IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2019, No. 4, pp. 119–146. DOI:10.13154/tosc.v2019.i4.119-146

Release of Unverified Plaintext: Tight Unified Model and Application to ANYDAE

Donghoon Chang[1,2], Nilanjan Datta[3], Avijit Dutta[3], Bart Mennink[4], Mridul Nandi[3], Somitra Sanadhya[5] and Ferdinand Sibleyras[6]

[1] Indraprastha Institute of Information Technology, Delhi, India
[2] National Institute of Standards and Technology, Gaithersburg, Maryland, USA
[3] Indian Statistical Institute, Kolkata, India
[4] Radboud University, Nijmegen, The Netherlands
[5] Indian Institute of Technology, Ropar, India
[6] Inria, Paris, France
pointchang@gmail.com, nilanjan_isi_jrf@yahoo.com, avirocks.dutta13@gmail.com, b.mennink@cs.ru.nl, mridul.nandi@gmail.com, somitra@iitrpr.ac.in, ferdinand.sibleyras@inria.fr

Abstract. Authenticated encryption schemes are usually expected to offer confidentiality and authenticity. In case of release of unverified plaintext (RUP), an adversary gets separated access to the decryption and verification functionality, and has more power in breaking the scheme. Andreeva et al. (ASIACRYPT 2014) formalized RUP security using plaintext awareness, informally meaning that the decryption functionality gives no extra power in breaking confidentiality, and INT-RUP security, covering authenticity in case of RUP. We describe a single, unified model, called AERUP security, that ties together these notions: we prove that an authenticated encryption scheme is AERUP secure if and only if it is conventionally secure, plaintext aware, and INT-RUP secure. We next present ANYDAE, a generalization of SUNDAE of Banik et al. (ToSC 2018/3). ANYDAE is a lightweight deterministic scheme that is based on a block cipher with block size n and arbitrary mixing functions that all operate on an n-bit state. It is particularly efficient for short messages, it does not rely on a nonce, and it provides maximal robustness to a lack of secure state. Whereas SUNDAE is not secure under release of unverified plaintext (a fairly simple attack can be mounted in constant time), ANYDAE is. We make handy use of the AERUP security model to prove that ANYDAE achieves both conventional security as RUP security, provided that certain modest conditions on the mixing functions are met. We describe two simple instances, called MONDAE and TUESDAE, that conform to these conditions and that are competitive with SUNDAE, in terms of efficiency and optimality.

Keywords: authenticated encryption · release of unverified plaintext · AERUP · generalization · SUNDAE · ANYDAE · MONDAE · TUESDAE

1 Introduction

The rise of the Internet of Things comes with high demands on and constrictive conditions for cryptographic schemes. Such constraints may come in various types, as these small interconnected devices may have to operate with low power, low area, low memory, or otherwise. Lightweight cryptography is about developing cryptographic solutions for such constrained environments, and partly ignited by the upcoming NIST lightweight competition [Nat18], the field is gaining momentum. Specifically, in recent years, various lightweight

Received: 2019-01-09, Accepted: 2019-01-11, Published: 2020-31-01

authenticated encryption schemes appeared, such as CLOC [IMGM14], JAMBU [WH16], COFB [CIMN17], SAEB [NMSS18], Beetle [CDNY18], and SUNDAE [BBLT18].

1.1 Release of Unverified Plaintext

The core idea of authenticated encryption is that plaintext is not only encrypted, but at the same time authenticated. This renders two security models relevant for the setting of authenticated encryption. The first one is *confidentiality*, where an adversary is given access to the authenticated encryption functionality $\mathcal{E}$ or a random function, and it has to distinguish both. The second one is *authenticity*, where an adversary is given access to the authenticated encryption $\mathcal{E}$ and the decryption/verification $\mathcal{DV}$ and tries to provide an authenticated ciphertext that is approved by $\mathcal{DV}$. Refer to Section 3.1 for the conventional security models.

Authenticated encryption thus suits a two-fold goal. Upon decryption, however, this means that a ciphertext needs to be decrypted *and its corresponding plaintext needs to be verified* before the plaintext is released to the user. In some applications, where the tag is computed over plaintext and not over the ciphertext, this means that the decrypted plaintext should be kept in secure memory before the authentication verification is completed. Only at that point in time, the plaintext can be released. The exigency of secure memory can however be a serious issue in constrained devices in the Internet of Things. Security under release of unverified plaintext may also be a necessity in case of high performance requirements (e.g., high speed, low latency, and long messages): real-time streaming protocols and Optical Transport Networks sometimes release plaintexts on the fly in order to reduce the end-to-end latency and storage.

Andreeva et al. [ABL+14] formalized security under release of unverified plaintext. The approach consists of keeping the encryption functionality $\mathcal{E}$ as is, but separating the decryption/verification functionality $\mathcal{DV}$ into a decryption functionality $\mathcal{D}$ and a verification functionality $\mathcal{V}$. The security model of confidentiality remains unchanged, noting that in the conventional confidentiality model the adversary has access to encryption $\mathcal{E}$ only. For authenticity, where the adversary tries to forge an authenticated encryption, it is now allowed to query *both* $\mathcal{D}$ *and* $\mathcal{V}$. Andreeva et al. also introduced plaintext awareness. In plaintext awareness, the adversary is given access to $\mathcal{E}$ and to either $\mathcal{D}$ or a simulator. It complements conventional confidentiality in the sense that it measures the advantage an adversary can gain from actually having access to $\mathcal{D}$. Refer to Section 3.2 for the RUP security models.

In the model of RUP security, the adversary turns out to have significantly more power: since the introduction of the model, various schemes have been forged in the RUP model [CDN16, DLMN17, IMI18], and schemes that have been proven to achieve RUP authenticity often come at a cost [ZWH+17, HSY17, ADL17]. For plaintext awareness, the situation is even more poignant: the model is often ignored, which arguably comes from the fact that the model is more complicated (it requires the description of a simulator). To our knowledge, the only article directly proving plaintext awareness of a construction is that of Andreeva et al. [ABL+14].

1.2 Generalized AERUP Security

In Section 3.3, we describe AERUP security. AERUP unifies the notions of RUP authenticity and plaintext awareness in the setting where one does not rely on nonces, and delivers a simple and handy model that allows for easily proving security in both notions *in one go*. In AERUP security, one considers an adversary that has access to either $\mathcal{E}$, $\mathcal{D}$, and $\mathcal{V}$, or it has access to a random function, a simulator, and a function that always rejects. The generalization reminds of the unification of conventional authenticated encryption – one simply considers an adversary that has access to either $\mathcal{E}$ and $\mathcal{D}$ or a random function and

a function that always rejects (see Definition 6) – but for AERUP security, proving that the unification works well is a bit more involved. Nevertheless, we prove in Sections 3.4 and 3.5 that AERUP security is equivalent to conventional nonce-independent security plus RUP authenticity plus plaintext awareness.

The equivalence, in particular, means that AERUP more closely matches the idea of RUP security, in contrast to robust authenticated encryption of Hoang et al. [HKR15], that considers a strong notion of security against a maximum misuse of a nonce, subtle authenticated encryption of Barwell et al. [BPS15], that considers different types of leakage oracles, and RUPAE of Ashur et al. [ADL17], that mostly focuses on the strength of the decryption function in case of nonce-based encryption. Note, particularly, that the models of Barwell et al. and Ashur et al. concentrate on nonce-based authenticated encryption. Keeping the defense in depth setting – CAESAR use case 3 – in mind, both authenticity and limited privacy damage are desirable under nonce misuse and release of unverified plaintext.

A further comparison with existing models is given in Section 3.6.

1.3 SUNDAE and Its RUP Security

Banik et al. [BBLT18] introduced SUNDAE as a deterministic authenticated encryption scheme [RS06]. It is a block cipher based construction that is developed to meet various of the constraints put forward by the applications of lightweight cryptography. More specifically, SUNDAE internally uses an n-bit block cipher. It has an n-bit state, and makes $a+2m+1$ block cipher calls for the authenticated encryption of associated data of a blocks and message of m blocks. Banik et al. proved that SUNDAE achieves confidentiality and authenticity up to attack complexity around $2^{n/2}$. We discuss the SUNDAE mode in detail in Section 4.

Despite its recent introduction, SUNDAE found quick adoption. It serves as mode in no less than five authenticated encryption schemes submitted to the NIST Lightweight Cryptography Standardization Process: ESTATE [CDJ+19], SIV-Rijndael256 [BGIS19a], SIV-TEM-PHOTON [BGIS19b], SUNDAE-GIFT [BBP+19], and TRIFLE [DGM+19].

SUNDAE is particularly efficient for short messages, it has a state size as small as the block size of the underlying cipher, and it offers good implementation characteristics both on lightweight and high-performance platforms. One of the properties of the SUNDAE mode is that it "provides maximal robustness to a lack of proper randomness or secure state" [BBLT18]. This is achieved by making SUNDAE deterministic, i.e., not depending on a nonce for security. In the context of lightweight cryptography, this property of SUNDAE is particularly important as the mode may be ran in environments with extremely limited state to store a counter, or it may be ran in an environment that has no access to an entropy source to generate a random nonce.

The developers of SUNDAE stated that for their construction, "unverified plaintext from the decryption algorithm should not be released." This is somewhat in contradiction with the claimed "maximal robustness to a lack of [...] secure state." We provide clarity in this ambivalence by demonstrating that SUNDAE is *not* RUP secure (Section 4.2). The attack is quite simple: in an authenticated encryption evaluation of SUNDAE, the tag simply equals the state after data absorption. In an unverified decryption evaluation, an adversary $\mathcal{A}$ may set the tag to an arbitrary value and learn the encryption of this value. This allows it to "reconstruct" a verification evaluation by itself, and to produce a forgery. Admittedly, Banik et al. explicitly mentioned that unverified plaintext should not be released, and our RUP forgery does *not* invalidate SUNDAE's security.

1.4 ANYDAE: Salvaging SUNDAE in the RUP Setting

Motivated by our attack on SUNDAE, we investigate the possibilities to improve SUNDAE to achieve RUP security. To do this, we present a generalization of SUNDAE, dubbed ANYDAE, in Section 5. Like SUNDAE, ANYDAE consists of a sequential evaluation of an n-bit block cipher, but it is interlaced with evaluations of arbitrary mixing functions. We derive sufficient conditions on these mixing functions in order for ANYDAE to be secure in the new AERUP security model (Section 5.2). The level of security that is achieved is the same as that of SUNDAE, up to constant.

ANYDAE is thus AERUP secure under certain modest assumptions on the mixing functions. The proof sees the notion of AERUP security in action: it consists of describing a proper simulator, and bounding the distance between $\mathcal{E}$, $\mathcal{D}$, and $\mathcal{V}$ on the one hand and the random function, the simulator, and the rejection function on the other hand. This bounding, in turn, is performed using the H-coefficient technique [Pat08, CS14].

Finally, we map the result to two simple instantiations. The first instantiation is MONDAE (Section 5.3). MONDAE resolves the issue of RUP security in SUNDAE quite elegantly: it simply fixes one bit of the state *after* generation of the tag to 1. This appears as a modest change – efficiency-wise – to achieve stronger security. The second instantiation is TUESDAE (Section 5.4). TUESDAE is designed to be optimal in the number of block cipher invocations for most of its inputs. For example, for associated data of a blocks and a message of m blocks, with $a + m > 1$, TUESDAE makes exactly $a + 2m$ block cipher invocations [CDN18] and works well along with an $n = 128$-bit block cipher.

1.5 Related Work

We recall that the goal of the SUNDAE developers was to investigate what minimality limits can be met by using a block ciphers. ANYDAE, likewise, centers around the same problem. As block ciphers are still widely used, this is a valid question to consider.

An alternative approach to design an authenticated encryption scheme is by using a cryptographic permutation in a keyed duplex mode [BDPV11, MRV15, DMV17]. State of the art shows that a keyed duplex with b-bit permutation can absorb (de facto, authenticate) data at b bits per permutation call and squeeze (de facto, encrypt) data at $b - 2s$ bits per permutation, where s is the targeted security level. We remark that for a block cipher, the state size is $k + n$ bits, where k is the key size and n the block size.

One can see SUNDAE and ANYDAE as variants of the keyed duplex in the following way: (i) instead of a cryptographic permutation, a keyed block cipher is used, and (ii) due to the secrecy of this primitive, full-block squeezing is possible. The differences in domain separation and state mixing are minor. From this point of view, one may dispute the use of a block cipher in SUNDAE and MONDAE: after all, a block cipher is designed to be efficient both in forward and inverse direction, but only evaluated in forward direction. Efficiency of the schemes may be sped-up by using cryptographic primitives that are specifically developed to be efficient in forward direction.

2 Preliminaries

For any finite set $\mathcal{X}$, we denote by $X \xleftarrow{\$} \mathcal{X}$ the random selection of an element X from $\mathcal{X}$. Let $m, n \in \mathbb{N}$. We let $\mathcal{X}^n$ denote the set of n tuples of $\mathcal{X}$ and $\mathcal{X}^*$ denotes the set $\cup_{i \geq 1} \mathcal{X}^i$. Using this notation, we let $\{0,1\}^n$ denote the set of n-bit strings, $\{0,1\}^{\leq n}$ the set of bit strings of length at most n bits, and $\{0,1\}^*$ the set of arbitrarily length strings. We denote by $|X|$ the bit length of X. The empty string is denoted by ε. We often refer to an n bit binary string as a block. We write $X[1 \ldots \ell]$ to denote the sequence $X[1] \| \ldots \| X[\ell]$. We denote by $\mathrm{F}(n)$ (resp., $\mathrm{P}(n)$) the set of all n-bit functions (resp. permutations). We denote by $\mathrm{F}(*, n)$ the set of all functions that on input of a value $M \in \{0,1\}^*$ output a

value of size $|M| + n$ bits. Note that any such function can be defined by lazy sampling, and, overloading notation, by $\$ \xleftarrow{\$} \mathrm{F}(*, n)$ we denote the event of defining a function $\$$ that for every fresh input $M \in \{0,1\}^*$ returns a random bit string of length $|M| + n$.

If $m \leq n$, for $X \in \{0,1\}^n$ we denote by $\lfloor X \rfloor_m$ (resp., $\lceil X \rceil_m$) the m left-most (resp., right-most) bits of X. We let $\mathsf{pad}_n \colon \{0,1\}^{\leq n} \to \{0,1\}^n$ be the function that takes as input a bit string X of size at most n bits, and transforms it into a string of size n bits as follows:

$$\mathsf{pad}_n(X) = \begin{cases} X \parallel 10^{n-|X|-1}\,, \text{ if } |X| < n\,, \\ X\,, \text{ otherwise}\,. \end{cases}$$

For $X \in \{0,1\}^*$, we denote by $X[1 \ldots \ell] \xleftarrow{n} X$ the partitioning of X into $\ell = \lceil |X|/n \rceil$ blocks, where $X[i] \in \{0,1\}^n$ for $i = 1, \ldots, \ell - 1$ and $X[\ell] \in \{0,1\}^{\leq n}$. We define the function $\mathsf{fix}_1 \colon \{0,1\}^n \to \{0,1\}^n$ that fixes the rightmost bit of its input to 1:

$$\mathsf{fix}_1(X) = \lfloor X \rfloor_{n-1} \| 1\,. \tag{1}$$

The function $\mathsf{fix}_{10} \colon \{0,1\}^n \to \{0,1\}^n$ likewise fixes the rightmost two bits of its input to 10. For a function f, we denote its range by $\mathrm{range}(f)$. The expression "a ? b : c" equals b if a holds and c if a does not hold.

Let $\mathrm{GF}(2^n)$ be the finite field of order 2^n. Let $a = a_{n-1} \cdots a_1 a_0 \in \{0,1\}^n$. The value a can be represented as a polynomial $a(\mathtt{x}) = a_{n-1}\mathtt{x}^{n-1} + \cdots + a_1\mathtt{x} + a_0 \in \mathrm{GF}(2^n)$. It can likewise be represented by an integer in $\{0, \ldots, 2^n - 1\}$, by evaluating the polynomial a at $\mathtt{x} = 2$. For two elements $a, b \in \{0,1\}^n$, addition $a \oplus b$ is defined as addition of their polynomials, $a(\mathtt{x}) + b(\mathtt{x}) \in \mathrm{GF}(2^n)$, and multiplication $a \otimes b$ is defined with respect to the irreducible polynomial $f(\mathtt{x})$ used to represent $\mathrm{GF}(2^n)$: $a(\mathtt{x}) \cdot b(\mathtt{x}) \bmod f(\mathtt{x})$.

An adversary $\mathcal{A}$ is a probabilistic algorithm. By $\mathcal{A}^{\mathcal{O}} \to 1$ we denote that $\mathcal{A}$ has query access to its oracle $\mathcal{O}$, and after its communication outputs 1. Note that the oracle $\mathcal{O}$, itself, may be defined by a list of oracles.

2.1 Block Ciphers

Let $k, n \in \mathbb{N}$. A block cipher $E \colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is a function that takes as input a key $K \in \{0,1\}^k$ and a message block $X \in \{0,1\}^n$ and transforms it into a ciphertext $Y \in \{0,1\}^n$. We write $E_K(X) = E(K, X)$. The transformation is bijective, i.e., for fixed key K the function E_K is invertible, however, we will not be concerned with inverse evaluations of E. As such, security of block ciphers is measured by its PRP (pseudorandom permutation) security.

Definition 1. Let $k, n \in \mathbb{N}$, and let $E \colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher. The PRP security of E against an adversary $\mathcal{A}$ is defined as

$$\mathsf{PRP}_E(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{E_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{P} \to 1\right)\,, \tag{2}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$, $P \xleftarrow{\$} \mathrm{P}(n)$, and over the random choices of $\mathcal{A}$.

2.2 Differential-Uniform and Regular Functions

Let $n \in \mathbb{N}$ and let $\mathcal{T}$ be a (possibly empty) finite set. We describe the concepts of differential-uniformity and regularity of a function $\rho \colon \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n$ (if $\mathcal{T}$ is an empty set, then ρ is a function from $\{0,1\}^n$ to $\{0,1\}^n$).

Definition 2. Let $n \in \mathbb{N}$ and let $\mathcal{T}$ be a finite set. A function $\rho \colon \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n$ is ϵ-differential-uniform if for any $t \neq t' \in \mathcal{T}$, and $Y \in \{0,1\}^n$, then for $X \xleftarrow{\$} \{0,1\}^n$ we have

$$\mathbf{Pr}\left(\rho(X,t) \oplus \rho(X,t') = Y\right) \leq \epsilon\,.$$

Definition 3. Let $n \in \mathbb{N}$ and let $\mathcal{T}$ be a finite set. A function $\rho\colon \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n$ is ϵ-regular if for any $t \in \mathcal{T}$ and $Y \in \{0,1\}^n$, then for $X \xleftarrow{\$} \{0,1\}^n$ we have

$$\mathbf{Pr}\left(\rho(X,t) = Y\right) \leq \epsilon\,.$$

Based on Definition 3, we obtain the following corollary.

Corollary 1. *Let $n \in \mathbb{N}$ and let $\mathcal{T}$ be a finite set. Let $\rho\colon \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n$ be ϵ-regular. Then, for any $t, t' \in \mathcal{T}$, $Y \in \{0,1\}^n$, and $X, X' \xleftarrow{\$} \{0,1\}^n$ we have*

$$\mathbf{Pr}\left(\rho(X,t) \oplus \rho(X',t') = Y\right) \leq \epsilon\,.$$

2.3 Patarin's H-Coefficient Technique

Consider a computationally unbounded deterministic adaptive adversary $\mathcal{A}$ that interacts with either a real oracle $\mathcal{O}_{\mathrm{re}}$ or an ideal oracle $\mathcal{O}_{\mathrm{id}}$. After its interaction, $\mathcal{A}$ outputs a decision bit. The collection of all queries-responses obtained by $\mathcal{A}$ during its interaction with its oracle are summarized in a transcript τ. This transcript may, in addition, contain additional information about the random oracle that is revealed to the adversary after its interaction but before it outputs its decision bit. This is without loss of generality: the adversary gains more knowledge and hence more distinguishing power.

Let X_{re} and X_{id} be the random variables that take a transcript τ induced by the real and the ideal world respectively. The probability of realizing a transcript τ in the ideal world (i.e., $\mathbf{Pr}\left(X_{\mathrm{id}} = \tau\right)$) is called the *ideal interpolation probability* and the probability of realizing it in the real world is called the *real interpolation probability*. A transcript τ is said to be *attainable* if the ideal interpolation probability is non zero. We denote the set of all attainable transcripts by Θ. Following these notations, we state the main theorem of the H-coefficient technique as follows [Pat08, CS14].

Theorem 1 (H-coefficient technique)**.** *Let $\mathcal{A}$ be a fixed computationally unbounded deterministic adversary that has access to either the real oracle $\mathcal{O}_{\mathrm{re}}$ or the ideal oracle $\mathcal{O}_{\mathrm{id}}$. Let $\Theta = \Theta_{\mathrm{g}} \sqcup \Theta_{\mathrm{b}}$ be some partition of the set of all attainable transcripts into* good *and* bad *transcripts. Suppose there exists $\epsilon_{\mathrm{ratio}} \geq 0$ such that for any $\tau \in \Theta_{\mathrm{g}}$,*

$$\frac{\mathbf{Pr}\left(X_{\mathrm{re}} = \tau\right)}{\mathbf{Pr}\left(X_{\mathrm{id}} = \tau\right)} \geq 1 - \epsilon_{\mathrm{ratio}}\,,$$

and there exists $\epsilon_{\mathrm{bad}} \geq 0$ such that $\mathbf{Pr}\left(X_{\mathrm{id}} \in \Theta_{\mathrm{b}}\right) \leq \epsilon_{\mathrm{bad}}$. Then,

$$\mathbf{Pr}\left(\mathcal{A}^{\mathcal{O}_{\mathrm{re}}} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\mathcal{O}_{\mathrm{id}}} \to 1\right) \leq \epsilon_{\mathrm{ratio}} + \epsilon_{\mathrm{bad}}\,. \tag{3}$$

3 Authenticated Encryption

We consider deterministic authenticated encryption in the context of potential release of unverified plaintext (RUP). Following Andreeva et al. [ABL+14], we separate the decryption algorithm into plaintext computation and tag verification. Formally, let $k, t \in \mathbb{N}$. An authenticated encryption scheme AE is a family of algorithms $(\mathcal{E}, \mathcal{D}, \mathcal{V})$ where

$$\begin{aligned}
\mathcal{E}&\colon (K, A, M) \mapsto (T, C)\,,\\
\mathcal{D}&\colon (K, A, T, C) \mapsto M\,,\\
\mathcal{V}&\colon (K, A, T, C) \mapsto \top/\bot\,.
\end{aligned}$$

Here, $K \in \{0,1\}^k$ is a key, $A \in \{0,1\}^*$ associated data, $M \in \{0,1\}^*$ a message, $T \in \{0,1\}^t$ the verification tag, and $C \in \{0,1\}^{|M|}$ the ciphertext. The authenticated encryption scheme is required to be correct, i.e., should satisfy that $\mathcal{D}(K, A, \mathcal{E}(K, A, M)) = M$ and $\mathcal{V}(K, A, \mathcal{E}(K, A, M)) = \top$ for any K, A, M.

3.1 Conventional Security Models

Conventionally, an authenticated encryption scheme should offer confidentiality, meaning that its ciphertexts are computationally indistinguishable from random, and integrity, meaning that its tags are unforgeable. For confidentiality, we stay with the standard security model that measures the adversarial power to distinguish $\mathcal{E}_K$ for random key $K \xleftarrow{\$} \{0,1\}^k$ from a random function $\$ \xleftarrow{\$} \mathrm{F}(*,t)$.

Definition 4. Let $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme. The CONF security of AE against an adversary $\mathcal{A}$ is defined as

$$\mathsf{CONF}_{\mathrm{AE}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\$} \to 1\right), \tag{4}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$, $\$ \xleftarrow{\$} \mathrm{F}(*,t)$, and over the random choices of $\mathcal{A}$.

For integrity, the adversary is given access to $\mathcal{E}_K$ for random key $K \xleftarrow{\$} \{0,1\}^k$, and it additionally gets access to the verification oracle $\mathcal{V}_K$, and it succeeds if it manages to find a forgery for $\mathcal{V}_K$, i.e., if the oracle ever returns with $\top$ on input of some query (A,T,C) that was not the result of an earlier encryption query.

Definition 5. Let $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme. The INT security of AE against an adversary $\mathcal{A}$ is defined as

$$\mathsf{INT}_{\mathrm{AE}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K, \mathcal{V}_K} \text{ forges}\right), \tag{5}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$ and over the random choices of $\mathcal{A}$. The event "forges" happens if $\mathcal{V}_K$ ever returns $\top$ on some query (A,T,C), and (A,T,C) is not the result of an earlier encryption query (formally, there is no M such that an encryption call (A,M) resulted in (T,C)).

Typically, we combine the above two security notions of an authenticated encryption into an unified one defined as:

Definition 6. Let $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme. The AE security of AE against an adversary $\mathcal{A}$ is defined as

$$\mathsf{AE}_{\mathrm{AE}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K, \mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\$, \perp} \to 1\right), \tag{6}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$, $\$ \xleftarrow{\$} \mathrm{F}(*,t)$, and over the random choices of $\mathcal{A}$, and where $\perp$ is an oracle that rejects on every input.

The well-established result about the security of an authenticated encryption scheme is the following, which we take from Rogaway and Shrimpton [RS06, Propositions 8 and 9].

Proposition 1 ($\mathsf{CONF} + \mathsf{INT} \Leftrightarrow \mathsf{AE}$)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be an authenticated encryption scheme. For any adversaries* $\mathcal{A}_1$ *with query complexity* q_1*,* $\mathcal{A}_2$ *with query complexity* q_2 *(total query complexity of encryption and verification oracle), and* $\mathcal{A}_3$ *with query complexity* q_3 *(total query complexity of encryption and verification oracle),*

$$\begin{aligned} \mathsf{CONF}_{\mathrm{AE}}(\mathcal{A}_1) &\le \mathsf{AE}_{\mathrm{AE}}(\mathcal{B}_1), \\ \mathsf{INT}_{\mathrm{AE}}(\mathcal{A}_2) &\le \mathsf{AE}_{\mathrm{AE}}(\mathcal{B}_2), \\ \mathsf{AE}_{\mathrm{AE}}(\mathcal{A}_3) &\le \mathsf{CONF}_{\mathrm{AE}}(\mathcal{B}_3) + \mathsf{INT}_{\mathrm{AE}}(\mathcal{B}_4), \end{aligned}$$

where $\mathcal{B}_1$*,* $\mathcal{B}_2$*,* $\mathcal{B}_3$*, and* $\mathcal{B}_4$ *are some adversaries with query complexities* q_1*,* q_2*,* q_3*, and* q_3 *respectively.*

Note that, although Proposition 1 is taken from [RS06], the security definitions and notations are different. Nevertheless, the translation to our definitions is straightforward, with the following differences. For proving the first inequality, $\mathcal{B}_1$ runs $\mathcal{A}_1$ and answers $\mathcal{A}_1$'s queries through its own oracle, i.e, either through $\mathcal{E}_K$ or \$. For the second inequality, $\mathcal{B}_2$ runs $\mathcal{A}_2$ and answers its encryption queries either through $\mathcal{E}_K$ or \$, and it answers its forging attempt either through $\mathcal{V}_K$ or $\perp$. The third inequality holds the same way.

3.2 RUP Security Model

Andreeva et al. [ABL+14] proposed two versions for confidentiality in the RUP setting: plaintext awareness 1 (PA1) and plaintext awareness 2 (PA2). In both models, the real world is constituted of both $\mathcal{E}_K$ and $\mathcal{D}_K$ for random $K \xleftarrow{\$} \{0,1\}^k$. In the ideal world, the decryption algorithm is replaced by a *simulator*[1] $\mathcal{S}$ that mimics the outputs of $\mathcal{D}_K$. In PA1, the simulator has insight in the queries that the adversary made to $\mathcal{E}_K$, whereas in PA2, it has no insight in these queries (and, of course, the adversary is not allowed to make trivial queries). We will consider PA1 security, where the simulator has insight to the queries that the adversary makes to $\mathcal{E}_K$.

Definition 7. Let $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme. Let $\mathcal{S}$ be a simulator. The PA1 security of AE against an adversary $\mathcal{A}$ is defined as

$$\mathsf{PA1}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{S}} \to 1\right) , \tag{7}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$ and over the random choices of $\mathcal{S}$ and $\mathcal{A}$.

We consider integrity in the RUP setting, which differs from the conventional notion of integrity in the fact that the adversary has access to the plaintext computation algorithm $\mathcal{D}$, and can use its outputs to improve its advantage in forging a tag. The notion is taken from Andreeva et al. [ABL+14].

Definition 8. Let $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme. The INT-RUP security of AE against an adversary $\mathcal{A}$ is defined as

$$\mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \text{ forges}\right) , \tag{8}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$ and over the random choices of $\mathcal{A}$. The event "forges" happens if $\mathcal{V}_K$ ever returns top on some query (A, T, C), and (A, T, C) is not the result of an earlier encryption query (formally, there is no M such that an encryption call (A, M) resulted in (T, C)).

3.3 Generalized AERUP Security

In this section, we define an indistinguishability framework for AE security in the released unverified plaintext setting, dubbed as AERUP. The new model, in itself, basically combines RUP confidentiality (i.e., PA1) and integrity (i.e., INT-RUP), in the exact same way as AE unified CONF and INT (see Definition 6). This way it is a handy model for delivering a RUP security proof *in one single go*, so without delivering multiple different but related security proofs.

As before, the security model considers a distinguisher that has access to either of two worlds, the real or ideal world. In the real world, it can query the encryption, decryption, and verification oracles of the AE algorithm, $(\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K)$, for random key $K \xleftarrow{\$} \{0,1\}^k$. In the ideal world, it has access to a random function $\$ \xleftarrow{\$} \mathrm{F}(*, t)$, a simulator $\mathcal{S}$ that has access to the history of encryption queries, and a reject oracle $\perp$.

[1] Called an *extractor* by Andreeva et al. [ABL+14].

Definition 9. Let $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ be an authenticated encryption scheme. Let $\mathcal{S}$ be a simulator. The AERUP security of AE against an adversary $\mathcal{A}$ is defined as

$$\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) = \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\$, \mathcal{S}, \perp} \to 1\right), \tag{9}$$

where the randomness is taken over $K \xleftarrow{\$} \{0,1\}^k$, $\$ \xleftarrow{\$} \mathrm{F}(*, t)$, and over the random choices of $\mathcal{S}$ and $\mathcal{A}$. The adversary is not allowed to relay an earlier response from the first oracle to the third oracle.

Note that, as the simulator has access to the query history that $\mathcal{A}$ made to its first oracle, we can safely allow $\mathcal{A}$ to relay an earlier response from the first oracle to the second oracle.

3.4 Reductions From AERUP

The model of AERUP security is general: if an authenticated encryption scheme AE is AERUP secure, then it is AE secure in terms of Definition 6, PA1 secure in terms of Definition 7, and INT-RUP secure in terms of Definition 8. Stated differently,

$$\mathsf{AE} + \mathsf{PA1} + \mathsf{INT\text{-}RUP} \Leftarrow \mathsf{AERUP}\,.$$

The three reductions are given in below three propositions.

Proposition 2 ($\mathsf{AE} \Leftarrow \mathsf{AERUP}$)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be an authenticated encryption scheme. Let* $\mathcal{S}$ *be a simulator. For any adversary* $\mathcal{A}$ *with query complexity* q *which is the sum of encryption and verification query complexities,*

$$\mathsf{AE}_{\mathrm{AE}}(\mathcal{A}) \leq \mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B})\,,$$

where $\mathcal{B}$ *is some adversary with query complexity* q*.*

Proof. The proof is trivial as the notion of AE is identical to that of AERUP without access to the decryption oracle. Stated differently, $\mathcal{B}$ simulates the oracles of $\mathcal{A}$ by simply ignoring its second oracle and relaying all other oracle queries. □

Note by Propositions 1 and 2, $\mathsf{CONF} \Leftarrow \mathsf{AERUP}$. We will use this observation in the next reduction.

Proposition 3 ($\mathsf{PA1} \Leftarrow \mathsf{AERUP}$)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be an authenticated encryption scheme. Let* $\mathcal{S}$ *be a simulator. For any adversary* $\mathcal{A}$ *that makes* q_e *encryption queries and* q_d *decryption queries,*

$$\mathsf{PA1}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) \leq \mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B}_1) + \mathsf{CONF}_{\mathrm{AE}}(\mathcal{B}_2)\,,$$

where $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *are some adversaries where the query complexity of* $\mathcal{B}_1$ *is* $q_e + q_d$ *and the query complexity of* $\mathcal{B}_2$ *is* q_e*.*

Proof. Let $\mathcal{A}$ be an adversary, making q_e encryption and q_d decryption queries, that breaks the PA1 security of AE with respect to any simulator $\mathcal{S}$ with advantage $\mathsf{PA1}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A})$. We define an adversary $\mathcal{B}_1$ that has the same encryption and decryption query complexity as $\mathcal{A}$, interacts with either $(\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K)$ or $(\mathcal{E}_K, \mathcal{S}, \perp)$: it defines the simulator for $\mathcal{A}$ to be that for itself, and relays queries of $\mathcal{A}$ to its own oracles the obvious way (i.e., $\mathcal{B}_1$ never queries its third oracle). Adversary $\mathcal{B}_2$ with query complexity q_e, in turn, only has access

to an encryption oracle and simulates a simulator $\mathcal{S}$ and the $\perp$-oracle. By definition of $\mathcal{B}_1$ and $\mathcal{B}_2$, we have

$$\begin{aligned}
\mathsf{PA1}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) &= \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{S}} \to 1\right) \\
&\le \mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{S},\perp} \to 1\right) \\
&= \left(\mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}_1^{\$,\mathcal{S},\perp} \to 1\right)\right) \\
&\qquad + \left(\mathbf{Pr}\left(\mathcal{B}_1^{\$,\mathcal{S},\perp} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{S},\perp} \to 1\right)\right) \\
&= \mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B}_1) + \left(\mathbf{Pr}\left(\mathcal{B}_2^{\$} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}_2^{\mathcal{E}_K} \to 1\right)\right) \\
&= \mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B}_1) + \mathsf{CONF}_{\mathrm{AE}}(\mathcal{B}_2)\,,
\end{aligned}$$

where $\$ \xleftarrow{\$} \mathrm{F}(*,t)$ is a random function. □

Proposition 4 ($\mathsf{INT\text{-}RUP} \Leftarrow \mathsf{AERUP}$)**.** *Let* $\mathrm{AE} = (\mathcal{E},\mathcal{D},\mathcal{V})$ *be an authenticated encryption scheme. For any adversary* $\mathcal{A}$ *that makes altogether* q *encryption, decryption, and verification queries,*

$$\mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{A}) = \mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B})\,,$$

where $\mathcal{B}$ *is some adversary with query complexity* q *that involves encryption, decryption, and verification queries, and* $\mathcal{S}$ *is any simulator.*

Proof. Let $\mathcal{A}$ be an adversary that breaks the $\mathsf{INT\text{-}RUP}$ security of AE with advantage $\mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{A})$. We define an adversary $\mathcal{B}$ against the AERUP security of AE that has the same query complexity as $\mathcal{A}$: it relays queries of $\mathcal{A}$ to its own oracles the obvious way, and at the end of the game, it returns 1 if and only if $\mathcal{A}$ made a successful verification query. By definition of $\mathcal{B}$, we have

$$\begin{aligned}
\mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{A}) &= \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \text{ forges}\right) \\
&\overset{i}{=} \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \text{ forges}\right) - \mathbf{Pr}\left(\mathcal{A}^{\$,\mathcal{S},\perp} \text{ forges}\right) \\
&= \mathbf{Pr}\left(\mathcal{B}^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}^{\$,\mathcal{S},\perp} \to 1\right) \\
&= \mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B})\,,
\end{aligned}$$

where $\overset{i}{=}$ follows from the fact that $\mathbf{Pr}\left(\mathcal{A}^{\$,\mathcal{S},\perp} \text{ forges}\right) = 0$ as $\perp$ oracle always returns $\perp$. □

3.5 Reduction To AERUP

Likewise, we can prove that if an authenticated encryption scheme AE is AE secure, $\mathsf{PA1}$ secure, and $\mathsf{INT\text{-}RUP}$ secure, it is also AERUP secure. Stated differently,

$$\mathsf{AE} + \mathsf{PA1} + \mathsf{INT\text{-}RUP} \Rightarrow \mathsf{AERUP}\,.$$

We will prove this reduction below. Here, we note that Propositions 1 implies that $\mathsf{CONF} \Leftarrow \mathsf{AE}$.

Proposition 5 ($\mathsf{AE}+\mathsf{PA1}+\mathsf{INT\text{-}RUP} \Rightarrow \mathsf{AERUP}$)**.** *Let* $\mathrm{AE} = (\mathcal{E},\mathcal{D},\mathcal{V})$ *be an authenticated encryption scheme. Let* $\mathcal{S}$ *be a simulator. For any adversary* $\mathcal{A}$ *that makes* q_e *encryption,* q_d *decryption, and* q_v *verification queries,*

$$\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) \le \mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{B}_1) + \mathsf{PA1}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B}_2) + \mathsf{CONF}_{\mathrm{AE}}(\mathcal{B}_3)\,,$$

where $\mathcal{B}_1$ is adversary with total query complexity $q_e + q_d + q_v$ that includes encryption, decryption, and verification queries to its own oracles, $\mathcal{B}_2$ is an adversary with total query complexity $q_e + q_d$ that includes encryption and decryption queries to its own oracles, and $\mathcal{B}_3$ is an adversary with query complexity q_e.

Proof. Let $\mathcal{A}$ be an adversary that breaks the AERUP security of AE with advantage $\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A})$. We define an adversary $\mathcal{B}_1$ that has the same query complexity (i.e., $q_e + q_d + q_v$) as $\mathcal{A}$, interacts with a triplet of encryption, decryption, and verification oracles: it relays queries of $\mathcal{A}$ to its own oracles in the obvious way. If $\mathcal{A}$ does not make any valid verification query to $\mathcal{V}_K$, $\mathcal{B}_1$'s success is zero; otherwise, $\mathcal{B}_1$ forges with that message and tag coming from $\mathcal{A}$'s successful verification. Adversary $\mathcal{B}_2$, in turn, only has access to a pair of encryption and decryption oracles, and simulates the $\perp$ oracle: it relays encryption and decryption queries of $\mathcal{A}$ to its own oracles and for any verification query of $\mathcal{A}$, it always rejects. Finally, $\mathcal{B}_2$ returns whatever $\mathcal{A}$ returns. We also define adversary $\mathcal{B}_3$ that has access to an encryption oracle and simulates a simulator $\mathcal{S}$ and the $\perp$-oracle. $\mathcal{B}_3$ returns whatever $\mathcal{A}$ returns. By definition of $\mathcal{B}_1$, $\mathcal{B}_2$, and $\mathcal{B}_3$, we have

$$\begin{aligned}
\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) &= \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\$,\mathcal{S},\perp} \to 1\right) \\
&= \left(\mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\perp} \to 1\right)\right) \\
&\qquad + \left(\mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{D}_K,\perp} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{S},\perp} \to 1\right)\right) \\
&\qquad + \left(\mathbf{Pr}\left(\mathcal{A}^{\mathcal{E}_K,\mathcal{S},\perp} \to 1\right) - \mathbf{Pr}\left(\mathcal{A}^{\$,\mathcal{S},\perp} \to 1\right)\right) \\
&\overset{i}{=} \left(\mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{D}_K,\mathcal{V}_K} \text{ forges}\right) - \mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{D}_K,\perp} \text{ forges}\right)\right) \\
&\qquad + \left(\mathbf{Pr}\left(\mathcal{B}_2^{\mathcal{E}_K,\mathcal{D}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}_2^{\mathcal{E}_K,\mathcal{S}} \to 1\right)\right) \\
&\qquad + \left(\mathbf{Pr}\left(\mathcal{B}_3^{\mathcal{E}_K} \to 1\right) - \mathbf{Pr}\left(\mathcal{B}_3^{\$} \to 1\right)\right) \\
&= \mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{B}_1) + \mathsf{PA1}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{B}_2) + \mathsf{CONF}_{\mathrm{AE}}(\mathcal{B}_3)\,,
\end{aligned}$$

where $\overset{i}{=}$ follows from the fact that $\mathbf{Pr}\left(\mathcal{B}_1^{\mathcal{E}_K,\mathcal{D}_K,\perp} \text{ forges}\right) = 0$ as $\perp$ oracle always returns $\perp$. □

3.6 Comparison with Existing Notions

The comparison with conventional AE security (Definition 6) as well as the comparison with the standalone definitions of PA1 and INT-RUP security (Definitions 7 and 8) are already discussed. We compare AERUP security with various standalone authenticated encryption security notions *beyond* conventional security.

Hoang et al. [HKR15] introduced the notion of robust authenticated encryption, RAE. This notion covers a strong level of security, where the authenticated encryption should withstand attackers that do not obey to the nonce uniqueness (either there is no nonce or the nonce can be reused). In the RAE security notion, the goal of an adversary is to distinguish an authenticated encryption scheme from a random injective function. It bridges the gap between AE and SPRP security through a ciphertext expanding parameter λ. Examples of schemes satisfying this notion are AEZ and Encode-then-SPRP [HKR15]. Hoang et al. also considered a variant with decryption leakage, $\mathsf{RAE}_{\mathsf{sim}}$, in which a simulator simulates the decryption leakage without having any access to the query history. Note that this is a stronger model than PA1, where the simulator sees the communication between the adversary and $\mathcal{E}_K$. Our model of AERUP security is a variant of $\mathsf{RAE}_{\mathsf{sim}}$ where the simulator has access to the query-response history. This particularly means that $\mathsf{RAE}_{\mathsf{sim}} \Rightarrow \mathsf{AERUP}$.

Barwell et al. [BPS15] presented subtle authenticated encryption, SAE. The model is a generalization, basically a refinement, of RAE for nonce-based authenticated encryption, and covers several types of security definitions by varying the decryption oracle choices in the ideal world: it may be Γ_K (for some key-dependent leakage) or Γ (independent of the key). Note that the latter case implies $\mathsf{RAE}_{\mathsf{sim}}$.

Ashur et al. [ADL17] presented an alternative notion of RUP security, RUPAE. The notion focuses on nonce-based authenticated encryption, and RUPAE combines PA1 and INT-RUP in this setting with the restriction that the ideal model decryption is simply a random function. Ashur et al. proved GCM-RUP to be secure in the described nonce-based model. For comparison, in our case we are *not* relying on a nonce but neither are we simplifying the simulator by replacing it by a random function.

Although RAE and RUPAE are stronger than ours, it appears that these models may only be achieved by an "Encode then SPRP" construction, which is two-pass in both encryption and decryption (where "pass" refers to the number of times a message is sequentially fed to the construction). In contrast, nonce misuse security can be achieved in our relaxed model more efficiently, namely with a scheme whose decryption is single-pass. In summary, AERUP more closely matches the notion of RUP security than RAE, SAE, or RUPAE, in light of the equivalence

$$\mathsf{AE} + \mathsf{PA1} + \mathsf{INT\text{-}RUP} \Leftrightarrow \mathsf{AERUP}$$

proven in Sections 3.4 and 3.5. This equivalence also clearly exposes the main goal of the AERUP security model: to have one single security proof representing all different security notions required to hold for a RUP secure authenticated encryption scheme.

We note that Fouque et al. [FJMV03, FMP03] considered (concurrent) block-wise adaptive security of authenticated encryption. In this case, the adversary can see block-wise outputs of the scheme before committing to the next block input. From a decryption perspective, it is a stronger model than any of the models of discussed so far.

4 SUNDAE

We describe the specification of SUNDAE of Banik et al. [BBLT18] in the context of potential release of unverified plaintext, ergo with separated decryption and verification, in Section 4.1. We present our RUP attack against SUNDAE in Section 4.2.

4.1 Specification

The SUNDAE authenticated encryption scheme is built on top of a block cipher $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ for some $k, n \in \mathbb{N}$. Its key space is identical to that of E, $\{0,1\}^k$, but it accepts arbitrarily sized associated data A and message M. The encryption of SUNDAE is length preserving, meaning that the responded ciphertext is of size $|M|$. The tag is of size $\{0,1\}^n$. The encryption, decryption, and verification algorithms of SUNDAE are specified in Figure 1. It can be seen as a MAC-then-Encrypt construction based on GCBC1 (with minor adjustments) [Nan09] and OFB [FIP80]. An evaluation of SUNDAE on input of (A, M) with $|A|, |M| > 0$ is depicted in Figure 2.

4.2 RUP Insecurity

We describe a simple universal forgery attack against SUNDAE in the RUP setting.

Proposition 6 (INT-RUP insecurity of SUNDAE)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be the SUNDAE authenticated encryption scheme. There exists an adversary* $\mathcal{A}$ *such that*

$$\mathsf{INT\text{-}RUP}_{\mathrm{AE}}(\mathcal{A}) = 1\,.$$

Algorithm $\mathsf{MAC}(A, M)$

1. $b_1 \leftarrow |A| > 0 \ ? \ 1 \ : \ 0$
2. $b_2 \leftarrow |M| > 0 \ ? \ 1 \ : \ 0$
3. $T \leftarrow E_K(b_1 \| b_2 \| 0^{n-2})$
4. **if** $|A| > 0$ **then**
5. $\quad A[1 \dots a] \xleftarrow{n} A$
6. $\quad$ **for** $i = 1$ **to** $a - 1$
7. $\quad\quad T \leftarrow E_K(T \oplus A[i])$
8. $\quad X \leftarrow |A[a]| < n \ ? \ 2 \ : \ 4$
9. $\quad T \leftarrow E_K\big(X \cdot \big(T \oplus \mathsf{pad}_n(A[a])\big)\big)$
10. **if** $|M| > 0$ **then**
11. $\quad M[1 \dots m] \xleftarrow{n} M$
12. $\quad$ **for** $i = 1$ **to** $m - 1$
13. $\quad\quad T \leftarrow E_K(T \oplus M[i])$
14. $\quad X \leftarrow |M[m]| < n \ ? \ 2 \ : \ 4$
15. $\quad T \leftarrow E_K\big(X \cdot \big(T \oplus \mathsf{pad}_n(M[m])\big)\big)$
16. **return** T

Algorithm $\mathsf{OFB}(T, M)$

1. $M[1 \dots m] \xleftarrow{n} M$
2. $Z[0] \leftarrow T$
3. **for** $i = 1$ **to** m
4. $\quad Z[i] \leftarrow E_K(Z[i-1])$
5. $\quad C[i] \leftarrow \lfloor Z[i] \rfloor_{|M[i]|} \oplus M[i]$
6. **return** $C[1 \dots m]$

Algorithm $\mathcal{E}_K(A, M)$

1. $T \leftarrow \mathsf{MAC}(A, M)$
2. $C \leftarrow \mathsf{OFB}(T, M)$
3. **return** (T, C)

Algorithm $\mathcal{D}_K(A, T, C)$

1. $M \leftarrow \mathsf{OFB}(T, C)$
2. **return** M

Algorithm $\mathcal{V}_K(A, T, C)$

1. $M \leftarrow \mathsf{OFB}(T, C)$
2. $T' \leftarrow \mathsf{MAC}(A, M)$
3. **return** $T' = T \ ? \ \top \ : \ \bot$

Figure 1: Encryption, decryption and verification algorithm of SUNDAE.

The adversary makes 1 *query to* $\mathcal{E}$*,* 3 *queries to* $\mathcal{D}$*, and* 1 *query to* $\mathcal{V}$*.*

Proof. We show a simple universal forgery attack for arbitrary associated data $A = A[1 \dots a]$ and message $M = M[1 \dots m]$ with $a \geq 2$ and $m \geq 1$:

- $\mathcal{D}(\varepsilon, T_1, C_1[1])$ with $T_1 = 110^{n-2}$. Get $M_1[1]$;
- $\mathcal{D}(\varepsilon, T_2, C_2[1])$ with $T_2 = M_1[1] \oplus C_1[1] \oplus A[1]$. Get $M_2[1]$;
- $\mathcal{D}(\varepsilon, T_3, C_3[1])$ with $T_3 = M_1[1] \oplus C_1[1] \oplus A'[1]$ for some $A'[1] \neq A[1]$. Get $M_3[1]$;
- Let $\Delta := M_2[1] \oplus C_2[1] \oplus M_3[1] \oplus C_3[1]$;
- $\mathcal{E}(A'[1] \parallel A[2] \oplus \Delta \parallel A[3] \parallel \cdots \parallel A[a], M)$. Get the tag T and ciphertext C;
- $\mathcal{V}(A, T, C)$ is a valid forgery for the data A and message M.

In the attack, $C_1[1]$, $C_2[1]$, and $C_3[1]$ may take any n-bit value. □

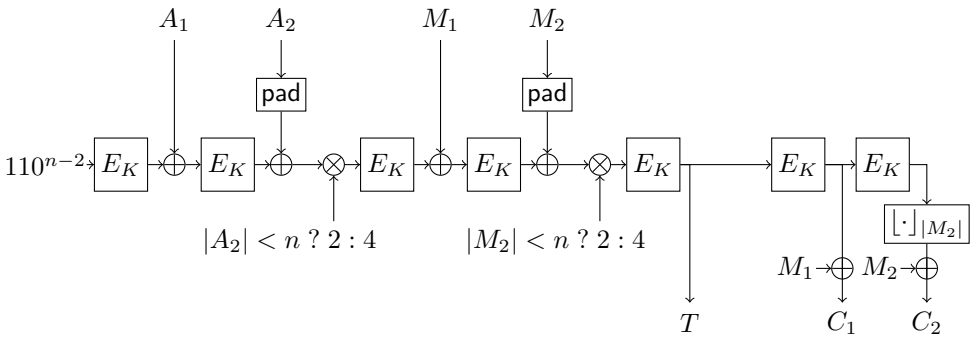

Figure 2: The authenticated encryption scheme SUNDAE (Section 4.1), for $0 < |A| \leq 2n$ and $0 < |M| \leq 2n$.

The core idea of the attack is that, in fact, a $\mathcal{D}$ query can be seen as a direct query to the underlying block cipher E_K which allows us to simulate step by step the tag generation process in $\mathcal{E}$. The attack works because from the first step we get $M_1[1] \oplus C_1[1] = E_K(110^{n-2})$ and the second and third queries allow us to compute Δ that is the internal state difference after the second block cipher call between processing $A[1]$ and $A'[1]$. After processing $A'[1] \parallel A[2] \oplus \Delta$ in the first case or $A[1] \parallel A[2]$ in the second case we get a full internal state collision for arbitrary $A[2]$ as we corrected the difference Δ. Hence the tag and ciphertext will remain equal whenever what follows is the same, which leads to this simple forgery.

The strategy is also applicable in the case where $A = \varepsilon$ or $M = \varepsilon$: we just need to use the proper starting value $b_1 \| b_2 \| 0^{n-2}$ instead of 110^{n-2}, and we can do the same trick with the first two processed blocks. The case of a single block data or message is also trivial as it generates the tag with 2 block cipher calls that we can fully simulate with 2 $\mathcal{D}$ queries.

The attack does not contradict the security claims of the original SUNDAE, but it is something to be aware of especially when implementing IoT products. Indeed, as already mentioned in the introduction, SUNDAE is claimed to offer maximal robustness to a lack of secure state.

5 ANYDAE

SUNDAE admits a RUP attack as the adversary can learn $E_K(T)$ for any choice $T \in \{0,1\}^n$. This allows it to "reconstruct" the forgery starting from the initial value 110^{n-2}, block cipher call by block cipher call. There are various solutions to counter this problem, which all center around adjusting the way the value T is generated or the value T is used to generate the encryption stream.

Naive solutions like transforming it through a block cipher call are undesirable: they increase state size (for the computation of the encrypted mask) or implementation size (for the implementation of the block cipher inverse). Instead, our focus is on updating the way T is used to generate the key stream. We do so by considering a generalized construction dubbed ANYDAE. The mode is specified in Section 5.1, we prove security of ANYDAE in Section 5.2, and we give two example instantiations in Sections 5.3 and 5.4.

5.1 Specification

The generalized ANYDAE authenticated encryption scheme is also built on top of a block cipher $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ for some $k, n \in \mathbb{N}$. In addition, it uses a formatting function to parse data to be encrypted, and mixing functions to process the state. Let $\mathcal{T}$

Algorithm $\mathsf{MAC}(A, M)$

1. $\big(B[1 \dots \ell], \delta[1 \dots \ell - 1]\big) \leftarrow \mathsf{Fmt}(A, M)$
2. $X[1] \leftarrow B[1]$
3. **for** $i = 1$ **to** $\ell - 1$
4. $\quad Y[i] \leftarrow E_K(X[i])$
5. $\quad X[i+1] \leftarrow \rho_1(Y[i], \delta[i]) \oplus B[i+1]$
6. $T \leftarrow E_K(X[\ell])$
7. **return** T

Algorithm $\mathsf{OFB}(T, M)$

1. $M[1 \dots m] \xleftarrow{n} M$
2. $U[1] \leftarrow \rho_2(T)$
3. **for** $i = 1$ **to** m
4. $\quad V[i] \leftarrow E_K(U[i])$
5. $\quad C[i] \leftarrow \lfloor V[i] \rfloor_{|M[i]|} \oplus M[i]$
6. $\quad U[i+1] \leftarrow \rho_3(V[i])$
7. **return** $C[1 \dots m]$

Algorithm $\mathcal{E}_K(A, M)$

1. $T \leftarrow \mathsf{MAC}(A, M)$
2. $C \leftarrow \mathsf{OFB}(T, M)$
3. **return** (T, C)

Algorithm $\mathcal{D}_K(A, T, C)$

1. $M \leftarrow \mathsf{OFB}(T, C)$
2. **return** M

Algorithm $\mathcal{V}_K(A, T, C)$

1. $M \leftarrow \mathsf{OFB}(T, C)$
2. $T' \leftarrow \mathsf{MAC}(A, M)$
3. **return** $T' = T \; ? \; \top \; : \; \bot$

Figure 3: Encryption, decryption and verification algorithm of ANYDAE.

be a (possibly empty) finite set. Let $\mathsf{Fmt}\colon \{0,1\}^* \to (\{0,1\}^n)^\ell \times (\mathcal{T})^{\ell-1}$ for any $\ell > 0$ be a *formatting function*, that takes an arbitrarily length bit string and generates a sequence of n-bit blocks of certain length and a sequence of elements of $\mathcal{T}$ of the same length minus one. Furthermore, consider the following three state processing functions:

$$\rho_1\colon \{0,1\}^n \times \mathcal{T} \to \{0,1\}^n\,, \qquad \rho_2\colon \{0,1\}^n \to \{0,1\}^n\,, \qquad \rho_3\colon \{0,1\}^n \to \{0,1\}^n\,.$$

ANYDAE has, just like SUNDAE, a key space identical to that of E, and it is also length preserving. The encryption, decryption, and verification algorithms of SUNDAE are specified in Figure 3. The message authentication part is simplified significantly due to the use of the formatting function to pre-process A and M. Then, ρ_1 is used to process the state in message authentication and ρ_2 and ρ_3 are used to process the state in the OFB encryption mode. The scheme is depicted in Figure 4.

5.2 Security of ANYDAE

We will prove that ANYDAE is an AERUP secure authenticated encryption scheme if Fmt is prefix-free, ρ_1, ρ_2, ρ_3 are sufficiently regular and/or differential-uniform, and certain conditions on the set of first block outputs of Fmt are satisfied.

Theorem 2 (AERUP security of ANYDAE)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be the* $\mathrm{ANYDAE}[\mathsf{Fmt}, \rho_1, \rho_2, \rho_3]$ *authenticated encryption scheme based on block cipher* $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$*. Denote by* $\mathcal{F}_1$ *the set of first block outputs of* Fmt*. If*

1. Fmt *is injective and prefix-free;*[2]
2. ρ_1 *is* ϵ_1*-differential-uniform and* γ_1*-regular;*

[2]For Fmt, prefix-freeness means that for any two elements $\big(B[1 \dots \ell], \delta[1 \dots \ell - 1]\big), \big(B'[1 \dots \ell'], \delta'[1 \dots \ell' - 1]\big) \in \mathrm{range}(\mathsf{Fmt})$ with $\ell < \ell'$, $\big(B[1 \dots \ell], \delta[1 \dots \ell - 1]\big) \neq \big(B'[1 \dots \ell], \delta'[1 \dots \ell - 1]\big)$.

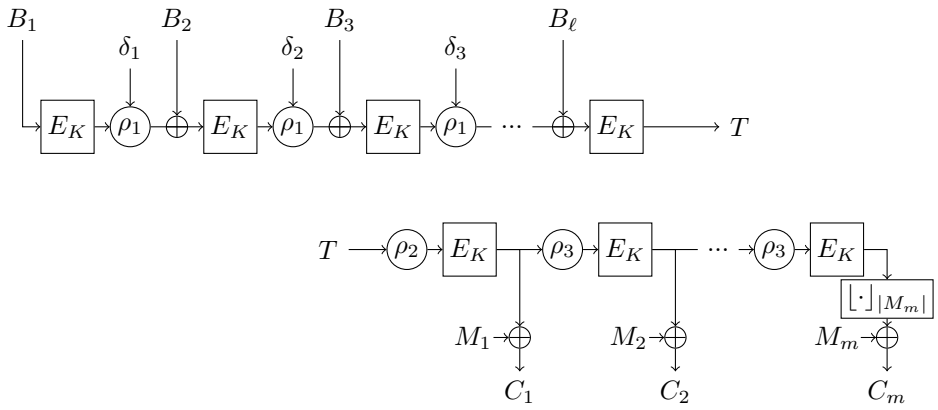

Figure 4: The authenticated encryption scheme ANYDAE (Section 5.1), where for authentication, A and M are preprocessed as $\big(B[1\dots\ell], \delta[1\dots\ell-1]\big) \leftarrow \mathsf{Fmt}(A, M)$.

3. *ρ_2 is γ_2-regular;*
4. *ρ_3 is γ_3-regular;*
5. $|\mathcal{F}_1 \cap \mathrm{range}(\rho_2)| = 0$,

then, for any adversary $\mathcal{A}$ having total complexity σ and making q_v verification queries, and operating in time t,

$$\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) \le \mathsf{PRP}_E(\mathcal{B}) + \binom{\sigma}{2}\Big(\frac{1}{2^n} + \max\{\epsilon_1, \gamma_1, \gamma_2, \gamma_3\}\Big) + \Omega\sigma\cdot\gamma_3 + \frac{q_v}{2^n}\,,$$

where $\Omega := |\mathcal{F}_1 \cap \mathrm{range}(\rho_3)|$ and $\mathcal{B}$ is some adversary that makes σ queries to its oracle and operates in time $t' \approx t$, and $\mathcal{S}$ is some simulator that is described in the proof.

The proof of Theorem 2 is given in Section 6. The simulator that is used, is described within the proof. Although the theorem looks complex due to the conditions dictated on Fmt, ρ_1, ρ_2, and ρ_3, it can easily be mapped to simple examples.

SUNDAE of Section 4 is an instance of ANYDAE, but Theorem 2 does not apply as condition 5 is violated: the cardinality of the intersection of $\mathcal{F}_1$ and range(ρ_2) is at least 1. In Section 5.3, we describe MONDAE, arguably the simplest instance of ANYDAE that in addition only differs from SUNDAE of Banik et al. [BBLT18] by a simple fix_1 function. In Section 5.4, we describe TUESDAE, a variant that is optimal in the number of block cipher calls for most of the possible inputs.

5.3 Example 1: MONDAE (RUP Secure SUNDAE)

Recall from Section 4.2 that SUNDAE is not RUP secure. It appears that it is surprisingly simple to salvage SUNDAE. To demonstrate this, we propose MONDAE, that differs *only in* the fact that we insert a fix_1 function right after the value T is output. More detailed, line 2 in the OFB encryption part of Figure 1 gets replaced by

$$Z[0] \leftarrow \mathsf{fix}_1(T)\,.$$

For the case of positive-length associated data and message, the MONDAE construction is depicted in Figure 5. The tweak works because, simply said, it prevents the adversary from learning $E_K(T)$ for any choice of $T \in \{0,1\}^n$.

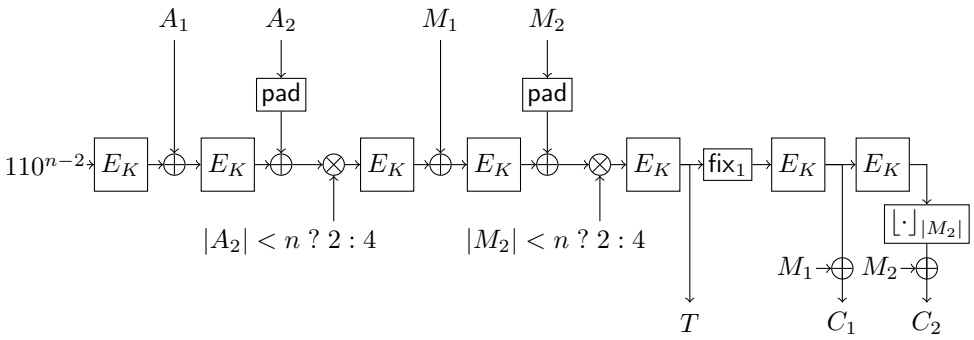

Figure 5: The authenticated encryption scheme MONDAE (Section 5.3), for $0 < |A| \leq 2n$ and $0 < |M| \leq 2n$.

Function $\mathsf{Fmt}(A, M)$

1. $b_1 \leftarrow |A| > 0 \ ? \ 1 \ : \ 0$
2. $b_2 \leftarrow |M| > 0 \ ? \ 1 \ : \ 0$
3. $\ell \leftarrow a + m + 1$
4. $B[1] \leftarrow b_1 \| b_2 \| 0^{n-2}$
5. **for** $i = 2$ **to** a
6. $\quad \delta[i-1] \leftarrow 1$
7. $\quad B[i] \leftarrow A[i-1]$
8. $\delta[a] \leftarrow |A[a]| < n \ ? \ 2 \ : \ 4$
9. $B[a+1] \leftarrow \delta[a] \cdot \mathsf{pad}_n(A[a])$
10. **for** $i = 2$ **to** m
11. $\quad \delta[a+i-1] \leftarrow 1$
12. $\quad B[a+i] \leftarrow M[i-1]$
13. $\delta[\ell - 1] \leftarrow |M[m]| < n \ ? \ 2 \ : \ 4$
14. $B[\ell] \leftarrow \delta[\ell - 1] \cdot \mathsf{pad}_n(M[m])$
15. **return** $\big(B[1 \ldots \ell], \delta[1 \ldots \ell - 1]\big)$

Figure 6: The formatting function Fmt of MONDAE.

It is clear to see that, in fact, MONDAE is a specific instantiation of the generalized construction ANYDAE, namely by defining

$$\rho_1(S, X) = X \cdot S\,, \qquad \rho_2(S) = \mathsf{fix}_1(S)\,, \qquad \rho_3(S) = S\,,$$

and taking the formatting function Fmt of Figure 6. This means that MONDAE is not only unforgeable in the RUP setting: it meets AERUP security of Definition 9. The result is a direct corollary of Theorem 2, noting that for MONDAE, we have $\epsilon_1 = \gamma_1 = \gamma_3 = 2^{-n}$, $\gamma_2 = 2 \cdot 2^{-n}$, and $\Omega = 4$ (as $|\mathcal{F}_1| = 4$, where $\mathcal{F}_1 = \{0^n, 10^{n-1}, 010^{n-2}, 110^{n-2}\}$ for accommodating empty or non-empty associated data and empty or non-empty message blocks, and $|\operatorname{range}(\rho_3)| = 2^n$).

Corollary 2 (AERUP security of MONDAE)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be the* MONDAE *authenticated encryption scheme based on block cipher* $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$. *For any adversary* $\mathcal{A}$ *having total complexity* σ *and making* q_v *verification queries, and operating in time* t*,*

$$\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) \leq \mathsf{PRP}_E(\mathcal{B}) + \frac{1.5\sigma^2}{2^n} + \frac{2.5\sigma}{2^n} + \frac{q_v}{2^n}\,,$$

where $\mathcal{B}$ *is some adversary that makes* σ *queries to its oracle and operates in time* $t' \approx t$*, and* $\mathcal{S}$ *is some simulator that is described in the proof of Theorem 2.*

Function $\mathsf{Fmt}(A, M)$

1. $D \xleftarrow{n} A \| M$
2. **if** $a = m = 0$ **then**
3. $\quad B[1] \leftarrow 0^n$
4. $\quad$ **return** $B[1]$ ▷ **Case A**
5. **if** $a + m = 1$ and $|D| \leq n - 5$ **then**
6. $\quad B[1] \leftarrow \mathsf{pad}_{n-4}(D[1]) \| \mathsf{type} \| 100$
7. $\quad$ **return** $B[1]$ ▷ **Case B**
8. $B[1] \leftarrow \mathsf{fix}_1(\mathsf{pad}_n(D[1]))$
9. $b \leftarrow \lfloor \mathsf{pad}_n(D[1]) \rfloor_1$
10. **if** $a + m = 1$ and $|D| \geq n - 4$ **then**
11. $\quad \delta[1] \leftarrow \mathsf{full}_{D[1]} \| 011 \| b$
12. $\quad B[2] \leftarrow 0^* \| \mathsf{type} \| 010$
13. $\quad$ **return** $(B[1 \dots 2], \delta[1])$ ▷ **Case C**
14. **if** $a + m = 2$ **then**
15. $\quad \delta[1] \leftarrow \mathsf{bin}(a)_2 \| \mathsf{full}_{M[m]} \| \mathsf{full}_{A[a]} \| b$
16. $\quad B[2] \leftarrow \mathsf{pad}_n(D[2])$
17. $\quad$ **return** $(B[1 \dots 2], \delta[1])$ ▷ **Case D**
18. **if** $a + m > 2$ **then**
19. $\quad$ **if** $a < 3$ **then**
20. $\quad\quad \delta[1] \leftarrow \mathsf{bin}(a + 5)_3 \| \mathsf{full}_{A[a]} \| b$
21. $\quad$ **else**
22. $\quad\quad \delta[1] \leftarrow 0001 \| b$
23. $\quad \ell \leftarrow a + m$
24. $\quad$ **for** $i = 2$ **to** ℓ
25. $\quad\quad B[i] \leftarrow \mathsf{pad}_n(D[i])$
26. $\quad$ **for** $i = 2$ **to** $\ell - 1$
27. $\quad\quad \delta[i] \leftarrow 00 \| \mathsf{empty}_i \| \mathsf{final}_{D[i+1]} \| \mathsf{full}_{D[i+1]}$
28. $\quad$ **return** $(B[1 \dots \ell], \delta[1 \dots \ell - 1])$ ▷ **Case E**

Figure 7: The formatting function Fmt of TUESDAE.

5.4 Example 2: TUESDAE (RUP Secure Optimal ANYDAE)

We now consider the problem of optimizing ANYDAE, or stating differently, defining the functions ρ_1, ρ_2, ρ_3, and the formatting function in such a way that ANYDAE becomes optimal in the number of block cipher invocations for most inputs. We call the resulting construction TUESDAE. For associated data of a blocks and a message of m blocks with $a + m > 1$, one encryption using TUESDAE consists of exactly $a + 2m$ block cipher invocations. Moreover, if (a, m) is $(1, 0)$ or $(0, 1)$, TUESDAE requires exactly 1 block cipher invocation if the length of the data block (associated data in the former case and message in the latter case) is less than $n - 4$ bits. Note that, as we need to distinguish on the one hand between associated data and message and on the other hand between partial and full data, it is impossible for any deterministic authenticated encryption scheme to just have a single block cipher call for a single (full) data block. The choice we made for ρ_1 requires, for good security properties, $n - 1$ to be prime so it is well suited for use along with a typical $n = 128$-bit block cipher.

For $X \in \{0, 1\}^n$, write $X \ggg t$ as the right rotation of X by t bits, and write

$$X \overline{\ggg} t := \lfloor X \rfloor_1 \parallel (\lceil X \rceil_{n-1} \ggg t) \ .$$

TUESDAE is an instantiation of ANYDAE by defining

$$\rho_1(S, X) = S \overline{\ggg} X \,, \qquad \rho_2(S) = \mathsf{fix}_{10}(S) \,, \qquad \rho_3(S) = \mathsf{fix}_{10}(S) \,,$$

and taking the formatting function Fmt of Figure 7. Here, we use type to indicate whether the current data block is associated data ($\mathsf{type} = 0$) or message ($\mathsf{type} = 1$), we use full_X to indicate whether X is n-bit ($\mathsf{full}_X = 1$) or partial ($\mathsf{full}_X = 0$), and we use final_X to indicate whether X is a final block of its type ($\mathsf{final}_X = 1$) or not ($\mathsf{final}_X = 0$). We define empty_i to be 1 if and only if $i = a - 1$ and $m = 0$. Finally, $\mathsf{bin}(a)_i$ denotes the i-bit binary representation of an integer a.

It is easy to verify that Fmt is prefix-free: the three rightmost bits of $B[1]$ are 000 in case A, 100 in case B, and $**1$ in cases C, D, and E. For the last three cases, difference is in $\delta[1]$: it equals $*011*$ for case C, either of $\{00*0*, 01****, 100**\}$ for case D, and either of $\{1010*, 110**, 111**, 0001*\}$ for case E. Here, for case E, distinction is made using $\mathsf{empty}_i \| \mathsf{final}_{D[i+1]} \| \mathsf{full}_{D[i+1]}$.

Also note that for any $B[1]$, $\lfloor B[1] \rfloor_2 \neq 10$. This implies that $|\mathcal{F}_1 \cap \mathrm{range}(\rho_3)| = 0$. Security of TUESDAE is therefore a direct corollary of Theorem 2 for $\epsilon_1 = \gamma_2 = \gamma_3 = 4 \cdot 2^{-n}$ if $n-1$ is prime,[3], $\gamma_1 = 2^{-n}$, and $\Omega = 0$.

Corollary 3 (AERUP security of TUESDAE)**.** *Let* $\mathrm{AE} = (\mathcal{E}, \mathcal{D}, \mathcal{V})$ *be the* TUESDAE *authenticated encryption scheme based on block cipher* $E \colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$*, where* $n-1$ *is prime. For any adversary* $\mathcal{A}$ *having total complexity* σ *and making* q_v *verification queries, and operating in time* t*,*

$$\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{AE}}(\mathcal{A}) \leq \mathsf{PRP}_E(\mathcal{B}) + \frac{2.5\sigma^2}{2^n} + \frac{q_v}{2^n}\,,$$

where $\mathcal{B}$ *is some adversary that makes* σ *queries to its oracle and operates in time* $t' \approx t$*, and* $\mathcal{S}$ *is some simulator that is described in the proof of Theorem 2.*

Note that the result only applies for prime $n-1$, as in this case, ρ_1 is differential-universal. However, the choice of ρ_1 is not strict in TUESDAE; any good differential-universal function suits.

Remark 1. We briefly elaborate on the differences between MONDAE and TUESDAE. For any query with associated data of a blocks and message of m blocks, with $a + m > 1$, one encryption using TUESDAE consists of an optimal number of $a + 2m$ block cipher invocations, whereas MONDAE requires $a + 2m + 1$ invocations. Moreover, if (a, m) is $(1, 0)$ or $(0, 1)$, and the length of the data block is less than $(n-4)$ bits, then TUESDAE requires an optimal amount of one block cipher invocation, whereas MONDAE requires one additional block cipher invocation. To achieve this optimality, TUESDAE needs to execute a few conditional statements (see Figure 7). These would require some additional multiplexers that results in a slight increase of the hardware area.

6 Proof of Theorem 2

We consider any adversary $\mathcal{A}$ that has access to either $(\mathcal{E}_K, \mathcal{D}_K, \mathcal{V}_K)$ for $K \xleftarrow{\$} \{0,1\}^k$ or $(\$, \mathcal{S}, \bot)$ for $\$ \xleftarrow{\$} \mathrm{F}(*, t)$ and $\mathcal{S}$ some simulator that we will define later on, and tries to distinguish both worlds. The adversary has encryption complexity σ_e, decryption complexity σ_d, and verification complexity σ_v, with $\sigma_e + \sigma_d + \sigma_v = \sigma$, and operates in time t. As a first step, we replace E_K by a random permutation $P \xleftarrow{\$} \mathrm{P}(n)$, at the cost of $\mathsf{PRP}_E(\mathcal{B})$ for some distinguisher $\mathcal{B}$ that makes σ queries to its oracle and operates in time $t' \approx t$. As a second step, we switch from P to a random function $R \xleftarrow{\$} \mathrm{F}(n)$ at the cost of $\binom{\sigma}{2}/2^n$. For brevity, denote the resulting construction by $\Pi = (\mathcal{E}[R], \mathcal{D}[R], \mathcal{V}[R])$. We have thus obtained

$$\mathsf{AERUP}^{\mathcal{S}}_{\mathrm{ANYDAE}}(\mathcal{A}) \leq \mathsf{PRP}_E(\mathcal{B}) + \binom{\sigma}{2}/2^n + \mathsf{AERUP}^{\mathcal{S}}_{\Pi}(\mathcal{A})\,, \tag{10}$$

and our focus is on upper bounding the remaining distance $\mathsf{AERUP}^{\mathcal{S}}_{\Pi}(\mathcal{A})$.

[3] Contini and Yin [CY99] showed that if $|S|$ is prime and $t \leq |S|$, $S \ggg t$ is $(2^{|S|-1})^{-1}$-differential-uniform.

6.1 Defining Adversary and Oracles

Without loss of generality, $\mathcal{A}$ is deterministic. Suppose it makes q_e encryption queries $(A_i^+, M_i^+)_{i=1}^{q_e}$ to the encryption oracle, where the block lengths of A_i^+ and M_i^+ are denoted by a_i^+ and m_i^+, with an aggregate of total σ_e blocks, q_d decryption queries $(A_i^-, C_i^-, T_i^-)_{i=1}^{q_d}$ to the decryption oracle, where the block lengths of A_i^- and C_i^- are denoted by a_i^- and c_i^-, with an aggregate of total σ_d blocks, and q_v verification queries $(A_i^\star, C_i^\star, T_i^\star)_{i=1}^{q_v}$ to the verification oracle, where the block lengths of $A_i^\star$ and $C_i^\star$ are denoted by $a_i^\star$ and $c_i^\star$, with an aggregate of total σ_v blocks. We assume that $\mathcal{A}$ is non-trivial and non-repeating, which means that all queries are distinct and there is no $(A_i^\star, C_i^\star, T_i^\star)$ that is an answer of an earlier encryption query. By $(i, *)$, we mean the i-th message of type $*$, where $* \in \{+, -, \star\}$. We use the notation $(j, \circledast) \prec (i, *)$ to denote that j-th message of type $\circledast$ was queried prior to the i-th message of type $*$.

6.1.1 Description of the Real World

The real world $\mathcal{O}_{\text{re}}$ consists of the encryption oracle $\Pi.\mathcal{E}[R]$, the decryption oracle $\Pi.\mathcal{D}[R]$, and the verification oracle $\Pi.\mathcal{V}[R]$ as outlined above. After the adversary has made all its queries, the oracles release all the internal variables from Figure 3. The encryption and verification oracles reveal all (X, Y)'s corresponding to authentication and all (U, V)'s corresponding to encryption. The decryption oracle reveals all (U, V)'s corresponding to decryption (the oracle does not verify the MAC). Note that there is some redundancy in the values, as the U's can be deduced from the values M, C, and V, but we reveal these for completeness.

6.1.2 Description of the Ideal World

The ideal world $\mathcal{O}_{\text{id}}$ consists of three oracles $(\$, \mathcal{S}, \perp)$. The verification oracle $\perp$ simply responds with the $\perp$-sign for each input $(A_i^\star, C_i^\star, T_i^\star)$. We will elaborate on the remaining two oracles, encryption $\$$ and decryption $\mathcal{S}$, in detail. For these two oracles, we maintain an initially empty table $\mathcal{L}$ to store (U, V)-tuples. Note that, as we work in the PA1-setting, i.e., $\mathcal{S}$ has insight in the queries made to the encryption oracle, this is sound.

The encryption oracle $\$$ is a random function that for each input $(A_i^+, M_i^+) = (A_i^+[1 \dots a_i^+], M_i^+[1 \dots m_i^+])$ generates a ciphertext and tag as

$$C_i^+ = C_i^+[1 \dots m_i^+] \xleftarrow{\$} \{0,1\}^{|M_i^+|}\,,$$
$$T_i^+ \xleftarrow{\$} \{0,1\}^n\,.$$

For later purposes, $\$$ will *in addition* set the following internal variables, which correspond to the inputs and outputs of R that are determined by M_i^+, C_i^+, T_i^+:

$$\big(U_i^+[k]\,,\, V_i^+[k]\big) \leftarrow \begin{cases} \big(\rho_2(T_i^+)\,, & M_i^+[1] \oplus C_i^+[1]\big)\,, \text{ for } k = 1\,, \\ \big(\rho_3(V_i^+[k-1])\,, & M_i^+[k] \oplus C_i^+[k]\big)\,, \text{ for } k = 2, \dots, m_i^+\,. \end{cases}$$

It stores all the individual (U_i^+, V_i^+) tuples in table $\mathcal{L}$.

The decryption oracle $\mathcal{S}$ is a simulator that we define to operate as follows on input of a query $(A_i^-, C_i^-, T_i^-) = (A_i^-[1, \dots, a_i^-], C_i^-[1, \dots, c_i^-], T_i^-)$:

1. $k \leftarrow 1$
2. $U_i^-[1] \leftarrow \rho_2(T_i^-)$
3. **while** $U_i^-[k] \in \mathcal{L}$ **do**
4. $\quad V_i^-[k] \leftarrow \mathcal{L}(U_i^-[k])$
5. $\quad M_i^-[k] \leftarrow V_i^-[k] \oplus C_i^-[k]$
6. $\quad U_i^-[k+1] \leftarrow \rho_3(V_i^-[k])$
7. $\quad k \leftarrow k+1$
8. **for** $j = k$ **to** c_i^-
9. $\quad M_i^-[j] \xleftarrow{\$} \{0,1\}^n$
10. $\quad V_i^-[j] \leftarrow M_i^-[j] \oplus C_i^-[j]$
11. $\quad U_i^-[j] \leftarrow \rho_3(V_i^-[j-1])$
12. $\quad$ add $(U_i^-[j], V_i^-[j])$ to $\mathcal{L}$
13. **return** $M_i^-[1 \dots c_i^-]$

Once the adversary has made all queries, we move to an *offline* phase where the adversary will be given the internal values (X, Y) and (U, V), just like in the real world. Note that the (U, V)'s have already been defined for encryption and decryption oracle. For any input query $(A_i^\star, C_i^\star, T_i^\star)$, verification oracle $\perp$ defines (U, V) in exactly the similar way as the decryption oracle defines for an input query (A_i^-, C_i^-, T_i^-) and also determines the underlying message $M_i^\star[1 \dots c_i^\star]$ which is released to the adversary. For the (X, Y)'s we use the following technique to define them. Note that we only have to focus on the encryption and verification queries; we do not bother about the (X, Y)'s for decryption queries as a decryption call does not verify the tag. For any query $(i, *)$ with $* \in \{+, \star\}$, we first compute the function $\mathsf{Fmt}(A_i^*, M_i^*)$ to obtain the sequence

$$B_i^* := \left(B_i^*[1 \dots \ell_i^*], \delta_i^*[1 \dots, \ell_i^* - 1]\right).$$

Let $(j, \circledast) \prec (i, *)$ with $\circledast \in \{+, \star\}$ be a query for which B_i^* has the longest common prefix with $B_j^\circledast$. Let $p < \ell_i^*$ be the length of the longest common prefix of B_i^* and $B_j^\circledast$. Next, we set $Y_i^*[k] \leftarrow Y_j^\circledast[k]$ for $1 \le k \le p$, and $Y_i^*[k] \xleftarrow{\$} \{0,1\}^n$, for $p+1 \le k \le \ell_i^*$. Finally, we set $X_i^*[1] \leftarrow B_i^*[1]$ and for $2 \le k \le \ell_i^*$, we have $X_i^*[k] \leftarrow \rho(Y_i^*[k-1], \delta_i^*[k-1]) \oplus B_i^*[k]$.

Finally, when the sampling of internal values is over, $\mathcal{O}_{\mathrm{id}}$ returns all the internal values. These are

$$\begin{aligned}(X_i^+, Y_i^+) &= (X_i^+[1 \dots \ell_i^+], Y_i^+[1 \dots \ell_i^+]),\\ (U_i^+, V_i^+) &= (U_i^+[1 \dots m_i^+], V_i^+[1 \dots m_i^+])\end{aligned}$$

for each encryption query $(A_i^+, M_i^+, C_i^+, T_i^+)$,

$$(U_i^-, V_i^-) = (U_i^-[1 \dots c_i^-], Y_i^-[1 \dots c_i^-])$$

for each decryption query $(A_i^-, M_i^-, C_i^-, T_i^-)$, and

$$\begin{aligned}(X_i^\star, Y_i^\star) &= (X_i^\star[1 \dots \ell_i^\star], Y_i^\star[1 \dots \ell_i^\star]),\\ (U_i^\star, V_i^\star) &= (U_i^\star[1 \dots m_i^\star], V_i^\star[1 \dots m_i^\star])\end{aligned}$$

for each verification query $(A_i^\star, M_i^\star, C_i^\star, T_i^\star, b_i^\star)$.

6.1.3 Attainable Transcripts

The overall transcript of the attack is $\tau = (\tau_e, \tau_d, \tau_v)$, where

$$\begin{aligned}\tau_e &= (A_i^+, M_i^+, C_i^+, T_i^+, X_i^+, Y_i^+, U_i^+, V_i^+)_{i=1}^{q_e},\\ \tau_d &= (A_i^-, M_i^-, C_i^-, T_i^-, U_i^-, V_i^-)_{i=1}^{q_d},\\ \tau_v &= (A_i^\star, M_i^\star, C_i^\star, T_i^\star, X_i^\star, Y_i^\star, U_i^\star, V_i^\star, b_i^\star)_{i=1}^{q_v}.\end{aligned}$$

A transcript $\tau = (\tau_e, \tau_d, \tau_v)$ is said to be *attainable* (with respect to $\mathcal{A}$) if the probability to realize this transcript in the ideal world $\mathcal{O}_{\mathrm{id}}$ is non-zero. Note that, particularly, for an attainable transcript τ, any verification query in τ_v satisfies $b_i^\star = \bot$. Following Section 2.3, we denote by Θ the set of all attainable transcripts, and by X_{re} and X_{id} the probability distributions of transcript τ induced by the real world and ideal world, respectively.

6.2 Definition of Bad Transcripts

We say that an attainable transcript τ is *bad* if one of the following events hold:

$$\begin{aligned}
\mathsf{Coll}_{\mathsf{XX}}\colon \exists (j,\circledast) \preceq (i,*), k, k' \quad & \text{with } \big(k \neq k' \text{ or } B^*[1\dots k'] \neq B^{\circledast}[1\dots k']\big) \\
& \text{such that } X_i^*[k'] = X_j^{\circledast}[k]\,, \\
\mathsf{Coll}_{\mathsf{XU}}\colon \exists (j,\circledast), (i,*), k, k' \quad & \text{such that } U_i^*[k'] = X_j^{\circledast}[k]\,, \\
\mathsf{Coll}_{\mathsf{UU}}\colon \exists (j,\circledast) \preceq (i,*), k, k' \quad & \text{with } \big(* = + \text{ or } U_i^*[1] \neq U_j^{\circledast}[k-k'+1]\big) \\
& \text{such that } U_i^*[k'] = U_j^{\circledast}[k]\,, \\
\mathsf{Forge}\colon \exists (i,\star) \quad & \text{such that } Y_i^\star[\ell^\star] = T_i^\star\,.
\end{aligned}$$

Note that, considering the real world, $\mathsf{Coll}_{\mathsf{XX}}$ denotes the event of an accidental collision between two inputs to R in the authentication part, where we exclude trivial collisions due to common prefix. Event $\mathsf{Coll}_{\mathsf{XU}}$ corresponds to accidental collisions between an input to R in the authentication and one in the encryption part. Event $\mathsf{Coll}_{\mathsf{UU}}$ corresponds to accidental collisions between two inputs to R in the encryption part, where we exclude trivial collisions triggered by a decryption query for a known U-value. Event Forge corresponds to the event that for any verification query, the last block cipher output in the MAC function collides with the given tag in the verification query.

In line with the H-coefficient technique (Theorem 1), Θ_{b} denotes the set of all attainable transcripts that are bad.

6.3 Probability of Bad Transcripts

We now bound the probability of a bad event in the ideal world.

Lemma 1. *Let X_{id} and Θ_{b} be as defined as above. Then,*

$$\mathbf{Pr}\left(X_{\mathrm{id}} \in \Theta_{\mathrm{b}}\right) \leq \binom{\sigma}{2} \cdot \max\{\epsilon_1, \gamma_1, \gamma_2, \gamma_3\} + \Omega\sigma \cdot \gamma_3 + \frac{q_v}{2^n}\,.$$

Proof. By applying the union bound,

$$\mathbf{Pr}\left(X_{\mathrm{id}} \in \Theta_{\mathrm{b}}\right) \leq \mathbf{Pr}\left(\mathsf{Coll}_{\mathsf{XX}}\right) + \mathbf{Pr}\left(\mathsf{Coll}_{\mathsf{XU}}\right) + \mathbf{Pr}\left(\mathsf{Coll}_{\mathsf{UU}}\right) + \mathbf{Pr}\left(\mathsf{Forge}\right)\,,$$

and we bound the three probabilities individually. We let $\#X$ be the number of X's in the transcript and $\#U$ the number of U's.

Bounding $\mathsf{Coll}_{\mathsf{XX}}$. Note that $X_i^*[1] = B_i^*[1]$, and $X_i^*[k'] = \rho_1(Y_i^*[k'-1], \delta_i^*[k'-1]) \oplus B_i^*[k']$ for $k' \neq 1$. We consider the following cases:

(i) $k = k' = 1$. The event is a contradiction as we require $B_i^*[1] \neq B_j^{\circledast}[1]$. The event is set with probability 0;

(ii) $k = 1, k' \neq 1$. The event implies that $\rho_1(Y_i^*[k'-1], \delta_i^*[k'-1]) = B_j^{\circledast}[1] \oplus B_i^*[k']$. As ρ_1 is γ_1-regular, the probability that this event happens is at most γ_1;

(iii) $k \neq 1, k' = 1$. The event implies that $\rho_1(Y_j^{\circledast}[k-1], \delta_j^{\circledast}[k-1]) = B_i^*[1] \oplus B_j^{\circledast}[k]$. As ρ_1 is γ_1-regular, the probability that this event happens is at most γ_1;

(iv) $k \neq 1, k' \neq 1$. The event implies that

$$\rho_1(Y_i^*[k'-1], \delta_i^*[k'-1]) \oplus \rho_1(Y_j^{\circledast}[k-1], \delta_j^{\circledast}[k-1]) = B_i^*[k'] \oplus B_j^{\circledast}[k]\,.$$

To bound the above event, we split it into two different subcases:

Case (a): When $k = k'$ and $B_i^*[1] = B_j^{\circledast}[1], (B_i^*[s], \delta_i^*[s-1]) = (B_i^{\circledast}[s], \delta_j^{\circledast}[s-1])$ for $2 \leq s \leq k-1$. This implies that $Y_i^*[k-1] = Y_j^{\circledast}[k-1]$. Now there are two different subcases. If $\delta_i^*[s] = \delta_j^{\circledast}[s]$, then we require $B_i^*[k] \neq B_j^{\circledast}[k]$ and the probability of the above event is zero. Otherwise, the above event boils down to

$$\rho_1(Y_i^*[k-1], \delta_i^*[k-1]) \oplus \rho_1(Y_j^{\circledast}[k-1], \delta_j^{\circledast}[k-1]) = B_i^*[k] \oplus B_j^{\circledast}[k]\,.$$

This event is bounded by the ϵ_1-differential-uniformity of ρ_1, where the probability is calculated over the random sampling of $Y_i^*[k-1]$, where we assume that $(j, \circledast) \prec (i, *)$.

Case (b): Otherwise, $Y_i^*[k'-1]$ and $Y_j^{\circledast}[k-1]$ are independent and the above event is bounded by the regular probability γ_1 of ρ_1 function that directly follows from Corollary 1.

Combining all the four cases, we obtain

$$\mathbf{Pr}\left(\mathsf{Coll}_{\mathsf{XX}}\right) \leq \binom{\#X}{2} \cdot \max\{\epsilon_1, \gamma_1\}\,.$$

Bounding $\mathsf{Coll}_{\mathsf{XU}}$. We consider the following cases:

(i) $k = k' = 1$. We have $|\mathcal{F}_1 \cap \mathrm{range}(\rho_2)| = \emptyset$, and hence, the probability that $\mathsf{Coll}_{\mathsf{XU}}$ is set is 0;

(ii) $k' \neq 1, k = 1$. The event implies that $\rho_3(C_i^*[k'] \oplus M_i^*[k']) = B_j^{\circledast}[1]$, or more generally that $\rho_3(C_i^*[k'] \oplus M_i^*[k']) \in \mathcal{F}_1$. As $|\mathcal{F}_1 \cap \mathrm{range}(\rho_3)| = \Omega$, and as ρ_3 is γ_3-regular, the probability that this event happens is at most $\Omega \# U \cdot \gamma_3$, where we have already summed over all possible query choices;

(iii) $k' = 1, k \neq 1$. The event implies that $\rho_1(Y_j^{\circledast}[k-1], \delta_j^{\circledast}[k-1]) = U_i^*[1] \oplus B_j^{\circledast}[k]$. As ρ_1 is γ_1-regular, the probability that this event happens is at most γ_1;

(iv) $k \neq 1, k' \neq 1$. The event implies $\rho_3(C_i^*[k'] \oplus M_i^*[k']) = \rho_1(Y_j^{\circledast}[k-1], \delta_j^{\circledast}[k-1]) \oplus B_j^{\circledast}[k]$. If $(j, \circledast) \prec (i, *)$, we can bound this event by γ_3 due to the random sampling of $C_i^*[k']$ or $M_i^*[k']$ and ρ_3 being γ_3-regular. Otherwise, we bound the event by γ_1 as ρ_1 is γ_1-regular.

For the third case, we have already summed over all possible occurrences of the case. The second and fourth case together occur at most $\#X \cdot \#U$ times. We therefore obtain

$$\mathbf{Pr}\left(\mathsf{Coll}_{\mathsf{XU}}\right) \leq (\#X \cdot \#U) \cdot \max\{\gamma_1, \gamma_3\} + \Omega \# U \cdot \gamma_3\,.$$

Bounding $\mathsf{Coll}_{\mathsf{UU}}$. We consider the following cases:

(i) $k' = 1$.

(a) $* \neq +$. The event is a contradiction as we require $U_i^*[1] \neq U_j^{\circledast}[k]$. The event is set with probability 0;

(b) $* = +$. The event implies $\rho_2(T_i^+) = U_j^{\circledast}[k]$ for some previous request $(j, \circledast)$ and some k. Since T_i^+ is always sampled uniformly at random. As ρ_2 is γ_2-regular, the probability that this event happens is at most γ_2;

(ii) $k' \neq 1$.

(a) $* \neq +$. The event implies $U_i^*[1] \neq U_j^{\circledast}[k-k'+1]$. This implies that there is an index $h \leq k'$ where the sequence merges, that is $U_i^*[k'-h-1] \neq U_j^{\circledast}[k-h-1]$ and $U_i^*[k'-h] = U_j^{\circledast}[k-h]$. If $U_i^*[k'-h-1] \in \mathcal{L}$, the event already happened before and this was already a bad transcript. Else, if $U_i^*[k'-h-1] \notin \mathcal{L}$, the event occurs when $\rho_3(V_i^*[k'-h-1]) = U_j^{\circledast}[k-h]$ with $V_i^*[k'-h-1]$ sampled uniformly at random. As ρ_3 is γ_3-regular, the probability that this event happens is at most γ_3;

(b) $* = +$. The event implies $\rho_3(V_i^+[k'-1]) = U_j^{\circledast}[k]$ for some previous request $(j, \circledast)$ and some k. Since $C_i^+[k'-1]$ is always sampled uniformly at random, so is $V_i^+[k'-1]$. As ρ_3 is γ_3-regular, the probability that this event happens is at most γ_3.

We obtain

$$\mathbf{Pr}\left(\mathsf{Coll}_{\mathsf{UU}}\right) \leq \binom{\#U}{2} \cdot \max\{\gamma_2, \gamma_3\}\,.$$

Bounding Forge**.** For a fixed verification query, the event is trivially bounded by 2^{-n} as $Y_i^\star[\ell^\star]$ is sampled uniformly at random. Summing over all possible choices of the index i, we have

$$\mathbf{Pr}\left(\mathsf{Forge}\right) \leq \frac{q_v}{2^n}\,.$$

Conclusion. We obtain that

$$\mathbf{Pr}\left(X_{\mathrm{id}} \in \Theta_{\mathrm{b}}\right) \leq \left(\binom{\#X}{2} + (\#X \cdot \#U) + \binom{\#U}{2}\right) \cdot \max\{\epsilon_1, \gamma_1, \gamma_2, \gamma_3\} + \Omega \#U \cdot \gamma_3 + \frac{q_v}{2^n}\,.$$

This completes the proof, noting that

$$\binom{\#X}{2} + (\#X \cdot \#U) + \binom{\#U}{2} = \binom{\#X + \#U}{2} \leq \binom{\sigma}{2}\,,$$

and in addition $\#U \leq \sigma$. □

6.4 Analysis of Good Transcripts

In this section we show that for a good transcript τ, realizing τ is almost as likely in the real world as in the ideal world. Formally, we prove the following lemma.

Lemma 2. *Let X_{re}, X_{id}, and Θ_{b} be as defined as above. For any good transcript $\tau = (\tau_e, \tau_v, \tau_d) \in \Theta \backslash \Theta_{\mathrm{b}}$,*

$$\frac{\mathbf{Pr}\left(X_{\mathrm{re}} = \tau\right)}{\mathbf{Pr}\left(X_{\mathrm{id}} = \tau\right)} = 1\,.$$

Proof. Let $\tau = (\tau_e, \tau_v, \tau_d)$ be a good transcript. Let s_e be the number of distinct X values in $\mathbf{X}^+ := (X_1^+, \ldots, X_{q_e}^+)$ and s_v be the number of distinct X values in $\mathbf{X}^\star := (X_1^\star, \ldots, X_{q_v}^\star)$. Moreover, let k_i be the number of non-fresh blocks for the i-th decryption query and k_i' be

the number of non-fresh blocks for the i-th verification query. There are $\sigma'_d := \sigma_d - \sum_{i=1}^{q_d} k_i$ values M_i^{-} and $\sigma'_v := \sigma_v - \sum_{i=1}^{q_v} k'_i$ values $M_i^{\star}$ that have been sampled.

This allows us to compute the ideal interpolation probability as follows: in the online phase, the encryption oracle samples q_e tag values and σ_{q_e} ciphertext blocks uniformly at random. The decryption oracle samples σ'_d message blocks uniformly at random and the verification oracle samples σ'_v message blocks uniformly at random. In the offline phase, the ideal oracle samples a total amount of $s_e + s_v$ values Y. Hence,

$$\mathbf{Pr}\left(X_{\mathrm{id}} = \tau\right) = \left(\frac{1}{2^n}\right)^{q_e} \cdot \left(\frac{1}{2^n}\right)^{\sigma_e} \cdot \left(\frac{1}{2^n}\right)^{\sigma'_d} \cdot \left(\frac{1}{2^n}\right)^{\sigma'_v} \cdot \left(\frac{1}{2^n}\right)^{s_e+s_v}.$$

We now compute the real interpolation probability for τ. Since τ is a good transcript and Fmt is a prefix-free function, $X_i^+[\ell_i]$ is fresh. Therefore, the values T_i^+ are uniformly distributed. Moreover, we do not have any collision in the tuple $\mathbf{U}^+ := (U_1^+, \ldots, U_{q_e}^+)$ as τ is good. This means that the ciphertext blocks are also generated uniformly at random. It is easy to see that the decryption oracle samples exactly σ'_d message blocks and the verification oracle samples exactly σ'_v message blocks. Moreover, as there are $s_e + s_v$ distinct X values in encryption and verification query history, we have,

$$\mathbf{Pr}\left(X_{\mathrm{re}} = \tau\right) = \left(\frac{1}{2^n}\right)^{q_e} \cdot \left(\frac{1}{2^n}\right)^{\sigma_e} \cdot \left(\frac{1}{2^n}\right)^{\sigma'_d} \cdot \left(\frac{1}{2^n}\right)^{\sigma'_v} \cdot \left(\frac{1}{2^n}\right)^{s_e+s_v}.$$

We conclude that the ratio of the real to ideal interpolation probability equals 1. □

6.5 Conclusion

By the H-coefficient technique of Theorem 1, we obtain for the remaining distance of (10):

$$\mathsf{AERUP}^{\mathcal{S}}_{\Pi}(\mathcal{A}) \leq \epsilon_{\mathrm{ratio}} + \epsilon_{\mathrm{bad}}\,,$$

where $\epsilon_{\mathrm{ratio}} = 0$ given the bound of Lemma 2, and ϵ_{bad} is set to be the bound of Lemma 1.

7 Conclusion

In this paper, we first proposed AERUP, a unified RUP security notion for deterministic authenticated encryption schemes. Next, we considered a generalized version of SUNDAE, called ANYDAE, and derived necessary and sufficient conditions for ANYDAE to achieve AERUP security. We introduced two instantiations of ANYDAE, called MONDAE and TUESDAE. MONDAE exhibits a structural resemblance with SUNDAE, with a minimal change in the construction so that its RUP security is retained. Therefore, it is at par with SUNDAE in terms of efficiency and optimality. TUESDAE is an optimal construction in terms of the number of block cipher invocations at the cost of a little increase of hardware area. It is an interesting open question to investigate AERUP security for existing SIV based constructions.

Acknowledgements. This work was initiated during the group discussion in the Asian Symmetric Key Workshop, 2018. The authors would like to thank all the anonymous reviewers of ToSC 2019 for their valuable comments. Nilanjan Datta, Avijit Dutta and Mridul Nandi are supported by the project “Study and Analysis of IoT Security” under Government of India at R.C.Bose Centre for Cryptology and Security, Indian Statistical Institute, Kolkata. Bart Mennink is supported by a postdoctoral fellowship from the Netherlands Organisation for Scientific Research (NWO) under Veni grant 016.Veni.173.017. Ferdinand Sibleyras is partly supported by the French DGA.

References

[ABL+14] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. How to Securely Release Unverified Plaintext in Authenticated Encryption. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 105–125. Springer, 2014.

[ADL17] Tomer Ashur, Orr Dunkelman, and Atul Luykx. Boosting Authenticated Encryption Robustness with Minimal Modifications. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 3–33. Springer, 2017.

[BBLT18] Subhadeep Banik, Andrey Bogdanov, Atul Luykx, and Elmar Tischhauser. SUNDAE: Small Universal Deterministic Authenticated Encryption for the Internet of Things. *IACR Trans. Symmetric Cryptol.*, 2018(3):1–35, 2018.

[BBP+19] Subhadeep Banik, Andrey Bogdanov, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, Elmar Tischhauser, and Yosuke Todo. SUNDAE-GIFT v1.0, March 2019. Submission to NIST Lightweight Cryptography Standardization Process.

[BDPV11] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications. In Ali Miri and Serge Vaudenay, editors, *SAC 2011*, volume 7118 of *LNCS*, pages 320–337. Springer, 2011.

[BGIS19a] Zhenzhen Bao, Jian Guo, Tetsu Iwata, and Ling Song. SIV-Rijndael256 Authenticated Encryption and Hash Family, February 2019. Submission to NIST Lightweight Cryptography Standardization Process.

[BGIS19b] Zhenzhen Bao, Jian Guo, Tetsu Iwata, and Ling Song. SIV-TEM-PHOTON Authenticated Encryption and Hash Family, February 2019. Submission to NIST Lightweight Cryptography Standardization Process.

[BPS15] Guy Barwell, Daniel Page, and Martijn Stam. Rogue Decryption Failures: Reconciling AE Robustness Notions. In Jens Groth, editor, *IMACC 2015*, volume 9496 of *LNCS*, pages 94–111. Springer, 2015.

[CDJ+19] Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas Lopez, Mridul Nandi, and Yu Sasaki. ESTATE, March 2019. Submission to NIST Lightweight Cryptography Standardization Process.

[CDN16] Avik Chakraborti, Nilanjan Datta, and Mridul Nandi. INT-RUP Analysis of Block-cipher Based Authenticated Encryption Schemes. In Kazue Sako, editor, *CT-RSA 2016*, volume 9610 of *LNCS*, pages 39–54. Springer, 2016.

[CDN18] Avik Chakraborti, Nilanjan Datta, and Mridul Nandi. On the optimality of non-linear computations for symmetric key primitives. *J. Mathematical Cryptology*, 12(4):241–259, 2018.

[CDNY18] Avik Chakraborti, Nilanjan Datta, Mridul Nandi, and Kan Yasuda. Beetle Family of Lightweight and Secure Authenticated Encryption Ciphers. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(2):218–241, 2018.

[CIMN17] Avik Chakraborti, Tetsu Iwata, Kazuhiko Minematsu, and Mridul Nandi. Blockcipher-Based Authenticated Encryption: How Small Can We Go? In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 277–298. Springer, 2017.

[CS14] Shan Chen and John P. Steinberger. Tight Security Bounds for Key-Alternating Ciphers. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 327–350. Springer, 2014.

[CY99] Scott Contini and Yiqun Lisa Yin. On Differential Properties of Data-Dependent Rotations and Their Use in MARS and RC6, 1999.

[DGM+19] Nilanjan Datta, Ashrujit Ghoshal, Debdeep Mukhopadhyay, Sikhar Patranabis, Stjepan Picek, and Rajat Sadhukhan. TRIFLE, March 2019. Submission to NIST Lightweight Cryptography Standardization Process.

[DLMN17] Nilanjan Datta, Atul Luykx, Bart Mennink, and Mridul Nandi. Understanding RUP Integrity of COLM. *IACR Trans. Symmetric Cryptol.*, 2017(2):143–161, 2017.

[DMV17] Joan Daemen, Bart Mennink, and Gilles Van Assche. Full-State Keyed Duplex with Built-In Multi-user Support. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 606–637. Springer, 2017.

[FIP80] FIPS 81: DES Modes of Operation, 1980.

[FJMV03] Pierre-Alain Fouque, Antoine Joux, Gwenaëlle Martinet, and Frédéric Valette. Authenticated On-Line Encryption. In Mitsuru Matsui and Robert J. Zuccherato, editors, *SAC 2003*, volume 3006 of *LNCS*, pages 145–159. Springer, 2003.

[FMP03] Pierre-Alain Fouque, Gwenaëlle Martinet, and Guillaume Poupard. Practical Symmetric On-Line Encryption. In Thomas Johansson, editor, *FSE 2003*, volume 2887 of *LNCS*, pages 362–375. Springer, 2003.

[HKR15] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust Authenticated-Encryption AEZ and the Problem That It Solves. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 15–44. Springer, 2015.

[HSY17] Shoichi Hirose, Yu Sasaki, and Kan Yasuda. Rate-One AE with Security Under RUP. In Phong Q. Nguyen and Jianying Zhou, editors, *ISC 2017*, volume 10599 of *LNCS*, pages 3–20. Springer, 2017.

[IMGM14] Tetsu Iwata, Kazuhiko Minematsu, Jian Guo, and Sumio Morioka. CLOC: Authenticated Encryption for Short Input. In Carlos Cid and Christian Rechberger, editors, *FSE 2014*, volume 8540 of *LNCS*, pages 149–167. Springer, 2014.

[IMI18] Kazuya Imamura, Kazuhiko Minematsu, and Tetsu Iwata. Integrity analysis of authenticated encryption based on stream ciphers. *Int. J. Inf. Sec.*, 17(5):493–511, 2018.

[MRV15] Bart Mennink, Reza Reyhanitabar, and Damian Vizár. Security of Full-State Keyed Sponge and Duplex: Applications to Authenticated Encryption. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 465–489. Springer, 2015.

[Nan09] Mridul Nandi. Fast and Secure CBC-Type MAC Algorithms. In Orr Dunkelman, editor, *FSE 2009*, volume 5665 of *LNCS*, pages 375–393. Springer, 2009.

[Nat18] National Institute of Standards and Technology (NIST). Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process, August 2018.

[NMSS18] Yusuke Naito, Mitsuru Matsui, Takeshi Sugawara, and Daisuke Suzuki. SAEB: A Lightweight Blockcipher-Based AEAD Mode of Operation. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(2):192–217, 2018.

[Pat08] Jacques Patarin. The "Coefficients H" Technique. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *SAC 2008*, volume 5381 of *LNCS*, pages 328–345. Springer, 2008.

[RS06] Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 373–390. Springer, 2006.

[WH16] Hongjun Wu and Tao Huang. The JAMBU Lightweight Authentication Encryption Mode (v2.1), 2016. Submission to CAESAR competition.

[ZWH+17] Ping Zhang, Peng Wang, Honggang Hu, Changsong Cheng, and Wenke Kuai. INT-RUP Security of Checksum-Based Authenticated Encryption. In Tatsuaki Okamoto, Yong Yu, Man Ho Au, and Yannan Li, editors, *Provable Security 2017*, volume 10592 of *LNCS*, pages 147–166. Springer, 2017.