $(U', V'\|W')$ in τ^*_{H} such that $U \neq U'$ yet $V\|W = V'\|W'$;
- (B-5) there exists two distinct encryption queries $(u^{(i)}, N^{(i)}, A^{(i)}, \phi^{(i)}, C^{(i)})$ and $(u^{(j)}, N^{(j)}, A^{(j)}, \phi^{(j)}, C^{(j)})$ with hash query $(\mathsf{pad}(A^{(i)}\|N^{(i)}\|\overrightarrow{c}^{(i)}\|T_{u^{(i)}}), V^{(i)}\|W^{(i)})$ and $(\mathsf{pad}(A^{(j)}\|N^{(j)}\|\overrightarrow{c}^{(j)}\|T_{u^{(j)}}), V^{(j)}\|W^{(j)})$ in τ^*_{H} satisfying: $W^{(i)} = W^{(j)}$, and $Z^{(i)} = Z^{(j)}$;
- (B-6) *incorrect plaintext length:* there exists a query $(u^{(i)}, N^{(i)}, A^{(i)}, \phi^{(i)}, C^{(i)}) \in \tau_e$ such that $|\phi^{(i)}| \neq |C^{(i)}| - n$.

We can recycle most of the results in section H-A:

$$\Pr[\text{(B-1)}] \leq \frac{u^2}{2^{n+\mathbf{H}_\infty(\mathcal{K})}},\ \ \Pr[\text{(B-2)} \vee \text{(B-4)}] \leq \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{8q^{**}_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{1}{2^n},$$
$$\Pr[\text{(B-3)} \mid \text{(B-2)}] \leq \frac{n^2 q^{**}_{\widetilde{\mathsf{IC}}}}{2^n},\ \text{and}\ \ \Pr[\text{(B-5)}] \leq \frac{2q_e^2}{2^n(2^n - q^{**}_{\widetilde{\mathsf{IC}}})} \leq \frac{4q_e^2}{2^{2n}} \leq \frac{q^{**}_{\widetilde{\mathsf{IC}}}}{2^n}.$$

Finally, since $\overrightarrow{\phi}$ are assumed to be of fixed-length, we actually have $\Pr[\text{(B-6)}] = 0$. This condition was presented only for clearness. Therefore,

$$\Pr[T_{id} \in \mathcal{T}_{bad}] \leq \frac{u^2}{2^{n+\mathbf{H}_\infty(\mathcal{K})}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(n^2+9)q^{**}_{\widetilde{\mathsf{IC}}} + 1}{2^n}.$$

2) Ratio for Good Transcripts.: Fix a good τ. For each encryption query record $(u^{(i)}, N^{(i)}, A^{(i)}, \phi^{(i)}, C^{(i)})$, we "replace" the field $\phi^{(i)}$ by $M^{(i)} = \phi^{(i)}(\mathbf{K})$. Here we eschew the notation $\phi^{(i)}[\widetilde{\mathsf{IC}}]$ since it's oracle-free. In this vein, the set τ_e turns the same as the set τ_e analyzed in section H-B. Importantly, the message deriving functions would not affect the calculations since they are oracle-free. So we can recycle the result

$$\begin{aligned}\Pr[\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \vdash \tau_e \mid \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}] \geq& \left(1 - \frac{2\mu_T(\sigma + q^{**}_{\widetilde{\mathsf{IC}}}) + 2\mu_Y(\sigma + q^{**}_{\widetilde{\mathsf{IC}}})}{2^n}\right)\left(\frac{1}{2^n}\right)^{q_e + \sum_{i=1}^{q_e} \ell_i} \\ \geq& \left(1 - \frac{8nq^{**}_{\widetilde{\mathsf{IC}}}}{2^n}\right)\left(\frac{1}{2^n}\right)^{q_e + \sum_{i=1}^{q_e} \ell_i} \end{aligned} \tag{47}$$

We only need to lower bound

$$\Pr[\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \vdash \tau_d \mid \mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \vdash \tau_e \wedge \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}].$$

For this, we upper bound the probability

$$\Pr[\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \nvdash \tau_d \mid \mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \vdash \tau_e \wedge \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}].$$

This requires to prove any decryption query $\mathsf{Dec}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}}(i, N, A, C)$ results in 0 in the real world except with negligible probability. Let $C = (\overrightarrow{c}, Z)$, and let $V\|W = \mathsf{H}[\widetilde{\mathsf{IC}}](\mathsf{pad}(A\|N\|\overrightarrow{c}\|T_i))$. We distinguish two case:

a) Case 1: there exists an encryption query $(u^{(i)}, N^{(i)}, A^{(i)}, M^{(i)}, C^{(i)})$ such that $K_{u^{(i)}} = K_i$, and $D^{(i)}\|W^{(i)} = \mathsf{H}[\widetilde{\mathsf{IC}}](\mathsf{pad}(A^{(i)}\|N^{(i)}\|\overrightarrow{c}^{(i)}\|T_{u^{(i)}})) = V\|W$, where $C^{(i)} = (\overrightarrow{c}^{(i)}, Z^{(i)})$. By ¬(B-4) and Lemma 8, this implies $(N, A, \overrightarrow{c}, T_i) = (N^{(i)}, A^{(i)}, \overrightarrow{c}^{(i)}, T_{u^{(i)}})$. Then since we forbid trivial decryption query, it has to be $Z^{(i)} \neq Z$. By this, $\widetilde{\mathsf{IC}}^{W\|1}_{K_i}(V) = Z^{(i)} \neq Z$ implies this decryption query results the answer 0.

b) Case 2: otherwise, then conditioned on $\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \vdash \tau_e \wedge \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}$, the value $\widetilde{\mathsf{IC}}^{W\|1}_{K_i}(V)$ remains uniform in $\geq 2^n - q_e - q_d$ possibilities, and thus

$$\Pr[\mathsf{Dec}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}}(i, N, A, C) = 1] = \Pr[\widetilde{\mathsf{IC}}^{W\|1}_{K_i}(V) = Z] \leq \frac{1}{2^n - q_e - q_d} \leq \frac{2}{2^n}.$$

Taking a union bound over the q_d decryption queries yields

$$\Pr[\mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \nvdash \tau_d \mid \mathsf{TEDT}[\widetilde{\mathsf{IC}}]_{\mathbf{K},\mathbf{T}} \vdash \tau_e \wedge \widetilde{\mathsf{IC}} \vdash \tau^*_{\widetilde{\mathsf{IC}}}] \leq \frac{2q_d}{2^n}. \tag{48}$$

Gathering Eq. (47) and (48) and $\Pr[T_{id} \in \mathcal{T}_{bad}]$ yields the final bound

$$\begin{aligned}&\frac{u^2}{2^{n+\mathbf{H}_\infty(\mathcal{K})}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(n^2+9)q^{**}_{\widetilde{\mathsf{IC}}} + 1}{2^n} + \frac{8nq^{**}_{\widetilde{\mathsf{IC}}}}{2^n} + \frac{2q_d}{2^n} \\ \leq& \frac{u^2}{2^{n+\mathbf{H}_\infty(\mathcal{K})}} + \frac{1}{2n!} \cdot \left(\frac{4u}{2^n}\right)^n + \frac{(5n^2+9)q^{**}_{\widetilde{\mathsf{IC}}} + 2q_d + 1}{2^n} \quad (n \geq 2).\end{aligned}$$

E. Insecurity with a Single Nonce Reuse

We target a single user-key (K, T). We first use the identity function and an arbitrary nonce N for an encryption query: this cause TEDT compute $k_0 := \widetilde{\mathsf{E}}_K^T(P_0(N))$, $y_1 := \widetilde{\mathsf{E}}_{k_0}^T(Q_0(N))$, and $c_1 := y_1 \oplus K$. We don't care about the subsequent tag generation. We then use the constant function $\phi(K) = 0$ and the nonce N (*repeating*) for the second encryption query: the resulted ciphertext block is $c_1^* = y_1$. Then $K = c_1 \oplus c_1^*$ is recovered (even if we don't need to decide the pseudorandomness of c_1 and c_1^*).

The situation here deviates from section VII because TEDT doesn't ensure the confidentiality of messages encrypted by repeated nonce: this isn't harmful in the $\mathsf{muCCAm\$}$ setting, but is harmful here since the message may be the secrete key.

How to design a $\mathsf{muKDI\$}$ secure scheme in the misuse-resilience is out of the scope of this paper.