Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions[⋆]

Kazuhiko Minematsu

NEC Corporation, Japan
k-minematsu@ah.jp.nec.com

Abstract. This paper proposes a new scheme for authenticated encryption (AE) which is typically realized as a blockcipher mode of operation. The proposed scheme has attractive features for fast and compact operation. When it is realized with a blockcipher, it requires one blockcipher call to process one input block (i.e. rate-1), and uses the encryption function of the blockcipher for both encryption and decryption. Moreover, the scheme enables one-pass, parallel operation under two-block partition. The proposed scheme thus attains similar characteristics as the seminal OCB mode, without using the inverse blockcipher. The key idea of our proposal is a novel usage of two-round Feistel permutation, where the round functions are derived from the theory of tweakable blockcipher. We also provide basic software results, and describe some ideas on using a non-invertible primitive, such as a keyed hash function.

Keywords: Authenticated Encryption, Blockcipher Mode, Pseudorandom Function, OCB.

1 Introduction

Authenticated encryption. Authenticated encryption, AE for short, is a method to simultaneously provide message confidentiality and integrity (authentication) using a symmetric-key cryptographic function. Although a secure AE function can be basically obtained by an adequate composition of secure encryption and message authentication [14, 32], this requires at least two independent keys, and the composition methods in practice (say, AES + HMAC in TLS) frequently deviate from what proved to be secure [42]. Considering this situation, there have been numerous efforts devoted to efficient, one-key constructions. Among many approaches to AE, blockcipher mode of operation is one of the most popular ones. We have CCM [3], GCM [5], EAX [16], OCB [33, 44, 47] and the predecessors [26, 31], and CCFB [36], to name a few. We have some standards, such as NIST SP 800-38C (CCM) and 38D (GCM), and ISO/IEC 19772 [6].

This paper presents a new AE mode using a blockcipher, or more generally, a pseudorandom function (PRF). Our proposal has a number of desirable features for fast and compact operations. Specifically, when the underlying n-bit blockcipher is E_K (where K denotes the key), the properties of our proposal can be summarized as follows.

- The key is one blockcipher key, K.
- Encryption and decryption can be done by the encryption function of E_K.
- For s-bit input, the number of E_K calls is $\lceil s/n \rceil + 2$, i.e., rate-1 processing, for both encryption and decryption.
- On-line, one-pass, and parallel encryption and decryption, under two-block partition.
- Provable security up to about $2^{n/2}$ input blocks, based on the assumption that E_K is a pseudorandom function (PRF) or a pseudorandom permutation (PRP).

These features are realized with a novel usage of two-round Feistel permutation, where internal round functions are PRFs with input masking. From this we call our proposal OTR, for Offset Two-round. Table 1 provides a summary of properties of popular AE modes and ours, which shows that OTR attains similar characteristics as the seminal OCB mode, without using the inverse blockcipher. The proposed scheme generates

[⋆] A preliminary version appears at Eurocrypt 2014. This is the full version (version 20170602). This paper also contains additional sections in the appendix to define variants of the scheme presented in the proceeding of Eurocrypt 2014. These variants are included in the submission to CAESAR competition [38]. Bost and Sanders [21] reported a flaw in the proof of original specification with respect to masking constants. Reflecting [21], this paper presents the updated specification. Software implementation results (Section 6) are not updated but still useful as the update has negligible impact on efficiency.

input masks to E_K using GF(2^n) constant multiplications. This technique is called GF doubling [44], which is a quite popular tool for mode design. However, our core idea is rather generic and thus allows other masking methods. We also remark that Liting et al.'s iFeed mode [51] has similar properties to ours, without introducing 2-block partition. However, its decryption is inherently serial. In return for these attractive features, one potential drawback of OTR is that it inherently needs two-block partition (though the message itself can be of any length in bits), which implies more state memories required than that of OCB. The parallelizability of our scheme is up to the half of the message blocks, while OCB has full parallelizability, up to the number of message blocks. On-line processing capability is restrictive as it needs buffering of consecutive two input blocks.

We also warn that the security is proved for the standard nonce-respecting adversary [45], i.e. the encryption never processes duplicate nonces (or initial vectors), see Section 2.2. Some recent proposals have a provable security under nonce-reusing adversary, or even security without nonce (called on-line encryption) [9,25]. However we do not claim any security guarantee for such adversaries.

Table 1. A comparison of AE modes. Calls denotes the number of calls for m-block message and a-block header and one-block nonce, without constants.

Mode	Calls	On-line	Parallel	Primitive
CCM [3]	$a + 2m$	no	no	E
GCM [5]	m [E] and $a + m$ [Mul]	yes	yes	$E, \mathrm{Mul}^{\dagger}$
EAX [16]	$a + 2m$	yes	no	E
OCB [33,44,47]	$a + m$	yes	yes	E, E^{-1}
CCFB [36]	$a + cm$ for some $1 < c^{\ddagger}$	yes	no	E
OTR	$a + m$	yes¶	yes¶	E

† GF(2^n) multiplication
‡ Security degrades as c approaches 1
¶ two-block partition

Benefits of inverse-freeness. The use of blockcipher inversion, as in OCB, has mainly two drawbacks, as discussed by Iwata and Yasuda [30]. The first is efficiency. The integration of encryption and decryption functions increases size, e.g. footprint of hardware, or memory of software (See Section 6). Moreover, some ciphers have unequal speed for enc/dec. For AES, decryption is slower than encryption on some, typically constrained, platforms. For example, an AES implementation on Atmel AVR by Osvik et al. [41] has about 45% slower decryption than encryption. This property is the initial design choice [23], in preference of encryption-only mode, e.g., CTR, OFB, and CFB. IDEA is another example, where decryption is exceptionally slower than encryption on microcontrollers [43]. The uneven performance figures of blockcipher enc/dec functions is undesirable in practice, when the mode uses both functions.

The second is security. Usually the security of a mode using both enc/dec functions of a blockcipher, denoted by E and E^{-1}, needs (E, E^{-1}) to be a strong pseudorandom permutation (Strong PRP or SPRP). This holds true for the original security proofs of all versions of OCB [33, 44, 47], though a recent work of Aoki and Yasuda [11] showed a relaxation on the security condition for OCB. In contrast, when the mode uses only E, the security assumption is relaxed to PRP or PRF.

In addition, the inverse-freeness allows instantiations using non-blockcipher primitives, such as a hash function. Some basic ideas on this direction are explained in Section 7.4.

Hardware assistance. We remark that some software platforms have hardware-assisted blockcipher, most notably AES instructions called AESNI in Intel CPUs. AMD CPUs also have an equivalent set. AESNI enables the same performance for AES encryption and decryption. Therefore, when our proposal uses AESNI, the performance would be roughly similar to that of OCB-AES with AESNI, though the increased number of states may degrade the result. We have other SW platforms where hardware AES is available but decryption is slower (e.g., [27]). Basically, the value of our proposal is *not* to provide the fastest operation on modern CPUs, instead, to increase the availability of OCB-like performance for various platforms, using single algorithm.

Related works. Our scheme has a similar structure as OCB [33, 44, 47], which seems essential to integrate encryption and authentication keeping parallelizability. The idea of using Feistel rounds with pseudorandom round functions for building AE seems to go back to the proposal of ManTiCore [7], and they also described the idea of using two-round Feistel with hash function in [8], while this paper is an independent work.

2 Preliminaries

2.1 Basic Notations

Let $\mathbb{N} = \{1, 2, \ldots, \}$, and let $\{0,1\}^*$ be the set of all finite-length binary strings, including the empty string ε. The bit length of a binary string X is denoted by $|X|$, and let $|X|_a \overset{\text{def}}{=} \max\{\lceil |X|/a \rceil, 1\}$. Here, if $X = \varepsilon$ we have $|X|_a = 1$ for any $a \geq 1$ and $|X| = 0$. A concatenation of $X, Y \in \{0,1\}^*$ is written as $X \| Y$ or simply XY. A sequence of a zeros is denoted by 0^a. For $k \geq 1$, we denote $\bigcup_{i=1}^{k} \{0,1\}^i$ by $\{0,1\}^{\leq k}$. For $X \in \{0,1\}^*$, let $(X[1], \ldots, X[x]) \overset{n}{\leftarrow} X$ denote the n-bit block partitioning of X, i.e., $X[1] \| X[2] \| \ldots \| X[x] = X$ where $x = |X|_n$, and $|X[i]| = n$ for $i < x$ and $|X[x]| \leq n$. If $X = \varepsilon$ the parsing with any $n \geq 1$ makes $x = 1$, $X[1] = \varepsilon$. The sequence of first c bits of $X \in \{0,1\}^*$ is denoted by $\mathtt{msb}_c(X)$. We have $\mathtt{msb}_0(X) = \varepsilon$ for any X.

For a finite set $\mathcal{X}$, if X is uniformly chosen from $\mathcal{X}$ we write $X \overset{\$}{\leftarrow} \mathcal{X}$. We assume $X \oplus Y$ is ε if X or Y is ε. For a binary string X with $0 \leq |X| \leq n$, $\underline{X}$ denotes the padding written as $X \| 1 \| 0^{n-|X|-1}$. When $|X| = n$, $\underline{X}$ denotes X.

For keyed function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ with key $K \in \mathcal{K}$, we may simply write $F_K : \mathcal{X} \to \mathcal{Y}$ if key space is obvious, or even write as F if being keyed with K is obvious. If $E_K : \mathcal{X} \to \mathcal{X}$ is a keyed permutation, or a blockcipher, E_K is a permutation over $\mathcal{X}$ for every $K \in \mathcal{K}$. Its inverse is denoted by E_K^{-1}. A keyed function may have an additional parameter called tweak, in the sense of Liskov, Rivest and Wagner [34]. It is called a tweakable keyed function and written as $\widetilde{F} : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$ or $\widetilde{F}_K : \mathcal{T} \times \mathcal{X} \to \mathcal{Y}$, where $\mathcal{T}$ denotes the space of tweaks. Instead of writing $\widetilde{F}_K(T, X)$, we may write as $\widetilde{F}_K^{\langle T \rangle}(X)$. A tweakable keyed permutation, or a tweakable blockcipher (TBC), is defined analogously by requiring that every combination of (T, K) produces a permutation over $\mathcal{X}$.
Galois field. An n-bit string X may be viewed as an element of $\mathrm{GF}(2^n)$ by taking X as a coefficient vector of a polynomial in $\mathrm{GF}(2^n)$. Following Rogaway [44], we write $2X$ to denote the multiplication of 2 and X over $\mathrm{GF}(2^n)$, where 2 denotes the generator of the field $\mathrm{GF}(2^n)$, by seeing 2 as $\mathtt{x}$ in the polynomial representation. This operation is called *doubling*. Similarly we write $3X$ (where the corresponding polynomial is $\mathtt{x+1}$) and 2^2X to denote $2X \oplus X$ and $2(2X)$, and $7X$ to denote $2(2X) \oplus 2X \oplus X$. We may write $4X$ to denote 2^2X. The doubling can be efficiently computed by one-bit shift with conditional XOR of a constant, and other constant multiplications can be done by combining doubling and XOR, as shown above. There operations are frequently used as a tool to build efficient blockcipher modes, e.g. [16, 28, 44].

Throughout the paper we assume $n = 128$ and the corresponding field $\mathrm{GF}(2^n)$ is defined over the polynomial $\mathtt{x}^{128} + \mathtt{x}^7 + \mathtt{x}^2 + \mathtt{x}^1 + 1$, which is lexicographically-first primitive (thus irreducible) polynomial and is quite popular for doubling-based tweaks. The scheme itself is general and is easily extended to other block sizes, however care must be taken since masking coefficients are specific to the choice of n and the polynomial defining the field.

2.2 Random Function and Pseudorandom Function

Let $\mathrm{Func}(n, m)$ be the set of all functions $\{0,1\}^n \to \{0,1\}^m$. In addition, let $\mathrm{Perm}(n)$ be the set of all permutations over $\{0,1\}^n$. A uniform random function (URF) having n-bit input and m-bit output is uniformly distributed over $\mathrm{Func}(n, m)$. It is denoted by $\mathsf{R} \overset{\$}{\leftarrow} \mathrm{Func}(n, m)$. An n-bit uniform random permutation (URP), denoted by P, is similarly defined as $\mathsf{P} \overset{\$}{\leftarrow} \mathrm{Perm}(n)$.

We also define tweakable URF and URP. Let $\mathcal{T}$ be a set of tweak and $\mathrm{Func}^{\mathcal{T}}(n, m)$ be a set of functions $\mathcal{T} \times \{0,1\}^n \to \{0,1\}^m$. A tweakable URF with tweak $T \in \mathcal{T}$, and n-bit input, m-bit output is written as $\widetilde{\mathsf{R}} \overset{\$}{\leftarrow} \mathrm{Func}^{\mathcal{T}}(n, m)$. Note that if $\mathcal{T} = \{0,1\}^t$, $\mathrm{Func}^{\mathcal{T}}(n, m)$ has the same cardinality as $\mathrm{Func}(n + t, m)$, hence $\widetilde{\mathsf{R}}$ is simply realized with URF of $(n+t)$-bit input. In addition, let $\mathrm{Perm}^{\mathcal{T}}(n)$ be a set of functions $\mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ such that, for any $f \in \mathrm{Perm}^{\mathcal{T}}(n)$ and $t \in \mathcal{T}$, $f(t, *)$ is a permutation. A tweakable n-bit URP with tweak $T \in \mathcal{T}$ is defined as $\widetilde{\mathsf{P}} \overset{\$}{\leftarrow} \mathrm{Perm}^{\mathcal{T}}(n)$. We also define a URF having variable input length (VIL), denoted by $\mathsf{R}^{\infty} : \{0,1\}^* \to \{0,1\}^n$. This can be realized by stateful lazy sampling.
PRF. For c oracles, $O_1, O_2, \ldots, O_c$, we write $\mathcal{A}^{O_1, O_2, \ldots, O_c}$ to represent the adversary $\mathcal{A}$ accessing these c oracles in an arbitrarily order. If O and O' are oracles having the same input and output domains, we say they are compatible. Let $F_K : \{0,1\}^n \to \{0,1\}^m$ and $G_{K'} : \{0,1\}^n \to \{0,1\}^m$ be two compatible keyed functions, with $K \in \mathcal{K}$ and $K' \in \mathcal{K}'$ (key spaces are not necessarily the same). Let $\mathcal{A}$ be an adversary trying distinguish them using chosen-plaintext queries. Then the advantage of $\mathcal{A}$ is defined as

$$\mathtt{Adv}^{\mathtt{cpa}}_{F_K, G_{K'}}(\mathcal{A}) \overset{\text{def}}{=} \Pr[K \overset{\$}{\leftarrow} \mathcal{K} : \mathcal{A}^{F_K} \Rightarrow 1] - \Pr[K' \overset{\$}{\leftarrow} \mathcal{K}' : \mathcal{A}^{G_{K'}} \Rightarrow 1].$$

The above definition can be naturally extended to the case when $G_{K'}$ is a URF, $\mathsf{R} \overset{\$}{\leftarrow} \mathrm{Func}(n, m)$. We have

$$\mathtt{Adv}^{\mathtt{prf}}_{F_K}(\mathcal{A}) \overset{\mathrm{def}}{=} \mathtt{Adv}^{\mathtt{cpa}}_{F_K,\mathsf{R}}(\mathcal{A}).$$

If F_K is a VIL function we define $\mathtt{Adv}^{\mathtt{prf}}_{F_K}(\mathcal{A})$ as $\mathtt{Adv}^{\mathtt{cpa}}_{F_K,\mathsf{R}^\infty}(\mathcal{A})$. Similarly, for tweakable keyed function $\widetilde{F}_K : \mathcal{T} \times \{0,1\}^n \to \{0,1\}^m$ and $\widetilde{\mathsf{R}} \overset{\$}{\leftarrow} \mathrm{Func}^{\mathcal{T}}(n, m)$, we have

$$\mathtt{Adv}^{\mathtt{prf}}_{\widetilde{F}_K}(\mathcal{A}) \overset{\mathrm{def}}{=} \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{F}_K,\widetilde{\mathsf{R}}}(\mathcal{A}).$$

We stress that $\mathcal{A}$ in the above is allowed to choose tweaks, arbitrarily and adaptively. By convention we say F_K is a pseudorandom function (PRF) if $\mathtt{Adv}^{\mathtt{prf}}_{F_K}(\mathcal{A})$ is small (though the formal definition requires F_K to be a function family). Similarly we say F_K is a pseudorandom permutation (PRP) if $\mathtt{Adv}^{\mathtt{prp}}_{F_K}(\mathcal{A}) = \mathtt{Adv}^{\mathtt{cpa}}_{F_K,\mathsf{P}}(\mathcal{A})$ is small and F_K is invertible. A VIL-PRF is defined in a similar way.

2.3 Definition of Authenticated Encryption

Following [16, 45], we define nonce-based AE, or more formally, AE with associated data, called AEAD. We then introduce two security notions, privacy and authenticity, to model AE security.

Definition. Let $\mathsf{AE}[\tau]$ be an AE having τ-bit tag, where the encryption and decryption algorithms are $\mathsf{AE}\text{-}\mathcal{E}_\tau$ and $\mathsf{AE}\text{-}\mathcal{D}_\tau$. They are keyed functions. Besides the key, the input to $\mathsf{AE}\text{-}\mathcal{E}_\tau$ consists of a nonce $N \in \mathcal{N}_{ae}$, an associated data (AD, or a header) $A \in \mathcal{A}_{ae}$, and a plaintext $M \in \mathcal{M}_{ae}$. The output consists of $C \in \mathcal{M}_{ae}$ and $T \in \{0,1\}^\tau$, where $|C| = |M|$. The tuple (N, A, C, T) will be sent to the receiver. The decryption function is denoted by $\mathsf{AE}\text{-}\mathcal{D}_\tau$. It takes $(N, A, C, T) \in \mathcal{N}_{ae} \times \mathcal{A}_{ae} \times \mathcal{M}_{ae} \times \{0,1\}^\tau$, and outputs a plaintext M with $|M| = |C|$ if input is determined as valid, or error symbol $\perp$ if determined as invalid.

Security. A PRIV-adversary $\mathcal{A}$ against $\mathsf{AE}[\tau]$ accesses $\mathsf{AE}\text{-}\mathcal{E}_\tau$, where the i-th query consists of nonce N_i, header A_i, and plaintext M_i. We define $\mathcal{A}$'s parameter list to be (q, σ_A, σ_M), where q denotes the number of queries, and $\sigma_A \overset{\mathrm{def}}{=} \sum_{i=1}^{q} |A_i|_n$ and $\sigma_M \overset{\mathrm{def}}{=} \sum_{i=1}^{q} |M_i|_n$. We assume $\mathcal{A}$ is nonce-respecting, i.e., all N_is are distinct. We also define random-bit oracle, \$, which takes $(N, A, M) \in \mathcal{N}_{ae} \times \mathcal{A}_{ae} \times \mathcal{M}_{ae}$ and returns $(C, T) \overset{\$}{\leftarrow} \{0,1\}^{|M|} \times \{0,1\}^\tau$. The privacy notion for $\mathcal{A}$ is defined as

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathsf{AE}[\tau]}(\mathcal{A}) \overset{\mathrm{def}}{=} \Pr[K \overset{\$}{\leftarrow} \mathcal{K} : \mathcal{A}^{\mathsf{AE}\text{-}\mathcal{E}_\tau} \Rightarrow 1] - \Pr[\mathcal{A}^{\$} \Rightarrow 1]. \tag{1}$$

An AUTH-adversary $\mathcal{A}$ against $\mathsf{AE}[\tau]$ accesses $\mathsf{AE}\text{-}\mathcal{E}_\tau$ and $\mathsf{AE}\text{-}\mathcal{D}_\tau$, using q encryption queries and q_v decryption queries. Let $(N_1, A_1, M_1), \ldots, (N_q, A_q, M_q)$ and $(N'_1, A'_1, C'_1, T'_1), \ldots, (N'_{q_v}, A'_{q_v}, C'_{q_v}, T'_{q_v})$ be all the encryption and decryption queries made by $\mathcal{A}$. We define $\mathcal{A}$'s parameter list to be $(q, q_v, \sigma_A, \sigma_M, \sigma_{A'}, \sigma_{C'})$, where $\sigma_{A'} \overset{\mathrm{def}}{=} \sum_{i=1}^{q_v} |A'_i|_n$ and $\sigma_{C'} \overset{\mathrm{def}}{=} \sum_{i=1}^{q_v} |C'_i|_n$, in addition to σ_A and σ_M. The authenticity notion for the AUTH-adversary $\mathcal{A}$ is defined as

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathsf{AE}[\tau]}(\mathcal{A}) \overset{\mathrm{def}}{=} \Pr[K \overset{\$}{\leftarrow} \mathcal{K} : \mathcal{A}^{\mathsf{AE}\text{-}\mathcal{E}_\tau, \mathsf{AE}\text{-}\mathcal{D}_\tau} \text{ forges }], \tag{2}$$

where $\mathcal{A}$ forges if $\mathsf{AE}\text{-}\mathcal{D}_\tau$ returns a bit string (other than $\perp$) for a decryption query (N'_i, A'_i, C'_i, T'_i) for some $1 \le i \le q_v$ such that $(N'_i, A'_i, C'_i, T'_i) \neq (N_j, A_j, C_j, T_j)$ for all $1 \le j \le q$. We assume AUTH-adversary $\mathcal{A}$ is always nonce-respecting with respect to encryption queries; using the same N for encryption and decryption queries is allowed, and the same N can be repeated within decryption queries, i.e. N_i is different from N_j for any $j \neq i$ but $N'{}_i$ may be equal to N_j or $N'_{i'}$ for some j and $i' \neq i$.

Moreover, when F_K and $G_{K'}$ are compatible with $\mathsf{AE}\text{-}\mathcal{E}_\tau$, let $\mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{F,G}(\mathcal{A})$ be the same function as $\mathtt{Adv}^{\mathtt{cpa}}_{F,G}(\mathcal{A})$ but $\mathcal{A}$ is restricted to be nonce-respecting. Note that $\mathtt{Adv}^{\mathtt{priv}}_{\mathsf{AE}[\tau]}(\mathcal{A}) = \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathsf{AE}\text{-}\mathcal{E}_\tau,\$}(\mathcal{A})$ holds for any nonce-respecting $\mathcal{A}$. In addition when F_K and $G_{K'}$ are the pairs of encryption and decryption functions written as $F_K = (F^e_K, F^d_K)$ and $G_{K'} = (G^e_{K'}, G^d_{K'})$ and they are compatible with $(\mathsf{AE}\text{-}\mathcal{E}_\tau, \mathsf{AE}\text{-}\mathcal{D}_\tau)$, we define

$$\mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{F,G}(\mathcal{A}) \overset{\mathrm{def}}{=} \Pr[K \overset{\$}{\leftarrow} \mathcal{K} : \mathcal{A}^{F^e_K, F^d_K} \Rightarrow 1] - \Pr[K' \overset{\$}{\leftarrow} \mathcal{K}' : \mathcal{A}^{G^e_{K'}, G^d_{K'}} \Rightarrow 1], \tag{3}$$

where $\mathcal{A}$ is assumed to be nonce-respecting for encryption queries. Then we have

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathsf{AE}[\tau]}(\mathcal{A}) \le \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathsf{AE}[\tau],\mathsf{AE}'[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{auth}}_{\mathsf{AE}'[\tau]}(\mathcal{A}) \tag{4}$$

for any AE scheme $\mathsf{AE}'[\tau]$ and any AUTH-adversary $\mathcal{A}$.

3 Specification of OTR

We present an AE scheme based on an $E_K : \{0,1\}^n \to \{0,1\}^n$, which is denoted by OTR$[E, \tau]$, where $\tau \in \{1, \ldots, n\}$ denotes the length of tag. The encryption function and decryption function of OTR$[E, \tau]$ are denoted by OTR-$\mathcal{E}_{E,\tau}$ and OTR-$\mathcal{D}_{E,\tau}$. Here OTR-$\mathcal{E}_{E,\tau}$ (OTR-$\mathcal{D}_{E,\tau}$) has the same interface as AE-$\mathcal{E}_\tau$ (AE-$\mathcal{D}_\tau$) of Section 2.3, with nonce space $\mathcal{N}_{ae} = \{0,1\}^{\le n-1}$ (which excludes ε by definition), header space $\mathcal{A}_{ae} = \{0,1\}^*$, message space $\mathcal{M}_{ae} = \{0,1\}^*$, and tag space $\{0,1\}^\tau$. The functions OTR-$\mathcal{E}_{E,\tau}$ and OTR-$\mathcal{D}_{E,\tau}$ are further decomposed into the encryption and decryption cores, EF_E, DF_E, and the authentication core, AF_E. Figs. 1 and 2 show the scheme. As shown by Fig. 2, OTR consists of two-round Feistel permutations using a blockcipher taking a distinct input mask in each round. To authenticate the plaintext a check sum is computed for the right part of two-round Feistel (namely the even plaintext blocks), and the tag is derived from encrypting the check sum with an input mask. The overall structure has a similarity to OCB, and the function AF_E is a variant of PMAC [44]. We also show Fig. 6 as an alternative representation of Fig. 1 to make explicit the used constants over GF(2^n) for masking.

4 Security Bounds

We provide the security bounds of OTR. Here we assume the underlying blockcipher is an n-bit URP, P. The bounds when the underlying blockcipher is a PRP are easily derived from our bounds, using a standard technique, thus omitted.

Theorem 1. *Fix $\tau \in \{1, \ldots, n\}$. For any PRIV-adversary $\mathcal{A}$ with parameter (q, σ_A, σ_M),*

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathrm{OTR}[\mathsf{P},\tau]}(\mathcal{A}) \le \frac{6\sigma^2_{\mathtt{priv}}}{2^n}$$

holds for $\sigma_{\mathtt{priv}} = q + \sigma_A + \sigma_M$.

Theorem 2. *Fix $\tau \in \{1, \ldots, n\}$. For any AUTH-adversary $\mathcal{A}$ with parameter $(q, q_v, \sigma_A, \sigma_M, \sigma_{A'}, \sigma_{C'})$,*

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathrm{OTR}[\mathsf{P},\tau]}(\mathcal{A}) \le \frac{6\sigma^2_{\mathtt{auth}}}{2^n} + \frac{q_v}{2^\tau}$$

holds for $\sigma_{\mathtt{auth}} = q + q_v + \sigma_A + \sigma_M + \sigma_{A'} + \sigma_{C'}$.

5 Proofs of Theorems 1 and 2

Overview. The proofs of Theorems 1 and 2 consist of two steps, where in the first step we interpret OTR as a mode of TBC and in the second step we prove the indistinguishability between the tweakable URF and the TBC used in OTR. This structure is essentially the same as original OCB proofs, say by Rogaway [44], as well as many other schemes based on TBC.
First step: TBC-based design. In the first step, we define an AEAD scheme denoted by $\mathbb{OTR}[\tau]$, which we may abbreviate $\mathbb{OTR}$ if τ is obvious. It is compatible with OTR$[E, \tau]$ and uses a tweakable n-bit URF, $\widetilde{\mathsf{R}} : \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$. Here, tweak $T \in \mathcal{T}$ is written as $T = (x, i, \omega) \in \mathcal{N}'_{ae} \times \mathbb{N} \times \Omega$, where $\mathcal{N}'_{ae} = \mathcal{N}_{ae} \cup \{0^n\}$ and $\Omega \overset{\text{def}}{=} \{\mathtt{f}, \mathtt{s}, \mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2, \mathtt{h}, \mathtt{g}_1, \mathtt{g}_2\}$. The encryption and decryption functions of $\mathbb{OTR}[\tau]$ are $\mathbb{OTR}$-$\mathcal{E}_\tau$ and $\mathbb{OTR}$-$\mathcal{D}_\tau$, and they consist of encryption core $\mathbb{EF}_{\widetilde{\mathsf{R}}}$, decryption core $\mathbb{DF}_{\widetilde{\mathsf{R}}}$, and authentication core $\mathbb{AF}_{\widetilde{\mathsf{R}}}$, as shown by Fig. 4. For reference $\mathbb{OTR}$ is also illustrated in Fig. 5. They can be seen as counterparts of EF_E, DF_E, and AF_E of OTR$[E, \tau]$. Fig. 4 also defines $\mathbb{OTR}'[\tau]$, which uses an independent VIL-URF, $\mathsf{R}^\infty : \{0,1\}^* \to \{0,1\}^n$, instead of $\mathbb{AF}_{\widetilde{\mathsf{R}}}$. We first derive the security bounds of $\mathbb{OTR}'[\tau]$ in the following theorem. The proof of Theorem 3 is given in Appendix A.

Theorem 3. *Fix $\tau \in \{1, \ldots, n\}$. For any PRIV-adversary $\mathcal{A}$,*

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathbb{OTR}'[\tau]}(\mathcal{A}) = 0.$$

Moreover, for any AUTH-adversary $\mathcal{A}$ using q encryption queries and q_v decryption queries,

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'[\tau]}(\mathcal{A}) \le \frac{2q_v}{2^n} + \frac{q_v}{2^\tau}.$$

Algorithm OTR-$\mathcal{E}_{E,\tau}(N, A, M)$

```
1. $(C, TE) \leftarrow \mathrm{EF}_E(N, M)$
2. **if** $A \neq \varepsilon$ **then** $TA \leftarrow \mathrm{AF}_E(A)$
3. **else** $TA \leftarrow 0^n$
4. $T \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
5. **return** $(C, T)$
```

Algorithm OTR-$\mathcal{D}_{E,\tau}(N, A, C, T)$

```
1. $(M, TE) \leftarrow \mathrm{DF}_E(N, C)$
2. **if** $A \neq \varepsilon$ **then** $TA \leftarrow \mathrm{AF}_E(A)$
3. **else** $TA \leftarrow 0^n$
4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
5. **if** $\widehat{T} = T$ **return** $M$
6. **else return** $\perp$
```

Algorithm $\mathrm{EF}_E(N, M)$

```
1. $\Sigma \leftarrow 0^n$
2. $U \leftarrow E(\underline{N})$, $L \leftarrow U$, $L^\sharp \leftarrow 3U$
3. $(M[1], \ldots, M[m]) \xleftarrow{n} M$
4. **for** $i = 1$ **to** $\lceil m/2 \rceil - 1$ **do**
5.     $C[2i-1] \leftarrow E(L \oplus M[2i-1]) \oplus M[2i]$
6.     $C[2i] \leftarrow E(L^\sharp \oplus C[2i-1]) \oplus M[2i-1]$
7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
8.     $L \leftarrow L \oplus L^\sharp$, $L^\sharp \leftarrow 2L^\sharp$          // $L = 2^i U$, $L^\sharp = 2^i 3U$
9. **if** $m$ **is even**
10.    $Z \leftarrow E(L \oplus M[m-1])$
11.    $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
12.    $C[m-1] \leftarrow E(L^\sharp \oplus \underline{C[m]}) \oplus M[m-1]$
13.    $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
14.    $L^* \leftarrow L^\sharp$
15. **if** $m$ **is odd**
16.    $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(E(L)) \oplus M[m]$
17.    $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
18.    $L^* \leftarrow L$
19. **if** $|M[m]| \neq n$ **then** $TE \leftarrow E(3^2 L^* \oplus \Sigma)$
20. **else** $TE \leftarrow E(7L^* \oplus \Sigma)$
21. $C \leftarrow (C[1], \ldots, C[m])$
22. **return** $(C, TE)$
```

Algorithm $\mathrm{DF}_E(N, C)$

```
1. $\Sigma \leftarrow 0^n$
2. $U \leftarrow E(\underline{N})$, $L \leftarrow U$, $L^\sharp \leftarrow 3U$
3. $(C[1], \ldots, C[m]) \xleftarrow{n} C$
4. **for** $i = 1$ **to** $\lceil m/2 \rceil - 1$ **do**
5.     $M[2i-1] \leftarrow E(L^\sharp \oplus C[2i-1]) \oplus C[2i]$
6.     $M[2i] \leftarrow E(L \oplus M[2i-1]) \oplus C[2i-1]$
7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
8.     $L \leftarrow L \oplus L^\sharp$, $L^\sharp \leftarrow 2L^\sharp$          // $L = 2^i U$, $L^\sharp = 2^i 3U$
9. **if** $m$ **is even**
10.    $M[m-1] \leftarrow E(L^\sharp \oplus \underline{C[m]}) \oplus C[m-1]$
11.    $Z \leftarrow E(L \oplus M[m-1])$
12.    $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
13.    $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
14.    $L^* \leftarrow L^\sharp$
15. **if** $m$ **is odd**
16.    $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(E(L)) \oplus C[m]$
17.    $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
18.    $L^* \leftarrow L$
19. **if** $|C[m]| \neq n$ **then** $TE \leftarrow E(3^2 L^* \oplus \Sigma)$
20. **else** $TE \leftarrow E(7L^* \oplus \Sigma)$
21. $M \leftarrow (M[1], \ldots, M[m])$
22. **return** $(M, TE)$
```

Algorithm $\mathrm{AF}_E(A)$

```
1. $\Xi \leftarrow 0^n$
2. $Q \leftarrow E(0^n)$
3. $(A[1], \ldots, A[a]) \xleftarrow{n} A$
4. **for** $i = 1$ **to** $a - 1$ **do**
5.     $\Xi \leftarrow \Xi \oplus E(Q \oplus A[i])$
6.     $Q \leftarrow 2Q$
7. $\Xi \leftarrow \Xi \oplus \underline{A[a]}$
8. **if** $|A[a]| \neq n$ **then** $TA \leftarrow E(3Q \oplus \Xi)$
9. **else** $TA \leftarrow E(3^2 Q \oplus \Xi)$
10. **return** $TA$
```

Fig. 1. Algorithms of OTR. Tag bit size is $0 < \tau \leq n$, and $\underline{X}$ denotes the 10^* padding of X, see Section 2.1.

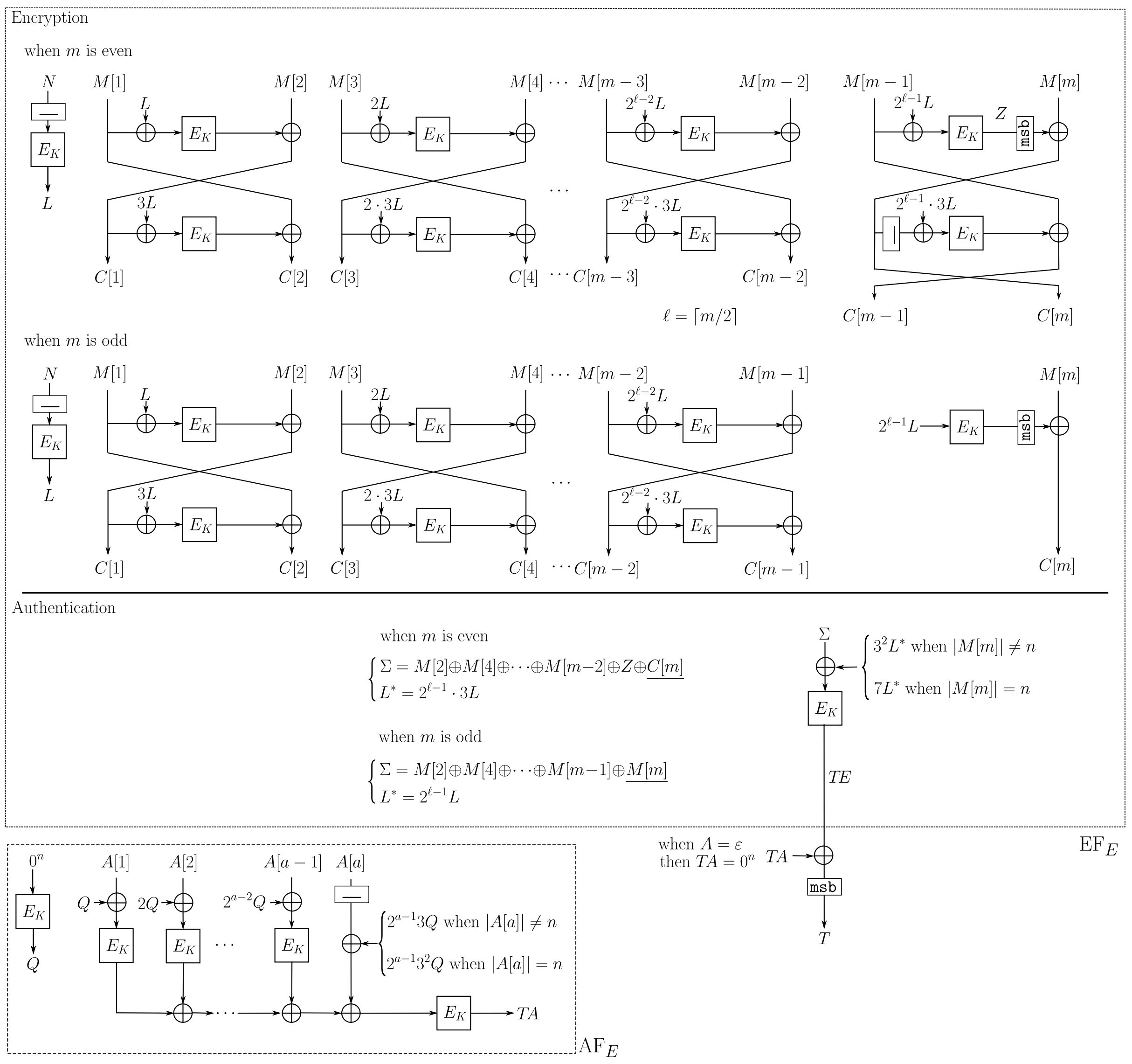

Encryption
when m is even
N
E_K
L
$M[1]$ $M[2]$ $M[3]$ $M[4]$ $\cdots$ $M[m-3]$ $M[m-2]$ $M[m-1]$ $M[m]$
L $2L$ $2^{\ell-2}L$ $2^{\ell-1}L$
$3L$ $2 \cdot 3L$ $2^{\ell-2} \cdot 3L$ $2^{\ell-1} \cdot 3L$
Z
msb
$C[1]$ $C[2]$ $C[3]$ $C[4]$ $\cdots C[m-3]$ $C[m-2]$
$\ell = \lceil m/2 \rceil$
$C[m-1]$ $C[m]$
when m is odd
$M[1]$ $M[2]$ $M[3]$ $M[4]$ $\cdots$ $M[m-2]$ $M[m-1]$ $M[m]$
$2^{\ell-1}L$
$C[1]$ $C[2]$ $C[3]$ $C[4]$ $\cdots C[m-2]$ $C[m-1]$ $C[m]$
Authentication
when m is even
$\Sigma = M[2] \oplus M[4] \oplus \cdots \oplus M[m-2] \oplus Z \oplus \underline{C[m]}$
$L^* = 2^{\ell-1} \cdot 3L$
when m is odd
$\Sigma = M[2] \oplus M[4] \oplus \cdots \oplus M[m-1] \oplus \underline{M[m]}$
$L^* = 2^{\ell-1}L$
Σ
3^2L^* when $|M[m]| \neq n$
$7L^*$ when $|M[m]| = n$
TE
when $A = \varepsilon$ then $TA = 0^n$
TA
T
EF_E
0^n $A[1]$ $A[2]$ $A[a-1]$ $A[a]$
Q $2Q$ $2^{a-2}Q$
$2^{a-1}3Q$ when $|A[a]| \neq n$
$2^{a-1}3^2Q$ when $|A[a]| = n$
TA
AF_E

Fig. 2. Encryption of OTR. Box with underline and $\underline{X}$ denote the 10^* padding of input X.

Proof intuition of Theorem 3. To understand Theorem 3, there are two important properties of a two-round Feistel permutation, denoted by $\phi_{f_1,f_2} : \{0,1\}^{2n} \to \{0,1\}^{2n}$. Here $\phi_{f_1,f_2}(X[1], X[2]) = (Y[1], Y[2])$ where $Y[1] = f_1(X[1]) \oplus X[2]$ and $Y[2] = f_2(Y[1]) \oplus X[1]$ and f_1 and f_2 are independent n-bit URFs. Then we have the followings.

Property 1. For any $(X[1], X[2]) \in \{0,1\}^{2n}$, $\phi_{f_1,f_2}(X[1], X[2])$ is uniformly random.
Property 2. Let $(Y[1], Y[2]) = \phi_{f_1,f_2}(X[1], X[2])$, and let $(Y'[1], Y'[2])$ be a function of $(X[1], X[2], Y[1], Y[2])$ satisfying $(Y'[1], Y'[2]) \neq (Y[1], Y[2])$. Then $X'[2]$, where $(X'[1], X'[2]) = \phi^{-1}_{f_1,f_2}(Y'[1], Y'[2])$, is uniform unless the event $\mathtt{Bad}_1 : X[1] = X'[1]$ occurs, which has the probability at most $1/2^n$.

Property 1 is simple because f_1 and f_2 are independent and the output of ϕ consists of those of f_1 and f_2. Property 2 needs some cares. It holds because if $X[1] \neq X'[1] = f_2(Y'[1]) \oplus Y'[2]$, $f_1(X'[1])$ is distributed uniformly random, independent of all other variables, and this makes $X'[2] = f_1(X'[1]) \oplus Y'[1]$ completely random. The $\mathtt{Bad}_1$ event has probability $1/2^n$ when $Y'[1] \neq Y'[1]$, and otherwise 0. Note that $(X[1], X[2], Y[1], Y[2])$ reveals corresponding I/O pairs of f_1 and f_2, however this does not help gain the probability of $\mathtt{Bad}_1$.

Intuitively, the privacy bound of Theorem 3 is simply obtained by the fact that all TBC calls in the game has distinct tweaks and all output blocks contain at least one TBC output with unique tweak. Combined with

Property 1, this makes all output blocks perfectly random, hence the privacy bound is 0. For the authenticity bound, suppose adversary $\mathcal{A}$ performs an encryption query (N, A, M) and obtains (C, T), and then performs a decryption query (N', A', C', T') for some $C \neq C'$ with $|C| = |C'|$, with $(N', A') = (N, A)$. This implies that there exists at least one chunk ($2n$-bit block) of C' different from the corresponding chunk in C, and from Property 2, the right half of the corresponding decrypted plaintext chunk is completely random, unless $\mathtt{Bad}_1$ occurs. There is another chance for the adversary to win, i.e. the checksum collision $\mathtt{Bad}_2 : \Sigma' = \Sigma$, which has probability $1/2^n$ provided $\mathtt{Bad}_1$ did not happen. Hence we have $\Pr[\mathtt{Bad}_1 \cup \mathtt{Bad}_2] \leq \Pr[\mathtt{Bad}_1] + \Pr[\mathtt{Bad}_2 | \overline{\mathtt{Bad}_1}] \leq 2/2^n$. When both events did not happen (i.e. given $\overline{\mathtt{Bad}_1 \cup \mathtt{Bad}_2}$), the final chance is to successfully guess the tag, where the probability is clearly bounded by $1/2^\tau$ because different checksums yield independent tags. Hence the authenticity bound is $2/2^n + 1/2^\tau$ for any $\mathcal{A}$ using $q_v = 1$ decryption query (of course we need to consider the existence of other encryption queries and many other cases for (N', A', C', T') as well, however the above bound holds for all cases). Finally we use a well-known result of Bellare, Goldreich and Mityagin [13] to obtain $2q_v/2^n + q_v/2^\tau$ for any $q_v \geq 1$.

```
Algorithm $\widetilde{G}[\mathsf{P}]^{\langle N,i,\omega\rangle}(X)$

 1. Preprocessing: $Q \leftarrow \mathsf{P}(0^n)$
 2. if $N \neq 0^n$ then $L \leftarrow \mathsf{P}(\underline{N})$
 3.    switch $\omega$
 4.    Case f : $\Delta \leftarrow 2^{i-1}L$
 5.    Case s : $\Delta \leftarrow 2^{i-1}3L$
 6.    Case a$_1$ : $\Delta \leftarrow 2^{i-1}3^3L$
 7.    Case a$_2$ : $\Delta \leftarrow 2^{i-1}3^17L$
 8.    Case b$_1$ : $\Delta \leftarrow 2^{i-1}3^2L$
 9.    Case b$_2$ : $\Delta \leftarrow 2^{i-1}7L$
10. else switch $\omega$
11.    Case h : $\Delta \leftarrow 2^{i-1}Q$
12.    Case g$_1$ : $\Delta \leftarrow 2^{i-1}3Q$
13.    Case g$_2$ : $\Delta \leftarrow 2^{i-1}3^2Q$
14. $Y \leftarrow \mathsf{P}(\Delta \oplus X)$
15. return $Y$
```

Fig. 3. Tweakable permutation of OTR$[\mathsf{P}, \tau]$, denoted by $\widetilde{G}[\mathsf{P}]$.

Second step: analysis of TBC. In Fig. 3 we define a TBC, $\widetilde{G}[\mathsf{P}]^{\langle N,i,\omega\rangle}(X)$, where (N, i, ω) is a tweak. It uses an n-bit URP, P. We remark that $\widetilde{G}[\mathsf{P}]$ slightly abuse N as it allows $N = 0^n$, making N to be an element of $\mathcal{N}'_{ae}$. For tweaks that do not appear in Fig. 3, we let them as undefined. We observe that $\widetilde{G}[\mathsf{P}]$ is compatible with $\widetilde{\mathsf{R}}$, the tweakable URF used by (components of) $\mathbb{OTR}[\tau]$ and $\mathbb{OTR}'[\tau]$, and in addition $\widetilde{G}[\mathsf{P}]$ is implicitly used by OTR$[\mathsf{P}, \tau]$, where the usage is the same as the way $\mathbb{OTR}[\tau]$ uses $\widetilde{\mathsf{R}}$. More formally, we have the following proposition.

Proposition 1. *If* $\mathbb{EF}_{\widetilde{\mathsf{R}}}$ *and* $\mathbb{DF}_{\widetilde{\mathsf{R}}}$ *use* $\widetilde{G}[\mathsf{P}]$ *instead of* $\widetilde{\mathsf{R}}$*, we obtain* EF_{P} *and* DF_{P}*. Similarly if* $\mathbb{AF}_{\widetilde{\mathsf{R}}}$ *uses* $\widetilde{G}[\mathsf{P}]$ *instead of* $\widetilde{\mathsf{R}}$ *we obtain* AF_{P}*.*

Note that $\widetilde{G}[\mathsf{P}]$ does not perform GF doublings in a sequential manner, instead a full multiplications for every input. This is inefficient in practice, however does not cause a problem for simulation purpose. We then prove that $\widetilde{G}[\mathsf{P}]$ is a secure tweakable URF, shown by the following lemma.

Lemma 1. *For any adversary* $\mathcal{A}$ *accessing* $\widetilde{G}[\mathsf{P}]$ *with* q *queries, we have* $\mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{A}) \leq 5q^2/2^n$*.*

We also provide the indistinguishability bound between $\mathbb{AF}_{\widetilde{\mathsf{R}}}$ and R^∞, which is as follows.

Lemma 2. *For any* $\mathcal{A}$ *with* σ *input blocks, we have* $\mathtt{Adv}^{\mathtt{prf}}_{\mathbb{AF}_{\widetilde{\mathsf{R}}}}(\mathcal{A}) \leq \sigma^2/2^{n+1}$*.*

The proof of Lemma 1 is given in Appendix B. The proof of Lemma 2 is the same as a part of PMAC proof, more specifically the last equation of Appendix E of [46].

Third step: deriving bounds. For privacy notion, there exist adversaries $\mathcal{B}$ against $\mathbb{AF}_{\widetilde{\mathsf{R}}}$ with σ_A input blocks, and $\mathcal{C}$ against $\widetilde{G}[\mathsf{P}]$ with $\sigma_{\mathtt{priv}}$ queries, satisfying

$$\begin{aligned}
\mathtt{Adv}^{\mathtt{priv}}_{\mathrm{OTR}[\mathsf{P},\tau]}(\mathcal{A}) &\le \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathrm{OTR}[\mathsf{P},\tau],\mathbb{OTR}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathbb{OTR}[\tau],\mathbb{OTR}'[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathbb{OTR}'[\tau],\$}(\mathcal{A}) && (5)\\
&\le \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathrm{OTR}[\mathsf{P},\tau],\mathbb{OTR}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{AF}_{\widetilde{\mathsf{R}}},\mathsf{R}^{\infty}}(\mathcal{B}) + \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathbb{OTR}'[\tau],\$}(\mathcal{A}) && (6)\\
&\le \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{C}) + \frac{\sigma_A^2}{2^{n+1}} && (7)\\
&\le \frac{5\sigma_{\mathtt{priv}}^2}{2^n} + \frac{\sigma_A^2}{2^{n+1}} && (8)\\
&\le \frac{6\sigma_{\mathtt{priv}}^2}{2^n}. && (9)
\end{aligned}$$

where the third inequality follows from Proposition 1, Lemma 2, and Theorem 3, and the fourth inequality follows from Lemma 1. Similarly, for authenticity notion, there exist $\mathcal{B}$ against $\mathbb{AF}_{\widetilde{\mathsf{R}}}$ with $\sigma_A + \sigma_{A'}$ input blocks, and $\mathcal{C}$ against $\widetilde{G}[\mathsf{P}]$ with $\sigma_{\mathtt{auth}}$ queries, satisfying

$$\begin{aligned}
\mathtt{Adv}^{\mathtt{auth}}_{\mathrm{OTR}[\mathsf{P},\tau]}(\mathcal{A}) &\le \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathrm{OTR}[\mathsf{P},\tau],\mathbb{OTR}'[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'[\tau]}(\mathcal{A}) && (10)\\
&\le \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathrm{OTR}[\mathsf{P},\tau],\mathbb{OTR}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathbb{OTR}[\tau],\mathbb{OTR}'[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'[\tau]}(\mathcal{A}) && (11)\\
&\le \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathrm{OTR}[\mathsf{P},\tau],\mathbb{OTR}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{AF}_{\widetilde{\mathsf{R}}},\mathsf{R}^{\infty}}(\mathcal{B}) + \mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'[\tau]}(\mathcal{A}) && (12)\\
&\le \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{C}) + \frac{(\sigma_A + \sigma_{A'})^2}{2^{n+1}} + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (13)\\
&\le \frac{5\sigma_{\mathtt{auth}}^2}{2^n} + \frac{(\sigma_A + \sigma_{A'})^2}{2^{n+1}} + \frac{2\sigma_{A'}}{2^n} + \frac{q_v}{2^\tau} && (14)\\
&\le \frac{6\sigma_{\mathtt{auth}}^2}{2^n} + \frac{q_v}{2^\tau}, && (15)
\end{aligned}$$

where the fourth inequality follows from Proposition 1, Lemma 2, and Theorem 3, and the fifth inequality follows from Lemma 1. This concludes the proof.

6 Experimental Results on Software

We implemented OTR on software. The purpose of this implementation is not to provide a fast code, but to see the effect of inverse-freeness in an experimental environment. We wrote a reference-like AES C code that takes byte arrays and uses 4Kbyte tables for combined S-box and Mixcolumn lookup, so-called T-tables. AES decryption of our code is slightly slower than encryption (see Table 2). We then wrote pure C code of OTR using the above AES code. All components, e.g. XOR of blocks and GF doubling, are byte-wise codes. For comparison we also wrote a C code of OCB2 [44] in the same manner, which is similar to a reference code by Krovetz [2].

We ran both codes on an x86 PC (Core i7 3770, Ivy bridge, 3.4GHz) with 64-bit Windows 7. We used Visual C++ 2012 (VC12) to obtain 32-bit and 64-bit executables and used GCC 4.7.1 for 32-bit executables, with option `-O2`. We measured speed for 4Kbyte messages and one-block header. We also tested the same code on an ARM board (Cortex-A8 1GHz) using GCC 4.7.3 with `-O2` option. Their speed figures in cycles per byte[1] are shown in the upper part of Table 2. For both OTR and OCB2, we can observe a noticeable slowdown from raw AES, however, OTR still receives the benefit of faster AES encryption. Another metric is the size, which is shown in the lower part of Table 2. For OTR we can remove the inverse T-tables and inverse S-box from AES code, as they are not needed for AES encryption, resulting in smaller AES objects.

We also measured the performance of these codes when AES is implemented using AESNI (on the Core i7 machine, using VC12). We simply substituted T-table AES with single-block AES routine using AESNI. In addition, two common functions to OCB2 and OTR, namely XOR of two 16-byte blocks and GF doubling, are substituted with SIMD intrinsic codes. Other byte-wise functions are unchanged. On our machine single-block AES ran at around 4.5 to 5.5 cycles per byte, for both encryption and decryption. Table 3 shows the results. It looks interesting, in that, although we did not write a parallel AESNI routine, we could observe the obvious

[1] As we were unable to use cycle counter in the ARM device, the measurement of ARM was based on a timer.

Table 2. Reference implementation results of OTR and OCB2. (Upper) Speed in cycles per byte. (Lower) Object size in Kbyte.

	x86			ARM
Algorithm	VC12(32-bit)	VC12(64-bit)	gcc 4.7.1(32-bit)	gcc 4.7.3
OTR Enc	27.59	18.94	22.02	69.88
OTR Dec	27.56	18.99	22.2	69.78
OCB2 Enc	27.38	19.93	22.69	71.22
OCB2 Dec	30.86	25.43	34.29	76.16
AES Enc	18.29	12.98	15.9	54.38
AES Dec	22.28	18.36	26.64	58.14

	x86			ARM
Object	VC12(32-bit)	VC12(64-bit)	gcc 4.7.1(32-bit)	gcc 4.7.3
OTR.o	19.9	21.3	5.4	5.9
OCB2.o	20.5	21.7	4.6	5.3
AES_Enc.o	20.2	20.7	6.7	7.1
AES_EncDec.o	45.4	46.2	17.3	17.9
OTR Total	40.1	42.0	12.1	13.0
OCB2 Total	65.9	67.9	21.9	23.2

effect of AESNI parallelism via compiler. Notably, both OTR and OCB2 achieved about 2 cycles per byte for 4K data, and OCB2 is slightly faster as expected. We think further optimization of OTR would be possible by using parallel AES routine with full utilization of SIMD instructions and a careful register handling in a similar manner to OCB, e.g. see a recent report by Bogdanov et al. [20].

These experiments, though quite naive, imply OTR's good performance under multiple platforms with a simple code. Of course, optimized implementations for various platforms are interesting future topics.

Table 3. Performance of codes with single-block AES routine using AES-NI. Data `x` denotes the plaintext length in bytes, and `a`/`b` denotes `a` (`b`) cycles per byte in 32-bit (64-bit) VC12 compilation.

Data (byte)	128	512	1024	2048	4096
OTR Enc	6.01/5.43	3.32/3.16	2.85/2.74	2.66/2.51	2.49/2.40
OTR Dec	7.22/5.60	3.81/3.15	3.06/2.72	2.79/2.51	2.59/2.39
OCB2 Enc	6.39/5.60	3.26/2.76	2.81/2.26	2.53/2.02	2.37/1.90
OCB2 Dec	6.36/5.86	3.04/2.80	2.59/2.26	2.28/2.03	2.11/1.91

7 Remarks

7.1 Remove Inverse from OCB

The abstract structure of OTR has a similarity to OCB, however, removing inverse is not a trivial task. Roughly, in OCB, each plaintext block is given to the ECB mode of an n-bit TBC $\widetilde{E}_K$ [34], namely $C[i] = \widetilde{E}_K^{\langle T \rangle}(M[i])$, where tweak T consists of nonce N and other parameters, based on a blockcipher E_K. The OCB decryption uses the inversion of TBC, $\widetilde{E}_K^{-1}$, and the security proof requires that $\widetilde{E}_K$ is a tweakable SPRP, i.e. $(\widetilde{E}_K, \widetilde{E}_K^{-1})$ and $(\widetilde{\mathsf{P}}, \widetilde{\mathsf{P}}^{-1})$ are hard to distinguish when $\widetilde{\mathsf{P}} \overset{\$}{\leftarrow} \mathrm{Perm}^{\mathcal{T}}(n)$. Since $\widetilde{E}_K^{-1}$ needs a computation of E_K^{-1}, a natural way to remove E_K^{-1} from OCB is to compose $\widetilde{E}_K$ from a PRP or a PRF. For example we can do this by using a $2n$-bit 4-round Feistel cipher as $\widetilde{E}_K$, based on an n-bit PRF, F_K. Then, the resulting mode (of F_K) is inverse-free and provably secure, since 4-round Feistel cipher is an SPRP, as shown by Luby and Rackoff [35] (it is easy to turn a SPRP into a tweakable SPRP). However, we then need four F_K calls per two blocks, i.e. the rate is degraded to two. Considering this, the two-round Feistel is seemingly a bad choice, since it even fails to provide a (tweakable) PRP. As explained in Section 5, the crucial observation is that, the encryption of two-round Feistel in OTR is invoked only once for each tweak, and that the authenticity needs only an n-bit unpredictable value in the decryption, rather than $2n$ bits. Two-round Feistel fulfills these requirements, which makes OTR provably secure.

7.2 Design Rationale for Masking

We remark that using the same mask for the two round functions, i.e. using $2^i L$ for the first and second rounds of a two-round Feistel, does not work. This is because Property 2 of Section 5 does not hold anymore since the two-round Feistel becomes an involution. Once you query $(X[1], X[2])$ and receive $(Y[1], Y[2]) = \phi_{f_1,f_2}(X[1], X[2])$, you know $X'[2] = Y[2]$ always holds (where $(X'[1], X'[2]) = \phi^{-1}_{f_1,f_2}(Y'[1], Y'[2])$), when $(Y'[1], Y'[2]) = (X[1], X[2])$. This implies that the adversary can control the checksum value in the decryption, hence breaks authenticity.

We also remark that the masks for EF_E depend on N, hence do not allow precomputation. In contrast the latest OCB3 allows mask precomputation by using $E_K(0^n)$ [33]. The reason is that we want our scheme not to generate $E_K(0^n)$ for header-less usage (i.e. when A is always empty). As a result our scheme has a rather similar structure as OCB2 and an AEAD mode based on OCB2, called AEM [44]. Recent studies reported that the doubling is not too slow [10], hence we employ on-the-fly doubling as a practical masking option.

7.3 Comparison with Other Inverse-free Modes

Section 6 only considers a comparison with OCB. Here we provide a basic comparison with other modes, in particular those not using the blockcipher inverse. Table 1 shows examples of such inverse-free modes. Among them, CCM, GCM, and EAX are rate-2, assuming the speed of field multiplication in GCM is comparable with blockcipher encryption. At least in theory, OTR is faster for sufficiently long messages for its rate-1 computation. For CCFB, the rate c is a variable satisfying $1 < c$ and $c \approx 1$ is impractical for weak security guarantee[2]. For memory consumption, all inverse-free modes including OTR have a similar profile, as long as the blockcipher encryption is the dominant factor. An exception is GCM since field multiplication usually needs large memories. At the same time, a potential disadvantage of OTR is the complexity introduced by the two-round Feistel, such as a limited on-line/parallel capability, and a slight complex design compared with simple designs reusing existing modes like CTR, CFB, and CMAC.

7.4 Other Instantiations

As the core idea of our proposal is general, it allows various instantiations, by seeing $\mathbb{OTR}$ or $\mathbb{OTR}'$ as a prototype. What we need is just to instantiate $\widetilde{\mathsf{R}}$ accepting n-bit input and tweak (N, i, ω), and producing n-bit output. While we employ GF doubling, one can use a different masking scheme, such as Gray code [33,47], or word-oriented LFSR [22,33,50], or bit-rotation of a special prime length [39]. Moreover, we can use non-invertible cryptographic primitives, typically a Hash-based PRF such as HMAC, or a permutation of Keccak [17] with Even-Mansour conversion [24] for implementing a component of OTR. In the latter case the resulting scheme does not need an inversion of the permutation, which is different from the permutation-based OCB described at [40], and there is no output loss like "capacity" bits of SpongeWrap [18]. In these settings, it is possible that the underlying primitive accepts longer input than output. Then a simple tweaking method by tweak prepending can be an option. For example we take SipHash [12], which is a VIL-PRF with 64-bit output. A SipHash-based scheme would be obtained by replacing $\widetilde{\mathsf{R}}^{\langle N,i,\omega\rangle}(X)$ of $\mathbb{OTR}'$ (Fig. 4) with $\mathrm{SipHash}_K(N\|i\|\omega\|X)$, and replacing $\mathsf{R}^{\infty}(X)$ with $\mathrm{SipHash}_K(0^n\|0\|\mathtt{h}\|X)$, accompanied with an appropriate input encoding. As SipHash has an iterative structure, a caching of an internal value allows efficient computation of $\mathrm{SipHash}_K(N\|i\|\omega\|X)$ from $\mathrm{SipHash}_K(N\|i'\|\omega'\|X')$. We remark that this scheme has roughly 64-bit security. The proof is trivial from Theorem 3, combined with the assumption that SipHash is a VIL-PRF.

8 Conclusion

This paper has presented an authenticated encryption scheme using a PRF. This scheme enables rate-1, on-line, and parallel processing for both encryption and decryption. The core idea of our proposal is to use two-round

[2] More formally, the security bound is roughly $\sigma^2/2^{n/c}$ for privacy and $(\sigma^2/2^{n/c} + 1/2^{n(1-(1/c))})$ for authenticity, with single decryption query and σ total blocks.

Feistel permutation with input masking, combined with a message check sum. As a concrete instantiation we provide a blockcipher mode, called OTR, entirely based on a blockcipher encryption function, which may be seen as an “inverse-free” version of OCB. Our proposal has a higher complexity than OCB outside the blockcipher, hence it will not outperform OCB when the blockcipher enc/dec functions are natively supported and equally fast (say CPU with AESNI), despite the relaxed security assumption. Still, our proposal would be useful for various other environments where the use of blockcipher inverse imposes a non-negligible cost, or when the available crypto function is simply not invertible.

Acknowledgments. The author would like to thank anonymous reviewers at Eurocrypt 2014 for careful reading and invaluable suggestions, which greatly improved the presentation of the paper. The author also would like to thank Tetsu Iwata for fruitful discussions, and Sumio Morioka and Tomoyasu Suzaki for useful comments on implementation aspects, and Martin M. Lauridsen for feedback on the initial ePrint report, and Raphael Bost and Olivier Sanders for their report and the subseqent paper on the tweak of initial specification.

References

1. CAESAR (Competition for Authenticated Encryption: Security, Applicability, and Robustness), `http://competitions.cr.yp.to/index.html/`
2. Reference C code of OCB2, `http://www.cs.ucdavis.edu/~rogaway/ocb/code-2.0.htm/`
3. Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality . NIST Special Publication 800-38C (2004), national Institute of Standards and Technology.
4. Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. NIST Special Publication 800-38B (2005), national Institute of Standards and Technology.
5. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D (2007), national Institute of Standards and Technology.
6. Information Technology - Security techniques - Authenticated encryption, ISO/IEC 19772:2009. International Standard ISO/IEC 19772 (2009)
7. Anderson, E., Beaver, C.L., Draelos, T., Schroeppel, R., Torgerson, M.: ManTiCore: Encryption with Joint Cipher-State Authentication. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP. Lecture Notes in Computer Science, vol. 3108, pp. 440–453. Springer (2004)
8. Anderson, E., Beaver, C.L., Draelos, T., Schroeppel, R., Torgerson, M.: Manticore and CS mode: parallelizable encryption with joint Cipher-State authentication (2014)
9. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and Authenticated Online Ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT (1). Lecture Notes in Computer Science, vol. 8269, pp. 424–443. Springer (2013)
10. Aoki, K., Iwata, T., Yasuda, K.: How Fast Can a Two-Pass Mode Go? A Parallel Deterministic Authenticated Encryption Mode for AES-NI. DIAC 2012: Directions in Authenticated Ciphers (2012), available from `http://hyperelliptic.org/DIAC/`
11. Aoki, K., Yasuda, K.: The Security of the OCB Mode of Operation without the SPRP Assumption. In: Susilo and Reyhanitabar [49], pp. 202–220
12. Aumasson, J.P., Bernstein, D.J.: SipHash: A Fast Short-Input PRF. In: Galbraith, S.D., Nandi, M. (eds.) INDOCRYPT. Lecture Notes in Computer Science, vol. 7668, pp. 489–508. Springer (2012)
13. Bellare, M., Goldreich, O., Mityagin, A.: The Power of Verification Queries in Message Authentication and Authenticated Encryption. Cryptology ePrint Archive, Report 2004/309 (2004), `http://eprint.iacr.org/`
14. Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In: Okamoto, T. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 1976, pp. 531–545. Springer (2000)
15. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 4004, pp. 409–426. Springer (2006)
16. Bellare, M., Rogaway, P., Wagner, D.: The EAX Mode of Operation. In: Roy and Meier [48], pp. 389–407
17. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: The Keccak SHA-3 submission (January 2011), `http://keccak.noekeon.org/`
18. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In: Miri, A., Vaudenay, S. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 7118, pp. 320–337. Springer (2011)
19. Black, J., Rogaway, P.: CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions. In: Bellare, M. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 1880, pp. 197–215. Springer (2000)
20. Bogdanov, A., Lauridsen, M.M., Tischhauser, E.: AES-Based Authenticated Encryption Modes in Parallel High-Performance Software. IACR Cryptology ePrint Archive 2014, 186 (2014)

21. Bost, R., Sanders, O.: Trick or Tweak: On the (In)security of OTR's Tweaks. Cryptology ePrint Archive, Report 2016/234 (2016), `http://eprint.iacr.org/`
22. Chakraborty, D., Sarkar, P.: A General Construction of Tweakable Block Ciphers and Different Modes of Operations. IEEE Transactions on Information Theory 54(5), 1991–2006 (2008)
23. Daemen, J., Rijmen, V.: AES Proposal: Rijndael (1999)
24. Even, S., Mansour, Y.: A Construction of a Cipher From a Single Pseudorandom Permutation. In: Imai, H., Rivest, R.L., Matsumoto, T. (eds.) ASIACRYPT. Lecture Notes in Computer Science, vol. 739, pp. 210–224. Springer (1991)
25. Fleischmann, E., Forler, C., Lucks, S.: McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In: Canteaut, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 7549, pp. 196–215. Springer (2012)
26. Gligor, V.D., Donescu, P.: Fast Encryption and Authentication: XCBC Encryption and XECB Authentication Modes. In: Matsui, M. (ed.) FSE. Lecture Notes in Computer Science, vol. 2355, pp. 92–108. Springer (2001)
27. Gouvêa, C.P.L., López, J.: High Speed Implementation of Authenticated Encryption for the MSP430X Microcontroller. In: Hevia, A., Neven, G. (eds.) LATINCRYPT. Lecture Notes in Computer Science, vol. 7533, pp. 288–304. Springer (2012)
28. Iwata, T., Kurosawa, K.: OMAC: One-Key CBC MAC. In: Johansson, T. (ed.) FSE. Lecture Notes in Computer Science, vol. 2887, pp. 129–153. Springer (2003)
29. Iwata, T., Minematsu, K., Guo, J., Morioka, S.: CLOC: Authenticated Encryption for Short Input. In: FSE. Lecture Notes in Computer Science, vol. 8540, pp. 149–167. Springer (2014)
30. Iwata, T., Yasuda, K.: BTM: A Single-Key, Inverse-Cipher-Free Mode for Deterministic Authenticated Encryption. In: Jr., M.J.J., Rijmen, V., Safavi-Naini, R. (eds.) Selected Areas in Cryptography. Lecture Notes in Computer Science, vol. 5867, pp. 313–330. Springer (2009)
31. Jutla, C.S.: Encryption Modes with Almost Free Message Integrity. In: Pfitzmann, B. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 2045, pp. 529–544. Springer (2001)
32. Krawczyk, H.: The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?). In: Kilian, J. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 2139, pp. 310–331. Springer (2001)
33. Krovetz, T., Rogaway, P.: The Software Performance of Authenticated-Encryption Modes. In: Joux, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 306–327. Springer (2011)
34. Liskov, M., Rivest, R.L., Wagner, D.: Tweakable Block Ciphers. In: Yung, M. (ed.) CRYPTO. Lecture Notes in Computer Science, vol. 2442, pp. 31–46. Springer (2002)
35. Luby, M., Rackoff, C.: How to Construct Pseudorandom Permutations from Pseudorandom Functions. SIAM J. Comput. 17(2), 373–386 (1988)
36. Lucks, S.: Two-Pass Authenticated Encryption Faster Than Generic Composition. In: Gilbert, H., Handschuh, H. (eds.) FSE. Lecture Notes in Computer Science, vol. 3557, pp. 284–298. Springer (2005)
37. Maurer, U.M.: Indistinguishability of Random Systems. In: Knudsen, L.R. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 2332, pp. 110–132. Springer (2002)
38. Minematsu, K.: AES-OTR (A submission to CAESAR), `http://competitions.cr.yp.to/round1/aesotrv1.pdf/`
39. Minematsu, K.: A Short Universal Hash Function from Bit Rotation, and Applications to Blockcipher Modes. In: Susilo and Reyhanitabar [49], pp. 221–238
40. Namprempre, C., Rogaway, P., Shrimpton, T.: Reconsidering Generic Composition. DIAC 2013: Directions in Authenticated Ciphers (2013), available from `http://2013.diac.cr.yp.to/`
41. Osvik, D.A., Bos, J.W., Stefan, D., Canright, D.: Fast Software AES Encryption. In: Hong, S., Iwata, T. (eds.) FSE. Lecture Notes in Computer Science, vol. 6147, pp. 75–93. Springer (2010)
42. Paterson, K.: Authenticated Encryption in TLS. DIAC 2013: Directions in Authenticated Ciphers (2013), available from `http://2013.diac.cr.yp.to/`
43. Rinne, S.: Performance Analysis of Contemporary Light-Weight Cryptographic Algorithms on a Smart Card Microcontroller. SPEED – Software Performance Enhancement for Encryption and Decryption (2007), available from `http://www.hyperelliptic.org/SPEED/start07.html`
44. Rogaway, P.: Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 3329, pp. 16–31. Springer (2004)
45. Rogaway, P.: Nonce-Based Symmetric Encryption. In: Roy and Meier [48], pp. 348–359
46. Rogaway, P.: Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. Full version (2013), available from `http://www.cs.ucdavis.edu/~rogaway/papers/`
47. Rogaway, P., Bellare, M., Black, J.: OCB: A block-cipher mode of operation for efficient authenticated encryption. ACM Trans. Inf. Syst. Secur. 6(3), 365–403 (2003)
48. Roy, B.K., Meier, W. (eds.): Fast Software Encryption, 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004, Revised Papers, Lecture Notes in Computer Science, vol. 3017. Springer (2004)
49. Susilo, W., Reyhanitabar, R. (eds.): Provable Security - 7th International Conference, ProvSec 2013, Melaka, Malaysia, October 23-25, 2013. Proceedings, Lecture Notes in Computer Science, vol. 8209. Springer (2013)
50. Zeng, G., Han, W., He, K.: High Efficiency Feedback Shift Register: σ-LFSR. Cryptology ePrint Archive, Report 2007/114 (2007), `http://eprint.iacr.org/`
51. Zhang, L., Han, S., Wu, W., Wang, P.: iFeed: the Input-Feed AE Modes. Rump Session of FSE 2013 (2013), slides from `http://fse.2013.rump.cr.yp.to/`

A Proof of Theorem 3

PRIV bound. We observe that the any output block of encryption oracle $\mathbb{OTR}'$-$\mathcal{E}_\tau$ contains an output block of $\widetilde{\mathsf{R}}^{\langle N,i,\omega\rangle}$, where the tweak (N,i,ω) is uniquely used throughout the attack by PRIV-adversary $\mathcal{A}$. For example any odd ciphertext block contains an output of $\widetilde{\mathsf{R}}^{\langle N,i,\mathtt{f}\rangle}$ for odd i and any even ciphertext block contains an output of $\widetilde{\mathsf{R}}^{\langle N,i,\mathtt{s}\rangle}$ for even i, and a tag T contains TE, which is an output of $\widetilde{\mathsf{R}}^{\langle N,i,\omega\rangle}$ for some $\omega \in \{\mathtt{a}_1,\mathtt{a}_2,\mathtt{b}_1,\mathtt{b}_2\}$ and thus random. Tag T is an XOR of TE and TA and the latter is $\mathsf{R}^\infty(A)$ if $A \neq \varepsilon$ and 0^n if $A=\varepsilon$, therefore, T is also independent and random. This implies that the output blocks in (C,T) is completely random and independent of the adversary's choice (except the length), thus indistinguishable from those of \$ oracle. PRIV bound is naturally derived from this observation.

AUTH bound. We first consider the case $q_v = 1$. Let $\mathcal{A}$ be AUTH-adversary against $\mathbb{OTR}'$ with q encryption queries and a decryption query. Without loss of generality we can assume $\mathcal{A}$ first performs all encryption queries before the decryption query, which is the best strategy for maximizing the probability of successful forgery.

Following Section 2.2, we denote the i-th encryption query and the answer as (N_i,A_i,M_i) and (C_i,T_i). Here $|M_i| = |C_i|$ and $N_i \neq N_j$ for any $1 \le i < j \le q$ from the assumption. Let $(M_i[1],M_i[2],\dots,M_i[m_i]) \xleftarrow{n} M_i$ and $(MM_i[1],MM_i[2],\dots,MM_i[\ell_i]) \xleftarrow{2n} M_i$, where $M_i[j]$ is called a j-th block and $MM_i[j]$ is called a j-th chunk for M_i. Note that $m_i = |M_i|_n$ and $\ell_i = |M_i|_{2n}$ (which equals to $\lceil m_i/2\rceil$). For ciphertext we similarly define $C_i[j]$ and $CC_i[j]$. The decryption query (or forgery attempt) is denoted by (N',A',C',T'). We require $(N',A',C') \neq (N_i,A_i,C_i)$ for all $i=1,\dots,q$, since forgery attempt with $(N',A',C') = (N_i,A_i,C_i)$ and $T' \neq T_i$ for some i is always rejected.

Let T^* be the true tag value for the forgery attempt. Similarly we define TE^*, TA^* and Σ^* for the corresponding values produced in the decryption of the forgery attempt, which uses (N',A',C'). The forgery attempt is accepted as valid iff $T^* = T'$, where

$$T^* = \mathtt{msb}_\tau(TE^* \oplus TA^*),\ \text{and}\ TE^* = \mathtt{lsb}_n(\mathbb{DF}_{\widetilde{\mathsf{R}}}(N',C')),\ \text{and}\ TA^* = \mathsf{R}^\infty(A'), \tag{16}$$

where $\mathtt{lsb}_n(X)$ denotes the last (rightmost) n bits of X. Let $m' = |C'|_n$ and $\ell' = |C'|_{2n}$. We consider parsings, $(C'[1],\dots,C'[m']) \xleftarrow{n} C'$ and $(CC'[1],\dots,CC'[\ell']) \xleftarrow{2n} C'$. Note that TE^* is equal to $\widetilde{\mathsf{R}}^{\langle N',\ell',\omega'\rangle}(\Sigma^*)$, where Σ^* is generated as an internal variable of $\mathbb{DF}_{\widetilde{\mathsf{R}}}(N',C')$ for some $\omega' \in \{\mathtt{a}_1,\mathtt{a}_2,\mathtt{b}_1,\mathtt{b}_2\}$ uniquely determined by the length of C'. Application of function $\widetilde{\mathsf{R}}^{\langle N',\ell',\omega'\rangle}$ is called a finalization and the tweak (N',ℓ',ω') is called a finalization tweak.

Let $\mathbf{Z} = \{(N_i,A_i,M_i,C_i,T_i)\}_{i=1,\dots,q}$ be the transcript obtained by encryption queries. Seeing $\mathbf{Z}$ as a random variable, the forgery probability is written as

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'}(\mathcal{A}) = \Pr_{\mathcal{A},\mathbb{OTR}'}[T'=T^*] = \sum_{\mathbf{z}} \Pr_{\mathcal{A},\mathbb{OTR}'}[T'=T^*|\mathbf{Z}=\mathbf{z}] \cdot \Pr_{\mathcal{A},\mathbb{OTR}'}[\mathbf{Z}=\mathbf{z}], \tag{17}$$

where the probability space is defined by the interactive game involving $\mathcal{A}$ and $\mathbb{OTR}'$ (also applies to all probabilities hereafter). In deriving the authenticity bound, we fix adversary $\mathcal{A}$ and define $\mathrm{FP}_{\mathbf{z}}$ as $\Pr[T'=T^*|\mathbf{Z}=\mathbf{z}]$, and bound a maximum of $\mathrm{FP}_{\mathbf{z}}$ for all possible $\mathbf{z}$ with $\mathcal{A}$. This provides the upper bound of $\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'}(\mathcal{A})$. Here we can assume that $\mathcal{A}$ produces a decryption query (N',A',C',T') deterministically from $\mathbf{z}$ so that it maximizes $\mathrm{FP}_{\mathbf{z}}$. Note that, the transcript reveals all the input-output pairs for $\widetilde{\mathsf{R}}$ invoked at all encryption queries, except the residual bits of Z in the check sum for the case of even plaintext blocks (the security proof does not rely on this fact though). Hence (N',A',C',T') can be any function of these input/output pairs of $\widetilde{\mathsf{R}}$. We perform a case analysis for (N',A',C').

Case 1: $N' \neq N_i$ **for all** $1 \le i \le q$**.**
The finalization tweak is new, hence the TE^* is independent and uniformly random. Thus $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$.

Case 2: $(N',C') = (N_\alpha,C_\alpha)$ **for some** $1 \le \alpha \le q$**, and** $A' \neq A_\alpha$**.**
We have $T^* = \mathtt{msb}_\tau(TE_\alpha \oplus TA^*)$. First we observe that, throughout the attack the adversary obtains no knowledge about TA_α for all *non-empty* A_α, since TA_α is xored with TE_α, and $TE_1,\dots,TE_q$, including TE_α, are independent and uniform. Note that for any $i \le q$ with $A_i = \varepsilon$ we always have $TA_i = 0^n$, and for $A_\alpha \neq \varepsilon$ we have $TA_\alpha = \mathsf{R}^\infty(A_\alpha)$, which is random. If we have $A_\alpha = A_\beta \neq \varepsilon$ the adversary only knows that TA_α is uniformly random over $\{0,1\}^n$, thus completely unpredictable, and the equation $TA_\beta = TA_\alpha$. This means that the adversary can not predict TA_α for any non-empty A_α beyond random guess. Using this observation we do a further case analysis with respect to A'.

Case 2-1: $A' \neq A_i$ **for all** $i = 1, \ldots, q,$ **and** $A' \neq \varepsilon$**.**
We observe that $TA^* = \mathsf{R}^{\infty}(A')$ is uniformly random, thus $\mathrm{FP}_{\mathbf{z}} \leq 1/2^{\tau}$.

Case 2-2: $A' \neq A_i$ **for all** $i = 1, \ldots, q,$ **and** $A' = \varepsilon$**.**
We observe that $TA^* = 0^n$ and $T^* = \mathtt{msb}_{\tau}(TE_{\alpha}) = \mathtt{msb}_{\tau}(TA_{\alpha} \oplus T_{\alpha})$ for non-empty A_{α}. Then TA_{α} is completely unpredictable to the adversary. Thus T^* is also completely unpredictable, and we have $\mathrm{FP}_{\mathbf{z}} \leq 1/2^{\tau}$.

Case 2-3: $A' = A_{\beta} \neq \varepsilon$ **for some** $\beta \neq \alpha$**.**
We observe that $TA^* = TA_{\beta}$ and $T^* = \mathtt{msb}_{\tau}(TA_{\beta} \oplus TE_{\alpha})$. As TA_{β} for non-empty A_{β} is completely unpredictable, we have $\mathrm{FP}_{\mathbf{z}} \leq 1/2^{\tau}$.

Case 2-4: $A' = A_{\beta} = \varepsilon$ **for some** $\beta \neq \alpha$**.**
We observe that $TA^* = TA_{\beta} = 0^n$ and $T^* = \mathtt{msb}_{\tau}(TE_{\alpha}) = \mathtt{msb}_{\tau}(T_{\alpha} \oplus TA_{\alpha})$. As $A_{\alpha} \neq \varepsilon$ holds TA_{α} is completely unpredictable, and we have $\mathrm{FP}_{\mathbf{z}} \leq 1/2^{\tau}$.

Therefore, for all cases we have $\mathrm{FP}_{\mathbf{z}} \leq 1/2^{\tau}$. We then consider cases with $N' = N_{\alpha}$ for some α and $C' \neq C_{\alpha}$.

Case 3: $N' = N_{\alpha}$**,** $|C'| = |C_{\alpha}|$ **and** $C' \neq C_{\alpha}$ **for some** $1 \leq \alpha \leq q$**.**
Let $(C_{\alpha}[1], \ldots, C_{\alpha}[m_{\alpha}]) \xleftarrow{n} C_{\alpha}$ and $(CC_{\alpha}[1], \ldots, CC_{\alpha}[\ell_{\alpha}]) \xleftarrow{2n} C_{\alpha}$. Similarly, let $(C'[1], \ldots, C'[m']) \xleftarrow{n} C'$, and $(CC'[1], \ldots, CC'[\ell']) \xleftarrow{2n} C'$. Here we have $m' = m_{\alpha}$ and $\ell' = \ell_{\alpha}$ as $|C'| = |C_{\alpha}|$ holds. Note that we made no assumption on A'.

Case 3-1: $|CC'[\ell']| = 2n$**.**
We observe that the finalization tweaks for α-th query and the forgery attempt are the same, i.e. $(N_{\alpha}, \ell_{\alpha}, \mathtt{a}_2)$. This means that, there exists at least one chunk different, i.e., we must have $CC'[i] \neq CC_{\alpha}[i]$, for some $1 \leq i \leq \ell'$. We first consider the case $i < \ell_{\alpha}$. Then we obtain

$$M^*[2i-1] = \widetilde{\mathsf{R}}^{\langle N', i, \mathtt{s}\rangle}(C'[2i-1]) \oplus C'[2i], \text{ and} \tag{18}$$

$$M^*[2i] = \widetilde{\mathsf{R}}^{\langle N', i, \mathtt{f}\rangle}(M^*[2i-1]) \oplus C'[2i-1] \tag{19}$$

in the decryption process of the forgery attempt. Let e_1 denote the event $M^*[2i-1] = M_{\alpha}[2i-1]$. If $C'[2i-1] \neq C_{\alpha}[2i-1]$, e_1 occurs with probability $1/2^n$, and if $C'[2i-1] = C_{\alpha}[2i-1]$ and $C'[2i] \neq C_{\alpha}[2i]$, the probability is zero. The event $\overline{e_1}$, i.e. $M^*[2i-1] \neq M_{\alpha}[2i-1]$, implies that the input to $\widetilde{\mathsf{R}}^{\langle N', i, \mathtt{f}\rangle}$ is new, thus $M^*[2i]$ is uniformly random and independent of any other variables in the transcript. This makes the computed check sum in the decryption of forgery attempt, written as $\varSigma^*$, independent and uniformly random under the event $\overline{e_1}$. Let e_2 be the event that $\varSigma^* = \varSigma_{\alpha}$, where $\varSigma_{\alpha}$ equals to $M_{\alpha}[2] \oplus M_{\alpha}[4] \oplus \cdots \oplus M_{\alpha}[m_{\alpha}]$. The above analysis implies that $\Pr(e_1|\mathbf{Z} = \mathbf{z}) = 1/2^n$ and $\Pr(e_2|\overline{e_1}, \mathbf{Z} = \mathbf{z}) = 1/2^n$ hold for any $\mathbf{z}$. In addition, given $\overline{e_2}$, $TE^* = \widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{a}_2\rangle}(\varSigma^*)$ is uniformly random and independent of all previously generated values, since $\widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{a}_2\rangle}$ is only invoked once with input $\varSigma_{\alpha}$ in the encryption queries. Hence we have

$$\mathrm{FP}_{\mathbf{z}} = \Pr(\mathtt{msb}_{\tau}(TE^* \oplus TA^*) = T'|\mathbf{Z} = \mathbf{z}) \tag{20}$$

$$\leq \Pr(\mathtt{msb}_{\tau}(\widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{a}_2\rangle}(\varSigma^*) \oplus TA^*) = T'|\overline{e_1 \vee e_2}, \mathbf{Z} = \mathbf{z}) \cdot \Pr(\overline{e_1 \vee e_2}|\mathbf{Z} = \mathbf{z}) + \Pr(e_1 \vee e_2|\mathbf{Z} = \mathbf{z}) \tag{21}$$

$$\leq \max_{x \in \{0,1\}^{\tau}} \Pr(\mathtt{msb}_{\tau}(\widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{a}_2\rangle}(\varSigma^*)) = x|\overline{e_1} \wedge \overline{e_2}, \mathbf{Z} = \mathbf{z}) + \Pr(e_2|\overline{e_1}, \mathbf{Z} = \mathbf{z}) + \Pr(e_1|\mathbf{Z} = \mathbf{z}) \tag{22}$$

$$\leq \frac{1}{2^{\tau}} + \frac{2}{2^n}, \tag{23}$$

where the third inequality is obtained by taking the maximum for all possible values of TA^*. We then consider the case $i = \ell_{\alpha}$, i.e. the difference is in the last chunks. For this case the same analysis holds when we exchange $C'[2i-1]$ and $C'[2i]$. Thus $\mathrm{FP}_{\mathbf{z}}$ is bounded by $\frac{1}{2^{\tau}} + \frac{2}{2^n}$ as well.

Case 3-2: $n < |CC'[\ell']| < 2n$**.**
The finalization tweak is $(N_{\alpha}, \ell_{\alpha}, \mathtt{a}_1)$, for both α-th encryption query and the forgery attempt. We have $CC'[i] \neq CC_{\alpha}[i]$ for some $1 \leq i \leq \ell'$. If $i < \ell' \ (= \ell_{\alpha})$ the case is the same as Case 3-1. Otherwise we have $CC'[j] = CC_{\alpha}[j]$ for all $j = 1, \ldots, \ell' - 1$ and $CC'[\ell'] \neq CC_{\alpha}[\ell']$. If we have $C'[m'] \neq C_{\alpha}[m']$ (i.e. the difference is in the last partial blocks), the event $M^*[m'-1] = M_{\alpha}[m'-1]$, which we denote by event e_1, has probability $1/2^n$. This is because $M^*[m'-1] = \widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{s}\rangle}(\underline{C'[m']}) \oplus C'[m'-1]$ and $\underline{C'[m']}$ is a new input to $\widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{s}\rangle}$. If we have $C'[m'] = C_{\alpha}[m']$ and $C'[m'-1] \neq C_{\alpha}[m'-1]$ (i.e. the difference is in the last-but-one blocks), we always have $M^*[m'-1] \neq M_{\alpha}[m'-1]$, hence e_1 never occurs. When $\overline{e_1}$ occurs, $M^*[m'-1]$ is a new input to produce $Z^* = \widetilde{\mathsf{R}}^{\langle N_{\alpha}, \ell_{\alpha}, \mathtt{f}\rangle}(M^*[m'-1])$, which makes Z^* completely random. As $\varSigma^*$ contains $Z^* \oplus \underline{C'[m']}$, $\varSigma^*$ is also

random. Therefore, by defining event e_2 as $\Sigma^* = \Sigma_\alpha$, $\mathrm{FP}_{\mathbf{z}}$ is bounded as

$$\begin{aligned}
\mathrm{FP}_{\mathbf{z}} &\le \Pr(\mathtt{msb}_\tau(\widetilde{\mathsf{R}}^{\langle N_\alpha,\ell_\alpha,\mathtt{a}_1\rangle}(\Sigma^*) \oplus TA^*) = T' | \overline{e_1 \vee e_2}, \mathbf{Z} = \mathbf{z}) + \Pr(e_2 \vee e_1 | \mathbf{Z} = \mathbf{z}) \\
&\le \Pr(\mathtt{msb}_\tau(\widetilde{\mathsf{R}}^{\langle N_\alpha,\ell_\alpha,\mathtt{a}_1\rangle}(\Sigma^*) \oplus TA^*) = T' | \overline{e_1} \wedge \overline{e_2}, \mathbf{Z} = \mathbf{z}) + \Pr(e_2 | \overline{e_1}, \mathbf{Z} = \mathbf{z}) + \Pr(e_1 | \mathbf{Z} = \mathbf{z}) \\
&\le \max_{x \in \{0,1\}^\tau} \Pr(\mathtt{msb}_\tau(\widetilde{\mathsf{R}}^{\langle N_\alpha,\ell_\alpha,\mathtt{a}_1\rangle}(\Sigma^*)) = x | \overline{e_1} \wedge \overline{e_2}, \mathbf{Z} = \mathbf{z}) + \frac{2}{2^n} \\
&\le \frac{1}{2^\tau} + \frac{2}{2^n},
\end{aligned} \tag{24}$$

in the same manner as Case 3-1.

Case 3-3: $|\mathit{CC}'[\ell']| = n$.
The finalization tweak is $(N_\alpha, \ell_\alpha, \mathtt{b}_2)$, for both α-th encryption query and the forgery attempt. We have $\mathit{CC}'[i] \neq \mathit{CC}_\alpha[i]$ for some $1 \le i \le \ell'$. If $i < \ell'$ $(= \ell_\alpha)$ the case is the same as Case 3-1. Otherwise we have $\mathit{CC}'[j] = \mathit{CC}_\alpha[j]$ for all $j = 1, \ldots, \ell' - 1$ and $\mathit{CC}'[\ell'] \neq \mathit{CC}_\alpha[\ell']$, which implies $C'[m'] \neq C_\alpha[m']$ (i.e. the difference is in the last blocks). Since $M^*[m'] = C'[m'] \oplus Z^*$ with $Z^* = \widetilde{\mathsf{R}}^{\langle N_\alpha,\ell_\alpha,\mathtt{f}\rangle}(0^n)$, and $M_\alpha[m'] = C_\alpha[m'] \oplus Z_\alpha$ with $Z_\alpha = Z^*$, $M^*[m']$ is always different from $M_\alpha[m']$. As variables contained in Σ_α and Σ^* other than $M_\alpha[m_\alpha]$ and $M^*[m']$ are the same, we always have $\Sigma_\alpha \neq \Sigma^*$. Thus $TE^* = \widetilde{\mathsf{R}}^{\langle N_\alpha,\ell_\alpha,\mathtt{b}_2\rangle}(\Sigma^*)$ is random and independent of TE_α, implying $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$. Thus $\mathrm{FP}_{\mathbf{z}}$ is bounded by $1/2^\tau + 2/2^n$ in Case 3-3.

Case 3-4: $|\mathit{CC}'[\ell']| < n$.
The finalization tweak is $(N_\alpha, \ell_\alpha, \mathtt{b}_1)$, for both α-th encryption query and the forgery attempt. The analysis is similar to Case 3-3, and we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau + 2/2^n$.

Case 4: $N' = N_\alpha$, $|C'| \neq |C_\alpha|$ for some $1 \le \alpha \le q$.

Case 4-1: $|\mathit{CC}_\alpha[\ell_\alpha]| = 2n$.
The finalization tweak for the forgery attempt is $(N_\alpha, \ell', \omega)$ for $\omega \in \{\mathtt{a}_1, \mathtt{a}_1, \mathtt{b}_1, \mathtt{b}_2\}$, and that for the α-th encryption query is $(N_\alpha, \ell_\alpha, \mathtt{a}_2)$. Note that ℓ' may or may not equal to ℓ_α. As we have $(\ell_\alpha, \mathtt{a}_2) \neq (\ell', \omega)$ (otherwise $|C'| = |C_\alpha|$ holds) and $\{N_1, \ldots, N_q\}$ contains no collision, the finalization tweak $(N_\alpha, \ell', \omega)$ is not invoked in the encryption queries. Hence $TE^* = \widetilde{\mathsf{R}}^{\langle N_\alpha,\ell',\omega\rangle}(\Sigma^*)$ is independent and random irrespective of Σ^*. This implies $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$.

Case 4-2: $n < |\mathit{CC}_\alpha[\ell_\alpha]| < 2n$.
The finalization tweak for the forgery attempt is $(N_\alpha, \ell', \omega)$ for $\omega \in \{\mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$, and that for the α-th encryption query is $(N_\alpha, \ell_\alpha, \mathtt{a}_1)$. If $(\ell_\alpha, \mathtt{a}_1) \neq (\ell', \omega)$, we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$ as with Case 4-1. If $(\ell_\alpha, \mathtt{a}_1) = (\ell', \omega)$, then we must have $m_\alpha = m'$ and $|C_\alpha[m_\alpha]| \neq |C'[m']|$ (i.e. the number of blocks are the same and the last blocks have different lengths). This means that $\underline{C_\alpha[m_\alpha]} \neq \underline{C'[m']}$, i.e., the inputs to $\widetilde{\mathsf{R}}^{\langle N_\alpha,\ell_\alpha,\mathtt{s}\rangle}$ are different due to the padding. Defining two bad events, e_1 and e_2, in the same manner to Case 3-2, we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau + 2/2^n$.

Case 4-3: $|\mathit{CC}_\alpha[\ell_\alpha]| = n$.
The finalization tweak for the forgery attempt is $(N_\alpha, \ell', \omega)$ for $\omega \in \{\mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$, and that for the α-th encryption query is $(N_\alpha, \ell_\alpha, \mathtt{b}_2)$. We have $(\ell_\alpha, \mathtt{b}_2) \neq (\ell', \omega)$, and thus $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$ holds as Case 4-1.

Case 4-4: $|\mathit{CC}_\alpha[\ell_\alpha]| < n$.
The finalization tweak for the forgery attempt is $(N_\alpha, \ell', \omega)$ for $\omega \in \{\mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$, and that for the α-th encryption query is $(N_\alpha, \ell_\alpha, \mathtt{b}_1)$. If $(\ell_\alpha, \mathtt{b}_1) \neq (\ell', \omega)$, we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$ as Case 4-1, and if $(\ell_\alpha, \mathtt{b}_1) = (\ell', \omega)$ and there exists $\mathit{CC}'[i] \neq \mathit{CC}_\alpha[i]$ for some $i < \ell'$, the analysis is the same as Case 3-1, and we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau + 2/2^n$. If $(\ell_\alpha, \mathtt{b}_1) = (\ell', \omega)$ and $\mathit{CC}'[i] = \mathit{CC}_\alpha[i]$ for all $i < \ell'$, we must have $m_\alpha = m'$ and $|C'[m']|, |C_\alpha[m_\alpha]| < n$ and $|C'[m']| \neq |C_\alpha[m_\alpha]|$. Then we have $\underline{M^*[m']} \neq \underline{M_\alpha[m_\alpha]}$. This implies that $\Sigma^* \oplus \Sigma_\alpha$ is $\underline{M^*[m']} \oplus \underline{M_\alpha[m_\alpha]} \neq 0$, hence Σ^* and Σ_α are different. Therefore, we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$.

Summarizing all cases. In all cases, we have $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau + 2/2^n$. From Equation (17) this proves

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'}(\mathcal{A}) \le \sum_{\mathbf{z}} \mathrm{FP}_{\mathbf{z}} \cdot \Pr[\mathbf{Z} = \mathbf{z}] \le \frac{2}{2^n} + \frac{1}{2^\tau} \tag{25}$$

for AUTH-adversary $\mathcal{A}$ with $q_v = 1$. Combining Equation (25) with the result of Bellare, Goldreich and Mityagin [13], we have $\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTR}'}(\mathcal{A}) \le 2q_v/2^n + q_v/2^\tau$ for any $\mathcal{A}$ with $q_v \ge 1$. This completes the derivation of AUTH bound.

B Proof of Lemma 1

For proving the security bound of $\widetilde{G}[\mathsf{P}]$, the crucial observation is that the input mask, denoted by $\varDelta$ in Fig. 3, is differentially uniform for any two distinct inputs. From Fig. 3 this is easily confirmed that $\widetilde{G}[\mathsf{P}]$ is identical to Rogaway's XE mode [44] for $n = 128$. More specifically, all the coefficients of $\mathsf{P}(\underline{N})$ or $\mathsf{P}(0^n)$ appeared in Fig. 3 are written as $2^i 3^j 7^k$. [44] shows that[3] $2^i 3^j 7^k \neq 2^{i'} 3^{j'} 7^{k'}$ for any distinct (i, j, k) and (i', j', k') from $\mathcal{I} \times \mathcal{J} \times \mathcal{K}$, where $\mathcal{I} = [-2^{108}..2^{108}]$ and $\mathcal{J} = [-2^7..2^7]$ and $\mathcal{K} = [-2^7..2^7]$.

In our case, (i, j, k) is always in $\mathcal{I} \times \mathcal{J} \times \mathcal{K}$, assuming i, which represents the block index of plaintext/ciphertext/AD, is at most $2^{n/2} = 2^{64}$. Then, Theorem 7 of [44] proves that $\mathtt{Adv}^{\mathsf{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{P}}}(\mathcal{A}) \leq 4.5q^2/2^n$.

Finally, a generalized variant of PRP/PRF switching lemma (e.g., Lemma 1 of [15]) tells that $\mathtt{Adv}^{\mathsf{cpa}}_{\widetilde{\mathsf{P}},\widetilde{\mathsf{R}}}(\mathcal{A}) \leq 0.5q^2/2^n$ for q CPA queries, hence the proof is completed as $\mathtt{Adv}^{\mathsf{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{A}) \leq \mathtt{Adv}^{\mathsf{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{P}}}(\mathcal{A}) + \mathtt{Adv}^{\mathsf{cpa}}_{\widetilde{\mathsf{P}},\widetilde{\mathsf{R}}}(\mathcal{A}) \leq 4.5q^2/2^n + 0.5q^2/2^n$. $\square$

[3] This result does depend on the choice of primitive polynomial defining GF(2^n).

```
Algorithm $\mathbb{OTR}'\text{-}\mathcal{E}_\tau(N, A, M)$

 1. $(C, TE) \leftarrow \mathbb{EF}_{\widetilde{\mathsf{R}}}(N, M)$
 2. if $A \neq \varepsilon$ then $TA \leftarrow \mathsf{R}^\infty(A)$
 3. else $TA \leftarrow 0^n$
 4. $T \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
 5. return $(C, T)$
```

```
Algorithm $\mathbb{OTR}'\text{-}\mathcal{D}_\tau(N, A, C, T)$

 1. $(M, TE) \leftarrow \mathbb{DF}_{\widetilde{\mathsf{R}}}(N, C)$
 2. if $A \neq \varepsilon$ then $TA \leftarrow \mathsf{R}^\infty(A)$
 3. else $TA \leftarrow 0^n$
 4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
 5. if $\widehat{T} = T$ return $M$
 6. else return $\perp$
```

```
Algorithm $\mathbb{OTR}\text{-}\mathcal{E}_\tau(N, A, M)$

 1. $(C, TE) \leftarrow \mathbb{EF}_{\widetilde{\mathsf{R}}}(N, M)$
 2. if $A \neq \varepsilon$ then $TA \leftarrow \mathbb{AF}_{\widetilde{\mathsf{R}}}(A)$
 3. else $TA \leftarrow 0^n$
 4. $T \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
 5. return $(C, T)$
```

```
Algorithm $\mathbb{OTR}\text{-}\mathcal{D}_\tau(N, A, C, T)$

 1. $(M, TE) \leftarrow \mathbb{DF}_{\widetilde{\mathsf{R}}}(N, C)$
 2. if $A \neq \varepsilon$ then $TA \leftarrow \mathbb{AF}_{\widetilde{\mathsf{R}}}(A)$
 3. else $TA \leftarrow 0^n$
 4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
 5. if $\widehat{T} = T$ return $M$
 6. else return $\perp$
```

```
Algorithm $\mathbb{EF}_{\widetilde{\mathsf{R}}}(N, M)$

 1. $\Sigma \leftarrow 0^n$
 2. $(M[1], \ldots, M[m]) \xleftarrow{n} M$
 3. $\ell \leftarrow \lceil m/2 \rceil$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $C[2i-1] \leftarrow \widetilde{\mathsf{R}}^{\langle N,i,\mathtt{f}\rangle}(M[2i-1]) \oplus M[2i]$
 6.     $C[2i] \leftarrow \widetilde{\mathsf{R}}^{\langle N,i,\mathtt{s}\rangle}(C[2i-1]) \oplus M[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8. if $m$ is even
 9.     $Z \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{f}\rangle}(M[m-1])$
10.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
11.     $C[m-1] \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{s}\rangle}(\underline{C[m]}) \oplus M[m-1]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(\widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{f}\rangle}(0^n)) \oplus M[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. if $m$ is even and $|M[m]| \neq n$
17.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{a}_1\rangle}(\Sigma)$
18. if $m$ is even and $|M[m]| = n$
19.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{a}_2\rangle}(\Sigma)$
20. if $m$ is odd and $|M[m]| \neq n$
21.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{b}_1\rangle}(\Sigma)$
22. if $m$ is odd and $|M[m]| = n$
23.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{b}_2\rangle}(\Sigma)$
24. $C \leftarrow (C[1], \ldots, C[m])$
25. return $(C, TE)$
```

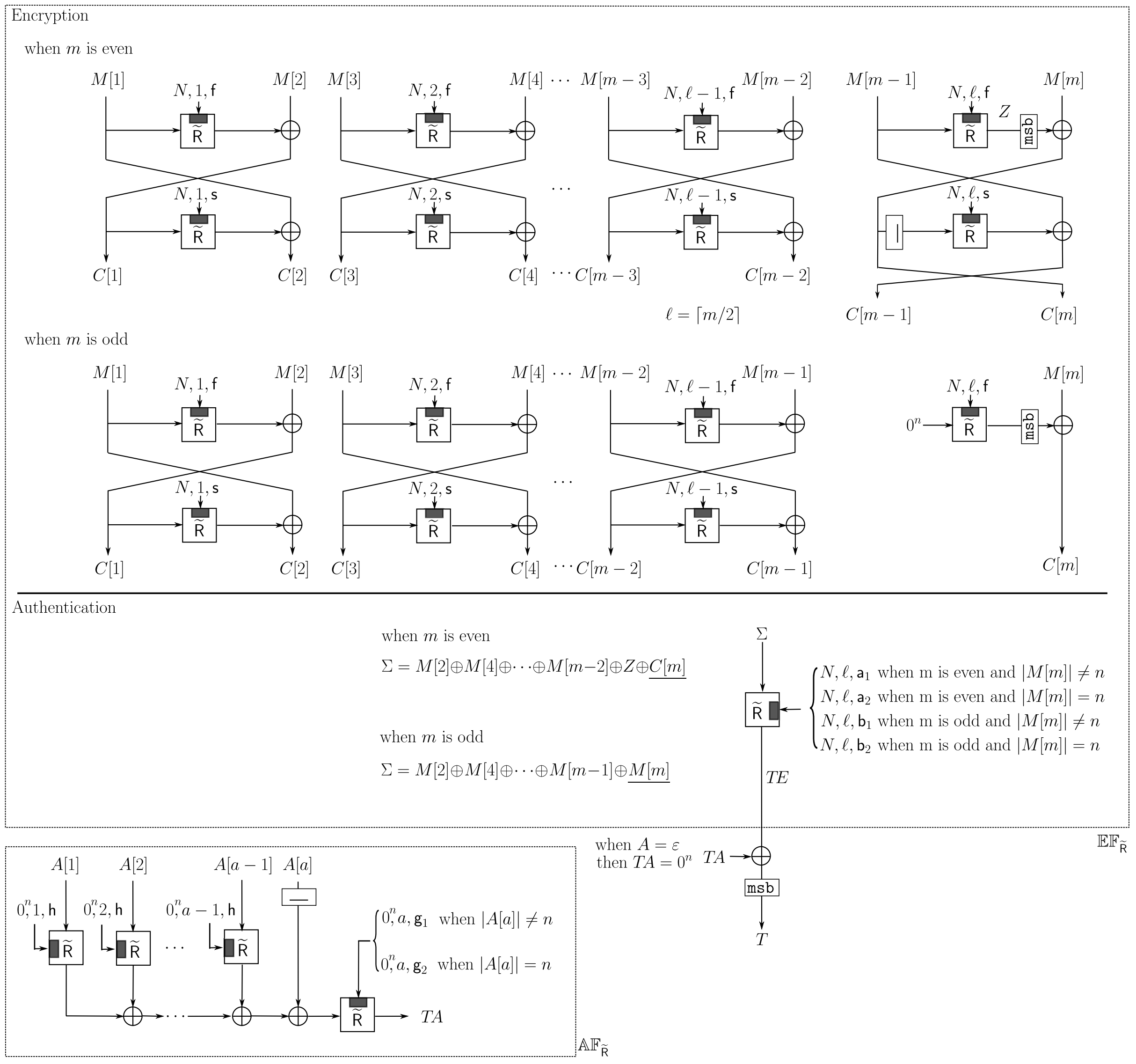
```
Algorithm $\mathbb{DF}_{\widetilde{\mathsf{R}}}(N, C)$

 1. $\Sigma \leftarrow 0^n$
 2. $(C[1], \ldots, C[m]) \xleftarrow{n} C$
 3. $\ell \leftarrow \lceil m/2 \rceil$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $M[2i-1] \leftarrow \widetilde{\mathsf{R}}^{\langle N,i,\mathtt{s}\rangle}(C[2i-1]) \oplus C[2i]$
 6.     $M[2i] \leftarrow \widetilde{\mathsf{R}}^{\langle N,i,\mathtt{f}\rangle}(M[2i-1]) \oplus C[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8. if $m$ is even
 9.     $M[m-1] \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{s}\rangle}(\underline{C[m]}) \oplus C[m-1]$
10.     $Z \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{f}\rangle}(M[m-1])$
11.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(\widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{f}\rangle}(0^n)) \oplus C[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. if $m$ is even and $|C[m]| \neq n$
17.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{a}_1\rangle}(\Sigma)$
18. if $m$ is even and $|C[m]| = n$
19.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{a}_2\rangle}(\Sigma)$
20. if $m$ is odd and $|C[m]| \neq n$
21.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{b}_1\rangle}(\Sigma)$
22. if $m$ is odd and $|C[m]| = n$
23.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle N,\ell,\mathtt{b}_2\rangle}(\Sigma)$
24. $M \leftarrow (M[1], \ldots, M[m])$
25. return $(M, TE)$
```

```
Algorithm $\mathbb{AF}_{\widetilde{\mathsf{R}}}(A)$

 1. $\Xi \leftarrow 0^n$
 2. $(A[1], \ldots, A[a]) \xleftarrow{n} A$
 3. for $i = 1$ to $a - 1$ do
 4.     $\Xi \leftarrow \Xi \oplus \widetilde{\mathsf{R}}^{\langle 0^n,i,\mathtt{h}\rangle}(A[i])$
 5. $\Xi \leftarrow \Xi \oplus \underline{A[a]}$
 6. if $|A[a]| \neq n$ then $TA \leftarrow \widetilde{\mathsf{R}}^{\langle 0^n,a,\mathtt{g}_1\rangle}(\Xi)$
 7. else $TA \leftarrow \widetilde{\mathsf{R}}^{\langle 0^n,a,\mathtt{g}_2\rangle}(\Xi)$
 8. return $TA$
```

Fig. 4. The components of $\mathbb{OTR}'[\tau]$ and $\mathbb{OTR}[\tau]$.



Encryption
when m is even
$M[1]$ $N,1,\mathsf{f}$ $M[2]$ $M[3]$ $N,2,\mathsf{f}$ $M[4]$ $\cdots$ $M[m-3]$ $N,\ell-1,\mathsf{f}$ $M[m-2]$ $M[m-1]$ N,ℓ,f $M[m]$
Z
msb
$N,1,\mathsf{s}$ $N,2,\mathsf{s}$ $N,\ell-1,\mathsf{s}$ N,ℓ,s
$\widetilde{\mathsf{R}}$
$C[1]$ $C[2]$ $C[3]$ $C[4]$ $\cdots C[m-3]$ $C[m-2]$
$\ell = \lceil m/2 \rceil$
$C[m-1]$ $C[m]$
when m is odd
$M[1]$ $N,1,\mathsf{f}$ $M[2]$ $M[3]$ $N,2,\mathsf{f}$ $M[4]$ $\cdots$ $M[m-2]$ $N,\ell-1,\mathsf{f}$ $M[m-1]$ N,ℓ,f $M[m]$
0^n
$N,1,\mathsf{s}$ $N,2,\mathsf{s}$ $N,\ell-1,\mathsf{s}$
$C[1]$ $C[2]$ $C[3]$ $C[4]$ $\cdots C[m-2]$ $C[m-1]$ $C[m]$
Authentication
when m is even
$\Sigma = M[2]\oplus M[4]\oplus\cdots\oplus M[m-2]\oplus Z\oplus \underline{C[m]}$
when m is odd
$\Sigma = M[2]\oplus M[4]\oplus\cdots\oplus M[m-1]\oplus \underline{M[m]}$
Σ
N,ℓ,a_1 when m is even and $|M[m]| \neq n$
N,ℓ,a_2 when m is even and $|M[m]| = n$
N,ℓ,b_1 when m is odd and $|M[m]| \neq n$
N,ℓ,b_2 when m is odd and $|M[m]| = n$
TE
$\mathbb{EF}_{\widetilde{\mathsf{R}}}$
when $A=\varepsilon$ then $TA = 0^n$
TA
msb
T
$A[1]$ $A[2]$ $A[a-1]$ $A[a]$
$0^n, 1, \mathsf{h}$ $0^n, 2, \mathsf{h}$ $0^n, a-1, \mathsf{h}$
$0^n, a, \mathsf{g}_1$ when $|A[a]| \neq n$
$0^n, a, \mathsf{g}_2$ when $|A[a]| = n$
TA
$\mathbb{AF}_{\widetilde{\mathsf{R}}}$

Fig. 5. $\mathbb{OTR}$ function.

```
**Algorithm** OTR-$\mathcal{E}_{E,\tau}(N, A, M)$

1. $(C, TE) \leftarrow \mathrm{EF}_E(N, M)$
2. **if** $A \neq \varepsilon$ **then** $TA \leftarrow \mathrm{AF}_E(A)$
3. **else** $TA \leftarrow 0^n$
4. $T \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
5. **return** $(C, T)$
```

```
**Algorithm** OTR-$\mathcal{D}_{E,\tau}(N, A, C, T)$

1. $(M, TE) \leftarrow \mathrm{DF}_E(N, C)$
2. **if** $A \neq \varepsilon$ **then** $TA \leftarrow \mathrm{AF}_E(A)$
3. **else** $TA \leftarrow 0^n$
4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE \oplus TA)$
5. **if** $\widehat{T} = T$ **return** $M$
6. **else return** $\perp$
```

```
**Algorithm** $\mathrm{EF}_E(N, M)$

1.  $\Sigma \leftarrow 0^n$
2.  $L \leftarrow E(\underline{N})$
3.  $(M[1], \ldots, M[m]) \overset{n}{\leftarrow} M$, $\ell \leftarrow \lceil m/2 \rceil$
4.  **for** $i = 1$ **to** $\ell - 1$ **do**
5.      $C[2i-1] \leftarrow E(2^{i-1}L \oplus M[2i-1]) \oplus M[2i]$
6.      $C[2i] \leftarrow E(2^{i-1}3L \oplus C[2i-1]) \oplus M[2i-1]$
7.      $\Sigma \leftarrow \Sigma \oplus M[2i]$
8.  **if** $m$ **is even**
9.      $Z \leftarrow E(2^{\ell-1}L \oplus M[m-1])$
10.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
11.     $C[m-1] \leftarrow E(2^{\ell-1}3L \oplus \underline{C[m]}) \oplus M[m-1]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. **if** $m$ **is odd**
14.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(E(2^{\ell-1}L)) \oplus M[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. **if** $m$ **is even and** $|M[m]| \neq n$
17.     **then** $TE \leftarrow E(2^{\ell-1}3^3L \oplus \Sigma)$
18. **if** $m$ **is even and** $|M[m]| = n$
19.     **then** $TE \leftarrow E(2^{\ell-1}3^1 7L \oplus \Sigma)$
20. **if** $m$ **is odd and** $|M[m]| \neq n$
21.     **then** $TE \leftarrow E(2^{\ell-1}3^2L \oplus \Sigma)$
22. **if** $m$ **is odd and** $|M[m]| = n$
23.     **then** $TE \leftarrow E(2^{\ell-1}7L \oplus \Sigma)$
24. $C \leftarrow (C[1], \ldots, C[m])$
25. **return** $(C, TE)$
```

```
**Algorithm** $\mathrm{DF}_E(N, C)$

1.  $\Sigma \leftarrow 0^n$
2.  $L \leftarrow E(\underline{N})$
3.  $(C[1], \ldots, C[m]) \overset{n}{\leftarrow} C$, $\ell \leftarrow \lceil m/2 \rceil$
4.  **for** $i = 1$ **to** $\ell - 1$ **do**
5.      $M[2i-1] \leftarrow E(2^{i-1}3L \oplus C[2i-1]) \oplus C[2i]$
6.      $M[2i] \leftarrow E(2^{i-1}L \oplus M[2i-1]) \oplus C[2i-1]$
7.      $\Sigma \leftarrow \Sigma \oplus M[2i]$
8.  **if** $m$ **is even**
9.      $M[m-1] \leftarrow E(2^{\ell-1}3L \oplus \underline{C[m]}) \oplus C[m-1]$
10.     $Z \leftarrow E(2^{\ell-1}L \oplus M[m-1])$
11.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. **if** $m$ **is odd**
14.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(E(2^{\ell-1}L)) \oplus C[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. **if** $m$ **is even and** $|C[m]| \neq n$
17.     **then** $TE \leftarrow E(2^{\ell-1}3^3L \oplus \Sigma)$
18. **if** $m$ **is even and** $|C[m]| = n$
19.     **then** $TE \leftarrow E(2^{\ell-1}3^1 7L \oplus \Sigma)$
20. **if** $m$ **is odd and** $|C[m]| \neq n$
21.     **then** $TE \leftarrow E(2^{\ell-1}3^2L \oplus \Sigma)$
22. **if** $m$ **is odd and** $|C[m]| = n$
23.     **then** $TE \leftarrow E(2^{\ell-1}7L \oplus \Sigma)$
24. $M \leftarrow (M[1], \ldots, M[m])$
25. **return** $(M, TE)$
```

```
**Algorithm** $\mathrm{AF}_E(A)$

1. $\Xi \leftarrow 0^n$
2. $Q \leftarrow E(0^n)$
3. $(A[1], \ldots, A[a]) \overset{n}{\leftarrow} A$
4. **for** $i = 1$ **to** $a - 1$ **do**
5.     $\Xi \leftarrow \Xi \oplus E(2^{i-1}Q \oplus A[i])$
6. $\Xi \leftarrow \Xi \oplus \underline{A[a]}$
7. **if** $|A[a]| \neq n$ **then** $TA \leftarrow E(2^{a-1}3Q \oplus \Xi)$
8. **else** $TA \leftarrow E(2^{a-1}3^2Q \oplus \Xi)$
9. **return** $TA$
```

Fig. 6. Alternative representation of Fig. 1 for explicitly showing the masking coefficients.

C A Variant with Serial AD Processing

Motivation. While OTR of the main body (Section 3) enables parallel processing for both message and associated data (AD), we naturally want to simplify it when the underlying computing environment is serial. We find that PMAC-like AF_E for AD processing (ADP) is not well suited to serial computing, causing GF doubling for each AD block. Moreover, when AD is processed first (which is normal), we need to keep TA until we compute TE, which requires additional memory state. With this in mind, this section defines a variant of OTR of the main body, which uses CMAC [28] for ADP. We call this variant as "OTR with serial ADP", while the original scheme in the main body may be called "OTR with parallel ADP", if needed, for a consistency with a submission document of OTR [38] to CAESAR competition [1]. In addition, for the notational purpose this section uses the name OTRS for an alias of "OTR with serial ADP", and uses the name OTR to denote the scheme of the main body.

The security bounds of OTRS and corresponding proofs are also given in this section. Although the security bounds are mostly the same as OTR, proving the security bounds of OTRS requires partially different steps due to the differences in authentication.

C.1 Specification

We write $\mathrm{OTRS}[E,\tau]$ to denote OTRS with blockcipher E, tag bit size τ (and this corresponds to $\mathrm{OTR}[E,\tau,s]$ in [38]). The encryption and decryption functions are denoted by $\mathrm{OTRS}\text{-}\mathcal{E}_{E,\tau}$ and $\mathrm{OTRS}\text{-}\mathcal{D}_{E,\tau}$, and shown in Fig. 7. Encryption of OTRS is also illustrated in Fig 8, and an alternative representation of Fig. 1 is shown by Fig. 6.

The interfaces of $\mathrm{OTRS}\text{-}\mathcal{E}_{E,\tau}$ and $\mathrm{OTRS}\text{-}\mathcal{D}_{E,\tau}$ are the same as $\mathrm{OTR}\text{-}\mathcal{E}_{E,\tau}$ and $\mathrm{OTR}\text{-}\mathcal{D}_{E,\tau}$ of OTR. The algorithms of Fig. 7 are further decomposed into encryption core $\mathrm{EF\text{-}S}_E$, decryption core $\mathrm{DF\text{-}S}_E$, and authentication core $\mathrm{AF\text{-}S}_E$. For $\mathrm{EF\text{-}S}_E$ and $\mathrm{DF\text{-}S}_E$ the pseudocodes are the same as EF_E and DF_E of Fig. 4 except line 2. Here, TA is generated by $\mathrm{AF\text{-}S}_E$, a variant of OMAC [28], also known as CMAC [4].

C.2 Security Analysis

Extended security notion. PRIV and AUTH notions are slightly extended in that the adversary is allowed to perform encryption queries $(N_1, A_1, M_1), \dots, (N_q, A_q, M_q)$ as long as $(N_i, A_i) \neq (N_j, A_j)$ holds for any $i \neq j$. That is, the security is preserved as long as the uniqueness of (A, N) pairs is guaranteed for encryptions, even if the uniqueness of original nonce, N, is not guaranteed. In a word we can deem (A, N) as nonce. This comes from the structure of the scheme for combining the result of ADP and the encryption, and has a similarity to CLOC [29]. In conjunction with this extension we extend the definitions of PRIV-adversary, AUTH-adversary, and the supplemental notions $\mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{F,G}(\mathcal{A})$ and $\mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{F,G}(\mathcal{A})$ so that the adversary is allowed to perform encryption queries as long as the uniqueness of pairs (A, N) is guaranteed for encryptions. The parameters for adversaries, such as q and σ_A, are similarly defined as in the main body.
Bounds. We provide the security bounds of OTRS. For simplicity we assume the underlying blockcipher is an n-bit URP, P. The computational counterparts are fairly straightforward, thus omitted.

Theorem 4. *Fix $\tau \in \{1, \dots, n\}$. For any PRIV-adversary $\mathcal{A}$ with parameter (q, σ_A, σ_M),*

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathrm{OTRS}[\mathsf{P},\tau]}(\mathcal{A}) \leq \frac{5\sigma^2_{\mathtt{priv}}}{2^n}$$

holds for $\sigma_{\mathtt{priv}} = q + \sigma_A + \sigma_M$.

Theorem 5. *Fix $\tau \in \{1, \dots, n\}$. For any AUTH-adversary $\mathcal{A}$ with parameter $(q, q_v, \sigma_A, \sigma_M, \sigma_{A'}, \sigma_{C'})$,*

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathrm{OTRS}[\mathsf{P},\tau]}(\mathcal{A}) \leq \frac{7\sigma^2_{\mathtt{auth}}}{2^n} + \frac{q_v}{2^\tau}$$

holds for $\sigma_{\mathtt{auth}} = q + q_v + \sigma_A + \sigma_M + \sigma_{A'} + \sigma_{C'}$.

```
**Algorithm** OTRS-$\mathcal{E}_{E,\tau}(N, A, M)$

 1. **if** $A \neq \varepsilon$ **then** $TA \leftarrow$ AF-S$_E(A)$
 2. **else** $TA \leftarrow 0^n$
 3. $(C, TE) \leftarrow$ EF-S$_E(N, M, TA)$
 4. $T \leftarrow \mathtt{msb}_\tau(TE)$
 5. **return** $(C, T)$
```

```
**Algorithm** OTRS-$\mathcal{D}_{E,\tau}(N, A, C, T)$

 1. **if** $A \neq \varepsilon$ **then** $TA \leftarrow$ AF-S$_E(A)$
 2. **else** $TA \leftarrow 0^n$
 3. $(M, TE) \leftarrow$ DF-S$_E(N, C, TA)$
 4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE)$
 5. **if** $\widehat{T} = T$ **return** $M$
 6. **else return** $\perp$
```

```
**Algorithm** EF-S$_E(N, M, TA)$

 1. $\Sigma \leftarrow 0^n$
 2. $U \leftarrow 2(E(\underline{N}) \oplus TA)$, $L \leftarrow U$, $L^\sharp \leftarrow 3U$
 3. $(M[1], \ldots, M[m]) \overset{n}{\leftarrow} M$
 4. **for** $i = 1$ **to** $\lceil m/2 \rceil - 1$ **do**
 5.     $C[2i-1] \leftarrow E(L \oplus M[2i-1]) \oplus M[2i]$
 6.     $C[2i] \leftarrow E(L^\sharp \oplus C[2i-1]) \oplus M[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8.     $L \leftarrow L \oplus L^\sharp$, $L^\sharp \leftarrow 2L^\sharp$          // $L = 2^iU$, $L^\sharp = 2^i3U$
 9. **if** $m$ **is even**
10.     $Z \leftarrow E(L \oplus M[m-1])$
11.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
12.     $C[m-1] \leftarrow E(L^\sharp \oplus \underline{C[m]}) \oplus M[m-1]$
13.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
14.     $L^* \leftarrow L^\sharp$
15. **if** $m$ **is odd**
16.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(E(L)) \oplus M[m]$
17.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
18.     $L^* \leftarrow L$
19. **if** $|M[m]| \neq n$ **then** $TE \leftarrow E(3^2L^* \oplus \Sigma)$
20. **else** $TE \leftarrow E(7L^* \oplus \Sigma)$
21. $C \leftarrow (C[1], \ldots, C[m])$
22. **return** $(C, TE)$
```

```
**Algorithm** DF-S$_E(N, C, TA)$

 1. $\Sigma \leftarrow 0^n$
 2. $U \leftarrow 2(E(\underline{N}) \oplus TA)$, $L \leftarrow U$, $L^\sharp \leftarrow 3U$
 3. $(C[1], \ldots, C[m]) \overset{n}{\leftarrow} C$
 4. **for** $i = 1$ **to** $\lceil m/2 \rceil - 1$ **do**
 5.     $M[2i-1] \leftarrow E(L^\sharp \oplus C[2i-1]) \oplus C[2i]$
 6.     $M[2i] \leftarrow E(L \oplus M[2i-1]) \oplus C[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8.     $L \leftarrow L \oplus L^\sharp$, $L^\sharp \leftarrow 2L^\sharp$          // $L = 2^iU$, $L^\sharp = 2^i3U$
 9. **if** $m$ **is even**
10.     $M[m-1] \leftarrow E(L^\sharp \oplus \underline{C[m]}) \oplus C[m-1]$
11.     $Z \leftarrow E(L \oplus M[m-1])$
12.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
13.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
14.     $L^* \leftarrow L^\sharp$
15. **if** $m$ **is odd**
16.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(E(L)) \oplus C[m]$
17.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
18.     $L^* \leftarrow L$
19. **if** $|C[m]| \neq n$ **then** $TE \leftarrow E(3^2L^* \oplus \Sigma)$
20. **else** $TE \leftarrow E(7L^* \oplus \Sigma)$
21. $M \leftarrow (M[1], \ldots, M[m])$
22. **return** $(M, TE)$
```

```
**Algorithm** AF-S$_E(A)$

 1. $\Xi \leftarrow 0^n$
 2. $Q \leftarrow E(0^n)$
 3. $(A[1], \ldots, A[a]) \overset{n}{\leftarrow} A$
 4. **for** $i = 1$ **to** $a - 1$ **do**
 5.     $\Xi \leftarrow E(A[i] \oplus \Xi)$
 6. $\Xi \leftarrow \Xi \oplus \underline{A[a]}$
 7. **if** $|A[a]| \neq n$ **then** $TA \leftarrow E(2Q \oplus \Xi)$
 8. **else** $TA \leftarrow E(4Q \oplus \Xi)$
 9. **return** $TA$
```

Fig. 7. Algorithms of OTRS. Tag bit size is $0 < \tau \leq n$, and $\underline{X}$ denotes the 10^* padding of X.

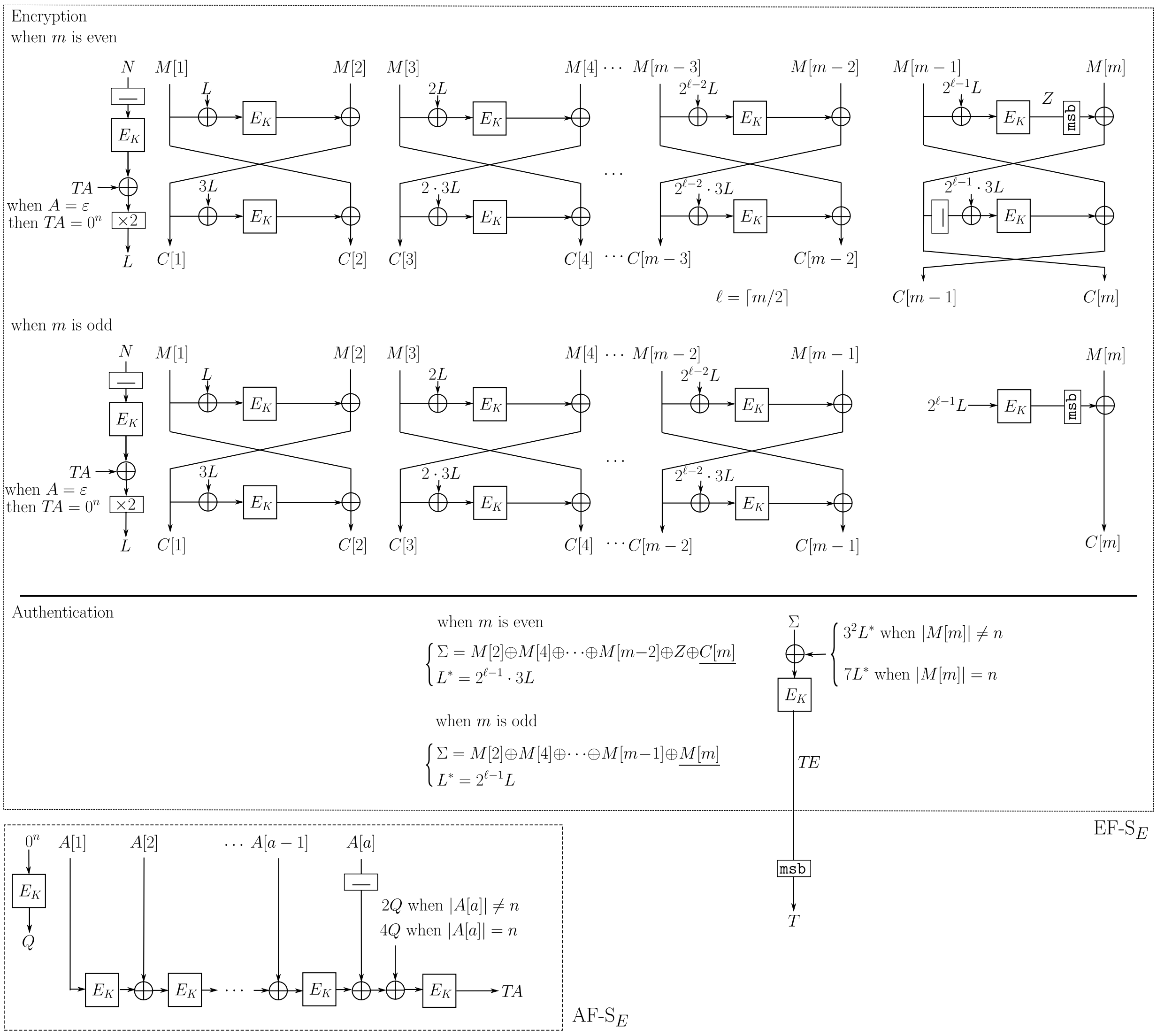

Encryption
when m is even
N $M[1]$ $M[2]$ $M[3]$ $M[4]$ $\cdots$ $M[m-3]$ $M[m-2]$ $M[m-1]$ $M[m]$
L $2L$ $2^{\ell-2}L$ $2^{\ell-1}L$ Z
E_K msb
TA
when $A=\varepsilon$ then $TA=0^n$
$\times 2$
$3L$ $2\cdot 3L$ $2^{\ell-2}\cdot 3L$ $2^{\ell-1}\cdot 3L$
L $C[1]$ $C[2]$ $C[3]$ $C[4]$ $\cdots C[m-3]$ $C[m-2]$
$\ell=\lceil m/2\rceil$ $C[m-1]$ $C[m]$
when m is odd
N $M[1]$ $M[2]$ $M[3]$ $M[4]$ $\cdots$ $M[m-2]$ $M[m-1]$ $M[m]$
$2^{\ell-1}L$
L $C[1]$ $C[2]$ $C[3]$ $C[4]$ $\cdots C[m-2]$ $C[m-1]$ $C[m]$
Authentication
when m is even
$\Sigma = M[2]\oplus M[4]\oplus\cdots\oplus M[m-2]\oplus Z\oplus \underline{C[m]}$
$L^* = 2^{\ell-1}\cdot 3L$
when m is odd
$\Sigma = M[2]\oplus M[4]\oplus\cdots\oplus M[m-1]\oplus \underline{M[m]}$
$L^* = 2^{\ell-1}L$
Σ
3^2L^* when $|M[m]|\neq n$
$7L^*$ when $|M[m]| = n$
TE
msb
T
EF-S$_E$
0^n $A[1]$ $A[2]$ $\cdots A[a-1]$ $A[a]$
Q
$2Q$ when $|A[a]|\neq n$
$4Q$ when $|A[a]| = n$
TA
AF-S$_E$

Fig. 8. Encryption of OTRS. Box with underline and $\underline{X}$ denote the 10^* padding of input X.

C.3 Proofs of Theorems 4 and 5

Overview. The proof structure is the same as those of Theorem 1 and Theorem 2. In the first step, we reinterpret the scheme as a mode of tweakable blockcipher, and then prove the security of the tweakable blockcipher in the second step, and the third step combines the results of previous two steps. However, the difference in the AD processing and the place to add TA makes some differences in proofs, mostly in the second step.

First step: TBC-based design. We define an AEAD scheme denoted by $\mathbb{OTRS}[\tau]$. It is compatible to OTRS$[E,\tau]$ and uses a tweakable n-bit URF, $\widetilde{\mathsf{R}} : \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$. Here, tweak T is represented as a vector, $T = (x, y, i, w) \in \mathcal{T} \overset{\text{def}}{=} \mathcal{A}_{ae} \times \mathcal{N}_{ae} \times \mathbb{N} \times \Omega$, where $\Omega = \{\mathtt{f}, \mathtt{s}, \mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$. The definition of $\mathbb{OTRS}$ is in Fig. 9 and its encryption is illustrated in Fig. 14. Here $\mathbb{OTRS}[\tau]$ consists of encryption function, $\mathbb{OTRS}\text{-}\mathcal{E}_\tau$, and decryption function, $\mathbb{OTRS}\text{-}\mathcal{D}_\tau$, and these functions use encryption and decryption cores, $\mathbb{EF}\text{-}\mathbb{S}_{\widetilde{\mathsf{R}}}$ and $\mathbb{DF}\text{-}\mathbb{S}_{\widetilde{\mathsf{R}}}$, shown in Fig. 9. These cores can be seen as counterparts to EF-S$_E$ and DF-S$_E$. The AD processing is absorbed, hence there is no counterpart to AF-S$_E$. The privacy and authenticity bounds of $\mathbb{OTRS}$ are shown in Theorem 6. The proof of Theorem 6 is in Section C.5.

Theorem 6. *Fix $\tau \in \{1,\ldots,n\}$. For any PRIV-adversary $\mathcal{A}$,*

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathbb{OTRS}[\tau]}(\mathcal{A}) = 0.$$

```
Algorithm $\mathbb{OTRS}\text{-}\mathcal{E}_\tau(N, A, M)$

 1. $(C, TE) \leftarrow \mathbb{EF}\text{-S}_{\widetilde{\mathsf{R}}}(N, A, M)$
 2. $T \leftarrow \mathtt{msb}_\tau(TE)$
 3. return $(C, T)$
```

```
Algorithm $\mathbb{OTRS}\text{-}\mathcal{D}_\tau(N, C, A, T)$

 1. $(M, TE) \leftarrow \mathbb{DF}\text{-S}_{\widetilde{\mathsf{R}}}(N, A, C)$
 2. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE)$
 3. if $\widehat{T} = T$ return $M$
 4. else return $\perp$
```

```
Algorithm $\mathbb{EF}\text{-S}_{\widetilde{\mathsf{R}}}(N, A, M)$

 1. $\Sigma \leftarrow 0^n$
 2. $(M[1], \dots, M[m]) \xleftarrow{n} M$
 3. $\ell \leftarrow \lceil m/2 \rceil$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $C[2i-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{f}\rangle}(M[2i-1]) \oplus M[2i]$
 6.     $C[2i] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{s}\rangle}(C[2i-1]) \oplus M[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8. if $m$ is even
 9.     $Z \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(M[m-1])$
10.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
11.     $C[m-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{s}\rangle}(\underline{C[m]}) \oplus M[m-1]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(\widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(0^n)) \oplus M[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. if $m$ is even and $|M[m]| \neq n$
17.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{a}_1\rangle}(\Sigma)$
18. if $m$ is even and $|M[m]| = n$
19.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{a}_2\rangle}(\Sigma)$
20. if $m$ is odd and $|M[m]| \neq n$
21.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{b}_1\rangle}(\Sigma)$
22. if $m$ is odd and $|M[m]| = n$
23.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{b}_2\rangle}(\Sigma)$
24. $C \leftarrow (C[1], \dots, C[m])$
25. return $(C, TE)$
```

```
Algorithm $\mathbb{DF}\text{-S}_{\widetilde{\mathsf{R}}}(N, A, C)$

 1. $\Sigma \leftarrow 0^n$
 2. $(C[1], \dots, C[m]) \xleftarrow{n} C$
 3. $\ell \leftarrow \lceil m/2 \rceil$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $M[2i-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{s}\rangle}(C[2i-1]) \oplus C[2i]$
 6.     $M[2i] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{f}\rangle}(M[2i-1]) \oplus C[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8. if $m$ is even
 9.     $M[m-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{s}\rangle}(\underline{C[m]}) \oplus C[m-1]$
10.     $Z \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(M[m-1])$
11.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(\widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(0^n)) \oplus C[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. if $m$ is even and $|C[m]| \neq n$
17.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{a}_1\rangle}(\Sigma)$
18. if $m$ is even and $|C[m]| = n$
19.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{a}_2\rangle}(\Sigma)$
20. if $m$ is odd and $|C[m]| \neq n$
21.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{b}_1\rangle}(\Sigma)$
22. if $m$ is odd and $|C[m]| = n$
23.     then $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{b}_2\rangle}(\Sigma)$
24. $M \leftarrow (M[1], \dots, M[m])$
25. return $(M, TE)$
```

Fig. 9. The encryption and decryption algorithms of $\mathbb{OTRS}[\tau]$ using a tweakable n-bit URF $\widetilde{\mathsf{R}}$.

Moreover, for any AUTH-adversary $\mathcal{A}$ using q encryption queries and q_v decryption queries,

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTRS}[\tau]}(\mathcal{A}) \leq \frac{2q_v}{2^n} + \frac{q_v}{2^\tau}.$$

Second step: analysis of TBC. In Fig. 10 we define a TBC, $\widetilde{G}'[\mathsf{P}]^{\langle A,N,i,\omega\rangle}(X)$, where $(A, N, i, \omega) \in \mathcal{T}$ denotes a tweak and X denotes an input. It uses an n-bit URP, P. For tweaks that do not appear in Fig. 10, we let them as undefined. Let $\widetilde{\mathsf{R}}$ be a tweakable URF compatible with $\widetilde{G}'[\mathsf{P}]$. Then, in the same manner to Proposition 1 we have the following proposition.

Proposition 2. *If* $\mathbb{EF}\text{-S}_{\widetilde{\mathsf{R}}}$ *(*$\mathbb{DF}\text{-S}_{\widetilde{\mathsf{R}}}$*) uses* $\widetilde{G}'[\mathsf{P}]$ *instead of* $\widetilde{\mathsf{R}}$*, we obtain* EF-S$_{\mathsf{P}}$ *(*DF-S$_{\mathsf{P}}$*).*

We remark that $\widetilde{G}'[\mathsf{P}]$ here does not perform GF doublings in a sequential manner, and performs AF-S$_{\mathsf{P}}$ for every input, however, this does not cause a problem for simulation purpose. We then prove that $\widetilde{G}'[\mathsf{P}]$ is a secure tweakable URF, shown by the following lemma. The proof is in Section C.4.

Lemma 3. *For adversary $\mathcal{A}$ accessing $\widetilde{G}'[\mathsf{P}]$ using q queries, we write the j-th query of $\mathcal{A}$ as $(X_j, A_j, N_j, i_j, \omega_j)$ and let σ be the total blocks of unique ADs, defined as $\sum_{j=j[1],\dots,j[J]} |A_j|_n$, where $A_{j[h]}$ is the first representative element in the h-th equivalent class (i.e. $A_{j[h]} \neq A_k$ for all $k < j[h]$), and J denotes the number of all equivalent*

```
Algorithm $\widetilde{G}'[\mathsf{P}]^{\langle A,N,i,\omega\rangle}(X)$

 1. if $A \neq \varepsilon$ then $TA \leftarrow$ AF-S$_\mathsf{P}(A)$
 2. else $TA \leftarrow 0^n$
 3. $L \leftarrow \mathsf{P}(\underline{N}) \oplus TA$
 4. $Y \leftarrow \mathsf{P}(g(i,\omega,L) \oplus X)$
 5. return $Y$
```

```
Function $g(i,\omega,L)$

 1. $L \leftarrow 2L$
 2.    Switch $\omega$
 3.    Case f : $\Delta \leftarrow 2^{i-1}L$
 4.    Case s : $\Delta \leftarrow 2^{i-1}3L$
 5.    Case a$_1$ : $\Delta \leftarrow 2^{i-1}3^3L$
 6.    Case a$_2$ : $\Delta \leftarrow 2^{i-1}3^17L$
 7.    Case b$_1$ : $\Delta \leftarrow 2^{i-1}3^2L$
 8.    Case b$_2$ : $\Delta \leftarrow 2^{i-1}7L$
 9. return $\Delta$
```

Fig. 10. Tweakable permutation implicitly used by OTRS[P,τ]. AF-S is described in Fig. 7.

classes. Note that $\sigma \le \sum_j |A_j|_n$ holds. Then we have

$$\mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}'[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{A}) \le \frac{(2q+\sigma)^2+q^2+\sigma^2}{2^n},$$

where $\widetilde{\mathsf{R}}$ is a tweakable URF compatible with $\widetilde{G}'[\mathsf{P}]$.

Third step: deriving bounds. In a similar manner to the proof of Theorem 1, we introduce adversary $\mathcal{B}$ against $\widetilde{G}'[\mathsf{P}]$ with $\sigma_M + q$ queries and σ_A total blocks of unique ADs, and derive the bound as

$$\begin{aligned}
\mathtt{Adv}^{\mathtt{priv}}_{\mathrm{OTRS}[\mathsf{P},\tau]}(\mathcal{A}) &= \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathrm{OTRS}[\mathsf{P},\tau],\$}(\mathcal{A}) && (26)\\
&\le \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathrm{OTRS}[\mathsf{P},\tau],\mathbb{OTRS}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathbb{OTRS}[\tau],\$}(\mathcal{A}) && (27)\\
&\le \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}'[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{B}) && (28)\\
&\le \frac{(2(q+\sigma_M)+\sigma_A)^2+(q+\sigma_M)^2+\sigma_A^2}{2^n} && (29)\\
&\le \frac{5\sigma^2_{\mathtt{priv}}}{2^n}, && (30)
\end{aligned}$$

where the second inequality follows from Theorem 6, and the third follows from Lemma 3. For proving AUTH bound, we similarly introduce $\mathcal{B}$ against $\widetilde{G}'[\mathsf{P}]$ with $\sigma_M + \sigma_{C'} + q + q_v$ queries and $\sigma_A + \sigma_{A'}$ total blocks of unique ADs, and derive the bound as

$$\begin{aligned}
\mathtt{Adv}^{\mathtt{auth}}_{\mathrm{OTRS}[\mathsf{P},\tau]}(\mathcal{A}) &\le \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathrm{OTRS}[\mathsf{P},\tau],\mathbb{OTRS}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTRS}[\tau]}(\mathcal{A}) && (31)\\
&\le \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}'[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{B}) + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (32)\\
&\le \frac{(2(q+\sigma_M+q_v+\sigma_{C'})+\sigma_A+\sigma_{A'})^2+(q+\sigma_M+q_v+\sigma_{C'})^2+(\sigma_A+\sigma_{A'})^2}{2^n} + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (33)\\
&\le \frac{5\sigma^2_{\mathtt{auth}}}{2^n} + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (34)\\
&\le \frac{7\sigma^2_{\mathtt{auth}}}{2^n} + \frac{q_v}{2^\tau}, && (35)
\end{aligned}$$

where the second inequality follows from Theorem 6, and the third follows from Lemma 3. This concludes the proof.

C.4 Proof of Lemma 3

We first consider $\widetilde{G}'[\mathsf{R}]$, a function obtained by substituting P in Fig. 10 with an n-bit URF, R. Then we decompose $\widetilde{G}'[\mathsf{R}]$ into a family of smaller functions written as $\mathbf{Q} = \{\mathbf{Q}_i\}_{i=1,\dots,10}$[4]. Let $\mathtt{Rnd} \in \{0,1\}^n$ be an

[4] The decomposition of OMAC here is slight different from the original proof [28], in that we explicitly separate the case when the first input block is all-zero from other cases. We employ this separation to reduce the following proof complexity, but the original decomposition of [28] would work as well.

```
Algorithm $\mathbb{G}[\mathbf{Q}]^{\langle A,N,i,\omega\rangle}(X)$

 1. $TA \leftarrow \mathrm{CMAC}'[\mathbf{Q}](A)$
 2. $L \leftarrow TA \oplus \mathbf{Q}_1(\underline{N})$
 3. $S \leftarrow g(i,\omega,L) \oplus X$
 4. $Y \leftarrow \mathbf{Q}_{10}^{(i,\omega)}(S)$
 5. return $Y$
```

```
Algorithm $\mathrm{CMAC}'[\mathbf{Q}](X)$

 1. if $X = \varepsilon$ then $Y \leftarrow 0^n$
 2. else
 3.   $(X[1], X[2], \ldots, X[x]) \xleftarrow{n} X$
 4.   if $x = 1$ then
 5.     if $|X[x]| \neq n$ then $Y \leftarrow \mathbf{Q}_5(\underline{X[x]})$
 6.     else $Y \leftarrow \mathbf{Q}_6(\underline{X[x]})$
 7.   if $x = 2$ then
 8.     if $X[1] = 0^n$ then
 9.       if $|X[x]| \neq n$ then $Y \leftarrow \mathbf{Q}_8(\underline{X[x]})$
10.       else $Y \leftarrow \mathbf{Q}_9(\underline{X[x]})$
11.     else $S \leftarrow \mathbf{Q}_1(\overline{X[1]})$                // $X[1] \neq 0^n$
12.     if $|X[x]| \neq n$ then $Y \leftarrow \mathbf{Q}_3(S \oplus \underline{X[x]})$
13.     else $Y \leftarrow \mathbf{Q}_4(S \oplus \underline{X[x]})$
14.   if $x > 2$ then
15.     if $X[1] = 0^n$ then $S \leftarrow \mathbf{Q}_7(X[2])$, $I \leftarrow 3$
16.     else $S \leftarrow \mathbf{Q}_1(X[1])$, $I \leftarrow 2$        // $X[1] \neq 0^n$
17.     for $i = I$ to $x - 1$
18.       do $S \leftarrow \mathbf{Q}_2(S \oplus X[i])$              // when $x \geq I + 1$
19.     if $|X[x]| \neq n$ then $Y \leftarrow \mathbf{Q}_3(\underline{X[x]})$
20.     else $Y \leftarrow \mathbf{Q}_4(\underline{X[x]})$
21. return $Y$
```

Fig. 11. An equivalent function to $\widetilde{G}'[\mathsf{R}]$, $\mathbb{G}[\mathbf{Q}]$.

independent and uniform variable, then we define

$$\mathbf{Q}_1(x) \stackrel{\text{def}}{=} \mathsf{R}(x) \oplus \mathtt{Rnd}, \qquad \mathbf{Q}_2(x) = \mathsf{R}(x \oplus \mathtt{Rnd}) \oplus \mathtt{Rnd} \tag{36}$$

$$\mathbf{Q}_3(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus \mathtt{Rnd} \oplus 2U), \qquad \mathbf{Q}_4(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus \mathtt{Rnd} \oplus 4U) \tag{37}$$

$$\mathbf{Q}_5(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus 2U), \qquad \mathbf{Q}_6(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus 4U) \tag{38}$$

$$\mathbf{Q}_7(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus U) \oplus \mathtt{Rnd}, \qquad \mathbf{Q}_8(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus U \oplus 2U) \tag{39}$$

$$\mathbf{Q}_9(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus U \oplus 4U), \qquad \mathbf{Q}_{10}^{(i,\omega)}(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus g(i,\omega,\mathtt{Rnd})) \tag{40}$$

where $U = \mathsf{R}(0^n)$. All functions have n-bit input and output, except $\mathbf{Q}_1$ and $\mathbf{Q}_{10}$. Here $\mathbf{Q}_1$ has input domain $\{0,1\}^n \setminus \{0^n\}$, and $\mathbf{Q}_{10}^{(*,*)}(*)$ has input domain $(\mathbb{N} \times \Omega) \times \{0,1\}^n$. Both have n-bit output. Function $g(*,*,*)$ is defined as Fig 10 and $g(i,\omega,x)$ is a multiplication over $\mathrm{GF}(2^n)$ with x and a constant depending on (i,ω), written as $c_{(i,\omega)} \in \mathrm{GF}(2^n)$. Therefore we may represent $g(i,\omega,x)$ by $c_{(i,\omega)} \cdot x$. For example $c_{(i,\mathtt{s})} = 2(2^{i-1}3) = 2^i 3$. In the rest of the proof we assume i in $g(i,\omega,x)$ is no more than $2^{n/2}$, which is needed for security. From Proposition 5 of [44] we have

Proposition 3. *In Fig 10, for any (i,ω) with $i = 1,\ldots,2^{n/2}$, $c_{(i,\omega)} \neq c_{(i',\omega')}$ for any $(i,\omega) \neq (i',\omega')$, and $c_{(i,\omega)}$ is not an identity element nor zero element.*

Using $\mathbf{Q}$ we build a function $\mathbb{G}[\mathbf{Q}]$ which is equivalent to $\widetilde{G}'[\mathsf{R}]$, shown by Fig. 20. It uses $\mathrm{CMAC}'[\mathbf{Q}]$, which is equivalent to the original CMAC [28] (instantiated by URF) except that the empty input produces 0^n output and the opposite GF coefficients for last-block mask, i.e. 2 (4) for the case the last input block is partial (full). Note that L in the algorithm of $\mathbb{G}[\mathbf{Q}]$ contains $\mathtt{Rnd}$ from the output of $\mathbf{Q}_1$, but has no effect on the final output Y, as $\mathtt{Rnd}$ is canceled out by input mask of $\mathbf{Q}_{10}^{(i,\omega)}$ thanks to the XOR-linearity of g. We may abbreviate $\mathbf{Q}_{10}^{(i,\omega)}(x)$ as $\mathbf{Q}_{10}(x)$ if underlying (i,ω) is obvious. We then show that $\mathbf{Q}$ is indistinguishable from a set of URFs.

Lemma 4. *Let $\widetilde{\mathbf{Q}} = \{\widetilde{\mathbf{Q}}_i\}_{i=1,\ldots,10}$ be the set of functions, where $\widetilde{\mathbf{Q}}_i$ is an independent URF compatible with $\mathbf{Q}_i$. We also consider $\mathbf{Q}$ and $\widetilde{\mathbf{Q}}$ as tweakable functions accepting tweak $t \in \{1,\ldots,10\}$. Then we have*

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbf{Q},\widetilde{\mathbf{Q}}}(\mathcal{A}) \leq \frac{q^2}{2^n},$$

for adversary $\mathcal{A}$ using q queries, where each query consists of tweak t and the corresponding input to $\mathbf{Q}_t$ or $\widetilde{\mathbf{Q}}_t$.

Proof. Let Δ_t and ∇_t be the input and output masks for $\mathbf{Q}_t$, defined as

$$\begin{aligned}&\Delta_1 = 0^n,\ \Delta_2 = \mathtt{Rnd},\ \Delta_3 = \mathtt{Rnd} \oplus 2U,\ \Delta_4 = \mathtt{Rnd} \oplus 4U,\ \Delta_5 = 2U,\\ &\Delta_6 = 4U,\ \Delta_7 = U,\ \Delta_8 = U \oplus 2U,\ \Delta_9 = U \oplus 4U,\ \Delta_{10}^{(i,\omega)} = g(i, \omega, \mathtt{Rnd}),\end{aligned} \tag{41}$$

and

$$\nabla_1 = \mathtt{Rnd},\ \nabla_2 = \mathtt{Rnd},\ \nabla_7 = \mathtt{Rnd}, \tag{42}$$

and other ∇_is are 0^n, including $\nabla_{10}^{(i,\omega)} = 0^n$ for any (i, ω), where $U = \mathsf{R}(0^n)$ and $\mathtt{Rnd}$ are independently random. Then $\mathbf{Q}_t(x) = \mathsf{R}(\Delta_t \oplus x) \oplus \nabla_t$ for $t \le 9$, and $\mathbf{Q}_{10}^{(i,\omega)}(x) = \mathsf{R}(\Delta_{10}^{(i,\omega)} \oplus x) \oplus \nabla_{10}^{(i,\omega)}$. We introduce Fig. 21 which defines two games, Game$\mathbf{Q}$ and Game$\widetilde{\mathbf{Q}}$. A query is (t, X, i, ω) and for any $t \neq 10$, i and ω are assumed to be fixed default values to avoid pointless queries. In Fig. 21 masks are written as functions taking all arguments, e.g. $\Delta_2(U, \mathtt{Rnd}, i, \omega) = \Delta_2 = \mathtt{Rnd}$ and $\Delta_{10}(U, \mathtt{Rnd}, i, \omega) = \Delta_{10}^{(i,\omega)} = g(i, \omega, \mathtt{Rnd})$. We observe that Game$\mathbf{Q}$ and Game$\widetilde{\mathbf{Q}}$ precisely simulate $\mathbf{Q}$ and $\widetilde{\mathbf{Q}}$. For Game$\widetilde{\mathbf{Q}}$ this is obvious, since the output of Game$\widetilde{\mathbf{Q}}$ is always independent and uniformly random. For Game$\mathbf{Q}$ the generation procedure of Y and YE in Game$\mathbf{Q}$ is opposite to that of $\mathbf{Q}$; in Game$\mathbf{Q}$, if XE is a new value, Y is uniformly sampled and then $Y = YE \oplus \nabla_t$ is determined, while $\mathbf{Q}$ first determines YE randomly and computes $Y = YE \oplus \nabla_t$. However, both yield the identical marginal distribution of (Y, YE). If XE has a collision, Game$\mathbf{Q}$ determines YE from the set of previously sampled values, and Y is determined as $Y \leftarrow YE \oplus \nabla_t$. Games of Fig. 21 define the flag *bad* to set (in line 13) when two inputs to ρ after the input maskings collide. Then, following the Game-Playing technique [15], both games are identical until *bad* gets set to $\mathtt{true}$, thus we have

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbf{Q},\widetilde{\mathbf{Q}}}(\mathcal{A}) \le \Pr[\mathcal{A}^{\mathrm{Game}\mathbf{Q}} \Rightarrow 1] - \Pr[\mathcal{A}^{\mathrm{Game}\widetilde{\mathbf{Q}}} \Rightarrow 1] \le \Pr[\mathcal{A}^{\mathrm{Game}\widetilde{\mathbf{Q}}} \text{ sets } bad]. \tag{43}$$

Hence what we need is to bound the last probability, which is derived from the pairwise collision probability of input masks. We see that

$$\max_{t\in\{2,\dots,9\}, d\in\{0,1\}^n} \Pr_{U,\mathtt{Rnd}}[\Delta_t = d] \le \frac{1}{2^n} \tag{44}$$

$$\max_{(i,\omega)\in\mathbb{N}\times\Omega, d\in\{0,1\}^n} \Pr_{U,\mathtt{Rnd}}[\Delta_{10}^{(i,\omega)} = d] \le \frac{1}{2^n} \tag{45}$$

$$\max_{t,t'\in\{2,\dots,9\}, t\neq t', d\in\{0,1\}^n} \Pr_{U,\mathtt{Rnd}}[\Delta_t \oplus \Delta_{t'} = d] \le \frac{1}{2^n} \tag{46}$$

$$\max_{t\in\{2,\dots,9\}, (i,\omega)\in\mathbb{N}\times\Omega, d\in\{0,1\}^n} \Pr_{U,\mathtt{Rnd}}[\Delta_t \oplus \Delta_{10}^{(i,\omega)} = d] \le \frac{1}{2^n} \tag{47}$$

$$\max_{(i,\omega),(i',\omega')\in\mathbb{N}\times\Omega, (i,\omega)\neq(i',\omega'), d\in\{0,1\}^n} \Pr_{U,\mathtt{Rnd}}[\Delta_{10}^{(i,\omega)} \oplus \Delta_{10}^{(i',\omega')} = d] \le \frac{1}{2^n} \tag{48}$$

where the probabilities are defined by U and $\mathtt{Rnd}$, which are independent and uniform over $\{0,1\}^n$ (as U is a sole output of URF involved in the event). (87) and (45) denote the collision probability of the form $[\Delta_1 \oplus X_i = \Delta_j \oplus X_k] \equiv [\Delta_j = X_i \oplus X_k]$ for $j \neq 1$, and (46) to (88) denote the other collision cases. Here (47) and (88) follow[5] from Proposition 3. These equations show that any collision occurs at most with probability $1/2^n$, and since q queries in the game yield at most $q+1$ accesses to ρ, the bound is derived as $\binom{q+1}{2}/2^n \le q^2/2^n$. This proves Lemma 7. □

We consider $\mathbb{G}[\widetilde{\mathbf{Q}}] : \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$, which is obtained by substituting $\mathbf{Q}_i$ with $\widetilde{\mathbf{Q}}_i$ in Fig. 20, for all $i = 1, \dots, 10$. In $\mathbb{G}[\widetilde{\mathbf{Q}}]$ the internal CMAC-like function is written as CMAC$'[\widetilde{\mathbf{Q}}]$. We then need to evaluate the indistinguishability between $\mathbb{G}[\widetilde{\mathbf{Q}}]$ and a URF compatible with $\mathbb{G}[\widetilde{\mathbf{Q}}]$, $\widetilde{\mathsf{R}}$. For doing this we consider another decomposition of $\mathbb{G}[\widetilde{\mathbf{Q}}]$. Let f_{NX} be a function using $\widetilde{\mathbf{Q}}_1$ and $\widetilde{\mathbf{Q}}_{10}$, such that

$$f_{NX}(N, i, \omega, X) \overset{\mathrm{def}}{=} \widetilde{\mathbf{Q}}_{10}^{(i,\omega)}(g(i, \omega, \widetilde{\mathbf{Q}}_1(N)) \oplus X). \tag{49}$$

[5] In more detail, (47) with $t = 1$ ($t = 2$) needs $c_{(i,\omega)} \cdot \mathtt{Rnd}$ ($\mathtt{Rnd} \oplus c_{(i,\omega)} \cdot \mathtt{Rnd}$) to be uniform, which holds if $c_{(i,\omega)}$ is not zero (identity element).

```
Initialization
00    $U \leftarrow \rho(0^n) \xleftarrow{\$} \{0,1\}^n$
01    $\mathtt{Rnd} \xleftarrow{\$} \{0,1\}^n$
On query $(t, X, i, \omega) \in \{1, \dots, 10\} \times \{0,1\}^n \times \mathbb{N} \times \Omega$          // $(i, \omega)$ works for $t = 10$
10    $XE \leftarrow \Delta_t(U, \mathtt{Rnd}, i, \omega) \oplus X$                                    // $X \neq 0^n$ when $t = 1$
11    $Y \xleftarrow{\$} \{0,1\}^n$
12    $YE \leftarrow Y \oplus \nabla_t(U, \mathtt{Rnd}, i, \omega)$
13    if $XE \in \mathrm{Dom}(\rho)$ then $\mathtt{bad} \leftarrow \mathtt{true}$, $\boxed{YE \leftarrow \rho(XE),\ Y \leftarrow YE \oplus \nabla_t(U, \mathtt{Rnd}, i, \omega)}$
14    else $\rho(XE) \leftarrow YE$
15    return $Y$
```

Fig. 12. Game$\mathbf{Q}$ contains the boxed arguments, while Game$\widetilde{\mathbf{Q}}$ does not.

We also define f_A as CMAC$'[\widetilde{\mathbf{Q}}]$. Then we have

$$\mathbb{G}[\widetilde{\mathbf{Q}}]^{\langle A,N,i,\omega\rangle}(X) = f_{NX}(N, i, \omega, X \oplus g(i, \omega, f_A(A))) \tag{50}$$

due to the linearity of g. We then claim that a pair of functions, (f_{NX}, f_A), is hard to distinguish from a pair of independent compatible URFs, respectively denoted by R_{NX} and R_A, where R_A is a VIL-URF that can accept an empty string (unlike a normal URF) and produces 0^n.

The proof is basically the same as the proof of OMAC [28] (more precisely the analysis of MOMAC function in [28]), however direct application of [28]'s proof is not possible due to the existence of f_{NX} which shares $\widetilde{\mathbf{Q}}_1$ with f_A. Fig. 13 shows how (f_{NX}, f_A) is queried. Throughout queries we assume that the oracle maintains the following lists. Let $\mathcal{L}^1_{i,\omega}$ be the list of n-bit (non-tweak) input blocks to $\widetilde{\mathbf{Q}}_{10}$ in f_{NX}, defined as $g(i, \omega, \widetilde{\mathbf{Q}}_1(N)) \oplus X$. Similarly, let $\mathcal{L}^2_\kappa$ be the list of input blocks to $\widetilde{\mathbf{Q}}_\kappa$ in f_A, for $\kappa \in \{3,4,5,6,8,9\}$, which denotes the index set of finalization function. Let ε_1 be the event that a collision of two values in the same list $\mathcal{L}^1_{i,\omega}$, for some queried (i, ω), and let ε_2 be the event that a collision of two values in the same list $\mathcal{L}^2_\kappa$, for some $\kappa \in \{3,4,5,6,8,9\}$. Consider adversary $\mathcal{A}$ accessing pair (f_{NX}, f_A), where a query specifies which function to access combined with an input to the specified function. Let q be the number of total queries and σ be the total input blocks to f_A. We naturally assume there is no duplicate queries. We observe that the finalization functions, i.e. $\widetilde{\mathbf{Q}}_i$ to generate the output Y in Fig. 13, used by f_{NX} and f_A are different. Therefore (f_{NX}, f_A) can be seen as a pair of Carter-Wegman MACs using independent finalizations, or more generally a Carter-Wegman MAC combining f_{NX} and f_A with one-bit tweak for additional input to specify which one is used. This implies that

$$\begin{aligned}\mathtt{Adv}^{\mathtt{cpa}}_{(f_{NX},f_A),(\mathsf{R}_{NX},\mathsf{R}_A)}(\mathcal{A}) &\le \Pr[\varepsilon_1 \cup \varepsilon_2], \text{ and hence} &(51)\\ &\le \Pr[\varepsilon_1] + \Pr[\varepsilon_2] &(52)\end{aligned}$$

holds true, where probabilities are defined by $\mathcal{A}$ and (f_{NX}, f_A), and the probability of the right hand side (rhs) of (51) (and thus (52)) for adaptive adversaries can be bounded by that of the non-adaptive adversaries, which is obtained by, e.g., applying [Theorem 2 and Corollary 1 of [37]] to a Carter-Wegman MAC function combining f_{NX} and f_A. Hence we can focus on the two probabilities of rhs of (52) for q inputs given by a non-adaptive adversary.

We first analyze $\Pr[\varepsilon_2]$. Let f_A^{-1} be the function defined as f_A without the finalization, i.e. whose output is obtained by substituting $\widetilde{\mathbf{Q}}_\kappa$ of f_A with identity function for all $\kappa \in \{3,4,5,6,8,9\}$. Let $h(X) : \{0,1\}^* \to \{3,4,5,6,8,9\}$ be the function that maps an input to f_A or f_A^{-1} to the index of the corresponding finalization function (e.g. $h(0^n \| 1^n) = 9$). Then we have

$$\Pr[\varepsilon_2] = \max_{\substack{X_1,\dots,X_q, |X_i|_n = m_i, \Sigma_i m_i = \sigma,\\ X_i \neq X_j \text{ for all } i<j}} \sum_{i \neq j, h(X_i) = h(X_j)} \Pr[f_A^{-1}(X_i) = f_A^{-1}(X_j)]. \tag{53}$$

Let X and X' be two distinct inputs with $|X|_n = m$, $|X'|_n = m'$, and $h(X) = h(X') = \kappa$. Without loss of generality we assume $m \ge m'$. Then $\Pr[f_A^{-1}(X_i) = f_A^{-1}(X_j)]$ is zero when $\kappa = 5$ or 6 (which implies $m = m' = 1$), and when $\kappa = 8$ or 9 (which implies $m = m' = 2$ with $X[1] = X'[1] = 0^n$, $X[2] \neq X'[2]$) since f_A^{-1} output is $\underline{X[2]}$. We then consider the remaining cases having $\kappa = 3$ or 4 (which include $m, m' = 2$ and $X[1], X'[1] \neq 0^n$,

```
Initialization
Set $\mathcal{L}^1_{i,\omega}$ as $\emptyset$ for all $(i,\omega)$, $\mathcal{L}^2_\kappa$ as $\emptyset$ for all $\kappa \in \{3,4,5,6,8,9\}$
When $f_{NX}$ is queried with $(N, X, i, \omega)$

 1. $S \leftarrow g(i,\omega,\widetilde{\mathbf{Q}}_1(N)) \oplus X$, update list $\mathcal{L}^1_{i,\omega} \leftarrow S$
 2. $Y \leftarrow \widetilde{\mathbf{Q}}^{(i,\omega)}_{10}(S)$
 3. return $Y$

When $f_A$ is queried with $X$

 1. if $X = \varepsilon$ then $Y \leftarrow 0^n$
 2. else
 3.    $(X[1], X[2], \ldots, X[x]) \xleftarrow{n} X$
 4.    if $x = 1$ then
 5.       if $|X[x]| \neq n$ then $Y \leftarrow \widetilde{\mathbf{Q}}_5(\underline{X[x]})$, update list $\mathcal{L}^2_5 \leftarrow \underline{X[x]}$
 6.       else $Y \leftarrow \widetilde{\mathbf{Q}}_6(\underline{X[x]})$, update list $\mathcal{L}^2_6 \leftarrow \underline{X[x]}$
 7.    if $x = 2$ then
 8.       if $X[1] = 0^n$ then
 9.          if $|X[x]| \neq n$ then $Y \leftarrow \widetilde{\mathbf{Q}}_8(\underline{X[x]})$, update list $\mathcal{L}^2_8 \leftarrow \underline{X[x]}$
10.          else $Y \leftarrow \widetilde{\mathbf{Q}}_9(\underline{X[x]})$, update list $\mathcal{L}^2_9 \leftarrow \underline{X[x]}$
11.       else $S \leftarrow \widetilde{\mathbf{Q}}_1(X[1])$                                  // $X[1] \neq 0^n$
12.          if $|X[x]| \neq n$ then $Y \leftarrow \widetilde{\mathbf{Q}}_3(S \oplus \underline{X[x]})$, update list $\mathcal{L}^2_3 \leftarrow S \oplus \underline{X[x]}$
13.          else $Y \leftarrow \widetilde{\mathbf{Q}}_4(S \oplus \underline{X[x]})$, update list $\mathcal{L}^2_4 \leftarrow S \oplus \underline{X[x]}$
14.    if $x > 2$ then
15.       if $X[1] = 0^n$ then $S \leftarrow \widetilde{\mathbf{Q}}_7(X[2])$, $I \leftarrow 3$
16.       else $S \leftarrow \widetilde{\mathbf{Q}}_1(X[1])$, $I \leftarrow 2$                      // $X[1] \neq 0^n$
17.       for $i = I$ to $x - 1$
18.          do $S \leftarrow \widetilde{\mathbf{Q}}_2(S \oplus X[i])$                           // when $x \geq I + 1$
19.       if $|X[x]| \neq n$ then $Y \leftarrow \widetilde{\mathbf{Q}}_3(S \oplus \underline{X[x]})$, update list $\mathcal{L}^2_3 \leftarrow S \oplus \underline{X[x]}$
20.       else $Y \leftarrow \widetilde{\mathbf{Q}}_4(S \oplus \underline{X[x]})$, update list $\mathcal{L}^2_4 \leftarrow S \oplus \underline{X[x]}$
21. return $Y$
```

Fig. 13. f_{NX} and f_A, with finalization input lists.

and the cases with at least one of m, m' is more than 2). Let CBC_F be the standard CBC-MAC function using n-bit function F as the internal blockcipher, working for any input in $(\{0,1\}^n)^i$ for $i = 1, 2, \ldots$. We now introduce the following lemma of Black and Rogaway [19].

Lemma 5. *([19]) For n-bit URF* R *and two distinct inputs to* $\mathrm{CBC}_{\mathsf{R}}$*,* X *and* X'*,* $|X| = mn$ *and* $|X'| = m'n$*, we have*

$$\Pr[\mathrm{CBC}_{\mathsf{R}}(X) = \mathrm{CBC}_{\mathsf{R}}(X')] \leq \frac{m \cdot m'}{2^n} + \frac{\max\{m, m'\}}{2^n}.$$

Note that the lemma also implies $\Pr[\mathrm{CBC}_{\mathsf{R}}(X) = c] \leq 2(m+1)/2^n$ for any $c \in \{0,1\}^n$ (by applying CBC for both $X \| 0^n$ and c), and $\Pr[\mathrm{CBC}_{\mathsf{R}}(X) \oplus \mathrm{CBC}_{\mathsf{R}}(X') = c] \leq (m+1) \cdot (m'+1)/2^n + \max\{m+1, m'+1\}/2^n$ for any $c \in \{0,1\}^n$ (by applying CBC for both $X \| c$ and $X' \| 0^n$). The remaining cases with $\kappa = 3$ or 4 can be further divided into the following sub-cases. Recall that we assumed $h(X) = h(X')$, thus both X and X' have either partial last blocks or full last blocks.

Case 1: $m = m' = 2$, $X[1], X'[1] \neq 0^n$**.** Then $f_A^{-1}(X) = \widetilde{\mathbf{Q}}_1(X[1]) \oplus \underline{X[2]}$ and $f_A^{-1}(X') = \widetilde{\mathbf{Q}}_1(X'[1]) \oplus \underline{X'[2]}$, hence $\Pr[f_A^{-1}(X_i) = f_A^{-1}(X_j)]$ is at most $1/2^n$.

Case 2: $m > 2$, $m' = 2$, $X[1], X'[1] \neq 0^n$**.** Then we have

$$\Pr[f_A^{-1}(X_i) = f_A^{-1}(X_j)] = \Pr[\mathrm{CBC}_{\widetilde{\mathbf{Q}}_2}(V, X[3], \ldots, X[m-1]) \oplus \underline{X[m]} = V'] \leq \frac{2(m-1)}{2^n}, \quad (54)$$

where $V = \widetilde{\mathbf{Q}}_1(X[1]) \oplus X[2]$ and $V' = \widetilde{\mathbf{Q}}_1(X'[1]) \oplus \underline{X'[2]}$, and the inequality follows from Lemma 5.

Case 3: $m = 3$, $m' = 2$, $X[1] = 0^n$, $X'[1] \neq 0^n$**.** Then $f_A^{-1}(X) = \widetilde{\mathbf{Q}}_7(X[2]) \oplus \underline{X[3]}$ and $f_A^{-1}(X') = \widetilde{\mathbf{Q}}_1(X'[1]) \oplus$

$\underline{X'[2]}$, hence the probability is $1/2^n$.

Case 4: $m > 3$**,** $m' = 2$**,** $X[1] = 0^n$**,** $X'[1] \neq 0^n$**.** The same analysis as Case 2 holds, with $V = \widetilde{\mathbf{Q}}_7(X[2]) \oplus X[3]$. The probability is bounded by $2(m-1)/2^n$.

Case 5: $m > 2$**,** $m' > 2$**,** $X[1], X'[1] \neq 0^n$**.** Let $V = \widetilde{\mathbf{Q}}_1(X[1]) \oplus X[2]$ and $V' = \widetilde{\mathbf{Q}}_1(X'[1]) \oplus X'[2]$. When $(X[1], X[2]) \neq (X'[1], X'[2])$, the probability of $V = V'$ is at most $1/2^n$, and given $V \neq V'$ we have two distinct inputs to $\mathrm{CBC}_{\widetilde{\mathbf{Q}}_2}$. Thus the conditional collision probability is

$$\begin{aligned}
&\Pr[\mathrm{CBC}_{\widetilde{\mathbf{Q}}_2}(V, X[3], \ldots, X[m-1]) \oplus \mathrm{CBC}_{\widetilde{\mathbf{Q}}_2}(V', X'[3], \ldots, X'[m'-1]) = \underline{X[m]} \oplus \underline{X'[m']} | V \neq V'] \\
&\leq \frac{(m-1)(m'-1)}{2^n} + \frac{m-1}{2^n} \qquad (55)
\end{aligned}$$

from Lemma 5 and the assumption $m \geq m'$. Therefore, by adding the V-collision probability, the bound is obtained as $(m-1)(m'-1)/2^n + (m'-1)/2^n + 1/2^n$, which is at most $mm'/2^n + m/2^n$. When $(X[1], X[2]) = (X'[1], X'[2])$, we have $V = V'$ and two distinct inputs to $\mathrm{CBC}_{\widetilde{\mathbf{Q}}_2}$, hence the same bound applies.

Case 6: $m > 2$**,** $m' > 2$**,** $X[1] = 0^n, X'[1] \neq 0^n$ **or** $X[1] \neq 0^n, X'[1] = 0^n$**.** The same as Case 5, and the bound is $mm'/2^n + m'/2^n$.

Therefore, summarizing all cases we have

$$\begin{aligned}
\Pr[\varepsilon_2] &\leq \max_{\substack{X_1,\ldots,X_q, |X_i|_n = m_i, \Sigma_i |X_i|_n = \sigma, \\ X_i \neq X_j \text{ for all } i<j}} \sum_{i \neq j, h(X_i) = h(X_j)} \Pr[f_A^{-1}(X_i) = f_A^{-1}(X_j)] \qquad (56) \\
&\leq \max_{\substack{X_1,\ldots,X_q, |X_i|_n = m_i, \Sigma_i |X_i|_n = \sigma, \\ X_i \neq X_j \text{ for all } i<j}} \sum_{i \neq j, h(X_i) = h(X_j)} \frac{m_i \cdot m_j}{2^n} + \frac{\max\{m_i, m_j\}}{2^n} \qquad (57) \\
&\leq \frac{\sigma^2}{2^n}, \qquad (58)
\end{aligned}$$

where the last inequality follows from [19].

The analysis of $\Pr[\varepsilon_1]$ is easy. For any two $(N, X, i, \omega) \neq (N', X', i', \omega')$ with $(i, \omega) = (i', \omega')$, the probability of collision (i.e. $g(i, \omega, \widetilde{\mathbf{Q}}_1(N)) \oplus X = g(i', \omega', \widetilde{\mathbf{Q}}_1(N')) \oplus X'$) is at most $1/2^n$, hence

$$\Pr[\varepsilon_1] \leq \binom{q}{2} \cdot \frac{1}{2^n} \leq \frac{q^2}{2^{n+1}}. \qquad (59)$$

Combining (58) and (59), we have

$$\mathtt{Adv}^{\mathtt{cpa}}_{(f_{NX}, f_A), (\mathsf{R}_{NX}, \mathsf{R}_A)}(\mathcal{A}) \leq \frac{0.5q^2 + \sigma^2}{2^n}. \qquad (60)$$

Finally, we consider $\mathbb{F}$, a function compatible with $\mathbb{G}[\mathbf{Q}]$ defined as

$$\mathbb{F}^{\langle A, N, i, \omega \rangle}(X) \stackrel{\text{def}}{=} \mathsf{R}_{NX}(N, i, \omega, X \oplus g(i, \omega, \mathsf{R}_A(A))). \qquad (61)$$

Recall that we assumed $\mathsf{R}_A(\varepsilon) = 0^n$ as f_A. Hence $g(i, \omega, \mathsf{R}_A(\varepsilon))) = 0^n$ holds. Then we focus the collision probability at inputs to R_{NX}. It is simple to observe that

$$\Pr[(N, i, \omega, X \oplus g(i, \omega, \mathsf{R}_A(A))) = (N', i', \omega', X' \oplus g(i', \omega', \mathsf{R}_A(A')))] \leq \frac{1}{2^n} \qquad (62)$$

for any two distinct (A, N, i, ω, X) and $(A', N', i', \omega', X')$, including the cases A and/or A' is empty. This is because we require $(N, i, \omega) = (N', i', \omega')$ to have non-zero probability for the left hand side of (62) and if $A \neq A'$ the sum $X \oplus g(i, \omega, \mathsf{R}_A(A)) \oplus X' \oplus g(i', \omega', \mathsf{R}_A(A'))$, which equals to $c_{(i,\omega)} \cdot (\mathsf{R}_A(A) \oplus \mathsf{R}_A(A')) \oplus X \oplus X'$, is uniform. When $A = A'$ and $X \neq X'$ the collision probability is apparently zero. This implies that

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{F}, \widetilde{\mathsf{R}}}(\mathcal{A}) \leq \binom{q}{2} \cdot \frac{1}{2^n} \leq \frac{q^2}{2^{n+1}} \qquad (63)$$

for any adversary with q queries, irrespective of lengths of A.

Then, for any adversary $\mathcal{A}$ querying $\widetilde{G}'[\mathsf{P}]$ using q queries and total blocks of unique ADs being σ, there exist adversary $\mathcal{B}$ querying $\mathbf{Q}$ (or $\widetilde{\mathbf{Q}}$) with $2q+\sigma$ queries and adversary $\mathcal{C}$ querying (f_{NX}, f_A) (or $(\mathsf{R}_{NX}, \mathsf{R}_A)$) with q queries with σ total blocks to the second function, satisfying

$$\mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}'[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{A}) \le \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{G}[\mathbf{Q}],\mathbb{G}[\widetilde{\mathbf{Q}}]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{G}[\widetilde{\mathbf{Q}}],\mathbb{F}}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{F},\widetilde{\mathsf{R}}}(\mathcal{A}) \tag{64}$$

$$\le \mathtt{Adv}^{\mathtt{cpa}}_{\mathbf{Q},\widetilde{\mathbf{Q}}}(\mathcal{B}) + \mathtt{Adv}^{\mathtt{cpa}}_{(f_{NX},f_A),(\mathsf{R}_{NX},\mathsf{R}_A)}(\mathcal{C}) + \frac{0.5q^2}{2^n} \tag{65}$$

$$\le \frac{(2q+\sigma)^2}{2^n} + \frac{0.5q^2+\sigma^2}{2^n} + \frac{0.5q^2}{2^n} = \frac{(2q+\sigma)^2+q^2+\sigma^2}{2^n}, \tag{66}$$

where the second inequality follows from (63), and the third follows Lemma 7 and (60). This concludes the proof.

C.5 Proof of Theorem 6

The proof is mostly a subset of proof of Theorem 3.

PRIV bound. We observe that any output block of encryption oracle $\mathbb{OTRS}$-$\mathcal{E}_\tau$ contains an output block of $\widetilde{\mathsf{R}}^{\langle A,N,i,\omega\rangle}$, where the tweak (A, N, i, ω) is uniquely used throughout the attack by PRIV-adversary $\mathcal{A}$ whose queries have unique pairs of (A, N). This implies that the output blocks in (C, T) is independent and uniform, thus indistinguishable from those of \$ oracle. PRIV bound being 0 is naturally derived from this observation.

AUTH bound. Following the proof of Theorem 3, we first consider the case $q_v = 1$. Let $\mathcal{A}$ be AUTH-adversary against $\mathbb{OTRS}$ with q encryption queries and a decryption query. Without loss of generality we can assume $\mathcal{A}$ first performs all encryption queries before the decryption query. As well as the proof of Theorem 3 we use the notations (N_i, A_i, M_i), and (C_i, T_i) for $i = 1, \dots, q$, and a decryption query (a forgery attempt) (N', A', C', T') satisfying $(N', A', C') \neq (N_i, A_i, C_i)$ for all $i = 1, \dots, q$. Here $|M_i| = |C_i|$ and $(A_i, N_i) \neq (A_j, N_j)$ for any $1 \le i < j \le q$ by assumption. Let T^* be the true tag value for the forgery attempt, and let TE^* and Σ^* be the corresponding values produced in the forgery attempt using (N', A', C'). The forgery attempt is accepted as valid iff $T^* = T'$, where

$$T^* = \mathtt{msb}_\tau(TE^*), \text{ and } TE^* = \mathtt{lsb}_n(\mathbb{DF}\text{-}\mathrm{S}_{\widetilde{\mathsf{R}}}(N', A', C')). \tag{67}$$

Let $m' = |C'|_n$ and $\ell' = |C'|_{2n}$. Note that TE^* is equal to $\widetilde{\mathsf{R}}^{\langle A',N',\ell',\omega'\rangle}(\Sigma^*)$, where Σ^* is generated as an internal variable of $\mathbb{DF}\text{-}\mathrm{S}_{\widetilde{\mathsf{R}}}(N', A', C')$ for some $\omega' \in \{\mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$ uniquely determined by the length of C'. Application of function $\widetilde{\mathsf{R}}^{\langle A',N',\ell',\omega'\rangle}$ is called a finalization and the tweak (A', N', ℓ', ω') is called a finalization tweak. We let $\mathbf{Z} = \{(N_i, A_i, M_i, C_i, T_i)\}_{i=1,\dots,q}$ be the transcript obtained by encryption queries, and by seeing $\mathbf{Z}$ as a random variable, the forgery probability is obtained as the maximum of $\mathrm{FP}_{\mathbf{z}}$ defined as $\Pr[T' = T^* | \mathbf{Z} = \mathbf{z}]$, for all transcripts. We can then perform a case analysis for (N', A', C'), which is a simplified version of the one provided for the proof of Theorem 3. Specifically we have two cases.

Case 1: $(A', N') \neq (A_i, N_i)$ **for all** $1 \le i \le q$**.**

The finalization tweak is new, hence TE^* is independent and uniformly random. Thus $\mathrm{FP}_{\mathbf{z}} \le 1/2^\tau$.

Case 2: $(A', N') = (A_\alpha, N_\alpha)$**, and** $C' \neq C_\alpha$ **for some** $1 \le \alpha \le q$**.**

The analysis is completely the same as Case 3 and Case 4 in the proof of Theorem 3. This is because analyses given in these cases work irrespective of the values of A, and $\mathbb{OTRS}$ assuming (A, N) as nonce and $\mathbb{OTR}'$ assuming $A = \varepsilon$ is essentially equivalent. Following the Case 3 and Case 4 in the proof of Theorem 3, we have $\mathrm{FP}_{\mathbf{z}} \le 2/2^n + 1/2^\tau$.

Finally, the case $q_v > 1$ is obtained by combining the above bound for $q_v = 1$ with the result of [13]. This provides AUTH bound $2q_v/2^n + q_v/2^\tau$, which completes the proof.

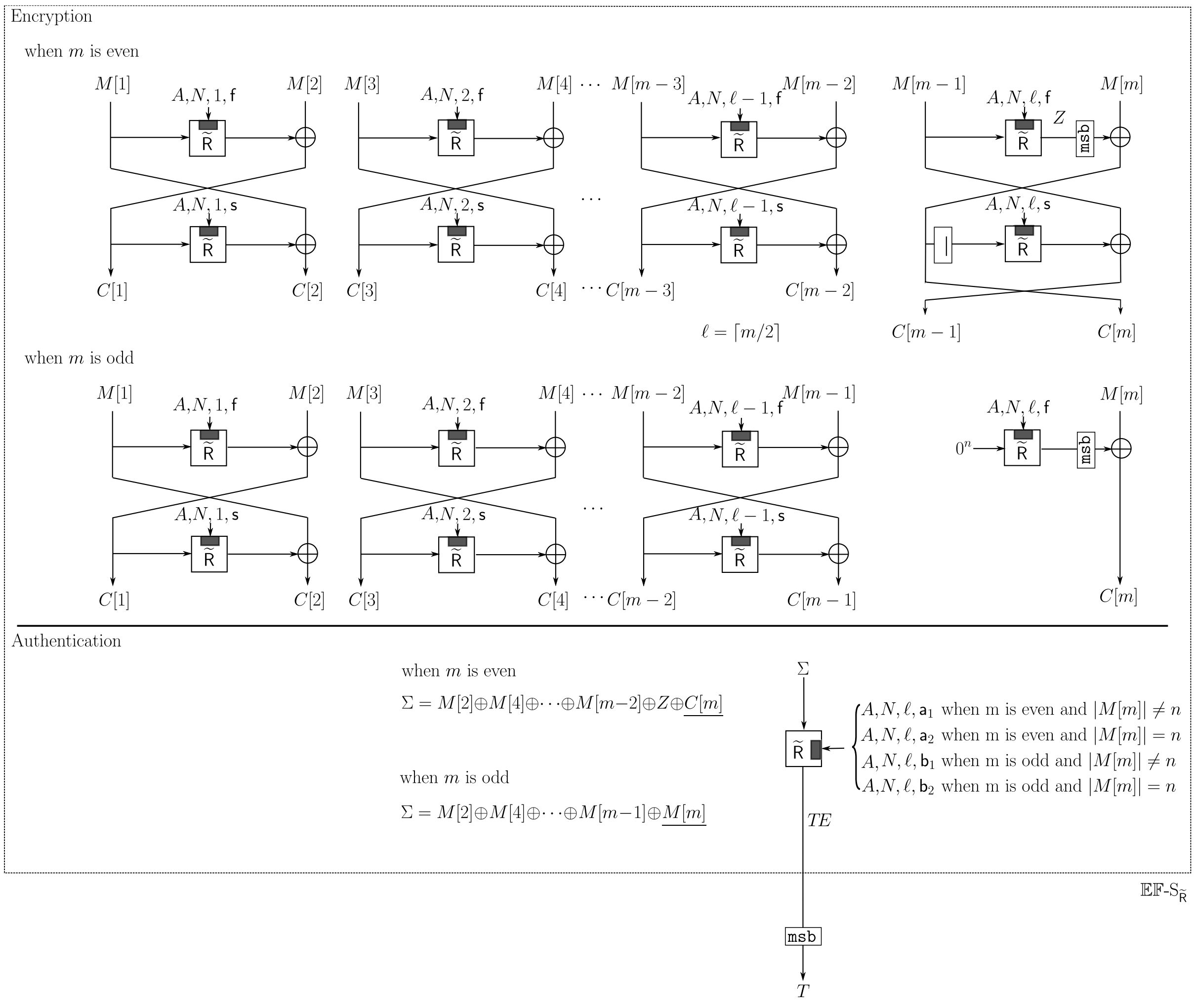


Fig. 14. Encryption of $\mathbb{OTRS}$ function.

```
Algorithm OTRS-$\mathcal{E}_{E,\tau}(N, A, M)$

 1. if $A \neq \varepsilon$ then $TA \leftarrow$ AF-S$_E(A)$
 2. else $TA \leftarrow 0^n$
 3. $(C, TE) \leftarrow$ EF-S$_E(N, M, TA)$
 4. $T \leftarrow \mathtt{msb}_\tau(TE)$
 5. return $(C, T)$
```

```
Algorithm OTRS-$\mathcal{D}_{E,\tau}(N, A, C, T)$

 1. if $A \neq \varepsilon$ then $TA \leftarrow$ AF-S$_E(A)$
 2. else $TA \leftarrow 0^n$
 3. $(M, TE) \leftarrow$ DF-S$_E(N, C, TA)$
 4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE)$
 5. if $\widehat{T} = T$ return $M$
 6. else return $\perp$
```

```
Algorithm EF-S$_E(N, M, TA)$

 1. $\Sigma \leftarrow 0^n$
 2. $L \leftarrow E(\underline{N}) \oplus TA, L \leftarrow 2L$
 3. $(M[1], \dots, M[m]) \xleftarrow{n} M,\ \ell \leftarrow \lceil m/2 \rceil$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $C[2i-1] \leftarrow E(2^{i-1}L \oplus M[2i-1]) \oplus M[2i]$
 6.     $C[2i] \leftarrow E(2^{i-1}3L \oplus C[2i-1]) \oplus M[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8. if $m$ is even
 9.     $Z \leftarrow E(2^{\ell-1}L \oplus M[m-1])$
10.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
11.     $C[m-1] \leftarrow E(2^{\ell-1}3L \oplus \underline{C[m]}) \oplus M[m-1]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(E(2^{\ell-1}L)) \oplus M[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. if $m$ is even and $|M[m]| \neq n$
17.     then $TE \leftarrow E(2^{\ell-1}3^3L \oplus \Sigma)$
18. if $m$ is even and $|M[m]| = n$
19.     then $TE \leftarrow E(2^{\ell-1}3^17L \oplus \Sigma)$
20. if $m$ is odd and $|M[m]| \neq n$
21.     then $TE \leftarrow E(2^{\ell-1}3^2L \oplus \Sigma)$
22. if $m$ is odd and $|M[m]| = n$
23.     then $TE \leftarrow E(2^{\ell-1}7L \oplus \Sigma)$
24. $C \leftarrow (C[1], \dots, C[m])$
25. return $(C, TE)$
```

```
Algorithm DF-S$_E(N, C, TA)$

 1. $\Sigma \leftarrow 0^n$
 2. $L \leftarrow E(\underline{N}) \oplus TA, L \leftarrow 2L$
 3. $(C[1], \dots, C[m]) \xleftarrow{n} C,\ \ell \leftarrow \lceil m/2 \rceil$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $M[2i-1] \leftarrow E(2^{i-1}3L \oplus C[2i-1]) \oplus C[2i]$
 6.     $M[2i] \leftarrow E(2^{i-1}L \oplus M[2i-1]) \oplus C[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8. if $m$ is even
 9.     $M[m-1] \leftarrow E(2^{\ell-1}3L \oplus \underline{C[m]}) \oplus C[m-1]$
10.     $Z \leftarrow E(2^{\ell-1}L \oplus M[m-1])$
11.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(E(2^{\ell-1}L)) \oplus C[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. if $m$ is even and $|C[m]| \neq n$
17.     then $TE \leftarrow E(2^{\ell-1}3^3L \oplus \Sigma)$
18. if $m$ is even and $|C[m]| = n$
19.     then $TE \leftarrow E(2^{\ell-1}3^17L \oplus \Sigma)$
20. if $m$ is odd and $|C[m]| \neq n$
21.     then $TE \leftarrow E(2^{\ell-1}3^2L \oplus \Sigma)$
22. if $m$ is odd and $|C[m]| = n$
23.     then $TE \leftarrow E(2^{\ell-1}7L \oplus \Sigma)$
24. $M \leftarrow (M[1], \dots, M[m])$
25. return $(M, TE)$
```

```
Algorithm AF-S$_E(A)$

 1. $\Xi \leftarrow 0^n$
 2. $Q \leftarrow E(0^n)$
 3. $(A[1], \dots, A[a]) \xleftarrow{n} A$
 4. for $i = 1$ to $a - 1$ do
 5.     $\Xi \leftarrow E(A[i] \oplus \Xi)$
 6. $\Xi \leftarrow \Xi \oplus \underline{A[a]}$
 7. if $|A[a]| \neq n$ then $TA \leftarrow E(2Q \oplus \Xi)$
 8. else $TA \leftarrow E(4Q \oplus \Xi)$
 9. return $TA$
```

Fig. 15. Alternative representation of Fig. 7 for explicitly showing the masking coefficients.

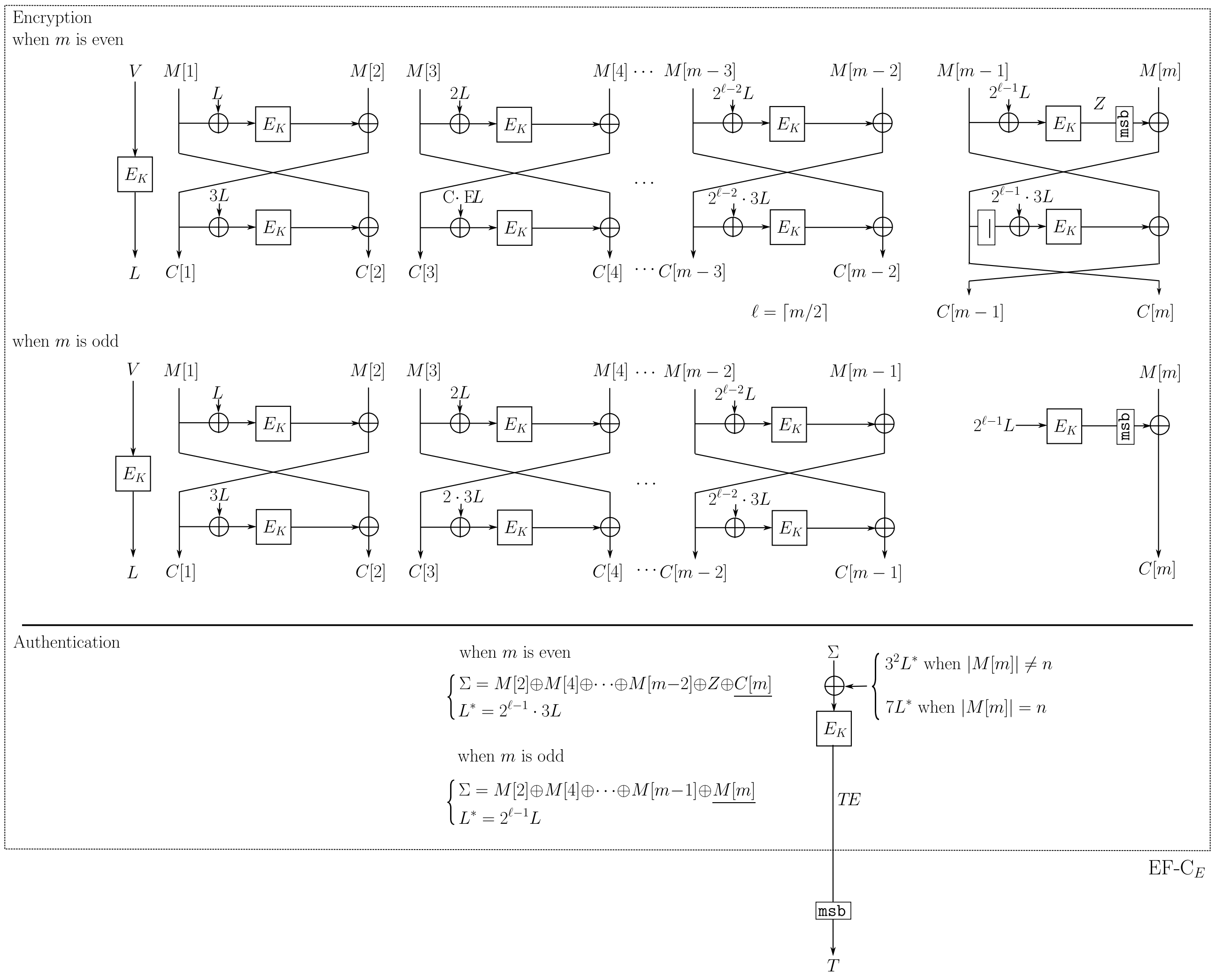


Fig. 16. Encryption of OTRC, where $V = \texttt{lsb}_n(\text{DF-C}_E(0^n, A)) \oplus \underline{N}$ if $A \neq \varepsilon$ and $V = \underline{N}$ if $A = \varepsilon$.

D A Variant with Combined AD Processing

D.1 Specification

This section specifies a variant of OTR with a novel combined method for authentication of associated data and decryption function. We call it OTR with combined ADP or OTRC for short. We write OTRC$[E, \tau]$ to denote OTRC with blockcipher E and tab bit-length τ. OTRC consists of the encryption and decryption core functions, defined as EF-C$_E$ and DF-C$_E$. They are almost the same as EF$_E$ and DF$_E$ of OTR, the only difference is in the generation of L, which is now simply an encryption of the first n-bit argument of the core functions written as V. For encryption, if AD A is non-empty, OTRC first computes $V = \texttt{lsb}_n(\text{DF-C}_E(0^n, A)) \oplus \underline{N}$ and if A is empty, $V = \underline{N}$. Then OTRC computes EF-C$_E(V, M) \to (C, TE)$. The tag is $\texttt{msb}_\tau(TE)$. Reuse of DF-C$_E$ for AD processing is beneficial to reducing the implmentation size, for both software and hardware, while enabling parallel AD processing. Since the global structure of OTRC is similar to OTRS, it allows the caching for static AD, and enjoys OTRS's extended security notion as well (i.e. (A, N) can be used as nonce). Fig. 16 shows the encryption algorithm of OTRC, and Fig. 17 shows the pseudocode of OTRC.

D.2 Security Analysis

Seucirty bounds. The security proof is done in a way similar to that of OTR with serial ADP. We provide the security bounds of OTRC for the extended security notions as explained by Appendix C, that is, the definitions

Algorithm OTRC-$\mathcal{E}_{E,\tau}(N, A, M)$

```
 1. $V \leftarrow \underline{N}$
 2. if $A \neq \varepsilon$ then $V \leftarrow V \oplus \mathtt{lsb}_n(\text{DF-C}_E(0^n, A))$
 3. $(C, TE) \leftarrow \text{EF-C}_E(V, M)$
 4. $T \leftarrow \mathtt{msb}_\tau(TE)$
 5. return $(C, T)$
```

Algorithm OTRC-$\mathcal{D}_{E,\tau}(N, A, C, T)$

```
 1. $V \leftarrow \underline{N}$
 2. if $A \neq \varepsilon$ then $V \leftarrow V \oplus \mathtt{lsb}_n(\text{DF-C}_E(0^n, A))$
 3. $(M, TE) \leftarrow \text{DF-C}_E(V, C)$
 4. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE)$
 5. if $\widehat{T} = T$ return $M$
 6. else return $\perp$
```

Algorithm EF-C$_E(V, M)$

```
 1. $\Sigma \leftarrow 0^n$
 2. $U \leftarrow E(V)$, $L \leftarrow U$, $L^\sharp \leftarrow 3U$
 3. $(M[1], \ldots, M[m]) \xleftarrow{n} M$
 4. for $i = 1$ to $\lceil m/2 \rceil - 1$ do
 5.     $C[2i-1] \leftarrow E(L \oplus M[2i-1]) \oplus M[2i]$
 6.     $C[2i] \leftarrow E(L^\sharp \oplus C[2i-1]) \oplus M[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8.     $L \leftarrow L \oplus L^\sharp$, $L^\sharp \leftarrow 2L^\sharp$
 9. if $m$ is even
10.     $Z \leftarrow E(L \oplus M[m-1])$
11.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
12.     $C[m-1] \leftarrow E(L^\sharp \oplus \underline{C[m]}) \oplus M[m-1]$
13.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
14.     $L^* \leftarrow L^\sharp$
15. if $m$ is odd
16.     $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(E(L)) \oplus M[m]$
17.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
18.     $L^* \leftarrow L$
19. if $|M[m]| \neq n$ then $TE \leftarrow E(3^2 L^* \oplus \Sigma)$
20. else $TE \leftarrow E(7L^* \oplus \Sigma)$
21. $C \leftarrow (C[1], \ldots, C[m])$
22. return $(C, TE)$
```

Algorithm DF-C$_E(V, C)$

```
 1. $\Sigma \leftarrow 0^n$
 2. $U \leftarrow E(V)$, $L \leftarrow U$, $L^\sharp \leftarrow 3U$
 3. $(C[1], \ldots, C[m]) \xleftarrow{n} C$
 4. for $i = 1$ to $\lceil m/2 \rceil - 1$ do
 5.     $M[2i-1] \leftarrow E(L^\sharp \oplus C[2i-1]) \oplus C[2i]$
 6.     $M[2i] \leftarrow E(L \oplus M[2i-1]) \oplus C[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
 8.     $L \leftarrow L \oplus L^\sharp$, $L^\sharp \leftarrow 2L^\sharp$
 9. if $m$ is even
10.     $M[m-1] \leftarrow E(L^\sharp \oplus \underline{C[m]}) \oplus C[m-1]$
11.     $Z \leftarrow E(L \oplus \underline{M[m-1]})$
12.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
13.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
14.     $L^* \leftarrow L^\sharp$
15. if $m$ is odd
16.     $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(E(L)) \oplus C[m]$
17.     $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
18.     $L^* \leftarrow L$
19. if $|C[m]| \neq n$ then $TE \leftarrow E(3^2 L^* \oplus \Sigma)$
20. else $TE \leftarrow E(7L^* \oplus \Sigma)$
21. $M \leftarrow (M[1], \ldots, M[m])$
22. return $(M, TE)$
```

Fig. 17. Algorithms of OTR with combined ADP (OTRC). Tag bit size is $0 < \tau \leq n$, and $\underline{X}$ denotes the 10^* padding of X, and $|N| \leq n-1$.

of $\mathtt{Adv}^{\mathtt{priv}}$, $\mathtt{Adv}^{\mathtt{auth}}$, $\mathtt{Adv}^{\mathtt{cpa\text{-}nr}}$ and $\mathtt{Adv}^{\mathtt{cca\text{-}nr}}$ are identical to those defined in Appendix C. For simplicity we assume the underlying blockcipher is an n-bit URP, P. The computational counterparts are fairly straightforward, thus omitted.

Theorem 7. *Fix $\tau \in \{1, \ldots, n\}$. For any PRIV-adversary $\mathcal{A}$ with parameter (q, σ_A, σ_M),*

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathrm{OTRC}[\mathsf{P},\tau]}(\mathcal{A}) \leq \frac{10\sigma^2_{\mathtt{priv}}}{2^n}$$

holds for $\sigma_{\mathtt{priv}} = q + \sigma_A + \sigma_M$.

Theorem 8. *Fix $\tau \in \{1, \ldots, n\}$. For any AUTH-adversary $\mathcal{A}$ with parameter $(q, q_v, \sigma_A, \sigma_M, \sigma_{A'}, \sigma_{C'})$,*

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathrm{OTRC}[\mathsf{P},\tau]}(\mathcal{A}) \leq \frac{12\sigma^2_{\mathtt{auth}}}{2^n} + \frac{q_v}{2^\tau}$$

holds for $\sigma_{\mathtt{auth}} = q + q_v + \sigma_A + \sigma_M + \sigma_{A'} + \sigma_{C'}$.

D.3 Proofs of Theorems 7 and 8

The proofs are largely the same as Appendix C with a separate analysis for the pseudorandomness of the last n-bit output of DF-C. This section has some overlaps with Appendix C, and thus some notations are reused, in order to make this section self-contained and accessible with minimum reference to Appendix C.

First step: TBC-based design.

In Fig. 18, we define an AEAD scheme denoted by $\mathbb{OTRC}[\tau]$. It is compatible to OTRC$[E,\tau]$ and uses a tweakable n-bit URF, $\widetilde{\mathsf{R}} : \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$. Here, tweak T is represented as a vector, $T = (x, y, i, w) \in \mathcal{T} \overset{\text{def}}{=} \mathcal{A}_{ae} \times \mathcal{N}_{ae} \times \mathbb{N} \times \varOmega$, where $\varOmega = \{\mathtt{f}, \mathtt{s}, \mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$. For convenience we let $\varOmega_1 = \{\mathtt{f}, \mathtt{s}\}$ and $\varOmega_2 = \{\mathtt{a}_1, \mathtt{a}_2, \mathtt{b}_1, \mathtt{b}_2\}$. Here $\mathbb{OTRC}[\tau]$ consists of encryption function, $\mathbb{OTRC}$-$\mathcal{E}_\tau$, and decryption function, $\mathbb{OTRC}$-$\mathcal{D}_\tau$, and these functions use encryption and decryption cores, $\mathbb{EF}$-C$_{\widetilde{\mathsf{R}}}$ and $\mathbb{DF}$-C$_{\widetilde{\mathsf{R}}}$. The privacy and authenticity bounds of $\mathbb{OTRC}$ are shown in Theorem 9. We remark that $\mathbb{OTRC}$ and $\mathbb{OTRS}$ in Appendix C are equivalent (though we introduce function η for convenience), hence the proof of Theorem 9 trivially follows from that of Theorem 6.

Theorem 9. *Fix $\tau \in \{1, \ldots, n\}$. For any PRIV-adversary $\mathcal{A}$,*

$$\mathtt{Adv}^{\mathtt{priv}}_{\mathbb{OTRC}[\tau]}(\mathcal{A}) = 0.$$

Moreover, for any AUTH-adversary $\mathcal{A}$ using q encryption queries and q_v decryption queries,

$$\mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTRC}[\tau]}(\mathcal{A}) \le \frac{2q_v}{2^n} + \frac{q_v}{2^\tau}.$$

Second step: analysis of TBC. In Fig. 19, we define a TBC, $\widetilde{G}[\mathsf{P}]^{\langle A,N,i,\omega\rangle}(X)$, where $(A, N, i, \omega) \in \mathcal{T}$ denotes a tweak and X denotes an input. It uses an n-bit URP, P. For tweaks that do not appear in Fig. 10, we let them as undefined. Let $\widetilde{\mathsf{R}}$ be a tweakable URF compatible with $\widetilde{G}[\mathsf{P}]$.

Proposition 4. *If* $\mathbb{EF}$-C *and* $\mathbb{DF}$-C *use $\widetilde{G}[\mathsf{P}]$ instead of $\widetilde{\mathsf{R}}$, we obtain* EF-C$_{\mathsf{P}}$ *and* DF-C$_{\mathsf{P}}$.

We remark that $\widetilde{G}[\mathsf{P}]$ here does not perform GF doublings in a sequential manner, and performs DF-C$_{\mathsf{P}}$ with the first argument set to 0^n for every input, however, this does not cause a problem for simulation purpose. We then prove that $\widetilde{G}[\mathsf{P}]$ is a secure tweakable URF, shown by the following lemma. The proof is in Appendix D.4.

Lemma 6. *For adversary $\mathcal{A}$ accessing $\widetilde{G}[\mathsf{P}]$ using q queries with σ_A total blocks for A, we have*

$$\mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{A}) \le \frac{10(\sigma_A^2 + q^2)}{2^n},$$

where $\widetilde{\mathsf{R}}$ is a tweakable URF compatible with $\widetilde{G}[\mathsf{P}]$.

Third step: deriving bounds. We show that, for $\mathcal{A}$ accessing OTRC$[\mathsf{P},\tau]$ using q encryption queries with σ_M total plaintext blocks and σ_A total AD blocks, there is an adversary $\mathcal{B}$ accessing $\widetilde{G}[\mathsf{P}]$ with $\sigma_M + q$ queries and σ_A total AD blocks, such that

$$\begin{aligned}
\mathtt{Adv}^{\mathtt{priv}}_{\mathrm{OTRC}[\mathsf{P},\tau]}(\mathcal{A}) &= \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathrm{OTRC}[\mathsf{P},\tau],\$}(\mathcal{A}) && (68)\\
&\le \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathrm{OTRC}[\mathsf{P},\tau],\mathbb{OTRC}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa\text{-}nr}}_{\mathbb{OTRC}[\tau],\$}(\mathcal{A}) && (69)\\
&\le \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{B}) && (70)\\
&\le \frac{10\sigma_A^2 + 10(q+\sigma_M)^2}{2^n} && (71)\\
&\le \frac{10\sigma_{\mathtt{priv}}^2}{2^n}, && (72)
\end{aligned}$$

where the second inequality follows from Proposition 4 and Theorem 9, and the third follows from Lemma 6. For proving AUTH bound, we similarly introduce $\mathcal{B}$ against $\widetilde{G}[\mathsf{P}]$ with $q + q_v + \sigma_M + \sigma_{C'}$ queries and $\sigma_A + \sigma_{A'}$ total AD blocks, and derive the bound as

$$\begin{aligned}
\mathtt{Adv}^{\mathtt{auth}}_{\mathrm{OTRC}[\mathsf{P},\tau]}(\mathcal{A}) &\le \mathtt{Adv}^{\mathtt{cca\text{-}nr}}_{\mathrm{OTRC}[\mathsf{P},\tau],\mathbb{OTRC}[\tau]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{auth}}_{\mathbb{OTRC}[\tau]}(\mathcal{A}) && (73)\\
&\le \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{\mathsf{R}}}(\mathcal{B}) + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (74)\\
&\le \frac{10(\sigma_A + \sigma_{A'})^2 + 10(q + q_v + \sigma_M + \sigma_{C'})^2}{2^n} + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (75)\\
&\le \frac{10\sigma_{\mathtt{auth}}^2}{2^n} + \frac{2q_v}{2^n} + \frac{q_v}{2^\tau} && (76)\\
&\le \frac{12\sigma_{\mathtt{auth}}^2}{2^n} + \frac{q_v}{2^\tau}, && (77)
\end{aligned}$$

```
Algorithm $\mathbb{OTRC}$-$\mathcal{E}_\tau(N, A, M)$

1. $(C, TE) \leftarrow \mathbb{EF}\text{-}\mathrm{C}_{\widetilde{\mathsf{R}}}(N, A, M)$
2. $T \leftarrow \mathtt{msb}_\tau(TE)$
3. return $(C, T)$
```

```
Algorithm $\mathbb{OTRC}$-$\mathcal{D}_\tau(N, C, A, T)$

1. $(M, TE) \leftarrow \mathbb{DF}\text{-}\mathrm{C}_{\widetilde{\mathsf{R}}}(N, A, C)$
2. $\widehat{T} \leftarrow \mathtt{msb}_\tau(TE)$
3. if $\widehat{T} = T$ return $M$
4. else return $\perp$
```

```
Algorithm $\mathbb{EF}\text{-}\mathrm{C}_{\widetilde{\mathsf{R}}}(N, A, M)$

1. $\Sigma \leftarrow 0^n$
2. $(M[1], \dots, M[m]) \xleftarrow{n} M$
3. $(\ell, \omega) \leftarrow \eta(M)$
4. for $i = 1$ to $\ell - 1$ do
5.     $C[2i-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{f}\rangle}(M[2i-1]) \oplus M[2i]$
6.     $C[2i] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{s}\rangle}(C[2i-1]) \oplus M[2i-1]$
7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
8. if $m$ is even
9.     $Z \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(M[m-1])$
10.    $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(Z) \oplus M[m]$
11.    $C[m-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{s}\rangle}(\underline{C[m]}) \oplus M[m-1]$
12.    $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.    $C[m] \leftarrow \mathtt{msb}_{|M[m]|}(\widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(0^n)) \oplus M[m]$
15.    $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\omega\rangle}(\Sigma)$
17. $C \leftarrow (C[1], \dots, C[m])$
18. return $(C, TE)$
```

```
Algorithm $\mathbb{DF}\text{-}\mathrm{C}_{\widetilde{\mathsf{R}}}(N, A, C)$

1. $\Sigma \leftarrow 0^n$
2. $(C[1], \dots, C[m]) \xleftarrow{n} C$
3. $(\ell, \omega) \leftarrow \eta(C)$
4. for $i = 1$ to $\ell - 1$ do
5.     $M[2i-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{s}\rangle}(C[2i-1]) \oplus C[2i]$
6.     $M[2i] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,i,\mathtt{f}\rangle}(M[2i-1]) \oplus C[2i-1]$
7.     $\Sigma \leftarrow \Sigma \oplus M[2i]$
8. if $m$ is even
9.     $M[m-1] \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{s}\rangle}(\underline{C[m]}) \oplus C[m-1]$
10.    $Z \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(M[m-1])$
11.    $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(Z) \oplus C[m]$
12.    $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{C[m]}$
13. if $m$ is odd
14.    $M[m] \leftarrow \mathtt{msb}_{|C[m]|}(\widetilde{\mathsf{R}}^{\langle A,N,\ell,\mathtt{f}\rangle}(0^n)) \oplus C[m]$
15.    $\Sigma \leftarrow \Sigma \oplus \underline{M[m]}$
16. $TE \leftarrow \widetilde{\mathsf{R}}^{\langle A,N,\ell,\omega\rangle}(\Sigma)$
17. $M \leftarrow (M[1], \dots, M[m])$
18. return $(M, TE)$
```

```
Algorithm $\eta(X)$

1. $(X[1], \dots, X[m]) \xleftarrow{n} X$
2. $\ell \leftarrow \lceil m/2 \rceil$
3. if $m$ is even and $|X[m]| \neq n$ then $\omega = \mathtt{a}_1$
4. if $m$ is even and $|X[m]| = n$ then $\omega = \mathtt{a}_2$
5. if $m$ is odd and $|X[m]| \neq n$ then $\omega = \mathtt{b}_1$
6. if $m$ is odd and $|X[m]| = n$ then $\omega = \mathtt{b}_2$
7. return $(\ell, \omega)$
```

Fig. 18. The encryption and decryption algorithms of $\mathbb{OTRC}[\tau]$ using a tweakable n-bit URF $\widetilde{\mathsf{R}}$. Here $\mathbb{OTRC}[\tau]$ is equivalent to $\mathbb{OTRS}[\tau]$ in Fig. 9.

where the second inequality follows from Proposition 4 and Theorem 9, and the third follows from Lemma 6. This concludes the proof.

D.4 Proof of Lemma 6

We first consider $\widetilde{G}[\mathsf{R}]$, a function obtained by substituting P in Fig. 10 with an n-bit URF, R. In the same manner to Appendix C, we define a family of small functions written as $\mathbf{Q} = \{\mathbf{Q}_i\}_{i=1,\dots,5}$, using two dummy variables, $\mathtt{Rnd}_1, \mathtt{Rnd}_2 \in \{0,1\}^n$, which are independent and uniform over n bits. Formally, we have

$$\mathbf{Q}_1^{(i,\omega)}(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus g(i, \omega, U)) \text{ for } \omega \in \Omega_1, \tag{78}$$

$$\mathbf{Q}_2^{(i,\omega)}(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus g(i, \omega, U)) \oplus \mathtt{Rnd}_1 \text{ for } \omega \in \Omega_2, \tag{79}$$

$$\mathbf{Q}_3(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus \mathtt{Rnd}_1) \oplus \mathtt{Rnd}_2, \tag{80}$$

$$\mathbf{Q}_4^{(i,\omega)}(x) \stackrel{\text{def}}{=} \mathsf{R}(x \oplus g(i, \omega, \mathtt{Rnd}_2)) \text{ for } \omega \in \Omega, \tag{81}$$

$$\mathbf{Q}_5(x) \stackrel{\text{def}}{=} \mathsf{R}(x) \oplus \mathtt{Rnd}_2, \tag{82}$$

```
Algorithm $\widetilde{G}[\mathsf{P}]^{\langle A,N,i,\omega\rangle}(X)$

 1. $V \leftarrow \underline{N}$
 2. if $A \neq \varepsilon$ then $V \leftarrow V \oplus \mathtt{lsb}_n(\text{DF-C}_{\mathsf{P}}(0^n, A))$
 3. $L \leftarrow \mathsf{P}(V)$
 4. $Y \leftarrow \mathsf{P}(g(i,\omega,L) \oplus X)$
 5. return $Y$
```

```
Function $g(i,\omega,L)$

 1. Switch $\omega$
 2.     Case $\mathtt{f}$ : $\Delta \leftarrow 2^{i-1}L$
 3.     Case $\mathtt{s}$ : $\Delta \leftarrow 2^{i-1}3L$
 4.     Case $\mathtt{a}_1$ : $\Delta \leftarrow 2^{i-1}3^3L$
 5.     Case $\mathtt{a}_2$ : $\Delta \leftarrow 2^{i-1}3^17L$
 6.     Case $\mathtt{b}_1$ : $\Delta \leftarrow 2^{i-1}3^2L$
 7.     Case $\mathtt{b}_2$ : $\Delta \leftarrow 2^{i-1}7L$
 8. return $\Delta$
```

Fig. 19. Tweakable permutation implicitly used by OTRC[P, τ]. DF-C is described in Fig. 17.

```
Algorithm $\mathbb{G}[\mathbf{Q}]^{\langle A,N,i,\omega\rangle}(X)$

 1. if $\underline{A} \neq \varepsilon$ then
 2.     $\widehat{V} \leftarrow \underline{N} \oplus \text{DF-C}'[\mathbf{Q}_1, \mathbf{Q}_2](A)$
 3.     $\widehat{L} \leftarrow \mathbf{Q}_3(\widehat{V})$
 4. else
 5.     $\widehat{V} \leftarrow \underline{N}$
 6.     $\widehat{L} \leftarrow \mathbf{Q}_5(\widehat{V})$
 7. $S \leftarrow g(i,\omega,\widehat{L}) \oplus X$
 8. $Y \leftarrow \mathbf{Q}_4^{(i,\omega)}(S)$
 9. return $Y$
```

```
Algorithm DF-C$'[\mathbf{Q}_1, \mathbf{Q}_2](A)$

 1. $\Sigma \leftarrow 0^n$
 2. $(X[1], \ldots, X[m]) \xleftarrow{n} X$
 3. $(\ell, \omega) \leftarrow \eta(X)$
 4. for $i = 1$ to $\ell - 1$ do
 5.     $Y[2i-1] \leftarrow \mathbf{Q}_1^{(i,\mathtt{s})}(X[2i-1]) \oplus X[2i]$
 6.     $Y[2i] \leftarrow \mathbf{Q}_1^{(i,\mathtt{f})}(Y[2i-1]) \oplus X[2i-1]$
 7.     $\Sigma \leftarrow \Sigma \oplus Y[2i]$
 8. if $m$ is even
 9.     $Y[m-1] \leftarrow \mathbf{Q}_1^{(\ell,\mathtt{s})}(\underline{X[m]}) \oplus X[m-1]$
10.     $Z \leftarrow \mathbf{Q}_1^{(\ell,\mathtt{f})}(Y[m-1])$
11.     $Y[m] \leftarrow \mathtt{msb}_{|X[m]|}(Z) \oplus X[m]$
12.     $\Sigma \leftarrow \Sigma \oplus Z \oplus \underline{X[m]}$
13. if $m$ is odd
14.     $Y[m] \leftarrow \mathtt{msb}_{|X[m]|}(\mathbf{Q}_1^{(\ell,\mathtt{f})}(0^n)) \oplus X[m]$
15.     $\Sigma \leftarrow \Sigma \oplus \underline{Y[m]}$
16. $T \leftarrow \mathbf{Q}_2^{(\ell,\omega)}(\Sigma)$
17. return $T$
```

Fig. 20. $\mathbb{G}[\mathbf{Q}]$, a function equivalent to $\widetilde{G}[\mathsf{R}]$.

where $U = \mathsf{R}(0^n)$, and $x \in \{0,1\}^n$ for $\mathbf{Q}_i$ for $i = 1,2,3,4$ and $x \in \{0,1\}^n \setminus \{0^n\}$ for $\mathbf{Q}_5$. For convenience, $\mathbf{Q}_3$ and $\mathbf{Q}_5$ may assume (i,ω) as a part of input but defined to be a default constant (dummy), and we write $\mathcal{I}_{\mathbf{Q}} \subseteq \{0,1\}^n \times \mathbb{N} \times \Omega \times \{1,\ldots,5\}$ to denote the set of all valid inputs to $\mathbf{Q}$, by seeing the last coordinate of $\mathcal{I}_{\mathbf{Q}}$ to specify the member of the family. Function $g(*,*,*)$ is as defined in Fig. 19. We remark that $g(i,\omega,x)$ is a multiplication over GF(2^n) with x and a constant depending on (i,ω) written as $c_{(i,\omega)} \in \text{GF}(2^n)$ (e.g. $c_{(3,\mathtt{s})} = 2^23$). Thus we can write $g(i,\omega,x) = c_{(i,\omega)} \cdot x$. In the rest of the proof we assume i in $g(i,\omega,x)$ is from 1 to $2^{n/2} - 1$, which is needed (and sufficient) for our security bounds. Recall that we assumed $n = 128$ and the lexicographically-first primitive polynomial for defining GF(2^n). Then, from Proposition 5 of [44] the following holds.

Proposition 5. *Let (i,ω) and (i',ω') be distinct elements over $\{1,\ldots,2^{n/2}-1\} \times \Omega$. Then we have $c_{(i,\omega)} \neq c_{(i',\omega')}$ and $c_{(i,\omega)}$ is not a zero element.*

Using $\mathbf{Q}$ we build a function $\mathbb{G}[\mathbf{Q}]$ shown by Fig. 20. It uses DF-C$'[\mathbf{Q}_1, \mathbf{Q}_2]$ for AD processing, which is equivalent to DF-C$_{\mathsf{R}}$ taking the first argument $V = 0^n$ and returning only the last n bits (i.e. *TE*). We observe that in $\mathbb{G}[\mathbf{Q}]$, $\mathtt{Rnd}_1$ and $\mathtt{Rnd}_2$ are always canceled between two consecutive calls of $\mathbf{Q}_i$. For example, suppose that the output of $\mathbf{Q}_3$ is $\widehat{L}$ and the consecutive input to $\mathbf{Q}_4$ is S as Fig. 20. Then the internal output of R in

$\mathbf{Q}_3$ is $L = \widehat{L} \oplus \mathtt{Rnd}_2$ and the internal input to R in $\mathbf{Q}_4^{(i,\omega)}$ is

$$\begin{aligned} S \oplus g(i,\omega,\mathtt{Rnd}_2) &= (g(i,\omega,\widehat{L}) \oplus X) \oplus g(i,\omega,\mathtt{Rnd}_2) \\ &= g(i,\omega,L \oplus \mathtt{Rnd}_2) \oplus g(i,\omega,\mathtt{Rnd}_2) \oplus X = g(i,\omega,L) \oplus X, \end{aligned} \tag{83}$$

from XOR-linearity of g, which precisely simulates the case of $\widetilde{G}[\mathsf{R}]$. Also, $\mathbf{Q}_5$ does not take 0^n in Fig. 20 since N is shorter than n bits and always padded by one-zero padding function. That is, we have the following proposition.

Proposition 6. *$\mathbb{G}[\mathbf{Q}]$ is equivalent to $\widetilde{G}[\mathsf{R}]$.*

We then show that $\mathbf{Q}$ is indistinguishable from a set of independent URFs.

Lemma 7. *Let $\widetilde{\mathbf{Q}} = \{\widetilde{\mathbf{Q}}_i\}_{i=1,\dots,5}$ be the set of functions, where $\widetilde{\mathbf{Q}}_i$ is an independent URF compatible with $\mathbf{Q}_i$. Taking $\mathbf{Q}$ and $\widetilde{\mathbf{Q}}$ as tweakable functions accepting tweak $t \in \{1,\dots,5\}$, we have*

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbf{Q},\widetilde{\mathbf{Q}}}(\mathcal{A}) \le \frac{(q+1)q}{2^{n+1}},$$

for adversary $\mathcal{A}$ using q queries, where each query is in $\mathcal{I}_{\mathbf{Q}}$.

Proof of Lemma 7. Let $\varDelta_t^{(i,\omega)}$ be the input masks for $\mathbf{Q}_t$, and let ∇_t be the output mask for $\mathbf{Q}_t$. They are defined as

$$\varDelta_1^{(i,\omega)} = g(i,\omega,U), \quad \varDelta_2^{(i,\omega)} = g(i,\omega,U), \quad \varDelta_3^{(i,\omega)} = \mathtt{Rnd}_1, \quad \varDelta_4^{(i,\omega)} = g(i,\omega,\mathtt{Rnd}_2), \quad \varDelta_5^{(i,\omega)} = 0^n, \tag{84}$$

$$\nabla_1 = 0^n, \quad \nabla_2 = \mathtt{Rnd}_1, \quad \nabla_3 = \mathtt{Rnd}_2, \quad \nabla_4 = 0^n, \quad \nabla_5 = \mathtt{Rnd}_2, \tag{85}$$

where $U = \mathsf{R}(0^n)$ and $\mathtt{Rnd}_1$ and $\mathtt{Rnd}_2$ are independently random. We remark that all masks are functions of $(U, \mathtt{Rnd}_1, \mathtt{Rnd}_2, i, \omega)$, thus we can write as $\mathbf{Q}_t^{(i,\omega)}(x) = \mathsf{R}(\varDelta_t^{(i,\omega)} \oplus x) \oplus \nabla_t$. We show Fig. 21 for defining two games, Game$\mathbf{Q}$ and Game$\widetilde{\mathbf{Q}}$. We observe that Game$\mathbf{Q}$ and Game$\widetilde{\mathbf{Q}}$ precisely simulate $\mathbf{Q}$ and $\widetilde{\mathbf{Q}}$ for adversary who does not repeat queries (here a query is in $\mathcal{I}_{\mathbf{Q}}$). For Game$\widetilde{\mathbf{Q}}$ this is obvious, since the output of Game$\widetilde{\mathbf{Q}}$ is always independent and uniformly random. For Game$\mathbf{Q}$ the generation procedure of Y and YE in Game$\mathbf{Q}$ is opposite to that of $\mathbf{Q}$; in Game$\mathbf{Q}$, if XE is a new value, Y is uniformly sampled and then $Y = YE \oplus \nabla_t$ is determined, while $\mathbf{Q}$ first determines YE randomly and computes $Y = YE \oplus \nabla_t$. However, both yield the identical marginal distribution of (Y, YE). If XE has a collision, Game$\mathbf{Q}$ determines YE from the set of previously sampled values, and Y is determined as $Y \leftarrow YE \oplus \nabla_t$. Games of Fig. 21 define the flag `bad` to set (in line 13) when two inputs to ρ after the input maskings collide. Then, following the Game-Playing technique [15], both games are identical until `bad` gets set to `true`, thus we have

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbf{Q},\widetilde{\mathbf{Q}}}(\mathcal{A}) \le \Pr[\mathcal{A}^{\mathrm{Game}\mathbf{Q}} \Rightarrow 1] - \Pr[\mathcal{A}^{\mathrm{Game}\widetilde{\mathbf{Q}}} \Rightarrow 1] \le \Pr[\mathcal{A}^{\mathrm{Game}\widetilde{\mathbf{Q}}} \text{ sets } bad]. \tag{86}$$

Hence, what we need is to bound the last probability, which is derived from the pairwise collision probability of inputs to ρ, including 0^n to produce U. From Proposition 5, we see that

$$\max_{(x,i,\omega,t)\in\mathcal{I}_{\mathbf{Q}}} \Pr_{U,\mathtt{Rnd}_1,\mathtt{Rnd}_2}[\varDelta_t^{(i,\omega)} = x] \le \frac{1}{2^n} \tag{87}$$

$$\max_{\substack{(x,i,\omega,t),(x',i',\omega',t')\in\mathcal{I}_{\mathbf{Q}} \\ (x,i,\omega,t)\neq(x',i',\omega',t')}} \Pr_{U,\mathtt{Rnd}_1,\mathtt{Rnd}_2}[\varDelta_t^{(i,\omega)} \oplus \varDelta_{t'}^{(i',\omega')} = x \oplus x'] \le \frac{1}{2^n}, \tag{88}$$

where (87) is for bounding the probability of $x \oplus \varDelta_t^{(i,\omega)} = 0^n$ for a query (x,i,ω,t) (note that this does not happen with $t = 5$ as $(0^n,i,\omega,5) \notin \mathcal{I}_{\mathbf{Q}}$), and (88) is for bounding the probability of $x \oplus \varDelta_t^{(i,\omega)} = x' \oplus \varDelta_{t'}^{(i',\omega')}$ for a pair of distinct, valid queries, (x,i,ω,t) and (x',i',ω',t').

These equations show that any collision occurs at most with probability $1/2^n$, and since q queries in the game yield at most $q+1$ accesses to ρ, the bound is derived as $\binom{q+1}{2}/2^n \le (q+1)q/2^{n+1}$. This proves Lemma 7. ∎

We next evaluate $\mathbb{G}[\widetilde{\mathbf{Q}}]$, i.e. using $\widetilde{\mathbf{Q}} = \{\widetilde{\mathbf{Q}}_1,\dots,\widetilde{\mathbf{Q}}_5\}$ instead of $\mathbf{Q}$. The core of $\mathbb{G}[\widetilde{\mathbf{Q}}]$ is DF-C$'[\widetilde{\mathbf{Q}}_1,\widetilde{\mathbf{Q}}_2]$, and we prove DF-C$'[\widetilde{\mathbf{Q}}_1,\widetilde{\mathbf{Q}}_2] : \{0,1\}^* \to \{0,1\}^n$ is a PRF. We remark that this lemma is slightly more general than what we need, as it includes the empty string as a valid input, however DF-C$'[\widetilde{\mathbf{Q}}_1,\widetilde{\mathbf{Q}}_2]$ in $\mathbb{G}[\widetilde{\mathbf{Q}}]$ never takes the empty string.

```
Initialization
00      $U \leftarrow \rho(0^n) \xleftarrow{\$} \{0,1\}^n$
01      $\mathtt{Rnd}_1 \xleftarrow{\$} \{0,1\}^n$, $\mathtt{Rnd}_2 \xleftarrow{\$} \{0,1\}^n$
On query $(X, i, \omega, t) \in \mathcal{I}_{\mathbf{Q}}$                    // $(i,\omega)$ is a dummy for $t = 3, 5$
10      $XE \leftarrow \Delta_t^{(i,\omega)} \oplus X$                              // $X \neq 0^n$ when $t = 5$
11      $Y \xleftarrow{\$} \{0,1\}^n$
12      $YE \leftarrow Y \oplus \nabla_t$
13      if $XE \in \mathrm{Dom}(\rho)$ then $\mathtt{bad} \leftarrow \mathtt{true}$, $\boxed{YE \leftarrow \rho(XE),\ Y \leftarrow YE \oplus \nabla_t}$
14      else $\rho(XE) \leftarrow YE$
15      return $Y$
```

Fig. 21. Game$\mathbf{Q}$ and Game$\widetilde{\mathbf{Q}}$. The latter does not contain the boxed argument.

Lemma 8. *For adversary $\mathcal{A}$ using q queries, we have*

$$\mathtt{Adv}^{\mathtt{prf}}_{\text{DF-C}'[\widetilde{\mathbf{Q}}_1, \widetilde{\mathbf{Q}}_2]}(\mathcal{A}) \leq \frac{q(q-1)}{2^n}. \tag{89}$$

Proof of Lemma 8. We decompose DF-C$'[\widetilde{\mathbf{Q}}_1, \widetilde{\mathbf{Q}}_2]$ into the message hashing, $f[\widetilde{\mathbf{Q}}_1]$, which takes input $X \in \{0,1\}^*$ to generate Σ (corresponding to the result of line 15 in Fig. 20) and $\eta(X)$, using $\widetilde{\mathbf{Q}}_1$ instead of $\mathbf{Q}_1$, and the finalization, $g[\widetilde{\mathbf{Q}}_2](\Sigma, \eta(X)) = \widetilde{\mathbf{Q}}_2^{(\eta(X))}(\Sigma)$. Then we have DF-C$'[\widetilde{\mathbf{Q}}_1, \widetilde{\mathbf{Q}}_2] = g[\widetilde{\mathbf{Q}}_2] \circ f[\widetilde{\mathbf{Q}}_1]$, and this can be seen as a Carter-Wegman MAC. We only need to know the maximum probability of

$$\mathrm{Coll}_1(X, X') = \Pr_{\widetilde{\mathbf{Q}}_1}[(\Sigma, \eta(X)) = (\Sigma', \eta(X'))], \tag{90}$$

where $(\Sigma, \eta(X)) = f[\widetilde{\mathbf{Q}}_1](X)$ and $(\Sigma', \eta(X')) = f[\widetilde{\mathbf{Q}}_1](X')$ for distinct $X, X' \in \{0,1\}^*$, including ε. In the following we perform a case analysis, which is mostly the same as Case 3 and Case 4 of Appendix A. Here, w.l.o.g. we assume $\eta(X) = \eta(X') = (\ell, \omega)$, which also implies $|X|_n = |X'|_n = m$ for some m and, $|X| \leq |X'|$. We employ the notations of Fig. 20 for internal variables, such as $X[1]$ and Z for input X and $X'[1]$ and Z' for X'.

Case 1 : $\ell = 1$. We have $\Sigma = \underline{Y[1]}$ and $\Sigma = \underline{Y'[1]}$. Let us first assume X and X' are non-empty. Then $|X| = |X[1]| = |Y[1]|$ and $|X'| = |X'[1]| = |Y'[1]|$. If $|X| \neq |X'|$, due to the definition of one-zero padding, $\Sigma \neq \Sigma'$ always holds. If $|X| = |X'|$, $\Sigma \oplus \Sigma' = \underline{Y[1]} \oplus \underline{Y'[1]} = (X \oplus X') \| 0^{n-|X|}$ is non-zero.

Next, when X is empty while X' is not, this implies $1 \leq |X'| < n$ so that both have $\omega = \mathtt{b}_1$, and we have $\Sigma = 10^{n-1}$ and $\Sigma' = \underline{Y'[1]}$, for $1 \leq |Y'[1]| < n$. Since the latter always contains bit 1 in the $(|X'|+1)$-th position, $\Sigma \oplus \Sigma' \neq 0^n$ holds. Thus $\mathrm{Coll}_1(X, X')$ is zero for both sub-cases.

Case 2 : $\ell = 2$. We have $\Sigma = \underline{X[2]} \oplus Z$, where $Z = \widetilde{\mathbf{Q}}_1^{(1,\mathtt{f})}(Y[1])$ and $Y[1] = \widetilde{\mathbf{Q}}_1^{(1,\mathtt{s})}(\underline{X[2]}) \oplus X[1]$. Similarly we have $\Sigma' = \underline{X'[2]} \oplus Z'$ and $Z' = \widetilde{\mathbf{Q}}_1^{(1,\mathtt{f})}(Y'[1])$ and $Y'[1] = \widetilde{\mathbf{Q}}_1^{(1,\mathtt{s})}(\underline{X'[2]}) \oplus X'[1]$. If $X[2] = X'[2]$ and $X[1] \neq X'[1]$, we have $\underline{X[2]} = \underline{X'[2]}$ and thus $Y[1] \neq Y'[1]$ always holds. Then Z and Z' are independent and uniform variables, which implies $\mathrm{Coll}_1(X, X') = 1/2^n$.

If $X[2] \neq X'[2]$, we always have $\underline{X[2]} \neq \underline{X'[2]}$ (as we assume ω is identical), thus $Y[1] = Y'[1]$ can happen with probability $1/2^n$. Given $Y[1] \neq Y'[1]$, Z and Z' are independent and uniform, thus $Z = Z'$ occurs with probability $1/2^n$. Thus $\mathrm{Coll}_1(X, X')$ is at most $2/2^n$.

Case 3 : $\ell > 2$, m is even. Suppose the difference is only in the last two blocks, i.e. $(X[2i-1], X[2i]) = (X'[2i-1], X'[2i])$ for $1 \leq i \leq \ell - 1$, and $(X[2\ell-1], X[2\ell]) \neq (X'[2\ell-1], X'[2\ell])$, where $2\ell = m$. We observe that $\Sigma \oplus \Sigma'$ is a sum of $Z \oplus Z' \oplus \underline{X[2\ell]} \oplus \underline{X'[2\ell]}$ and independent terms using $\widetilde{\mathbf{Q}}_1^{(i,\omega')}$ for $i < \ell$, $\omega' \in \Omega_1$. Here, $Z = \widetilde{\mathbf{Q}}_1^{(\ell,\mathtt{f})}(Y[2\ell-1])$ and $Y[2\ell-1] = \widetilde{\mathbf{Q}}_1^{(\ell,\mathtt{s})}(\underline{X[2\ell]}) \oplus X[2\ell-1]$. From the analysis similar to Case 2, $\mathrm{Coll}_1(X, X')$ is at most $2/2^n$.

Otherwise, we have $(X[2i-1], X[2i]) \neq (X'[2i-1], X'[2i])$ for some $1 \leq i \leq \ell - 1$. Then $\Sigma \oplus \Sigma'$ is a sum of $Y[2i] \oplus Y'[2i]$ and independent terms. From the analysis similar to Case 2, $Y[2i] \oplus Y'[2i] = c$ occurs with probability at most $2/2^n$ for any $c \in \{0,1\}^n$. Thus, again $\mathrm{Coll}_1(X, X')$ is at most $2/2^n$ in this case.

Case 4 : $\ell > 2$, m is odd. Suppose the difference is only in the last one block, i.e. $(X[2i-1], X[2i]) = (X'[2i-1], X'[2i])$ for $1 \leq i \leq \ell - 1$, and $X[m] \neq X'[m]$. Then $\Sigma \oplus \Sigma' = \underline{Y[m]} \oplus \underline{Y'[m]}$ and from the same analysis as Case 1, $\mathrm{Coll}_1(X, X')$ is zero. Otherwise, we have $(X[2i-1], X[2i]) \neq (X'[2i-1], X'[2i])$ for some $1 \leq i \leq \ell - 1$ and from the same analysis as Case 3, $\mathrm{Coll}_1(X, X')$ is at most $2/2^n$.

Summarizing all cases, $\mathrm{Coll}_1(X, X') \leq 2/2^n$ for any two distinct inputs, X and X'. Using this bound with (a variant of) Lemma 2 of Black and Rogaway [19], we have

$$\mathtt{Adv}^{\mathtt{prf}}_{\mathrm{DF\text{-}C'}[\widetilde{\mathbf{Q}}_1, \widetilde{\mathbf{Q}}_2]}(\mathcal{A}) \leq \max \sum_{1 \leq i < j \leq q} \mathrm{Coll}_1(X_i, X_j) \leq \binom{q}{2} \cdot \frac{2}{2^n} \leq \frac{q(q-1)}{2^n}, \tag{91}$$

where the maximum is taken for all (non-adaptive) choices of distinct q inputs, $(X_1, \ldots, X_q)$. This proves the Lemma 8. ■

We then show that $\mathbb{G}[\widetilde{\mathbf{Q}}]$ is a tweakable PRF. Let $\mathbb{G}_2[\widetilde{\mathbf{Q}}, \mathsf{R}_v]$ be the same as $\mathbb{G}[\widetilde{\mathbf{Q}}]$ except that $\mathrm{DF\text{-}C'}[\widetilde{\mathbf{Q}}_1, \widetilde{\mathbf{Q}}_2]$ is substituted with an independent variable-input-length random function, R_v. We prove the following.

Lemma 9. *For adversary $\mathcal{A}$ using q queries, we have*

$$\mathtt{Adv}^{\mathtt{prf}}_{\mathbb{G}_2[\widetilde{\mathbf{Q}}, \mathsf{R}_v]}(\mathcal{A}) \leq \frac{q(q-1)}{2^n}. \tag{92}$$

Proof of Lemma 9. We define

$$\mathrm{Coll}_2((A, N, i, \omega, X), (A', N', i', \omega', X')) \overset{\text{def}}{=} \Pr_{\mathsf{R}_v, \widetilde{\mathbf{Q}}_3, \widetilde{\mathbf{Q}}_5}[(S, i, \omega) = (S', i', \omega')] \tag{93}$$

as the probability of input collision for $\widetilde{\mathbf{Q}}_4$ in $\mathbb{G}_2[\widetilde{\mathbf{Q}}, \mathsf{R}_v]$, where the output is $Y = \widetilde{\mathbf{Q}}_4^{(i,\omega)}(S)$.

Then (93) is bounded as follows. W.l.o.g., we assume $(i, \omega) = (i', \omega')$ and focus on the collision $S = S'$.

- Case $[(A, N) = (A', N')$ and $X \neq X']$: We have $S \oplus S' = X \oplus X' \neq 0^n$.
- Case $[A = A' = \varepsilon, N \neq N']$: Here $S \oplus S' = g(i, \omega, \widetilde{\mathbf{Q}}_5(\underline{N})) \oplus g(i, \omega, \widetilde{\mathbf{Q}}_5(\underline{N'})) \oplus X \oplus X'$. From Proposition 5 and $\underline{N} \neq \underline{N'}$ (as both are shorter than n bits), the collision probability of S is $1/2^n$.
- Case $[A = A' \neq \varepsilon, N \neq N']$: Here, we have

$$\begin{aligned} S \oplus S' &= g(i, \omega, \widetilde{\mathbf{Q}}_3(\underline{N} \oplus \mathsf{R}_v(A))) \oplus g(i, \omega, \widetilde{\mathbf{Q}}_3(\underline{N'} \oplus \mathsf{R}_v(A'))) \oplus X \oplus X' \\ &= g(i, \omega, \widetilde{\mathbf{Q}}_3(\underline{N} \oplus \mathsf{R}_v(A)) \oplus \widetilde{\mathbf{Q}}_3(\underline{N'} \oplus \mathsf{R}_v(A))) \oplus X \oplus X'. \end{aligned} \tag{94}$$

  Since $\underline{N} \oplus \mathsf{R}_v(A) \neq \underline{N'} \oplus \mathsf{R}_v(A)$, the collision probability is $1/2^n$.
- Case $[A \neq A',\ A, A' \neq \varepsilon]$: Inputs to $\widetilde{\mathbf{Q}}_3$ are $\underline{N} \oplus \mathsf{R}_v(A)$ and $\underline{N'} \oplus \mathsf{R}_v(A')$. They will collide with probability $1/2^n$. If inputs to $\widetilde{\mathbf{Q}}_3$ are distinct, $S = S'$ occurs with probability $1/2^n$. Thus collision probability of S is at most $2/2^n$.
- Case $[A \neq A',\ A = \varepsilon]$: Here, $S \oplus S' = g(i, \omega, \widetilde{\mathbf{Q}}_5(\underline{N})) \oplus g(i, \omega, \widetilde{\mathbf{Q}}_3(\underline{N'} \oplus \mathsf{R}_v(A'))) \oplus X \oplus X'$. Since $\widetilde{\mathbf{Q}}_5(\underline{N})$ and $\widetilde{\mathbf{Q}}_3(V' \oplus \underline{N'})$ are independent and random, the collision probability of S is $1/2^n$.

Therefore, (93) is at most $2/2^n$ for any distinct inputs.

In the same manner as (91), this implies

$$\mathtt{Adv}^{\mathtt{prf}}_{\mathbb{G}_2[\widetilde{\mathbf{Q}}, \mathsf{R}_v]}(\mathcal{A}) \leq \binom{q}{2} \cdot \frac{2}{2^n} \leq \frac{q(q-1)}{2^n}, \tag{95}$$

which proves Lemma 9. ■

Now we are ready to bound $\mathtt{Adv}^{\mathtt{prf}}_{\widetilde{G}[\mathsf{P}]}(\mathcal{A})$. Recall that $\mathcal{A}$ uses q queries with σ_A total blocks for AD. Let $\sigma_p = \sigma_A + 3q$. We observe that $\sigma_p + 1$ is the maximum number of required P invocations in $\widetilde{G}[\mathsf{P}]$ queried by $\mathcal{A}$, including $\mathsf{P}(0^n)$ which can be cached. Summarizing all results, we have

$$\mathtt{Adv}^{\mathtt{prf}}_{\widetilde{G}[\mathsf{P}]}(\mathcal{A}) \leq \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}], \widetilde{G}[\mathsf{R}]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{R}], \mathbb{G}[\mathbf{Q}]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{G}[\mathbf{Q}], \mathbb{G}[\widetilde{\mathbf{Q}}]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{G}[\widetilde{\mathbf{Q}}], \mathbb{G}_2[\widetilde{\mathbf{Q}}, \mathsf{R}_v]}(\mathcal{A}) + \mathtt{Adv}^{\mathtt{prf}}_{\mathbb{G}_2[\widetilde{\mathbf{Q}}, \mathsf{R}_v]}(\mathcal{A}). \tag{96}$$

Each term is bounded as

$$\mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{P}],\widetilde{G}[\mathsf{R}]}(\mathcal{A}) \le \frac{(\sigma_p+1)\cdot\sigma_p}{2^{n+1}} \text{ from PRP-PRF switching lemma,} \tag{97}$$

$$\mathtt{Adv}^{\mathtt{cpa}}_{\widetilde{G}[\mathsf{R}],\mathbb{G}[\mathbf{Q}]}(\mathcal{A}) = 0 \text{ from Proposition 6,} \tag{98}$$

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{G}[\mathbf{Q}],\mathbb{G}[\widetilde{\mathbf{Q}}]}(\mathcal{A}) \le \mathtt{Adv}^{\mathtt{cpa}}_{\mathbf{Q},\widetilde{\mathbf{Q}}}(\mathcal{B}) \le \frac{(\sigma_p+1)\cdot\sigma_p}{2^{n+1}} \text{ from Lemma 7,} \tag{99}$$

$$\mathtt{Adv}^{\mathtt{cpa}}_{\mathbb{G}[\widetilde{\mathbf{Q}}],\mathbb{G}_2[\widetilde{\mathbf{Q}},\mathsf{R}_v]}(\mathcal{A}) \le \frac{q(q-1)}{2^n} \text{ from Lemma 8,} \tag{100}$$

$$\mathtt{Adv}^{\mathtt{prf}}_{\mathbb{G}_2[\widetilde{\mathbf{Q}},\mathsf{R}_v]}(\mathcal{A}) \le \frac{q(q-1)}{2^n} \text{ from Lemma 9,} \tag{101}$$

where $\mathcal{B}$ uses $\sigma_p + 1$ queries. Taking the sum of all bounds, the right hand side of (96) is at most

$$\frac{2(\sigma_p+1)\cdot\sigma_p}{2^{n+1}} + \frac{2q(q-1)}{2^n} = \frac{(\sigma_p+1)\cdot\sigma_p}{2^n} + \frac{2q(q-1)}{2^n} \tag{102}$$

$$= \frac{(\sigma_A+3q+1)\cdot(\sigma_A+3q)+2q(q-1)}{2^n} \tag{103}$$

$$\le \frac{\sigma_A^2+6q\sigma_A+\sigma_A+11q^2+q}{2^n} \tag{104}$$

$$\le \frac{8\sigma_A^2+12q^2}{2^n} \le \frac{10(\sigma_A^2+q^2)}{2^n}, \tag{105}$$

where the last two inequalities follow from the fact $\sigma_A \ge q$ (note that the empty string counts as 1 block). This proves Lemma 6. □