INT-RUP Security of SAEB and TinyJAMBU

Nilanjan Datta[1], Avijit Dutta[2], and Shibam Ghosh[3]

[1]TCG CREST, India
nilanjan.datta@tcgcrest.org
[2]TCG CREST, India
avirocks.dutta13@gmail.com
[3]Department of Computer Science, University Of Haifa, Israel
sghosh03@campus.haifa.ac.il

Abstract. The INT-RUP security of an authenticated encryption (AE) scheme is a well studied problem which deals with the integrity security of an AE scheme in the setting of releasing unverified plaintext model. Popular INT-RUP secure constructions either require a large state (e.g. GCM-RUP, LOCUS, Oribatida) or employ a two-pass mode (e.g. MON-DAE) that does not allow on-the-fly data processing. This motivates us to turn our attention to feedback type AE constructions that allow small state implementation as well as on-the-fly computation capability. In CT-RSA 2016, Chakraborti et al. have demonstrated a generic INT-RUP attack on rate-1 block cipher based feedback type AE schemes. Their results inspire us to study about feedback type AE constructions at a reduced rate. In this paper, we consider two such recent designs, SAEB and TinyJAMBU and we analyze their integrity security in the setting of releasing unverified plaintext model. We found an INT-RUP attack on SAEB with roughly 2^{32} decryption queries. However, the concrete analysis shows that if we reduce its rate to 32 bits, SAEB achieves the desired INT-RUP security bound without any additional overhead. Moreover, we have also analyzed TinyJAMBU, one of the finalists of the NIST LwC, and found it to be INT-RUP secure. To the best of our knowledge, this is the first work reporting the INT-RUP security analysis of the block cipher based single state, single pass, on-the-fly, inverse-free authenticated ciphers.

1 Introduction

In the last few years, the increasing growth of the Internet of Things (IoT) 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 CAESAR [16] and the ongoing NIST Lightweight Competition (LwC) [32]. As a result of these competitions, the cryptographic community has witnessed the rise of various lightweight authenticated encryption schemes in recent years. These include

block cipher based constructions such as CLOC [30], JAMBU [42], COFB [21], SAEB [36], SUNDAE [11], permutation based constructions such as ASCON [25], ACORN [41], Beetle [20] and tweakable block cipher based constructions such as Deoxys AEAD [31], Romulus and Remus [29], Skinny AEAD [12] etc. However, in the paper, we confine our discussion only to block cipher and permutation based AE schemes.

1.1 Designing Area-Efficient Authenticated Ciphers

Often Lightweight Authenticated Encryption (AE) schemes are sequential in nature. This is primarily due to the fact that sequential modes consume less state size (the memory needed for storing internal values), implying smaller hardware footprint. In addition, sequential modes usually offer inverse-free property (except a few construction such as CBC and its variants [1]) of the underlying primitive, which also suits one of the very basic needs of implementing ciphers in lightweight environments.

Block Cipher Based Designs The design of almost every block cipher based sequential AE schemes starts with a fixed initial state and processes each input block in sequence by applying a feedback function on the previous block cipher output, some secret auxiliary state, and the current input (message or associated data). This feedback function derives the next block cipher input, updated secret auxiliary state, and the current output (in case of message blocks). Thus, any block cipher based sequential AE scheme can be described by the underlying block cipher, the secret auxiliary state and the feedback function. Consequently, the AE scheme's efficiency and hardware footprint largely depend on the efficiency and the hardware footprint of the underlying block cipher, feedback function, and the secret auxiliary state. Zhang et al. [45] have proposed one such block cipher based sequential AE scheme called iFEED that uses plaintext feedback and achieves optimal rate [2] (i.e. rate-1). However, it requires a state size of $(3n + k)$-bits, where n and k are the block size and the key size of the underlying block cipher respectively. CPFB, proposed by Montes and Penazzi [34], is a notable scheme that reduces the state size to $(2n + k)$-bits, at the cost of reducing the rate to 3/4. In CHES'17, Chakraborti et al. proposed COFB [21], the first feedback type AE scheme, achieves rate-1 with a state size of $1.5n + k$-bits. Recently, it has been proven in [18] that the state size for any feedback type rate-1 AE scheme cannot be less than $1.5n + k$-bits. In the same paper, authors have also proposed a hybrid feedback type AE scheme called HyENA [18] that achieves rate-1 with the state size $1.5n + k$-bits but with a reduced XOR count.

Sponge Type Designs An alternative way to avoid the generation of the auxiliary states from the block cipher based designs is to use a public permu-

[1] Some inverse-free modes are not sequential, e.g., CTR, OTR, GCM etc.

[2] rate is defined as the inverse of the number of block cipher calls required to process a single block of message, where a block refers to the block size of the block cipher.

tation based sponge mode of operations. Since the selection of Ascon [25] in the final portfolio of the CAESAR [16] competition, sponge based designs have gained a huge momentum. In the ongoing NIST LwC competition [32], out of 57 submissions, 25 submissions are based on sponge type designs and out of 10 finalists, 5 candidates are sponge type designs [32]. The primary feature that one can get out of sponge type designs is that unlike block cipher based constructions, it does not require any key scheduling algorithm to invoke. This feature proves to be beneficial from the storage point of view when the data size of the underlying permutation for any sponge type design is less than the combination of the block size and key size of the underlying block cipher for any block cipher based designs. In such cases, sponge type mode becomes an excellent choice for area-efficient designs. Moreover, the additional feature of having no inverse call to the underlying permutation at the time of executing verified decryption algorithm, ensures an extremely low hardware footprint in a combined encryption-decryption implementation of the mode. By leveraging the advantages of sponge-type structure in block cipher based designs (albeit block cipher based schemes are required to store extra k-bit state for storing the keys), a few block cipher based sponge-type designs have recently been proposed. This includes CAESAR candidate JAMBU [42], and two NIST LwC candidates SAEAES [35] (which is an instantiation of SAEB [36] with AES-128 block cipher) and TinyJAMBU [43], where all the three AE schemes use a block size of 128-bits along with a block cipher key of 128-bits, employing an extremely small overall state of size 256-bits.

1.2 Authenticated Ciphers under Release of Unverified Plaintext (RUP) Setting

In traditional authenticated ciphers, the verification must be done prior to release of plaintexts to the user. However, in resource constrained environments with limited memory, it may not be feasible to store the whole plaintext and one might be forced to release the plaintext before verification. Further details can be found in the supplementary material (see Section B).

In [7], Andreeva et al. formalized the security notion of an authenticated encryption scheme under the release of unverified plaintext setting. In this model, the encryption functionality $\mathcal{E}$ remains, but it separates the decryption/verification functionality $\mathcal{DV}$ into a decryption functionality $\mathcal{D}$ and a verification functionality $\mathcal{V}$. Likewise the usual security notion of any AE scheme that ensures both confidentiality and integrity, Andreeva et al. [7] have suggested to achieve the confidentiality and integrity security for any AE scheme in the RUP model using IND-CPA + PA1 / PA2 notion and INT-RUP notion respectively. For the confidentiality model in PA1 setting, i.e., IND-CPA + PA1, the adversary is given access to $\mathcal{E}$ and to either $\mathcal{D}$ or a simulator. The purpose of this notion is to complement the conventional confidentiality in the sense that it measures the advantage an adversary can gain from actually having access to $\mathcal{D}$. The integrity notion of an authenticated encryption scheme under this model, i.e., INT-RUP, allows an adversary to interact with $\mathcal{E}, \mathcal{D}$, $\mathcal{V}$; and asks it to forge a

valid ciphertext, i.e., make a new, valid query to $\mathcal{V}$ oracle. The adversary potentially possesess significantly more power in this model due to the access to the decryption oracle $\mathcal{D}$.

Andreeva et al. [7] have shown that OCB [39], COPA [8] are insecure in the RUP security model. In [24], Datta et al. mounted an INT-RUP attack on any Encrypt-Linear mix-Encrypt type authenticated ciphers that includes the CAESAR standard COLM [6]. In another direction, Chakraborti et al. [19] mounted an INT-RUP attack on iFeed [44]. Adopting a similar attack strategy, they have shown a generic INT-RUP attack [19] on rate-1 block cipher based feedback type AEAD constructions. At the same time, they also proposed a scheme called mCPFB [19] and claimed that the INT-RUP security could be achieved at the cost of the rate of the construction. Similar approach have been used in OCBIC [46] and LOCUS [17], which builds upon OCB [39], and LOTUS [17], which builds upon OTR [33]. For both the constructions LOCUS and LOTUS, IND-CPA + PA1 and INT-RUP both security notions have been achieved at the cost of additional block cipher invocations, which halves the rate of the construction. Note that these modes are parallel in structure, and all of them require a state of size at least $3n + k$-bits, where n is the block size and k is the key size of the underlying block cipher. Ashur et al. [10] proposed an alternative notion of RUP security, called RUPAE. This notion focuses on nonce-based authenticated encryption, and proposed a RUP-variant of GCM [1], dubbed GCM-RUP [10], in the described nonce-based model. On the other extreme, Chang et al. [22] introduced the notion of AERUP which unifies the notions of RUP privacy (i.e., IND-CPA + PA1) and integrity (i.e., INT-RUP) for deterministic authenticated ciphers. They also proposed a simple variant of SUNDAE [11], dubbed MONDAE [22], that achieves confidentiality and integrity security in RUP setting under this newly introduced AERUP model. However, MONDAE is a two-pass authenticated encryption mode. Hence, it does not have the on-the-fly decryption feature. In a nutshell, while looking at the ciphers with RUP security, either the constructions lose on state size (e.g., LOCUS [17], mCPFB [19], GCM-RUP [10], requires at least $3n + k$-bits) or the construction does not have the on-the-fly decryption feature (e.g., MONDAE [22]). The above discussion makes us raise the question:

Can we have a block cipher based INT-RUP secure design with on-the-fly decryption feature with a total of $n + k$-bits state?

1.3 Towards RUP-Secure Single-state On-the-Fly Authenticated Encryption

The above question turns our attention to study the INT-RUP security of sponge based modes. In [15], Bhattacharjee et al. have studied the INT-RUP security of permutation based sponge type designs. They have presented an INT-RUP attack on generic duplex constructions, with attack complexity $O(q_d q_p/2^c)$, where q_d is the number of decryption queries, q_p is the number of primitive queries to the permutation, and c is the capacity part of the construction. They have

also shown that such attacks can be extended to other Sponge variants such as Beetle [20] and SPoC [2]. The main idea of the attack is to exploit a collision between an inner state of the construction and a primitive query. To resist such attacks, the authors used the concept of masking the previous state and proposed a new cipher called Oribatida [15] that achieves INT-RUP security of $O(q_d^2/2^c)$. However, this comes at the cost of an additional state.

When we move to block cipher based AEAD constructions, Chakraborti et al. [19] have shown that any feedback type rate-1 block cipher based AEAD construction is not INT-RUP secure. Adopting the idea used in iFeed, they have shown a generic INT-RUP attack on rate-1 block cipher based feedback type AEAD constructions. This result immediately rules out the popular area-efficient block cipher based designs such as COFB and HyENA to have INT-RUP security. Therefore, the focus goes to feedback type constructions with a lower rate. This makes us look into block cipher based sponge type constructions such as SAEB [36] and TinyJAMBU [43]. Due to the inverse-free implementation with $n+k$-bits state, these constructions are incredibly lightweight and ideally suited for resource constraint applications. At the same time, these constructions have the capability of on-the-fly computation of plaintext/ciphertext blocks. Thus, they are ideally suited for applications where RUP security would be of extreme relevance. However, the current literature does not say anything about these block cipher based constructions, and hence investigating their RUP security seems an exciting research direction. In this regard, we would like to mention that in a recent work, Andreeva et al. [4] have shown $2^{n/2}$ INT-RUP security bound on a forkcipher [9] based construction, called SAEF [3]. The structure of SAEF resembles to the CBC mode of operation, where one of the output blocks of the forkcipher is used to XOR-mask the input and sometimes output of the next primitive call.

1.4 Our Contribution and Significance of the Result

In this paper, we study the INT-RUP security of two constructions, namely SAEB [36] by Naito et al. and TinyJAMBU [43] by Wu et al. Our contribution is threefold:

(i) We have shown an INT-RUP attack on SAEB that uses a single encryption query, and roughly $2^{c/2}$ decryption queries, where c is the capacity part of the construction. The attack is applicable for any choices of rate and capacity.

(ii) We have investigated the INT-RUP security bound of SAEB. We have shown that it offers roughly $q_d^2/2^c$ INT-RUP security, where q_d is the number of decryption queries and c is the capacity part of the construction. Combining the proven security bound of SAEB with its attack complexity establishes the tightness of the security bound of SAEB. This result signifies that if we instantiate SAEB with a standard 128-bit block cipher and put a restriction that at a time 32-bits of the message will be injected to SAEB, the mode achieves INT-RUP security up to 2^{48} blocks, which satisfies the NIST criteria of having 2^{50} bytes of data complexity. However, for SAEAES, where we inject

64-bits of message at a time to the construction, achieves INT-RUP security up to 2^{32} blocks.

(iii) Finally, we consider the INT-RUP security of TinyJAMBU, one of the finalists of the NIST LwC. Interestingly TinyJAMBU has a unique structure, where message injection and ciphertext release occur from different parts of the state. We have proved that TinyJAMBU offers roughly $q_v\sigma_d/2^{n-r}$ INT-RUP security, where σ_d is the total number of blocks in all the decryption queries, r-bits of the message is injected at a time to the construction and n is the block size of the underlying block cipher.

Thus, in this work, we have obtained the INT-RUP security bounds of SAEB and TinyJAMBU. To the best of our knowledge, this is the first work that reports two single-pass [3], inverse-free AEAD constructions, achieving INT-RUP security, while keeping the on-the-fly decryption property intact. We would like to point here that both of these construction will not preserve confidentiality in the RUP setting as these are not two-pass modes. This paper solely focuses on studying the integrity security of the two constructions in the RUP setting.

2 Preliminaries

For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, 2, \ldots, n\}$ and for $a < b \in \mathbb{N}$, we write $[a, b]$ to denote the set $\{a, a+1, \ldots, b\}$. We write $(a, b) < (a', b')$ to denote that either $a < a'$ or $(a = a'$ and $b < b')$. For a finite set $\mathcal{X}$, $\mathsf{X} \leftarrow_{\$} \mathcal{X}$ denotes the uniform at random sampling of X from $\mathcal{X}$. For $n \in \mathbb{N}$, we write $\{0,1\}^+$ and $\{0,1\}^n$ to denote the set of all non-empty binary strings, and the set of all n-bit binary strings, respectively. We write $\emptyset$ to denote the empty string, and $\{0,1\}^* = \{0,1\}^+ \cup \{\emptyset\}$. For any two strings $X, Y \in \{0,1\}^*$, we write $X\|Y$ to denote the concatenation of the string X followed by the string Y. For $X \in \{0,1\}^*$, $|X|$ denotes the length (number of the bits) of X. For any non-empty binary string $X \in \{0,1\}^+$, $(X[1], \ldots, X[k]) \xleftarrow{n} X$ denotes the n-bit parsing of X, where $|X[i]| = n$ for $1 \leq i \leq k-1$, and $1 \leq |X[k]| \leq n$. We use the notation $X[a \ldots b]$ to denote bit string $X[a]\|X[a+1]\| \cdots \|X[b]$. If $a = 1$, then we write $X[\ldots b]$ to denote $X[1 \ldots b]$. In this paper, we fix a positive integer n and define the function ozs over the set of any binary string, as $\mathsf{ozs}(X) := X\|1\|0^{n-(|X| \mod n)-1}$. Note that the function is injective and maps all m-bit binary strings to a multiple of n-bit binary strings by appropriately padding the string with 10^*. For any real number X, $\lceil X \rceil$ denotes the smallest integer X' such that $X' \geq X$. For any $X \in \{0,1\}^+$ and an integer $i \leq |X|$, $\lfloor X \rfloor_i$ ($\lceil X \rceil_i$) returns the least significant (most significant, resp.) i-bits of X. For any integer i, we denote the n-bit unsigned representation of i as $\langle i \rangle_n$.

2.1 Authenticated Encryption

An authenticated encryption (AE) is an integrated scheme that provides both privacy and integrity of a plaintext $M \in \{0,1\}^*$ and integrity of an associ-

[3] GCM-RUP [10] also achieves inverse-free, INT-RUP security and on-the-fly decryption property but it requires two pass

ated data $A \in \{0,1\}^*$. Taking a nonce $N \in \mathcal{N}$ (which is a unique value for each encryption), where $\mathcal{N}$ is the nonce space, together with the associated data A and the plaintext M, the encryption function of AE, enc_K, produces a tagged-ciphertext $(C,T) \in \{0,1\}^* \times \{0,1\}^\tau$ with $|C| = |M|$. We denote the length in blocks of the associated data, message and ciphertext with a, m and c, respectively. The corresponding decryption function, dec_K, takes $(N,A,C,T) \in \mathcal{N} \times \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^\tau$ and returns a decrypted plaintext $M \in \{0,1\}^*$ when the authentication on (N,A,C,T) is successful; otherwise, it returns the atomic error symbol denoted by $\perp$. Following Andreeva et al. [7], we separate the decryption algorithm into plaintext computation and tag verification. Formally, the decryption interface provides two algorithms, a decryption function dec_K that takes (N,A,C) and returns a decrypted plaintext M irrespective of the authentication result (hence we drop the tag value), and a verification function ver_K that takes (N,A,C,T) and returns $\top$ when the authentication succeeds; otherwise, it returns $\perp$.

2.2 Integrity Security in RUP Setting

Following the definition of Andreeva et al. [7], we define the integrity security of an authenticated encryption scheme in the RUP setting. We consider an information theoretic adversary $\mathscr{A}$ with access to a triplet of oracles of an authenticated encryption scheme Θ - for a uniformly sampled secret key K, the encryption oracle $\Theta.\mathsf{enc}_K$, decryption oracle $\Theta.\mathsf{dec}_K$ and the verification oracle $\Theta.\mathsf{ver}_K$. We say that $\mathscr{A}$ forges Θ under the RUP setting if $\mathscr{A}$ can compute a tuple (N,A,C,T) satisfying $\Theta.\mathsf{ver}_K(N,A,C,T) \neq \perp$, without querying (N,A,M) to $\Theta.\mathsf{enc}_K$ and receiving (C,T) as a response, i.e. (N,A,C,T) is a non-trivial forgery. We assume that $\mathscr{A}$ can make decryption queries of the form (N,A,C) to the oracle $\Theta.\mathsf{dec}_K$, with no restriction on nonce repetitions, and receive the corresponding response M, whereas nonces should be distinct for every encryption queries to $\Theta.enc_K$. Then, the integrity security or equivalently the forging advantage of Θ for the adversary $\mathscr{A}$ in the RUP setting is defined as

$$\mathsf{Pr}[K \xleftarrow{\$} \{0,1\}^k : \mathscr{A}^{\Theta.\mathsf{enc}_K, \Theta.\mathsf{dec}_K, \Theta.\mathsf{ver}_K} \text{ forges }].$$

We assume that $\mathscr{A}$ does not make any query to the oracles for which it can compute the corresponding response on its own. We call such an adversary a *non-trivial* adversary. Following [7], we view the forging advantage of Θ in the RUP setting as an equivalent distinguishing game between two worlds. The real world consists of $(\Theta.\mathsf{enc}_K, \Theta.\mathsf{dec}_K, \Theta.\mathsf{ver}_K)$ for a uniformly chosen key K, whereas the ideal world consists of $(\Theta.\mathsf{enc}_K, \Theta.\mathsf{dec}_K, \perp)$, i.e., the verification oracle in the real world is replaced by the reject symbol. This means all verification attempts in the ideal world will lead to a rejection. Under this equivalent setting, the integrity advantage for any distinguisher $\mathscr{A}$ is defined as

$$\mathbf{Adv}^{\text{int-rup}}_{\Theta}(\mathscr{A}) := \left| \mathsf{Pr}[\mathscr{A}^{(\Theta.\mathsf{enc}_K, \Theta.\mathsf{dec}_K, \Theta.\mathsf{ver}_K)} = 1] - \mathsf{Pr}[\mathscr{A}^{(\Theta.\mathsf{enc}_K, \Theta.\mathsf{dec}_K, \perp)} = 1] \right|,$$

where $\perp$ denotes the degenerate oracle that always returns $\perp$ symbol and the probability is defined over the randomness of K. The integrity under RUP advantage of Θ is defined as

$$\mathbf{Adv}^{\mathsf{int\text{-}rup}}_{\Theta}(q_e, q_d, q_v, \sigma_e, \sigma_d, \sigma_v) := \max_{\mathscr{A}} \mathbf{Adv}^{\mathsf{int\text{-}rup}}_{\Theta}(\mathscr{A}),$$

where the maximum is taken over all distinguishers making q_e encryption queries, q_d decryption queries and q_v verification queries. Here σ_e, σ_d and σ_v denotes the total number of block cipher calls with distinct inputs in encryption, decryption, and verification queries, respectively. Throughout this paper, we write $(q_e, q_d, q_v, \sigma_e, \sigma_d, \sigma_v)$-distinguisher to represent a distinguisher that makes q_e encryption queries with a total of σ_e many primitive calls with distinct inputs in encryption queries, q_d decryption queries with a total of σ_d many primitive calls with distinct inputs in decryption queries and q_v verification queries with a total of σ_v many primitive calls with distinct inputs in verification queries.

Let Λ_1 and Λ_0 denote the random variable induced by the interaction of $\mathscr{A}$ with the real oracle and the ideal oracle, respectively. The probability of realizing a transcript ω in the ideal oracle (i.e., $\mathsf{Pr}[\Lambda_0 = \omega]$) is called the *ideal interpolation probability*. Similarly, one can define the *real interpolation probability*. A transcript ω is said to be *attainable* with respect to $\mathscr{A}$ if the ideal interpolation probability is non-zero (i.e., $\mathsf{Pr}[\Lambda_0 = \omega] > 0$). We denote the set of all attainable transcripts by Ω. Following these notations, we state the main result of the H-Coefficient Technique in Theorem 1. The proof of this theorem can be found in [37].

Theorem 1 (H-Coefficient Technique). *Suppose for some $\Omega_{\mathtt{bad}} \subseteq \Omega$, which we call the* bad *set of transcripts, the following conditions hold:*

1. $\mathsf{Pr}[\Lambda_0 \in \Omega_{\mathtt{bad}}] \leq \epsilon_1$,
2. *For any good transcript $\omega \in \Omega \setminus \Omega_{\mathtt{bad}}$,* $\mathsf{Pr}[\Lambda_1 = \omega] \geq (1 - \epsilon_2) \cdot \mathsf{Pr}[\Lambda_0 = \omega]$.

Then, we have

$$\mathbf{Adv}^{\mathsf{int\text{-}rup}}_{\Theta}(q_e, q_d, q_v, \sigma_e, \sigma_d, \sigma_v) \leq \epsilon_1 + \epsilon_2. \tag{1}$$

We will apply the H-Coefficient technique to bound the integrity security of the two block cipher based authenticated ciphers SAEB and TinyJAMBU in the RUP model. To do this, we first replace the underlying primitive of the construction, which is a block cipher, with a random permutation at the cost of the PRP advantage of the block cipher. Then, we bound the distinguishing advantage of the resulting construction (whose underlying primitive is a random permutation) from the ideal one. We bound this advantage against an adversary $\mathscr{A}$ that is computationally unbounded (i.e., no bound on the time complexity, but bounded on the number of queries that it can ask to the oracle) and hence deterministic. We call them *information-theoretic* adversary. Therefore, from now onwards, we skip the time parameter from their corresponding advantage definitions.

3 SAEB AEAD Mode and Its INT-RUP Security

SAEB [36] is a block cipher based AEAD scheme, proposed by Naito et al. in TCHES'18. The design principle of SAEB follows the sponge duplex mode based on block ciphers. Similar to permutation based sponge constructions, SAEB injects r-bits of the message at a time to the construction, called the message injection rate, c-bits capacity, and the overall block size is $n = r + c$-bits. The algorithmic description of the encryption function of SAEB is presented in Figure 1, and its schematic diagram is depicted in Figure 3. An instantiation of SAEB with AES-128, called SAEAES [35], was submitted in NIST LwC and is one of the second round candidates of the competition. In the original proposal of the scheme, the recommended parameters of SAEAES are $r = 64$ and $c = 64$.

$\mathsf{SAEB}(K, N, A, M)$

$A[1]\| \ldots \|A[a] \xleftarrow{r} A\|10^*$;
$M[1]\| \ldots \|M[m] \xleftarrow{r} M\|10^*$;
$s[1] \leftarrow A[1]\|0^c$;
`for` $i = 2$ `to` $a - 1$;
  $s[i] \leftarrow E_K(s[i-1]) \oplus A[i]\|0^\star$;
$s[a] \leftarrow E_K(s[a-1]) \oplus A[a]\|\mathsf{const}_1$;
$s[a+1] \leftarrow E_K(s[a]) \oplus N\|\mathsf{const}_2$;
`for` $i = 2$ `to` $m - 1$;
  $s[a+i] \leftarrow E_K(s[a+i-1]) \oplus M[i]\|0^\star$;
  $C[i] \leftarrow \lceil s[a+i] \rceil_r$;
$s[a+m+1] \leftarrow E_K(s[a+m]) \oplus M[m]\|\mathsf{const}_3$;
$T \leftarrow \lceil E_K(s[a+m+1]) \rceil_\tau$;
return (C, T);

Fig. 1: Encryption algorithm of SAEB authenticated encryption mode.

Naito et al. [36] have shown that SAEB achieves birthday bound security with the dominating term being $(\sigma_a + \sigma_d)/2^c + (\sigma_e + \sigma_d)^2/2^n$, where σ_a is the number of associated data blocks across all the queries, σ_e is the total number of block cipher calls with distinct inputs in encryption queries, and σ_d is the total number of block cipher calls with distinct inputs in decryption queries. However, the designers have not analyzed the construction in the RUP setting, and to the best of our knowledge, no prior work has addressed the issue of analyzing the security of this construction in the RUP setting. In the subsequent sections, we analyze the INT-RUP security of SAEB. In particular, we show in Sect. 3.1 that an adversary $\mathscr{A}$ with roughly $2^{c/2}$ decryption queries, can forge SAEB in the

release of unverified plaintext setting and in Sect. 4, we give an upper bound of the order $2^{c/2}$ on the INT-RUP security of SAEB.

3.1 INT-RUP Attack on SAEB

In this section, we show a forging attack on SAEB in the INT-RUP setting with $2^{c/2}$ decryption queries and a single encryption query. Here we describe the adversary $\mathcal{A}$ that primarily exploits the fact that during a decryption call, an adversary can control the rate part of the input of the block cipher by directly injecting the ciphertext into the rate part of the block cipher (see Figure 4) to mount the attack: First, we describe the attack when $r \geq c/2$, then extend it for $r < c/2$. Let $r \geq c/2$. We describe an adversary $\mathscr{A}$ that mounts an INT-RUP attack against SAEB with roughly $2^{c/2}$ decryption queries and a single encryption query as follows:

1. $\mathscr{A}$ chooses an arbitrary r-bit nonce N, an arbitrary r-bit associated data A and an arbitrary r-bit ciphertext data C, and then makes $2^{c/2+1}$ decryption queries of the form $(N,\ A,\ C_i[1] \parallel C[2] \parallel C[3] \parallel \ldots \parallel C[\ell+1])_{i=1,\ldots,2^{c/2}}$, with distinct r-bit $C_i[1]$ values such that $C[2] = C[3] = \ldots = C[\ell+1] = C$ and $\ell = \lceil c/r \rceil + 1$. Let the unverified released plaintext be $M_i[1] \parallel M_i[2] \parallel M_i[3] \parallel \ldots \parallel M_i[\ell+1]$.
2. Assume there exist two indices $j, k \in [2^{c/2}]$ for which $M_j[a] = M_k[a]$ for all $a \in [3, \ell+1]$.
3. $\mathscr{A}$ makes an encryption query with $(N,\ A,\ M_j[1] \parallel M_j[2] \parallel M_j[3] \parallel \ldots \parallel M_j[\ell+1])$. Let the tagged ciphertext be $(C_j[1] \parallel C[2] \parallel C[3] \parallel \ldots \parallel C[\ell+1],\ T)$, where $C[2] = C[3] = \ldots = C[\ell+1] = C$.
4. $\mathscr{A}$ forges with $(N,\ A,\ (C_k[1] \parallel C[2] \parallel C[3] \parallel \ldots \parallel C[\ell+1]),\ T)$, where $C[2] = C[3] = \ldots = C[\ell+1] = C$.

It is easy to see that $\mathcal{A}$ succeeds with probability $1/2$. For completeness, the details analysis is given in the Supplementary material (see Section C.2).

Attack when $r < c/2$. Now, we consider the case when $r < c/2$. Note that, when $r < c/2$, then varying just one r-bit ciphertext string would result in at most 2^r different values. This would not ensure a collision in the capacity part with high probability. To deal with this, we vary multiple consecutive r-bit ciphertext strings, say s many, which results in 2^{rs} many different values. If we appropriately choose s with $rs \geq c/2$, we expect a collision in the capacity part. Then a similar attack strategy, as described for $r \geq c/2$, will hold. For completeness, we formally present the attack algorithm in the supplementary material.

4 INT-RUP Security of SAEB

In this section, we show that SAEB is INT-RUP secure against all adversaries that make roughly $2^{c/2}$ decryption queries, where c is the capacity of the construction. We prove the security of the construction in the information-theoretic

setting, where a uniform random n-bit permutation P replaces the underlying block cipher of the construction at the cost of the prp advantage of the block cipher E and denote the resulting construction as $\mathsf{SAEB}^*[\mathsf{P}]$. In the following, we state and prove the int-rup security result of $\mathsf{SAEB}^*[\mathsf{P}]$.

Theorem 2. *Let* $\mathsf{P} \leftarrow\!\!\$\ \mathsf{Perm}(n)$ *be an uniformly sampled* n*-bit random permutation. The* INT-RUP *advantage for any* $(q_e, q_d, q_v, \sigma_e, \sigma_d, \sigma_v)$*-distinguisher* $\mathscr{A}$ *against the construction* $\mathsf{SAEB}^*[\mathsf{P}]$ *that makes at most* q_e *encryption,* q_d *decryption and* q_v *verification queries with at most* σ_e *primitive calls with distinct inputs in encryption queries,* σ_d *primitive calls with distinct inputs in decryption queries and* σ_v *primitive calls with distinct inputs in verification queries having a total of* $\sigma = \sigma_e + \sigma_d + \sigma_v$ *primitive calls with distinct inputs such that* $\rho \leq \sigma_e$*, where* ρ *is a parameter, is given by*

$$\mathbf{Adv}^{\mathsf{int\text{-}rup}}_{\mathsf{SAEB}^*[\mathsf{P}]}(\mathscr{A}) \leq \frac{\sigma^2}{2^{n+1}} + \frac{\sigma_e^{\rho}}{(2^r)^{\rho-1}} + \frac{q_v(2\rho+\sigma_d)}{2^c} + \frac{q_v}{2^{\tau}} + \frac{q_v(\sigma_e+\sigma_d)}{2^c}.$$

Proof. As the first step of the proof, we slightly modify the construction by replacing the random permutation with a random function $R \leftarrow\!\!\$\ \mathsf{Funcs}(\{0,1\}^n)$. We denote the resulting construction as Θ. This modification comes at the cost of birthday bound complexity due to the PRF-PRP switching lemma [13,14,23]. We consider a computationally unbounded non-trivial deterministic distinguisher $\mathscr{A}$ that interacts with a triplet of oracles in either of the two worlds: in the real world, it interacts with $(\Theta.\mathsf{enc}_R, \Theta.\mathsf{dec}_R, \Theta.\mathsf{ver}_R)$, and in the ideal world, it interacts with $(\Theta.\mathsf{enc}_R, \Theta.\mathsf{dec}_R, \perp)$, where $\perp$ denotes the oracle that always rejects the verification attempts. We summarize $\mathscr{A}$'s query-response in a transcript ω which is segregated into a transcript of encryption queries, decryption queries, verification queries. Basically, we segregate the transcript ω into three parts ω^+, ω^-, and $\omega^\times$, where $\omega^+ = \{(N_1^+, A_1^+, M_1^+, C_1^+, T_1^+), \ldots, (N_{q_e}^+, A_{q_e}^+, M_{q_e}^+, C_{q_e}^+, T_{q_e}^+)\}$ is a tuple of encryption queries, $\omega^- = \{(N_1^-, A_1^-, C_1^-, M_1^-), (N_2^-, A_2^-, C_2^-, M_2^-), \ldots, (N_{q_d}^-, A_{q_d}^-, C_{q_d}^-, M_{q_d}^-)\}$ is a tuple of decryption queries, and a tuple of verification queries $\omega^\times = \{(N_1^\times, A_1^\times, C_1^\times, T_1^\times, \perp_1), \ldots, (N_{q_v}^\times, A_{q_v}^\times, C_{q_v}^\times, T_{q_v}^\times, \perp_{q_v})\}$ such that $\omega = \omega^+ \cup \omega^- \cup \omega^\times$. We modify the experiment by releasing all internal state values to adversary $\mathscr{A}$ after it makes all the encryption, decryption and verification queries, and before it outputs the decision bit b. We denote the j-th internal state value in the i-th encryption, decryption and verification query as $s_i^+[j], s_i^-[j]$ and $s_i^\times[j]$ respectively. In general, we denote it as $s_i^\star[j]$, where $\star \in \{+,-,\times\}$. The length of associated data, message and ciphertext in the i-th query is denoted as $a_i^\star, m_i^\star$ and $c_i^\star$ respectively where $\star \in \{+,-,\times\}$. In the real world, these internal state variables for every encryption, decryption and verification queries are computed by the corresponding oracles that faithfully evaluate SAEB^*. Note that the sequence of internal state values, in the real world for the

i-th encryption query of length $a_i^+ + m_i^+ + 1$, are defined as follows:

$$\begin{cases} s_i^+[1] = A_i^+[1] \| \langle 0 \rangle_{c_1} \\ s_i^+[k] = A_i^+[k] \| \langle 0 \rangle_{c_1} \oplus R(s_i^+[k-1]), \text{ for } 2 \leq k < a_i^+ \\ s_i^+[a_i^+] = \mathsf{ozs}(A_i^+[a_i^+]) \| \langle \mathsf{const}_1 \rangle_c \oplus R(s_i^+[a_i^+ - 1]) \\ s_i^+[a_i^+ + 1] = N_i^+ \| \langle \mathsf{const}_2 \rangle_c \oplus R(s_i^+[a_i^+]) \\ s_i^+[a_i^+ + 1 + k] = M_i^+[k] \| \langle 0 \rangle_c \oplus R(s_i^+[k-1]), \text{ for } 1 \leq k < m_i^+ \\ s_i^+[a_i^+ + 1 + m_i^+] = \mathsf{ozs}(M_i^+[m_i^+]) \| \langle \mathsf{const}_3 \rangle_c \oplus R(s_i^+[a_i^+ + m_i^+]) \end{cases} \tag{2}$$

Similarly, we define the internal state values in the real world for decryption and verification queries. In the ideal world, as the encryption and decryption oracles are identical to that of the real world, the intermediate state variables for every encryption and decryption queries are faithfully evaluated by the corresponding oracles, and hence the sequence of state values for i-th encryption query is identically defined to Eqn. (2). Similar to the real world, we also define the internal state values in the ideal world for decryption queries. As the verification oracle $\perp$ in the ideal world always returns rejects and does not compute anything, the internal state variables are not defined. Therefore, we have to define the sampling of the intermediate state variables for every verification query in the ideal world. To achieve this, for every verification query (N, A, C, T), the verification oracle for the ideal world invokes the decryption oracle $\Theta.\mathsf{dec}_R$ with input (N, A, C) ignoring the output of r-bit plaintext strings. Finally, the verification oracle ignores the checking of the computed tag T^* with the given tag T. Thus, the sequence of internal state values is defined for verification queries. Let the modified attack transcripts be $\omega_{\mathtt{new}} = \omega_{\mathtt{new}}^+ \cup \omega_{\mathtt{new}}^- \cup \omega_{\mathtt{new}}^\times$, where

$$\begin{cases} \omega_{\mathtt{new}}^+ = \omega^+ \cup \{s_i^+[j] : i \in [q_e], j \in [a_i + m_i + 1]\} \\ \omega_{\mathtt{new}}^- = \omega^- \cup \{s_i^-[j] : i \in [q_d], j \in [a_i + m_i + 1]\} \\ \omega_{\mathtt{new}}^\times = \omega^\times \cup \{s_i^\times[j] : i \in [q_v], j \in [a_i + m_i + 1]\}. \end{cases}$$

For a given transcript $\omega_{\mathtt{new}}$, we reorder the transcript so that all the encryption queries appear first, followed by all the decryption queries and finally, all the verification queries. It is easy to see that a state collision occurs in $s_i^\star[j]$ and $s_{i'}^\circledast[j]$ with probability 1, where $\star, \circledast \in \{+, -, \times\}$, if

- $A_i^\star[\ldots j] = A_{i'}^\circledast[\ldots j]$, when $j \leq a_i^\star$,
- $A_i^\star = A_{i'}^\circledast$, $N_i^\star = N_{i'}^\circledast$, when $j = a_i^\star + 1$,
- $N_i^\star = N_{i'}^\circledast$, $A_i^\star = A_i^\circledast$, $M_i^\star[\ldots (j - a_i^\star - 1)] = M_{i'}^\circledast[\ldots (j - a_i^\star - 1)]$, when $a_i^\star + 1 < j \leq a_i^\star + 1 + m_i^\star$.

A state collision that happens with probability 1 is called a *trivial collision*, and any other state collision is *non-trivial*. For $j > 1$, we write $\mathsf{ancestor}(s_i^\star[j])$ to denote the sequence of state values $(s_i^\star[1], \ldots, s_i^\star[j-1])$ that leads to $s_i^\star[j]$ in the i-th query and $\mathsf{ancestor}(s_i^\star[1]) = \phi$, where $\star \in \{+, -, \times\}$. Using this notion, we say that a trivial state collision between $s_i^\star[j]$ and $s_{i'}^\circledast[j]$ occurs if and only if $\mathsf{ancestor}(s_i^\star[j]) = \mathsf{ancestor}(s_{i'}^\circledast[j])$ for some j. Let $D_i^\star := A_i^\star \| N_i^\star$ and $d_i^\star := a_i^\star + 1$

where $\star \in \{+, -, \times\}$. Let us consider two queries $\star, \circledast \in \{+, -, \times\}$ with distinct query indices i and i', where $i, i' \in [q_e + q_d + q_v]$. We define the *longest common prefix* of (i, i'), denoted as $\mathsf{LCP}(i, i')$

$$\begin{cases} j, \text{ if } j \in [d_i^\star], D_i^\star[1..j] = D_{i'}^\circledast[1..j], D_i^\star[j+1] \neq D_{i'}^\circledast[j+1] \\ d_i^\star, \text{ if } D_i^\star = D_{i'}^\circledast, M_i^\star[1] \neq M_{i'}^\circledast[1] \\ j, \text{ if } j \in [d_i^\star, d_i^\star + m_i^\star], D_i^\star = D_{i'}^\circledast, M_i^\star[1..j] = M_{i'}^\circledast[1..j], M_i^\star[j+1] \neq M_{i'}^\circledast[j+1] \\ d_i^\star + m_i^\star, \text{ if } D_i^\star = D_{i'}^\circledast, M_i^\star = M_{i'}^\circledast[1..m_i^\star] \end{cases}$$

Consequently, we define $\mathsf{LLCP}(i) \triangleq \max_{i'<i}\{\mathsf{LCP}(i, i')\}$, that denotes the longest common prefix of query index $i \in [q_e + q_d + q_v]$.

4.1 Definition and Probability of Bad Transcripts

In this section, we define and bound the probability of bad transcripts in the ideal world. Let Ω be the set of all attainable transcripts and $\omega_{\mathtt{new}} \in \Omega$ be one such attainable transcript. We say that transcript $\omega_{\mathtt{new}}$ is **bad**, i.e., $\omega_{\mathtt{new}} \in \Omega_{\mathtt{bad}}$, if at least one of the following holds:

1. Coll: there exists i, j, i', j' with $(i', j') < (i, j)$ with $\mathsf{LLCP}(i) < j \leq a_i + m_i + 1$ and $i \leq [q_e + q_d + q_v]$ such that $s_i^\star[j] = s_{i'}^\circledast[j']$, where $\star, \circledast \in \{+, -, \times\}$.
2. mColl: $\exists i_1, j_1, \ldots, i_\rho, j_\rho$ with $\{i_1, \ldots, i_\rho\} \subseteq [q_e]$ and for all $1 \leq k \leq \rho$, $j_k \in [m_{i_k}]$, such that $C_{i_1}^+[j_1] = \cdots = C_{i_\rho}^+[j_\rho]$.
3. Forge: This event happens if for some verification query, all its intermediate states prior to the final state matches with intermediate state from some encryption or decryption queries; and the final state of the verification query matches with the final state of an encryption query for which the tag matches. In other words, $\exists i \in [q_v]$ such that for the i-th verification query $(N_i^\times, A_i^\times, C_i^\times, T_i^+)$, the following two events hold:

$$\begin{cases} \forall j \in [a_i^\times + 1, (a_i^\times + c_i^\times)], \exists i', j' \text{ such that } s_i^\times[j] = s_{i'}^+[j'] \text{ or } s_i^\times[j] = s_{i'}^-[j'], \\ \exists f \in [q_e] \text{ such that } s_i^\times[a_i^\times + c_i^\times + 1] = s_f^+[a_f^+ + c_f^+ + 1] \text{ with } T_i^+ = T_f^+. \end{cases}$$

We now compute the probability of a transcript being bad in the ideal world. Using the union bound, we have

$$\mathsf{Pr}[\Lambda_0 \in \Omega_{\mathtt{bad}}] = \mathsf{Pr}[\mathsf{Coll} \vee \mathsf{mColl} \vee \mathsf{Forge}]. \tag{3}$$

Bounding Coll: For this event to happen, we know that there exists at least one pair of indices $(i', j') < (i, j)$ such that $\mathsf{LLCP}(i) < j \leq a_i + m_i + 1$ and $s_i^\star[j] = s_{i'}^\circledast[j']$. For any value of $j \in [1, a_i + m_i + 1]$, we have,

$$s_i^\star[j] = s_{i'}^\star[j] \Leftrightarrow R(s_i^\star[j-1]) \oplus R(s_{i'}^\star[j-1]) = x_i^\star[j] \oplus x_{i'}^\circledast[j] \tag{4}$$

where $x_i^\star[j]$ and $x_{i'}^\circledast[j]$ are two n-bit strings. Note that if $j > a_i + 1$, the first r-bits of $x_i^\star[j]$ is xored with message block if $\star = +$ or the first r-bits of $x_i^\star[j]$ is

replaced by the ciphertext block if $\star \in \{-, \times\}$. Similarly, if $j > a_i + 1$ the first r-bits of $x_{i'}^{\circledast}[j]$ is xored with message block if $\circledast = +$ or the first r-bits of $x_{i'}^{\circledast}[j]$ is replaced by a ciphertext block if $\circledast \in \{-, \times\}$. On the other hand, if $j \leq a_i + 1$, then first r-bits of $x_i^{\star}[j]$ and $x_{i'}^{\circledast}[j]$ is xored with associated data or nonce for $\star, \circledast \in \{+, -, \times\}$. First, let us consider the case when $j = j' = \mathsf{LLCP}(i) + 1$ and $j > 1$. Then there exists some i' such that $i' < i$ and $\mathsf{LLCP}(i)$ holds for i'. Now we consider the values of $s_i^{\star}[j-1]$ and $s_{i'}^{\circledast}[j'-1]$. Based on the values of j, we get following cases.

1. Case $j < a_i^{\star} + 1$. In this case, we have $D_i^{\star}[1..j-1] = D_{i'}^{\circledast}[1..j-1]$ i.e, $A_i^{\star}[1..j] = A_{i'}^{\circledast}[1..j]$ Then from the Equation (2), we have $s_i^{\star}[j-1] = s_{i'}^{\circledast}[j'-1]$.
2. Case $j = a_i^{\star} + 1$. In this case, $\mathsf{LLCP}(i) = a_i^{\star}$. Thus, $D_i^{\star}[1..a_i^{\star}] = D_{i'}^{\circledast}[1..a_i^{\star}]$ i.e, $A_i^{\star} = A_{i'}^{\circledast}$. Then from the Equation (2), we have $s_i^{\star}[j-1] = s_{i'}^{\circledast}[j'-1]$.
3. Case $j = a_i^{\star} + 2$. In this case, the associated date and nonce in i-th and i'-th query matches. So, we have $s_i^{\star}[j-1] = s_{i'}^{\circledast}[j'-1]$.
4. Case $a_i^{\star} + 2 \leq j \leq a_i^{\star} + m_i^{\star} + 1$. In this case, the nonce and associated data in the i-th and i'-th query matches. Also, the message/ciphertext in i-th and i'-th query matches up to $(j - a_i^{\star} - 2)$-th block. Thus from the Equation (15), we have $s_i^{\star}[j-1] = s_{i'}^{\circledast}[j'-1]$.

Therefore, for any value of $j \in [2, a_i + m_i + 1]$, $s_i^{\star}[j-1] = s_{i'}^{\circledast}[j'-1]$. Thus, the probability of the event,

$$s_i^{\star}[j] = s_{i'}^{\star}[j] \Leftrightarrow R(s_i^{\star}[j-1]) \oplus R(s_{i'}^{\star}[j-1]) = x_i^{\star}[j] \oplus x_{i'}^{\circledast}[j] \Leftrightarrow x_i^{\star}[j] \oplus x_{i'}^{\circledast}[j] = 0$$

is zero. On the other hand, for all $i' \leq i$ and $j' \neq j$ or $j \neq \mathsf{LLCP}(i) + 1$, $R(s_i^{\star}[j-1]) \oplus R(s_{i'}^{\star}[j-1]) = x_i^{\star}[j] \oplus x_{i'}^{\circledast}[j]$ holds with probability at most 2^{-n}. Thus, the probability that two states collide is 2^{-n}. Note that there are σ possible values of (i, j) in a transcript, each having no more than σ possible values of (i', j'), where σ is the total number of permutation calls including all encryption, decryption and verification queries, such that $s_i^{\star}[j] = s_{i'}^{\circledast}[j']$ holds. Therefore, we have

$$\mathsf{Pr}[\mathsf{Coll}] \leq \binom{\sigma}{2} \frac{1}{2^n} \leq \frac{\sigma^2}{2^{n+1}}. \tag{5}$$

Bounding mColl: We bound this event by conditioning the event that Coll does not occur. As there are no non-trivial collisions, $\mathsf{ancestor}(s_{i_1}^{+}[j_1])$, $\mathsf{ancestor}(s_{i_2}^{+}[j_2])$, ..., $\mathsf{ancestor}(s_{i_\rho}^{+}[j_\rho])$ are all distinct and fresh. Therefore, all the outputs $R(s_{i_1}^{+}[j_1])$, $R(s_{i_2}^{+}[j_2]), ..., R(s_{i_\rho}^{+}[j_\rho])$ are all uniformly sampled over $\{0,1\}^n$. Thus, from the randomness of R, we can view this event as throwing σ_e balls into 2^r bins (as we are seeking collisions in the rate part) uniformly at random, where σ_e denotes the total number of primitive calls including all encryption queries and we want to find the probability that there is a bin that contains ρ or more balls. In other words, ρ or more outputs take some constant value c. This event occurs with probability at most $(\frac{1}{2^r})^{\rho}$. Again, we have 2^r choices for the constant value c. Therefore, by varying the choices of all encryption queries, we have

$$\mathsf{Pr}[\mathsf{mColl} \mid \overline{\mathsf{Coll}}] \leq \binom{\sigma_e}{\rho} \times 2^r (\frac{1}{2^r})^{\rho} \leq \frac{\sigma_e^{\rho}}{(2^r)^{\rho-1}}, \tag{6}$$

where the last inequality follows from Stirling's approximation ignoring the constant term.

Bounding Forge: We fix a verification query $(N_i^{\times}, A_i^{\times}, C_i^{\times}, T_i^{\times})$ with associated data length $a_i^{\times}$ and ciphertext length $c_i^{\times}$ such that $\forall a_i^{\times}+1 \leq j \leq (a_i^{\times}+c_i^{\times}), \exists i', j'$ s.t $s_i^{\times}[j] = s_{i'}^{\star}[j']$ where $\star \in \{+,-\}$. Let $\overline{j}$ be the largest index for which $s_i^{\times}[\overline{j}]$ does have a trivial collision with $s_{i'}^{\star}[j]$. We bound the probability of Forge when Coll and mColl do not occur. Now, we consider the following two cases based on the values of $\overline{j}$.

(a) Consider $a_i^{\times}+1 \leq \overline{j} < a_i^{\times}+c_i^{\times}$. In this case the associated data, nonce and some parts of the message match with some previous query. Here, the adversary can control the rate part and so $s_i^{\times}[\overline{j}+1]$ matches with some encryption or decryption query with probability at most $\frac{\rho+\sigma_d}{2^c}$.
(b) Finally, consider the case $\overline{j} = a_i^{\times}+c_i^{\times}$. So the final state matches with some previous encryption query with probability at most $\frac{\rho}{2^c}$.

Combining everything together and by varying over the choices of all the verification queries,

$$\mathsf{Pr}[\mathsf{Forge} \mid \overline{\mathsf{Coll}} \wedge \overline{\mathsf{mColl}}] \leq \frac{q_v(2\rho+\sigma_d)}{2^c}. \tag{7}$$

From Eqn. (3)-Eqn. (7), we obtain the probability of a transcript being bad as,

$$\begin{aligned}\mathsf{Pr}[\Lambda_0 \in \Omega_{\mathsf{bad}}] &\leq \mathsf{Pr}[\mathsf{Coll}] + \mathsf{Pr}[\mathsf{mColl} \mid \overline{\mathsf{Coll}}] + \mathsf{Pr}[\mathsf{Forge} \mid \overline{\mathsf{Coll}} \wedge \overline{\mathsf{mColl}}] \\ &\leq \frac{\sigma^2}{2^{n+1}} + \frac{\sigma_e^{\rho}}{(2^r)^{\rho-1}} + \frac{q_v(2\rho+\sigma_d)}{2^c}. \end{aligned} \tag{8}$$

4.2 Analysis of the Good Transcripts

In this section, we show that for a good transcript $\omega_{\mathtt{new}} \in \Omega_{\mathtt{new}}$, the probability of realizing $\omega_{\mathtt{new}}$ in the real world is as likely as in the ideal world. It is easy to see that for a good transcript $\omega_{\mathtt{new}}$, we have that $\Pr[\Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^- = \omega_{\mathtt{new}}^-] = \Pr[\Lambda_0^+ = \omega_{\mathtt{new}}^+, \Lambda_0^- = \omega_{\mathtt{new}}^-]$. Thus, the ratio of interpolation probabilities is given by

$$\begin{aligned}\frac{\Pr[\Lambda_1 = \omega_{\mathtt{new}}]}{\Pr[\Lambda_0 = \omega_{\mathtt{new}}]} &= \Pr[\Lambda_1^{\times} = \omega_{\mathtt{new}}^{\times} | \Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^+ = \omega_{\mathtt{new}}^+] \\ &\geq 1 - \Pr[\Lambda_1^{\times} \neq \omega_{\mathtt{new}}^{\times} \mid \Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^- = \omega_{\mathtt{new}}^-],\end{aligned}$$

where we have used the fact that $\Pr[\Lambda_0^{\times} = \omega_{\mathtt{new}}^{\times} \mid \Lambda_0^+ = \omega_{\mathtt{new}}^+, \Lambda_0^- = \omega_{\mathtt{new}}^-] = 1$, because in the ideal world, the response to any verification query is $\perp$. For $i \in [q_v]$, let $\mathbf{E}_i$ denote the event $\mathbf{E}_i \triangleq \left(\bar{\lambda}^i \neq \perp \;\middle|\; \Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^- = \omega_{\mathtt{new}}^-\right)$, where $\bar{\lambda}^i$ be the random variable that denotes the response to the i-th verification query in the real world. Then, we have

$$\Pr[\Lambda_1^{\times} \neq \omega_{\mathtt{new}}^{\times} \mid \Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^- = \omega_{\mathtt{new}}^-] \leq \sum_{i \in [q_v]} \Pr[\mathbf{E}_i].$$

We fix $i \in [q_v]$ and let the i-th verification query be $(N_i^\times, A_i^\times, C_i^\times, T_i^\times)$, of associated data length $a_i^\times$ and ciphertext length $c_i^\times$. We want to bound $\Pr[\mathbf{E}_i]$. This probability is non-zero only if $\Theta.\mathsf{ver}_R$ returns anything other than $\perp$. If $s_i^\times[a_i^\times + c_i^\times + 1]$ does not match with the final state of any encryption query, $\Pr[\mathbf{E}_i] \leq \frac{1}{2^\tau}$ holds trivially. Suppose, there exists some encryption query such that the final state matches with $s_i^\times[a_i^\times + c_i^\times + 1]$. In this case, there must exist some j such that $s_i^\times[j]$ is fresh, otherwise Forge is true. Let j^* be the maximum of such j. Then, $s_i^\times[j^*+1]$ matches with some previous encryption or decryption state with probability at most $\frac{(\sigma_e+\sigma_d)}{2^c}$, as the adversary can control only the rate-part of these states. Putting everything together we get:

$$\Pr[\mathbf{E}_i] \leq \frac{1}{2^\tau} + \frac{(\sigma_e + \sigma_d)}{2^c}.$$

Therefore, we have

$$\Pr[\mathsf{\Lambda}_1^\times \neq \omega_{\mathtt{new}}^\times \mid \mathsf{\Lambda}_1^+ = \omega_{\mathtt{new}}^+, \mathsf{\Lambda}_1^- = \omega_{\mathtt{new}}^-] \leq \frac{q_v}{2^\tau} + \frac{q_v(\sigma_e + \sigma_d)}{2^c}. \tag{9}$$

The result follows as we combine Eqn. (8), Eqn. (9) and Theorem 1. □

Significance of INT-RUP Security of SAEB: If we instantiate SAEB with AES-128 block cipher and restrict its message *injection* rate to $r = 32$-bits, then the capacity c will be of 96-bits, and hence SAEB-32 will provide INT-RUP security up to 2^{48} decryption/verification blocks[4], which satisfies the NIST criteria of having 2^{50} byte of data-complexity. However, for SAEAES, where the message injection rate is $r = 64$-bits, we only achieve 2^{32} block of INT-RUP security. This result signifies that if we wish to have an INT-RUP secure variant of SAEB, we can simply use the same construction as SAEAES but with a lower message injection rate.

5 TinyJAMBU and Its INT-RUP Security

$\mathsf{TinyJAMBU}$ is one of the finalists of the NIST lightweight competition. The design principle of $\mathsf{TinyJAMBU}$ follows the sponge duplex mode based on keyed permutations derived from lightweight LFSRs. Unlike SAEB, $\mathsf{TinyJAMBU}$ injects the message in a specific part of the state and squeeze from a different part of the state to output the ciphertext. Therefore, we have a r-bit message injection part, r-bit squeezing part, and a c-bit unaltered capacity part, which is xored with the frame constants. Together, we have a total state size of $n = c + 2r$-bits. $\mathsf{TinyJAMBU}$ uses two different keyed permutations with the same key K. These two keyed permutations are similar in structure but differs only in the number of rounds. One permutation consists of 384 rounds of a LFSR, which we denote as $P_K^{(1)}$ and the another permutation consists of 1024 rounds of the same LFSR, which we denote as $P_K^{(2)}$. The encryption and the decryption algorithm

[4] Security bound of the SAEB is moot if the number of encryption blocks exceeds 2^{32}.

of TinyJAMBU starts with the Init function that mixes the key K and the nonce N to produce a pseudorandom state. In particular, the Init function consists of two steps: key set up phase and nonce set up phase. In the key setup phase, an 128-bit register is initialized with all 0 and update the state by the keyed permutation $P_K^{(2)}$. In the nonce setup phase, an 96-bits nonce N is splitted up into three 32-bits nonces $N[1]\|N[2]\|N[3]$. Followed by it, for each $i \in \{1,2,3\}$, it updates the intermediate state s by xoring $0\|\mathsf{const}_i\|0$ with the current value of s. Then, invoke $P_K^{(1)}$ on s and finally xor the output with $N[i]$ to update the intermediate state s.
The algorithmic description of the encryption function of TinyJAMBU is given in Fig. 2, and its schematic diagram is depicted in Fig. 5. In the original proposal of the scheme [43], the recommended parameters of TinyJAMBU are $r = 32$, $c = 64$. const_i denotes 3-bits frame constants for $i \in \{1,2,3,4\}$, where $\mathsf{const}_1 = 001, \mathsf{const}_2 = 011, \mathsf{const}_3 = 101, \mathsf{const}_4 = 111$. TinyJAMBU achieves birthday bound security with dominant terms being $(e\sigma_e/\rho 2^r)^\rho (2^r/\sqrt{\rho}) + (\sigma_a + \sigma_d)(\rho - 1)/2^{c+(r/2)+1}$, where ρ is a properly chosen constant. In the following, we state and prove the INT-RUP security result of TinyJAMBU*[P].

Theorem 3. *Let* $\mathsf{P} \leftarrow_{\$} \mathsf{Perm}(n)$ *be an* n*-bit uniform random permutation. Let* r, c *and* ρ *be three parameters such that* $n = c + 2r$ *and let* τ *be the bit size of the tag output by* TinyJAMBU. *The* INT-RUP *advantage for any* $(q_e, q_d, q_v, \sigma_e, \sigma_d, \sigma_v)$*-distinguisher* $\mathscr{A}$ *against the construction* TinyJAMBU*[P] *that makes at most* q_e *encryption,* q_d *decryption and* q_v *verification queries with at most* σ_e *primitive calls with distinct inputs in encryption queries,* σ_d *primitive calls with distinct inputs in decryption queries and* σ_v *primitive calls with distinct inputs in verification queries having a total of* $\sigma = \sigma_e + \sigma_d + \sigma_v$ *primitive calls with distinct inputs such that* $\rho \leq \sigma_e$ *is given by*

$$\mathbf{Adv}^{\mathsf{int\text{-}rup}}_{\mathsf{TinyJAMBU}^*}(\mathscr{A}) \leq \frac{\sigma^2}{2^{n+1}} + \frac{\sigma_e^\rho}{(2^r)^{\rho-1}} + \frac{q_v(2\rho + \sigma_d)}{2^{n-r}} + \frac{q_v}{2^\tau} + \frac{q_v(\sigma_e + \sigma_d)}{2^{n-r}}.$$

Remark 1. Recently two independent works by Sibleyras et al. [40] and Dunkelman et al. [26] have shown some vulnerabilities on the underlying permutation of TinyJAMBU. Note that these results do not imply any insecurity of the mode TinyJAMBU per se. In this paper, we prove the INT-RUP security of TinyJAMBU by viewing it as a mode which is build on top of some secure keyed permutations. Note that the construction uses two different keyed permutations with the same key but differs only in the number of rounds, we model the security proof of the construction in the standard setting, where we replace these two keyed permutations with an *n-bit uniform random permutation* and denote the resulting construction as TinyJAMBU*[P].

Proof. We proceed similar to the proof of Theorem 2. We modify the construction by replacing the permutations with random function $R \leftarrow_{\$} \mathsf{Funcs}(\{0,1\}^n)$ and denoting the resulting construction as Θ. We consider a distinguisher $\mathscr{A}$, that interacts with a triplet of oracles in either of the worlds: in the real world, it

TinyJAMBU(K, N, A, M)

1 : $A[1]\| \ldots \|A[a] \xleftarrow{r} A;\ M[1]\| \ldots \|M[m] \xleftarrow{r} M;$
2 : $s[0] \leftarrow \mathsf{Init}(K, N);$
3 : `for` $i = 1$ `to` $a - 1;$
4 : $\quad s[i-1] \leftarrow s[i-1] \oplus (0\|\mathsf{const}_2\|0);$
5 : $\quad s[i] \leftarrow P_K^{(1)}(s[i-1]) \oplus A[i];$
6 : $s[a-1] \leftarrow s[a-1] \oplus (0\|\mathsf{const}_2\|0);$
7 : $s[a] \leftarrow P_K^{(1)}(s[a-1]) \oplus A[a]\|10^{r-|A[a]|-1}\|lp(A[a]);$
8 : `for` $i = 1$ `to` $m - 1;$
9 : $\quad s[a+i-1] \leftarrow s[a+i-1] \oplus (0\|\mathsf{const}_3\|0);$
10 : $\quad s[a+i] \leftarrow P_K^{(2)}(s[a+i-1]) \oplus M[i];$
11 : $\quad C[i] \leftarrow M[i] \oplus s[a+i][64...(64+r-1)];$
12 : $s[a+m-1] \leftarrow s[a+m-1] \oplus (0\|\mathsf{const}_3\|0);$
13 : $s[a+m] \leftarrow P_K^{(2)}(s[a+m-1]) \oplus M[m]\|10^{r-|M[m]|-1}\|0^r\|lp(M[m]);$
14 : $C[m] \leftarrow M[m] \oplus s[a+m][64...(64+|M[m]|-1)];$
15 : $s[a+m] \leftarrow P_K^{(2)}(s[a+m] \oplus (0\|\mathsf{const}_4\|0));$
16 : $T_1 \leftarrow s[a+m][64...(64+\tau/2-1)];$
17 : $s[a+m] \leftarrow P_K^{(1)}(s[a+m] \oplus (0\|\mathsf{const}_4\|0));$
18 : $T_2 \leftarrow s[a+m][64...(64+\tau/2-1)];$
19 : $T \leftarrow T_1\|T_2;$
return $(C, T);$

Fig. 2: Formal Specification of TinyJAMBU authenticated encryption mode. The function $lp(X) := \lfloor |X|/8 \rfloor$ denotes the binary representation of the number of bytes present in a binary string X.

interacts with $(\Theta.\mathsf{enc}_R, \Theta.\mathsf{dec}_R, \Theta.\mathsf{ver}_R)$ and in the ideal world it interacts with $(\Theta.\mathsf{enc}_R, \Theta.\mathsf{dec}_R, \perp)$. We summarize the queries in a transcript ω and segregate the transcript ω into three parts ω^+, ω^-, and $\omega^\times$ as described for the analysis of SAEB and we have $\omega = \omega^+ \cup \omega^- \cup \omega^\times$. The length of associated data, message and ciphertext in the i-th query is denoted as $a_i^\star, m_i^\star$ and $c_i^\star$ respectively where $\star \in \{+, -, \times\}$. We denote the j-th input to R in the i-th encryption, decryption and verification query as $s_i^+[j], s_i^-[j]$ and $s_i^\times[j]$ respectively and in general $s_i^\star[j]$ where $\star \in \{+, -, \times\}$. In the above description, the nonce in the i-th encryption query is $N_i^+ = N_i^+[1]\|N_i^+[2]\|N_i^+[3]$ is processed in three steps. We modify the experiment by releasing all internal state values to the adversary $\mathscr{A}$ before it outputs the decision bit b, but after it makes all the queries. As the verification oracle $\perp$ in the ideal world always returns reject and does not compute anything, the internal state variables are not defined. Thus, similar to the proof of

Theorem 2, we have to define the sampling of the intermediate state variables for every verification query in the ideal world. To achieve this, for every verification query (N, A, C, T) made by adversary $\mathcal{A}$, the verification oracle for the ideal world invokes the decryption oracle $\Theta.\mathsf{dec}_R$ with input (N, A, C) ignoring the output of r-bit plaintext strings. Finally, the verification oracle ignores the checking of the computed tag T^* with the given tag T. Hence, the sequence of internal state values is defined for verification queries. Let the modified attack transcripts be $\omega_{\mathtt{new}} = \omega^{+}_{\mathtt{new}} \cup \omega^{-}_{\mathtt{new}} \cup \omega^{\times}_{\mathtt{new}}$. For a given transcript $\omega_{\mathtt{new}}$, we reorder the transcript in such a way that all the encryption queries appear first, followed by all the decryption queries and finally, all the verification queries. Now, it is easy to see that a state collision in $s^{\star}_i[j]$ and $s^{\circledast}_{i'}[j]$ occurs with probability 1 if

- $N^{\star}_i = N^{\circledast}_{i'}$, $A^{\star}_i[\ldots j] = A^{\circledast}_{i'}[\ldots j]$, when $j \leq a^{\star}_i$,
- $N^{\star}_i = N^{\circledast}_{i'}$, $A^{\star}_i = A^{\circledast}_i$, $M^{\star}_i[\ldots (j - a^{\star}_i)] = M^{\circledast}_{i'}[\ldots (j - a^{\star}_i)]$, when $a^{\star}_i < j \leq a^{\star}_i + m^{\star}_i$.

A state collision with probability 1 is called trivial, and any other state collision is called non-trivial. We denote $D^{\star}_i = N^{\star}_i \| A^{\star}_i$ and $d^{\star}_i = a^{\star}_i + 3$ where $\star \in \{+, -, \times\}$. Let us consider two query $\star, \circledast \in \{+, -, \times\}$ with distinct query indices i and i', where $i, i' \in [q_e + q_d + q_v]$. Similar to the proof of Theorem 2, we use the notations $\mathsf{ancestor}, \mathsf{LCP}(i, i')$ and $\mathsf{LLCP}(i)$.

5.1 Definition and Probability of Bad Transcripts

In this section, we define and bound the probability of bad transcripts in the ideal world. The idea of this proof is almost similar to that of Theorem 2. However, the bounds are different because the adversary cannot control the rate part from where the squeezing occurs. Let Ω be the set of all attainable transcripts and $\omega_{\mathtt{new}} \in \Omega$ be one such attainable transcript. We say that transcript $\omega_{\mathtt{new}}$ is **bad**, i.e., $\omega_{\mathtt{new}} \in \Omega_{\mathtt{bad}}$, if at least one of the following holds:

1. Coll: there exists i, j, i', j' with $(i', j') < (i, j)$ with $\mathsf{LLCP}(i) < j \leq a_i + m_i + 3$ and $i \leq [q_e + q_d + q_v]$ such that $s^{\star}_i[j] = s^{\circledast}_{i'}[j']$, where $\star, \circledast \in \{+, -, \times\}$.
2. mColl: $\exists i_1, j_1, \ldots, i_\rho, j_\rho$ with $\{i_1, \ldots, i_\rho\} \subseteq [q_e]$ and for all $1 \leq k \leq \rho$, $j_k \in [m_{i_k}]$, such that $C^{+}_{i_1}[j_1] = \cdots = C^{+}_{i_\rho}[j_\rho]$.
3. Forge: This event happens, if for some verification query, all its intermediate states prior to the final state match with the intermediate state from some encryption or decryption queries, and the final state of the verification query matches with the final state of the encryption query for which the tag matches. In other words, $\exists i \in [q_v]$ such that for the i-th verification query $(N^{\times}_i, A^{\times}_i, C^{\times}_i, T^{+}_i)$, the following two events hold:

$$\begin{cases} \forall j \in [a^{\times}_i, (a^{\times}_i + c^{\times}_i - 1)], \exists i', j' \text{ such that } s^{\times}_i[j] = s^{+}_{i'}[j'] \text{ or } s^{\times}_i[j] = s^{-}_{i'}[j'], \\ \exists f \in [q_e] \text{ such that } s^{\times}_i[a^{\times}_i + c^{\times}_i] = s^{+}_f[a^{+}_f + c^{+}_f] \text{ with } T^{+}_i = T^{+}_f. \end{cases}$$

We now compute the probability of a transcript being bad in the ideal world. If $\omega_{\mathtt{new}}$ is a transcript observed in the ideal world, we want to calculate the

probability of $\omega_{\mathtt{new}}$ to satisfy one of the above conditions. By applying the union bound, we have

$$\begin{aligned}\mathsf{Pr}[\Lambda_0 \in \Omega_{\mathtt{bad}}] &= \mathsf{Pr}[\mathsf{Coll} \vee \mathsf{mColl} \vee \mathsf{Forge}] \\ &\leq \mathsf{Pr}[\mathsf{Coll}] + \mathsf{Pr}[\mathsf{mColl} \mid \overline{\mathsf{Coll}}] + \mathsf{Pr}[\mathsf{Forge} \mid \overline{\mathsf{Coll}} \wedge \overline{\mathsf{mColl}}] \\ &\leq \frac{\sigma^2}{2^{n+1}} + \frac{\sigma_e^\rho}{(2^r)^{\rho-1}} + \frac{q_v(2\rho+\sigma_d)}{2^{n-r}}. \end{aligned} \tag{10}$$

The calculation for the bounds for each of the above terms are given in the supplementary section.

5.2 Analysis of the Good Transcripts

In this section, we show that for a good transcript $\omega_{\mathtt{new}} \in \Omega_{\mathtt{new}}$, the probability of realizing $\omega_{\mathtt{new}}$ in the real world is as likely as in the ideal world. It is easy to see that for a good transcript $\omega_{\mathtt{new}}$, we have $\Pr[\Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^- = \omega_{\mathtt{new}}^-] = \Pr[\Lambda_0^+ = \omega_{\mathtt{new}}^+, \Lambda_0^- = \omega_{\mathtt{new}}^-]$. Thus, the ratio of interpolation probabilities is given by

$$\begin{aligned}\frac{\Pr[\Lambda_1 = \omega_{\mathtt{new}}]}{\Pr[\Lambda_0 = \omega_{\mathtt{new}}]} &= \Pr[\Lambda_1^\times = \omega_{\mathtt{new}}^\times | \Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^+ = \omega_{\mathtt{new}}^+] \\ &\geq 1 - \Pr[\Lambda_1^\times \neq \omega_{\mathtt{new}}^\times \mid \Lambda_1^+ = \omega_{\mathtt{new}}^+, \Lambda_1^- = \omega_{\mathtt{new}}^-], \\ &\geq 1 - (\sum_{i \in [q_v]} \frac{1}{2^\tau} + \frac{(\sigma_e + \sigma_d)}{2^{n-r}}) \\ &\geq 1 - (\frac{q_v}{2^\tau} + \frac{q_v(\sigma_e+\sigma_d)}{2^{n-r}}). \end{aligned} \tag{11}$$

Finally, we obtain the result combining Eqn. (10), Eqn. (11) and using Theorem 1. □

6 Conclusion and Future Works

In this paper, we have analyzed the INT-RUP security of SAEB and TinyJAMBU. Our analysis on TinyJAMBU is particularly relevant from the NIST lightweight competition perspective, and we believe that our result may positively impact TinyJAMBU during the final portfolio selection. Our result on SAEB depicts that the INT-RUP security can be achieved by controlling the message injection rate without incurring any additional overheads. A similar analysis for the permutation based constructions may be considered as a future research direction. However, we would like to mention that the trick, similar to SAEB, cannot be applied on TinyJAMBU as message injection and ciphertext release occur from different parts of its state. It would be interesting to come up with a matching INT-RUP attack on TinyJAMBU. Note that Oribatida achieves INT-RUP security, but at the cost of maintaining an additional state. Investigating INT-RUP secure permutation based sponge constructions without any additional overhead seems a challenging open problem.

References

1. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D, 2007. National Institute of Standards and Technology.
2. Riham AlTawy, Guang Gong, Morgan He, Ashwin Jha, Kalikinkar Mandal, Mridul Nandi, and Raghvendra Rohit. SpoC: An Authenticated Cipher Submission to the NIST LWC Competition, 2019. `https://csrc.nist.gov/projects/lightweight-cryptography/round-2-candidates`.
3. Elena Andreeva, Amit Singh Bhati, and Damian Vizar. Nonce-misuse security of the saef authenticated encryption mode. Cryptology ePrint Archive, Report 2020/1524, 2020.
4. Elena Andreeva, Amit Singh Bhati, and Damian Vizar. Rup security of the saef authenticated encryption mode. Cryptology ePrint Archive, Report 2021/103, 2021.
5. Elena Andreeva, Begül Bilgin, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. Ape: Authenticated permutation-based encryption for lightweight cryptography. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption*, 2015.
6. Elena Andreeva, Andrey Bogdanov, Nilanjan Datta, Atul Luykx, Bart Mennink, Mridul Nandi, Elmar Tischhauser, and Kan Yasuda. COLM v1. CAESAR Competition.
7. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. How to securely release unverified plaintext in authenticated encryption. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I*, pages 105–125, 2014.
8. Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. AES-COPA v.2. Submission to CAESAR, 2015. `https://competitions.cr.yp.to/round2/aescopav2.pdf`.
9. Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. Forkcipher: A new primitive for authenticated encryption of very short messages. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part II*, volume 11922 of *Lecture Notes in Computer Science*, pages 153–182. Springer, 2019.
10. Tomer Ashur, Orr Dunkelman, and Atul Luykx. Boosting authenticated encryption robustness with minimal modifications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2017.
11. Subhadeep Banik, Andrey Bogdanov, Atul Luykx, and Elmar Tischhauser. Sundae: Small universal deterministic authenticated encryption for the internet of things. *IACR Transactions on Symmetric Cryptology*, 2018(3), 2018.
12. Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. SKINNY-AEAD and skinny-hash. *IACR Trans. Symmetric Cryptol.*, 2020(S1):88–131, 2020.

13. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining. In Yvo Desmedt, editor, *CRYPTO*, volume 839 of *LNCS*, pages 341–358. Springer, 1994.
14. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006.
15. Arghya Bhattacharjee, Cuauhtemoc Mancillas López, Eik List, and Mridul Nandi. The oribatida v1.3 family of lightweight authenticated encryption schemes. *Journal of Mathematical Cryptology*, 15(1), 2021.
16. CAESAR: Competition for Authenticated Encryption: Security, Applicability, and Robustness, 2014. `http://competitions.cr.yp.to/caesar.html`.
17. Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas-López, Mridul Nandi, and Yu Sasaki. INT-RUP secure lightweight parallel AE modes. *IACR Trans. Symmetric Cryptol.*, 2019(4):81–118, 2019.
18. Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Snehal Mitragotri, and Mridul Nandi. From combined to hybrid: Making feedback-based AE even smaller. *IACR Trans. Symmetric Cryptol.*, 2020(S1):417–445, 2020.
19. Avik Chakraborti, Nilanjan Datta, and Mridul Nandi. INT-RUP analysis of block-cipher based authenticated encryption schemes. In Kazue Sako, editor, *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 - March 4, 2016, Proceedings*, volume 9610 of *Lecture Notes in Computer Science*, pages 39–54. Springer, 2016.
20. 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.
21. Avik Chakraborti, Tetsu Iwata, Kazuhiko Minematsu, and Mridul Nandi. Blockcipher-based authenticated encryption: How small can we go? In *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pages 277–298, 2017.
22. Donghoon Chang, Nilanjan Datta, Avijit Dutta, Bart Mennink, Mridul Nandi, Somitra Sanadhya, and Ferdinand Sibleyras. Release of unverified plaintext: Tight unified model and application to ANYDAE. *IACR Trans. Symmetric Cryptol.*, 2019(4):119–146, 2019.
23. Donghoon Chang and Mridul Nandi. A short proof of the PRP/PRF switching lemma. *IACR Cryptol. ePrint Arch.*, 2008:78, 2008.
24. Nilanjan Datta, Atul Luykx, Bart Mennink, and Mridul Nandi. Understanding RUP integrity of COLM. *IACR Trans. Symmetric Cryptol.*, 2017(2):143–161, 2017.
25. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2. Submission to CAESAR, 2016. `https://competitions.cr.yp.to/round3/asconv12.pdf`.
26. Orr Dunkelman, Eran Lambooij, and Shibam Ghosh. Practical related-key forgery attacks on the full tinyjambu-192/256. Cryptology ePrint Archive, Paper 2022/1122, 2022.
27. Ewan Fleischmann, Christian Forler, and Stefan Lucks. Mcoe: A family of almost foolproof on-line authenticated encryption schemes. In Anne Canteaut, editor, *Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers*, volume 7549 of *Lecture Notes in Computer Science*, pages 196–215. Springer, 2012.

28. Viet Tung Hoang, Reza Reyhanitabar, Phillip Rogaway, and Damian Vizár. Online authenticated-encryption and its nonce-reuse misuse-resistance. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 493–517. Springer, 2015.
29. Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, and Thomas Peyrin. Duel of the titans: The romulus and remus families of lightweight AEAD algorithms. *IACR Trans. Symmetric Cryptol.*, 2020(1):43–120, 2020.
30. Tetsu Iwata, Kazuhiko Minematsu, Jian Guo, Sumio Morioka, and Eita Kobayashi. CAESAR Candidate CLOC. DIAC 2014.
31. Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. The deoxys AEAD family. *J. Cryptol.*, 34(3):31, 2021.
32. Kerry A. McKay, Larry Bassham, Meltem Sönmez Turan, and Nicky Mouha. Report on Lightweight Cryptography, 2017. `http://nvlpubs.nist.gov/nistpubs/ir/2017/NIST.IR.8114.pdf`.
33. Kazuhiko Minematsu. AES-OTR v3.1. Submission to CAESAR, 2016. `https://competitions.cr.yp.to/round3/aesotrv31.pdf`.
34. Miguel Montes and Daniel Penazzi. AES-CPFB v1. Submission to CAESAR. 2015. `https://competitions.cr.yp.to/round1/aescpfbv1.pdf`.
35. Yusuke Naito, Mitsuru Matsui, Yasuyuki Sakai, Daisuke Suzuki, Kazuo Sakiyama, and Takeshi Sugawara. SAEAES: Submission to NIST LwC, 2019. `https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/SAEAES-spec-round2.pdf`.
36. 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.
37. Jacques Patarin. The "coefficients h" technique. In *Selected Areas in Cryptography*, pages 328–345. 2008.
38. Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, Washington, DC, USA, November 18-22, 2002*, pages 98–107. ACM, 2002.
39. Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In *Advances in Cryptology - ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, pages 16–31, 2004.
40. Ferdinand Sibleyras, Yu Sasaki, Yosuke Todo, Akinori Hosoyamada, and Kan Yasuda. Birthday-bound slide attacks on tinyjambu's keyed-permutations for all key sizes. In Chen-Mou Cheng and Mitsuaki Akiyama, editors, *Advances in Information and Computer Security - 17th International Workshop on Security, IWSEC 2022, Tokyo, Japan, August 31 - September 2, 2022, Proceedings*, volume 13504 of *Lecture Notes in Computer Science*, pages 107–127. Springer, 2022.
41. Hongjun Wu. ACORN: A Lightweight Authenticated Cipher (v3). Submission to CAESAR, 2016. `https://competitions.cr.yp.to/round3/acornv3.pdf`.
42. Hongjun Wu and Tao Huang. The JAMBU Lightweight Authentication Encryption Mode (v2.1). Submission to CAESAR, 2016. `https://competitions.cr.yp.to/round3/jambuv21.pdf`.

43. Hongjun Wu and Tao Huang. TinyJAMBU: A Family of Lightweight Authenticated Encryption Algorithms: Submission to NIST LwC, 2019. https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/finalist-round/updated-spec-doc/tinyjambu-spec-final.pdf.
44. Liting Zhang, Wenling Wu, Han Sui, and Peng Wang. ifeed[aes] v1, 2014. https://competitions.cr.yp.to/round1/ifeedaesv1.pdf.
45. Liting Zhang, Wenling Wu, Han Sui, and Peng Wang. iFeed[AES] v1. Submission to CAESAR, 2014. https://competitions.cr.yp.to/round1/ifeedaesv1.pdf.
46. Ping Zhang, Peng Wang, and Honggang Hu. The INT-RUP Security of OCB with Intermediate (Parity) Checksum. *IACR Cryptology ePrint Archive*, 2017. https://eprint.iacr.org/2016/1059.pdf.

Supplementary Materials

A Security Notions

A.1 Block Cipher

Let $n, \kappa \in \mathbb{N}$. A block cipher $\mathsf{E} : \{0,1\}^{\kappa} \times \{0,1\}^{n} \to \{0,1\}^{n}$ is a function that takes as input a key $K \in \{0,1\}^{k}$ and a message $M \in \{0,1\}^{n}$ and transforms it into a ciphertext $C \in \{0,1\}^{n}$. We write $\mathsf{E}_K(M) := \mathsf{E}(K, M)$. The transformation is bijective, i.e., for each $K \in \{0,1\}^{k}$, the function E_K is invertible; however, we will not be concerned with inverse evaluations of E. We fix positive even integers n and k to denote the *block size* and the *key size* of the block cipher, respectively, in terms of the number of bits.

A.2 Pseudorandom Function Notion

Let $\mathsf{Funcs}(\{0,1\}^{n})$ be the set of all functions from $\{0,1\}^{n}$ to $\{0,1\}^{n}$ and $\mathsf{F} : \{0,1\}^{k} \times \{0,1\}^{n} \to \{0,1\}^{n}$ be a family of keyed functions from $\{0,1\}^{n}$ to $\{0,1\}^{n}$. We define the pseudorandom function (prf) advantage of F with respect to a distinguisher $\mathscr{A}$ as follows:

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(\mathscr{A}) \triangleq \left|\mathsf{Pr}[K \leftarrow_{\$} \{0,1\}^{k} : \mathscr{A}^{\mathsf{F}_K} = 1] - \mathsf{Pr}[R \leftarrow_{\$} \mathsf{Funcs}(\{0,1\}^{n}) : \mathscr{A}^{R} = 1]\right|.$$

We say that F is $(q, \mathtt{t}, \epsilon)$ secure if the maximum pseudorandom function advantage of F is ϵ where the maximum is taken over all distinguishers $\mathscr{A}$ that makes q queries to its oracle and runs for a time at most $\mathtt{t}$ [5], i.e.,

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(q, \mathtt{t}) \triangleq \max_{\mathscr{A} \in \mathcal{C}} \mathbf{Adv}^{\mathrm{prf}}_{\mathsf{F}}(\mathscr{A}),$$

where $\mathcal{C}$ is the class of all distinguishers $\mathscr{A}$ that makes at most q queries with run time at most $\mathtt{t}$.

A.3 Pseudorandom Permutation Notion

Let $\mathsf{Perm}(n)$ be the set of all permutations over $\{0,1\}^{n}$. We capture the security notion of a block cipher E with key size k and block size n in terms of indistinguishabilty advantage from a uniform random permutation. More formally, we define the pseudorandom permutation (prp) advantage of E of a distinguisher $\mathscr{A}$ as follows:

$$\mathbf{Adv}^{\mathrm{prp}}_{\mathsf{E}}(\mathscr{A}) \triangleq \left|\mathsf{Pr}[K \leftarrow_{\$} \{0,1\}^{k} : \mathscr{A}^{\mathsf{E}_K} = 1] - \mathsf{Pr}[P \leftarrow_{\$} \mathsf{Perm}(n) : \mathscr{A}^{P} = 1]\right|.$$

We say that E is $(q, \mathtt{t}, \epsilon)$ secure if the maximum pseudorandom permutation advantage of E is ϵ where the maximum is taken over all distinguishers $\mathscr{A}$ that makes q queries to its oracle and runs for a time at most $\mathtt{t}$.

[5] Time complexity of the adversary is defined over the usual RAM model of computations.

B Release of Unverified Plaintext (RUP) and Its Implication

The functionality of an AE scheme renders two security models [38]- (i) the first one is the confidentiality model, in which an adversary is given access to the authenticated encryption functionality $\mathcal{E}$ or a random function \$, and it has to distinguish one from the another, (ii) and the second one is the authenticity model in which an adversary is given access to the authenticated encryption functionality $\mathcal{E}$, and the decryption/verification functionality $\mathcal{DV}$, and the adversary tries to provide a valid ciphertext approved by $\mathcal{DV}$. Therefore, an authenticated encryption scheme suits a two-fold goal. Thus, upon decryption, 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 the plaintext and not over the ciphertext, enforce that the decrypted plaintext should be kept in secure memory before the verification of the plaintext is done. After successful verification, the plaintext can be released to the user. However, the necessity of this additional secure memory can be a severe issue in constrained devices in the Internet of Things (IoT). As a result, an attacker can get hold of the insecure memory of the IoT devices to get access of the unverified plaintext, which can leak significant informations about the cipher to break its security. Moreover, real-time streaming protocols (e.g., SRTP, SSH, and SRTCP) and Optical Transport Networks sometimes need to release *plaintexts in segments with intermediate tags* on the fly to reduce end-to-end latency and storage. Owing to these real-time applications, we require a security notion which should ensure that when a cryptographic scheme Π satifies the security notion, releasing the unverified plaintext will not lead to any threat to the system. As a result of that, one does not need to store the entire plaintext in a secure memory until the verification takes place. *Releasing Unverified Plaintext* (RUP) security fulfils the above demand.

Note that, RUP security can also allow one to have the on-the-fly decryption feature of a cryptographic scheme. That means, if a cryptographic scheme is RUP secure, then its decryption module can decrypt the incoming ciphertext and release the corresponding plaintext in block-by-block fashion without thinking of the security issue that may arise due to the releasing of the unverified plaintext blocks, thanks to the RUP security! On the other hand, on-the-fly encryption requires OAE-1 [27] or OAE-2 [28] security. Owing to these applications, studying the security of cryptographic schemes under the release of unverified plaintext and designing ciphers that preserve the security is gaining importance in the literature.

C SAEB and Its INT-RUP Security

C.1 Description and Features of SAEB

The pictorial representation of the encryption and the decryption algorithm of SAEB is given in Fig. 3 and Fig. 4 respectively.

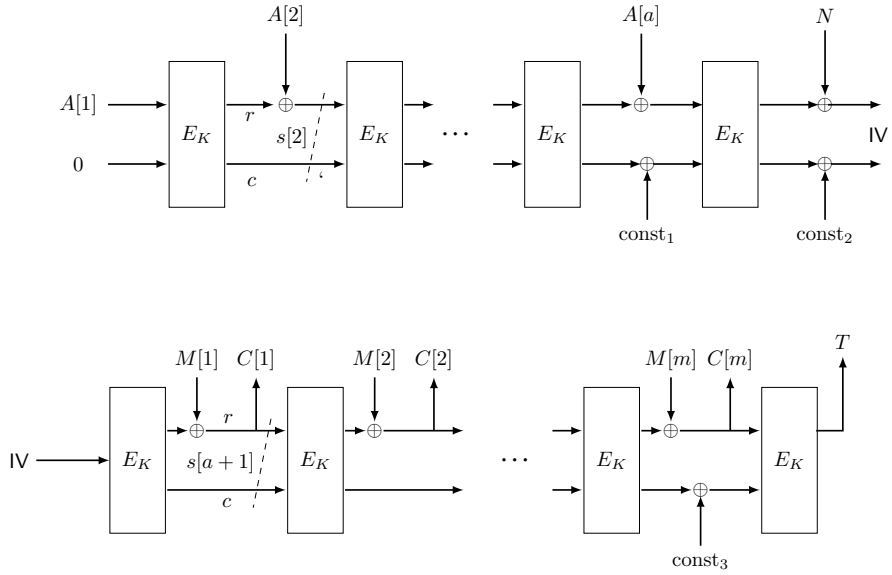

Fig. 3: Encryption algorithm of SAEB Authenticated Encryption for a block associated data, and m block message. const_1, const_2, const_3 are two-bit distinct constants used for domain separation. $\mathsf{const}_1 = 01/10$ based on whether $A[a]$ is a full or partial block. $\mathsf{const}_2 = 11$ and $\mathsf{const}_3 = 01/10$ based on whether $M[m]$ is a full or partial block.

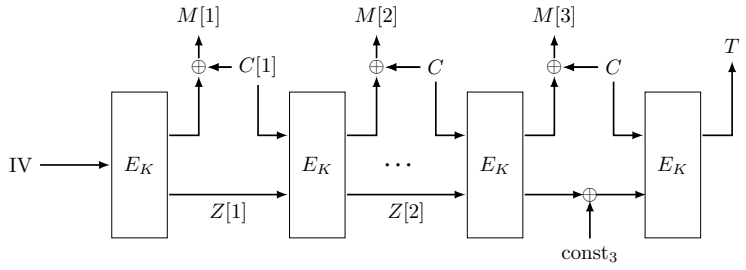

Fig. 4: Decryption of SAEB Authenticated Encryption.

The following features of SAEB make it highly efficient to be implemented in resource-constrained and ultra light-weight devices.

(i) **Minimum State Size:** SAEB is an authenticated cipher that requires the minimum possible state size of $n + k$-bits, where n is the block size of the block cipher and k is the key size of the block cipher.
(ii) **Inverse Free:** Due to the inherent duplex-sponge type structure, SAEB is an inverse-free design (exception is APE [5], which is a sponge type design but not inverse-free).
(iii) **On-the-fly Computation:** It is easy to see that the encryption algorithm of SAEB allows one to produce the i-th ciphertext block before the subsequent plaintext blocks are known, which makes the construction to encrypt messages on-the-fly. Similarly, one can also decrypt the ciphertext on-the-fly.
(iv) **XOR Only Linear Operation:** It says that apart from using the non-linear operation induced by the underlying block cipher invocation in the design, it uses only xor as a linear operation.
(v) **Efficient Static AD Processing:** In many practical scenarios, the associated data packets transmitted in a session remain unchanged. In such cases, it is desirable that while processing the data packets through any AEAD scheme, the associated data gets processed only once and then it is stored such that the stored value can be used in the subsequent processing of the data packets whose associated data has not been changed. Note that the encryption algorithm of SAEB achieves this property. It does not process the associated data each time it processes the data packet. Instead, it processes the associated data once and then it stores the processed value so that it can be reused in the subsequent processing of the data packets.

C.2 Analysis of the INT-RUP Attack against SAEB when $r \geq c/2$

We analyze the attack in two steps. In the first step, we show that the collision in $Z[2]$ values occur with high probability, and then we show that the forging is successful with high probability.

Stage-I: As adversary $\mathscr{A}$ makes decryption queries with uniformly random $C_i[1]$ values, $Z_i[2]$ values are uniformly distributed. Let $Z_{j,k}$ be the indicator random variable defined as follows:

$$Z_{j,k} = \begin{cases} 1, \text{ if } Z_j[2] = Z_k[2] \\ 0, \text{ otherwise.} \end{cases}$$

Let $Z = \sum_{j,k} Z_{j,k}$, and we have to show that $Z \geq 1$ with high probability. It is easy to see that $Z_{j,k}$'s are all pairwise independent random variables, and

therefore, using Chebyshev's inequality, we have

$$\begin{aligned}
&\Pr[\|Z - \mathsf{E}(Z)\| \geq \mathsf{E}(Z)] \leq \frac{\mathsf{V}(Z)}{\mathsf{E}(Z)^2} \\
\Rightarrow &\Pr[Z = 0] \leq \Pr[\|Z - \mathsf{E}(Z)\| \geq \mathsf{E}(Z)] \leq \frac{\mathsf{V}(Z)}{\mathsf{E}(Z)^2} \\
\Rightarrow &\Pr[Z \geq 1] \geq 1 - \frac{\mathsf{V}(Z)}{\mathsf{E}(Z)^2}. \qquad (12)
\end{aligned}$$

By using the linearity of expectation, we compute the expectation as follows:

$$\mathsf{E}(Z) = \mathsf{E}\Big[\sum_{j,k} Z_{j,k}\Big] = \sum_{j,k} \Pr[Z_{j,k} = 1] = \frac{\binom{2^{c/2+1}}{2}}{2^c}. \qquad (13)$$

Again, from the pairwise independence of the variables $Z_{j,k}$, we can compute the variance as follows:

$$\sum_{j,k} \mathsf{V}[Z_{j,k}] = \sum_{j,k} \mathsf{E}[Z_{j,k}^2] - (\mathsf{E}[Z_{j,k}])^2 = \sum_{j,k} \frac{1}{2^c} - \Big(\frac{1}{2^c}\Big)^2 \leq \sum_{j,k} \frac{1}{2^c} = \frac{\binom{2^{c/2+1}}{2}}{2^c}. \qquad (14)$$

Putting the values of Eqn. (13) and Eqn. (14) in Eqn. (12), we have

$$\Pr[Z \geq 1] \geq \Big(1 - \frac{2^c}{\binom{2^{c/2+1}}{2}}\Big) \geq \frac{1}{2},$$

which proves that the probability of getting at least one collision in $Z[2]$ values for any two decryption queries is at least 0.5. Let the query indices be j and k, i.e., $Z_j[2] = Z_k[2]$. As the successive r-bit ciphertext strings are identical for all chosen tuples, such a collision results in identical r-bit plaintext strings, which will be detected by adversary $\mathscr{A}$. Therefore, for the j-th and the k-th query, $\mathscr{A}$ observes whether $M_j[a] = M_k[a]$ holds or not for all $a \in [3, \ell + 1]$. Then, $\mathscr{A}$ makes an encryption query $(N, A, M_j[1 \ldots \ell+1])$ to obtain the tagged-ciphertext $(C_j[1] \parallel C[2] \parallel C[3] \parallel \ldots \parallel C[\ell+1], T)$, where $C[2] = C[3] = \ldots = C[\ell+1] = C$. This query allows $\mathscr{A}$ to forge with $(N, A, C_k[1] \parallel C[2] \parallel C[3] \parallel \ldots \parallel C[\ell+1], T)$, where $C[2] = C[3] = \ldots = C[\ell + 1] = C$. Note that $Z_j[2] = Z_k[2]$ ensures the collision in $M_j[3]$ and $M_k[3]$, which is propagated to the subsequent blocks of the block cipher.

Stage-II: Now we show that the forging is successful with high probability. According to step (4) of the attack algorithm, the kth decryption query is used in the forging if adversary detects $M_j[a] = M_k[a]$ for all $a \in [3, \ell + 1]$. However, the collision in r-bit plaintext strings, i.e., $M_j[a] = M_k[a]$ for $a \in [3, \ell + 1]$ and $j, k \in [2^{c/2+1}]$, also occurs without being $Z_j[2] = Z_k[2]$ to be true. Such a collision is called random collision, which occurs with probability 2^{-r} for each of the r-bit plaintext strings. Now, if the random collision occurs between the r-bit plaintext strings of the j-th and the k-th query, i.e., if $M_j[a] = M_k[a]$ occurs for

all $a \in [3, \ell+1]$ without being $Z_j[2] = Z_k[3]$ to be true, verification may not be successful. These are called *false-positive* cases. The probability with which one *false-positive* case will be detected is at most $\frac{1}{2^{r(\ell-1)}}$. Let $Y_{j,k}$ be the indicator random variable defined as follows:

$$Y_{j,k} = \begin{cases} 1, \text{ if } M_j[3..(\ell+1)] = M_k[3..(\ell+1)] \\ 0, \text{ otherwise} \end{cases}$$

and let $Y = \sum_{j,k} Y_{j,k}$ be the total number of false-positive cases. From the definition of Y and Z, we calculate the expected number of false-positive cases as follows:

$$\mathsf{E}(Y|Z=0) = \sum_{j,k} \Pr[Y_{j,k} = 1 | Z = 0] = \frac{\binom{2^{c/2+1}}{2}}{2^{r(\ell-1)}} \leq \frac{2^{c+1}}{2^{r(\ell-1)}} = 2,$$

where the last equality follows from the fact that $\ell = c/r+1$. Thus, the expected number of false-positive cases is at most 2.

C.3 INT-RUP Attack against SAEB, when $r < c/2$

Here we formally present the attack algorithm as follows:

1. Find minimum value of s such that $s \geq c/2r$.
2. $\mathscr{A}$ chooses an arbitrary r bit nonce N, an arbitrary r bit associated data A and an arbitrary r bit ciphertext data C, and then makes $2^{c/2}$ decryption queries in the form $(N,\ A,\ C_i[1] \parallel \ldots \parallel C_i[s] \parallel C[s+1] \parallel C[s+2] \parallel \ldots \parallel C[(s+\ell+1)])_{i=1,\ldots,2^{c/2}}$. Here the values of $C_i[1] \parallel \ldots \parallel C_i[s]$ are distinct, and $C[s+1] = C[s+2] = \ldots = C[(s+\ell+1)] = C$, where $\ell = c/r$. Let the corresponding plaintext be $M_i[1] \parallel \ldots \parallel M_i[(s+2)] \parallel \ldots \parallel M_i[(s+\ell+1)]$.
3. Assume there exists two indices $j, k \in [2^{c/2}]$ such that $M_j[a] = M_k[a]$ for all $a \in [(s+2), (s+\ell+1)]$.
4. $\mathscr{A}$ makes an encryption query with $(N,\ A,\ M_j[1] \parallel \ldots \parallel M_j[s] \parallel M_j[s+1] \parallel M_j[s+2] \parallel \ldots \parallel M_i[(s+\ell+1)])$. Let the tagged ciphertext be $(C_k[1] \parallel \ldots \parallel C_k[s] \parallel C[s+1] \parallel C[s+2] \parallel \ldots \parallel C[(s+\ell+1)],\ T)$, where $C[s+1] = C[s+2] = \ldots = C[(s+\ell+1)]$.
5. $\mathscr{A}$ forges with $(C_k[1] \parallel \ldots \parallel C_k[s] \parallel C[s+1] \parallel C[s+2] \parallel \ldots \parallel C[(s+\ell+1)],\ T)$, where $C[s+1] = C[s+2] = \ldots = C[(s+\ell+1)] = C$.

The analysis of the attack is the same as in the case of $r \geq c/2$. Using a similar analysis, one can prove that the expected number of false-positive cases is two. Hence, the number of verification attempts to be made is at most 2.

D TinyJAMBU and Its INT-RUP Security

D.1 Description and Properties of TinyJAMBU

The pictorial description of TinyJAMBU is presented in Figure 5. Similar to SAEB, TinyJAMBU also has the advantage of computing the data on-the-fly,

inverse-free, and maintains only an $n + k$-bit state with xor only linear operation. Additionally, TinyJAMBU achieves integrity even under the nonce-misuse scenario. Hence, this AEAD mode is considered a good choice in the *depth-in-defense* category. This additional benefit is achieved due to the property of message injection, and ciphertext release occurs from two different places of the construction. However, this property comes at the cost of the decreasing rate of the construction.

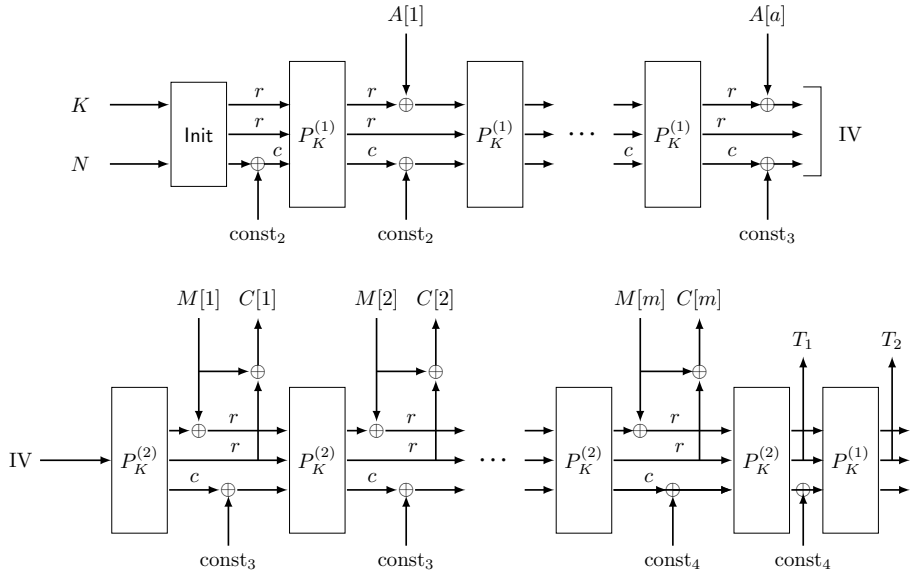

Fig. 5: TinyJAMBU Authenticated Encryption for a block associated data, and m block message. $\mathsf{const}_2 = 011$, $\mathsf{const}_3 = 101$, $\mathsf{const}_3 = 111$ are small distinct constants used for domain separation. 32-bits of the message and associated data are injected to the construction.

D.2 Internal State Values for the i-th Encryption Query in the Real World

The sequence of internal state values in the real world for i-th encryption query of length $a_i + m_i$ are defined according to the construction. as follows:

$$\begin{cases} s_i^+[0] = R(0) \\ s_i^+[1] = R(s_i^+[0] \oplus 0 \| \langle \mathsf{const}_1 \rangle_2 \| 0) \oplus N_i^+[1] \\ s_i^+[2] = R(s_i^+[1] \oplus 0 \| \langle \mathsf{const}_1 \rangle_2 \| 0) \oplus N_i^+[2] \\ s_i^+[3] = R(s_i^+[2] \oplus 0 \| \langle \mathsf{const}_1 \rangle_2 \| 0) \oplus N_i^+[3] \\ s_i^+[k+3] = R(s_i^+[k-1] \oplus 0 \| \langle \mathsf{const}_2 \rangle_2 \| 0) \oplus A_i^+[k], \text{ for } 1 \le k < a_i^+ \\ s_i^+[a_i+3] = R(s_i^+[a_i^+ - 1] \oplus 0 \| \langle \mathsf{const}_2 \rangle_2 \| 0) \oplus lp(A_i^+[a_i]) \\ s_i^+[a_i+3+k] = R(s_i^+[k-1] \oplus 0 \| \langle \mathsf{const}_3 \rangle_2 \| 0) \oplus M_i^+[k], \text{ for } 1 \le k < m_i \\ s_i^+[a_i+3+m_i] = R(s_i^+[a_i^+ + m_i^+] \oplus 0 \| \langle \mathsf{const}_3 \rangle_2 \| 0) \oplus lp(M_i^+[m_i]) \oplus 0 \| \langle \mathsf{const}_4 \rangle_2 \| 0 \end{cases} \tag{15}$$

D.3 Bounding the Probabilities of Bad Transcripts for TinyJAMBU

Here we bound the probabilities of each bad events case by case.

Bounding Coll: For this event to happen, we know that there exists at least one pair of indices $(i', j') < (i, j)$ such that $\mathsf{LLCP}(i) < j \le a_i + m_i + 3$ and $s_i^\star[j] = s_{i'}^{\circledast}[j']$. For any value of $j \in [1, a_i + m_i + 3]$, we have,

$$s_i^\star[j] = s_{i'}^\star[j] \Leftrightarrow R(s_i^\star[j-1]) \oplus R(s_{i'}^\star[j-1]) = x_i^\star[j] \oplus x_{i'}^{\circledast}[j] \tag{16}$$

where $x_i^\star[j]$ and $x_{i'}^{\circledast}[j]$ are two n-bit strings. Note that the first r-bits of $x_i^\star[j]$ is xored with nonce, associated data or message for the i-th query. Similarly, the first r-bits of $x_{i'}^{\circledast}[j]$ is xored with j-th nonce, associated data or message for the i'-th query.

First, let us consider the case when $j = j' = \mathsf{LLCP}(i) + 1$ and $j > 1$. Then there exists some i' such that $i' < i$ and $\mathsf{LLCP}(i)$ holds for i'. Now we consider the values of $s_i^\star[j-1]$ and $s_{i'}^{\circledast}[j'-1]$. Based on the values of j, we get following cases.

1. Case $j = 1$. In this case, we have $s_i^\star[0] = s_{i'}^{\circledast}[0] = R(0)$.
2. Case $j = 2$. In this case, a part of the nonce in the i-th and i'-th query matches, i.e., we have $N_i^\star[1] = N_i^{\circledast}[1]$. Then from the Equation (15), we have $s_i^\star[1] = s_{i'}^{\circledast}[1]$.
3. Case $3 \le j \le 4$. In this case, the nonce or part of the nonce in the i-th and i'-th query matches. As we show in the above case, we can similarly show that $s_i^\star[j-1] = s_{i'}^{\circledast}[j'-1]$.
4. Case $5 \le j \le a_i^\star + 4$. In this case, the nonce in the i-th and i'-th query matches. Also, the associated date in i-th and i'-th query matches up to $(j-4)$-th block. Thus from the Equation (15), we have $s_i^\star[j-1] = s_{i'}^{\circledast}[j'-1]$.

5. Case $a_i^\star + 5 \leq j \leq m_i^\star + 3$. In this case, the nonce and associated data in the i-th and i'-th query matches. Also, the message/ciphertext in i-th and i'-th query matches up to $(j - a_i^\star - 4)$-th block. Thus from the Equation (15), we have $s_i^\star[j-1] = s_{i'}^{\circledast}[j'-1]$.

Therefore, for any value of $j \in [1, a_i + m_i + 4]$, $s_i^\star[j-1] = s_{i'}^{\circledast}[j'-1]$. Thus, the probability of the event,

$$s_i^\star[j] = s_{i'}^\star[j] \Leftrightarrow R(s_i^\star[j-1]) \oplus R(s_{i'}^\star[j-1]) = x_i^\star[j] \oplus x_{i'}^{\circledast}[j] \Leftrightarrow x_i^\star[j] \oplus x_{i'}^{\circledast}[j] = 0$$

is zero.
On the other hand, for all $i' \leq i$ and $j' \neq j$ or $j \neq \mathsf{LLCP}(i) + 1$, $R(s_i^\star[j-1]) \oplus R(s_{i'}^\star[j-1]) = x_i^\star[j] \oplus x_{i'}^{\circledast}[j]$ holds with probability at most 2^{-n}. Thus, the probability that two states collide is 2^{-n}. Note that there are σ possible values of (i, j) in a transcript, each having no more than σ possible values of (i', j'), where σ is the total number of permutation calls including all encryption, decryption and verification queries, such that $s_i^\star[j] = s_{i'}^{\circledast}[j']$ holds. Therefore, we have

$$\mathsf{Pr}[\mathsf{Coll}] \leq \binom{\sigma}{2} \frac{1}{2^n} \leq \frac{\sigma^2}{2^{n+1}}. \tag{17}$$

Bounding mColl: We bound this event by conditioning the event that Coll does not occur. As there are no non-trivial collisions, $\mathsf{ancestor}(s_{i_1}^+[j_1])$, $\mathsf{ancestor}(s_{i_2}^+[j_2])$, ..., $\mathsf{ancestor}(s_{i_\rho}^+[j_\rho])$ are all distinct and fresh. Therefore, all the outputs $R(s_{i_1}^+[j_1])$, $R(s_{i_2}^+[j_2]), ..., R(s_{i_\rho}^+[j_\rho])$ are all uniformly sampled over $\{0,1\}^n$. Thus, from the randomness of R, we can view this event as throwing σ_e balls into 2^r bins (as we are seeking collisions in the rate part) uniformly at random, where σ_e denotes the total number of primitive calls with distinct inputs including all encryption queries and we want to find the probability that there is a bin that contains ρ or more balls. In other words, ρ or more outputs take some constant value c. This occurs with probability at most $(\frac{1}{2^r})^\rho$. Again, we have 2^r choices for the constant value c. Therefore, by varying over the choices of all encryption queries, we have

$$\mathsf{Pr}[\mathsf{mColl} \mid \overline{\mathsf{Coll}}] \leq \binom{\sigma_e}{\rho} \times 2^r (\frac{1}{2^r})^\rho \leq \frac{\sigma_e^\rho}{(2^r)^{\rho-1}}, \tag{18}$$

where the last inequality follows from Stirling's approximation and ignored the constant term in the Stirling's approximation.

Bounding Forge: Let us fix a verification query $(N_i^\times, A_i^\times, C_i^\times, T_i^\times)$ with associated data length $a_i^\times$ and ciphertext length $c_i^\times$ such that $\forall j \leq (a_i^\times + c_i^\times), \exists i', j'$ such that $s_i^\times[j] = s_{i'}^\star[j']$ where $\star \in \{+, -\}$. Let $\overline{j}$ be the largest index for which $s_i^\times[\overline{j}]$ does have a trivial collision with $s_{i'}^\star[j]$. We bound the probability of Forge when Coll and mColl do not occur. Now, we consider the following two cases based on the values of $\overline{j}$.

(a) Consider $a_i^{\times} + 3 \leq \bar{j} < a_i^{\times} + c_i^{\times} + 2$. In this case, the associated data, nonce and some parts of the message match with some previous queries. However, the adversary can control only the message injection part. So, $s_i^{\times}[\bar{j}+1]$ matches with some encryption or decryption query with probability at most $\frac{\rho+\sigma_d}{2^{n-r}}$.
(b) Finally, consider the case $\bar{j} = a_i^{\times} + c_i^{\times} + 2$. In this case, the final state matches with some previous encryption query with probability at most $\frac{\rho}{2^{n-r}}$.

Combining the cases and varying the choices of all the verification queries,

$$\mathsf{Pr}[\mathsf{Forge} \mid \overline{\mathsf{Coll}} \wedge \overline{\mathsf{mColl}}] \leq \frac{q_v(2\rho+\sigma_d)}{2^{n-r}}. \tag{19}$$

D.4 Bounding $\mathbf{Pr}[\mathbf{\Lambda}_1^{\times} \neq \boldsymbol{\omega}_{\mathtt{new}}^{\times} \mid \mathbf{\Lambda}_1^{+} = \boldsymbol{\omega}_{\mathtt{new}}^{+}, \mathbf{\Lambda}_1^{-} = \boldsymbol{\omega}_{\mathtt{new}}^{-}]$ in the Good Analysis

For $i \in [q_v]$, let $\mathbf{E}_i$ denote the following event

$$\mathbf{E}_i \triangleq \left(\bar{\lambda}^i \neq \perp \;\middle|\; \Lambda_1^{+} = \omega_{\mathtt{new}}^{+}, \Lambda_1^{-} = \omega_{\mathtt{new}}^{-}\right),$$

where $\bar{\lambda}^i$ be the random variable that denotes the response to the i-th verification query in the real world. Then, we have

$$\Pr[\Lambda_1^{\times} \neq \omega_{\mathtt{new}}^{\times} \mid \Lambda_1^{+} = \omega_{\mathtt{new}}^{+}, \Lambda_1^{-} = \omega_{\mathtt{new}}^{-}] \leq \sum_{i \in [q_v]} \Pr[\mathbf{E}_i].$$

We fix $i \in [q_v]$ and let the i-th verification query be $(N_i^{\times}, A_i^{\times}, C_i^{\times}, T_i^{\times})$. Assume that the associated data length is $a_i^{\times}$, and the ciphertext length is $c_i^{\times}$. We want to bound $\Pr[\mathbf{E}_i]$. This probability is non-zero only if $\Theta.\mathsf{ver}_{R_1,R_2}$ returns anything other than $\perp$. If $s_i^{\times}[a_i^{\times}+c_i^{\times}]$ does not match with the final state of any encryption query, we have $\Pr[\mathbf{E}_i] \leq \frac{1}{2^{\tau}}$. Suppose there is some encryption query such that the final state matches with $s_i^{\times}[a_i^{\times} + c_i^{\times}]$. In this case, there must be some j such that $s_i^{\times}[j]$ is fresh; otherwise Forge is true. Let j^* be the maximum of such j. Then, $s_i^{\times}[j^*+1]$ matches with some previous encryption or decryption states with probability at most $\frac{(\sigma_e+\sigma_d)}{2^{n-r}}$. Note that the adversary can control only the first r-bits of the state. Putting everything together we get:

$$\Pr[\mathbf{E}_i] \leq \frac{1}{2^{\tau}} + \frac{(\sigma_e+\sigma_d)}{2^{n-r}}.$$

Therefore, we have

$$\Pr[\Lambda_1^{\times} \neq \omega_{\mathtt{new}}^{\times} \mid \Lambda_1^{+} = \omega_{\mathtt{new}}^{+}, \Lambda_1^{-} = \omega_{\mathtt{new}}^{-}] \leq \sum_{i \in [q_v]} \frac{1}{2^{\tau}} + \frac{(\sigma_e+\sigma_d)}{2^{n-r}}.$$