π-Cipher with Intermediate Tags

Hristina Mihajloska[1], Bart Mennink[2], and Danilo Gligoroski[3]

[1]Faculty of Computer Science and Engineering, University "Ss. Cyril and Methodius", Skopje, Macedonia
[2]Dept. Electrical Engineering, ESAT/COSIC, KU Leuven, and iMinds, Belgium
[3]Dept. of Telematics, NTNU, Norwegian University of Science and Technology, Trondheim, Norway

May 13, 2016

Abstract

We introduce a new mode of operation for π-Cipher that supports intermediate tags. π-Cipher is one of the ciphers that participate in the second round of Competition for Authenticated Encryption: Security, Applicability, and Robustness - CAESAR. This mode allows π-Cipher to do a tag verification for a long message even on devices with limited memory capacities without releasing unverified plaintext. The mode is parallel, supports online encryption and decryption and gives an intermediate level of nonce-misuse resistance as an extra robustness feature of the cipher. This robustness is achieved by using the secret message number (SMN) as a part of the nonce. Also in this paper we provide a complete security proof for the privacy and integrity of the newly introduced mode of operation with intermediate tags.

Keywords: cryptography, on-line algorithms, authenticated encryption with associated data AEAD, intermediate tags, secret message number SMN.

1 Introduction

An authenticated encryption is a secret key technique that combines encryption and message authentication for simultaneously achieving both privacy and authenticity of the data. Additionally, by essentially providing them together in one mode of operation, where usually the same key and crypto primitive are used, it promotes efficiency and compactness. In [2], Bellare and Namprempre proved that the generic way to construct a secure authenticated encryption scheme is to first encrypt the data and then to compute the MAC, known as the "Encrypt-then-MAC" (EtM) paradigm. This paradigm is present in the most

known modes of operation for authenticated encryption, CCM [9], GCM [18], OCB [21], and EAX [3], where encryption and authentication are combined in single-pass or two-pass, fully parallelizable or sequential, with or without decryption function, online or offline modes of operation.

Nowadays, in the latest and still ongoing competition for authenticated encryption CAESAR [4], many questions about the onlineness and misuse resistance have revived. Most of the candidates in the second round of the competition are online, which means that encryption and authentication can be done on the fly. Instead of the encrypt-then-authenticate process from the sender side, from the receiver point of view, this process can be translated as verify-then-decrypt. So the way to transform verification and decryption to be done on the fly is getting more complex. It can be achieved in two different ways. The first one is done in two phases. In the first phase, only the tag is verified, and the second phase releases the decrypted blocks online. This approach is preferred for the memory constrained devices, where the memory of the device is not enough to collect all of the decrypted blocks before the tag is verified. The other way for secure online verification and decryption is by the use of intermediate tags. In this case, verification and decryption of the blocks is done on the fly without any special requirements of device memory or additional invocations.

In [22], Rogaway and Shrimpton defined a stronger authenticated encryption (AE) notion, which they called misuse-resistant AE (MRAE). They claimed that the price to create such a scheme is that the scheme *can't be online*. In [14], Hoang et al. give new definitions for online authenticated encryption and call them (corrected) OAE1 and OAE2. With these definitions in mind, none of the CAESAR candidates which are online, can be at the same time resist on the repetition of the nonce. This is due to the fact that the longest common prefix leakage cannot be avoided.

1.1 Related Work

No pre-CAESAR authenticated encryption schemes that could handle long messages without keeping the whole plaintext in memory have appeared, with the single exception of the duplex construction by Bertoni et al. [5], which turns an AE scheme into a single pass online authenticated encryption scheme. They proposed a mode with intermediate tags where a sequence of (header, message) pairs can be processed. There are many candidates in the CAESAR competition that are following the duplex construction such as ASCON [7], ICEPOLE [20], NORX [15], Keyak [13], STRIBOB [17], PRIMATEs [11] and π-Cipher [8]. Most of them are sequential and without option for online decryption.

There is another group of candidates, block cipher based, that are advertised as single pass, online, misuse resistant schemes, like COPA [10], ELmD [19] and POET [12]. All of them are characterized with parallelizability and two calls to the underlying primitive per block during the encryption process (in the case of POET it is one call of a block cipher and two calls to a hash function). Moreover, all of these schemes require the inverse primitive calls for decryption. Having in mind their specifications, POET and ELmD can also support intermediate

tags.

1.2 Our Contribution

In this paper, we introduce a new mode of operation for π-Cipher that supports intermediate tags. Without releasing unverified plaintext this technique allows us to do a tag verification for a long message even on devices with limited memory settings. This new mode is parallel, supports online encryption and decryption and gives an intermediate level of nonce-misuse resistance which manifests an extra feature of the robustness of the cipher. This robustness is achieved by using the secret message number (SMN) as a part of the nonce. Instead of the other online and parallel ciphers that support intermediate tags (ELmD and POET), our scheme does not make inverse primitive calls, so with just single implementation of the permutation function both encryption/authentication and decryption/verification are done.

1.3 Outline

We give a brief description of π-Cipher in Section 2. A formal definition of the new mode of π-Cipher with intermediate tags is presented in Section 3. Some preliminaries about the security model are given in Section 4. We provide a security proof of the privacy of the mode in Section 5 and in Section 6 we prove its integrity. The work is concluded in Section 7.

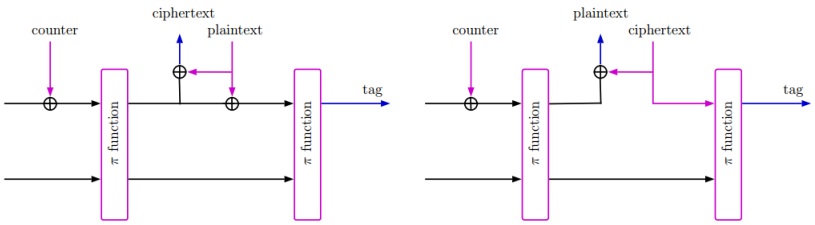
2 Brief Description of π-Cipher

We introduce π-Cipher at a level necessary to understand the proposal of the new mode of operation, and refer to [8] for the detailed and formal specification. π-Cipher is a permutation based scheme where the permutation function, denoted as π-function, plays the main role. As other sponge based permutation functions, this function has a b-bit internal state that consists of r-bit rate part (public one) and c-bit capacity part (secret one). Because of the nature of construction of the π-function, the rate and capacity parts are of the same size.

The encryption/authentication procedure of π-Cipher accepts key K of $klen$ bytes, associated data AD with $adlen$ bytes and message M with $mlen$ bytes. The cipher uses a public message number PMN and secret message number SMN. The output of the encryption/authentication procedure is a ciphertext C of $clen$ bytes and a tag T of τ bytes. The length of the ciphertext C is a sum of the byte length of the message, the authentication tag and the encrypted secret message number. The decryption/verification procedure accepts key K, associated data AD, ciphertext C, public message number PMN and tag T, and returns the decrypted pair (SMN, M) if the tag has been verified or $\perp$ otherwise.

The main building element in the operations of encryption/authentication and decryption/verification is our new construction related to the duplex sponge,

called triplex component. It uses the permutation function π twice, injects a counter into the internal state and digests an input string. The triplex component always outputs a tag. Optionally after the first call of the permutation function it can output a string (that can be a ciphertext block or a message block). Because of the differences in the encryption/authentication and decryption/verification procedures, there are two different variants of the triplex component. We call them *e-triplex* (for the phase of encryption) and *d-triplex* (for the phase of decryption). The only difference in these two components is how the input string is treated after the first call of the permutation function. They are shown in Figure 1.



(a) e-triplex (b) d-triplex

Figure 1: Triplex component

π-Cipher starts with the initialization phase, where by padding function, (K, PMN) is mapped to $(K||PMN||10^*)$ and processed by the π-function. As a result of this phase the counter and common internal state (CIS) become initialized.

Unlike other sponge-based schemes, π-Cipher offers parallelism both in the phase of processing the associated data and in the encryption phase. The number of parallel threads depends on the number of data blocks that every phase has (a threads in the phase of the associated data and m threads in the phase of the encryption). After the initialization phase, the state CIS is branched into a copies and associated data is processed. These a states are merged into one new state, that updates the CIS state and the newly obtained state CIS' is branched into m new copies where message blocks are encrypted and authenticated. After the computation of every block of ciphertext there is a production of a tag, called intermediate tag. At the end, all of these intermediate tags are summed into the final tag.

3 Mode of Operation of π-Cipher With Intermediate Tags

An authenticated encryption scheme with associated data (AEAD) provides encryption for a message M and authentication for M and associated data AD.

In the CAESAR competition [4], the traditional use of nonces has been modified to have two parts: a public message number (PMN) and a secret message number (SMN) i.e., the nonce N is represented as the pair $N = (PMN, SMN)$. The value of SMN is secret, and needs to be encrypted. In CAESAR, the use of an SMN was optional, and in fact, only two candidates have opted to facilitate it: π-Cipher [8] and ICEPOLE [20].

A precise formalization of the new variant of the authenticated encryption that uses the nonce as $N = (PMN, SMN)$ alongside the other arguments such as the key, the associated data and the message is given by Namprempre, Rogaway and Shrimpton in [6].

Definition 1 (AEAD with SMN)**.** *An authenticated encryption scheme with associated data (AEAD) and secret message number SMN is a tuple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ with $\mathcal{E}$ as a deterministic encryption algorithm that takes a key $K \in \mathcal{K} = \{0,1\}^k$, public message number $P \in \mathcal{P} = \{0,1\}^p$, associated data $A \in \mathcal{A} = \{0,1\}^*$, secret message number $S \in \mathcal{S} = \{0,1\}^s$, and message $M \in \mathcal{M} = \{0,1\}^*$, and returns a string $\mathcal{C} = \mathcal{E}_K^{P,A}(S, M)$. The decryption algorithm $\mathcal{D}$ takes strings K, P, A and $\mathcal{C} \subseteq \{0,1\}^*$ and returns either a string pair in $(\mathcal{S}, \mathcal{M})$ or the distinguished symbol $\perp$, $\mathcal{D}_K^{P,A}(\mathcal{C}) = (S, M)$ or $\mathcal{D}_K^{P,A}(\mathcal{C}) = \perp$. We require that $\mathcal{D}_K^{P,A}(\mathcal{E}_K^{P,A}(S, M)) = (S, M)$ for all $K \in \mathcal{K}$, $P \in \mathcal{P}$, $A \in \mathcal{A}$, $S \in \mathcal{S}$ and $M \in \mathcal{M}$. Thus, $|\mathcal{E}_K^{P,A}(S, M)| = l(|M|)$ for some linear-time computable length function l.*

Encryption and decryption functions can also be written as follows:

$$\mathcal{E} : \mathcal{K} \times \mathcal{P} \times \mathcal{A} \times \mathcal{S} \times \mathcal{M} \to \{0,1\}^*,$$

$$\mathcal{D} : \mathcal{K} \times \mathcal{P} \times \mathcal{A} \times \{0,1\}^* \to (\mathcal{S} \times \mathcal{M}) \cup \perp .$$

Hoang et al. [14] defined a segmented AE scheme with constant segment-expansion τ defined as follows:

Definition 2 (Segmented-AEAD scheme)**.** *A segmented-AEAD scheme is a tuple $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ where the key space $\mathcal{K}$ is a nonempty set with an associated distribution and both encryption $\mathcal{E} = (\mathcal{E}.init, \mathcal{E}.next, \mathcal{E}.last)$ and decryption $\mathcal{D} = (\mathcal{D}.init, \mathcal{D}.next, \mathcal{D}.last)$ specified by triples of deterministic algorithms. Associated to Π are its associated data space $\mathcal{A} \subseteq \{0,1\}^*$, its nonce space $\mathcal{N} \subseteq \{0,1\}^*$, its state space $\mathcal{S}$ and its segment-expansion τ. These parameters appear in the formal presentation of the components of $\mathcal{E}$ and $\mathcal{D}$:*

$$\begin{array}{ll}
\mathcal{E}.init : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \to \mathcal{S} & \mathcal{D}.init : \mathcal{K} \times \mathcal{N} \times \mathcal{A} \to \mathcal{S} \\
\mathcal{E}.next : \mathcal{K} \times \mathcal{S} \times \{0,1\}^{**} \to \{0,1\}^{**} \times \mathcal{S} & \mathcal{D}.next : \mathcal{K} \times \mathcal{S} \times \{0,1\}^{**} \to (\{0,1\}^{**} \times \mathcal{S}) \\
\mathcal{E}.last : \mathcal{K} \times \mathcal{S} \times \{0,1\}^* \to \{0,1\}^* & \mathcal{D}.last : \mathcal{K} \times \mathcal{S} \times \{0,1\}^* \to \{0,1\}^* \cup \perp,
\end{array}$$

where $\{0,1\}^{**} = (\{0,1\}^*)^*$ denotes the set of segmented-strings where each component of a segmented-string is a string.

The number of components in a segmented-string $\boldsymbol{M}$ is denoted as $|\boldsymbol{M}| = m$, while the i-th component of $\boldsymbol{M}$, $i \in [1..m]$, is denoted as $\boldsymbol{M}_i$. Note that indexing begins at 1. Thus, $\boldsymbol{M} = (M_1, M_2, \ldots M_m) \in \{0,1\}^{**}$. Here $\boldsymbol{M}$ gets transformed into a segmented ciphertext $\boldsymbol{C} = (C_1, C_2, \ldots, C_m) = \mathcal{E}(K, N, A, \boldsymbol{M})$ where

$|C_i| = |M_i| + \tau$ for all $i \in [1..m]$, and encryption algorithm is given with a triplet of algorithms as in the Definition 2.

π-Cipher $= (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is an AEAD scheme with SMN and it complies with the Definition 1. It relies on a permutation function $\pi : \{0,1\}^b \to \{0,1\}^b$ and correspondingly K-key, PMN-public message number, AD-associated data, SMN-secret message number, M-message and C-ciphertext. π-Cipher can work in different modes depending on the purpose of its use as it is described in its documentation [8]. Since one of the essential properties of π-Cipher is that it is online and parallel, it can be used in a mode of operation where the problem with secure release of unverified plaintext is addressed [1]. In order to precisely and mathematically define this mode of operation we naturally join the models of *AEAD with SMN* of Definition 1 and *Segmented-AEAD scheme* of Definition 2 into *Segmented-AEAD with SMN* where the authenticated segmentation is achieved by computing intermediate tags. With the model of intermediate tags, π-Cipher provides security against block-wise adversaries. In this model adversaries are allowed to send messages block-by-block to the cipher and receive the corresponding ciphertext blocks on-the-fly, authenticated.

The basic idea is for a message $M \in \{0,1\}^{mn}$, π-Cipher to generate a ciphertext $C \in \{0,1\}^{n+mn+m\tau+\tau}$, where $m \geqslant 0$ is the number of message/ciphertext blocks, $n \geqslant 0$ is the length of one block in bits and $\tau = k$ is the length of the tag in bits. In this case the ciphertext has an extension of $n + m\tau + \tau$-bits, where n bits correspond to the encrypted SMN, $m\tau$ represents the tag of every encrypted message block M_i and τ is reserved for the final tag. Processing the message part remains the same and also calculation of the ciphertext is the same. The intermediate tags t_j are generated and released after every block message. The final tag T is computed at the end of the entire message and it consists of the sum of the tag generated after the associated data phase T', the tag generated after SMN phase t_0 and all released intermediate tags of the message blocks t_j.

$$T = T' \boxplus t_0 \boxplus_{j=1}^{m} t_j.$$

The encryption and decryption functions can be described as follows:

$$\mathcal{E}(K, PMN, AD, SMN, M) \to (C, IT, T) \quad \text{and}$$
$$\mathcal{D}(K, PMN, AD, C, IT, T) \to (SMN, M) \cup \perp,$$

where $IT = (\underbrace{\lambda, \lambda, \ldots, \lambda}_{a+1}, t_1, t_2, \ldots, t_m)$ represents the sequence of intermediate tags. Here, we denote by λ the empty string, and in our model we do not release intermediate tags neither for the associated data nor for SMN. Thus, the total number of released intermediate tags is m (the same as the number of message blocks). The bit length of each intermediate tag is $|t_j| = k$ (as the length of the key).

The formal definition for this mode of π-Cipher can be formulated as following:

Definition 3 (Segmented-AEAD scheme with SMN)**.** *π-Cipher $= (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is a segmented AEAD scheme with constant segment-expansion and secret message number SMN, where the key space $\mathcal{K}$ is a nonempty set with an associated distribution and both encryption $\mathcal{E} = (\mathcal{E}.init, \mathcal{E}.smn, \mathcal{E}.msg)$ and decryption $\mathcal{D} = (\mathcal{D}.init, \mathcal{D}.smn, \mathcal{D}.msg)$ specified by triples of deterministic algorithms. Associated to π-Cipher are its AD space $\mathcal{A} \subseteq \{0,1\}^*$, its pubic message number PMN space $\mathcal{P} \subseteq \{0,1\}^*$, its secret message number SMN space $\mathcal{S}$, common internal state CIS space $\mathcal{IS}$ and its segment-expansion τ. These parameters appear in the formal presentation of the components of $\mathcal{E}$ and $\mathcal{D}$ as follows:*

$$\begin{aligned} \mathcal{E}.init &: \mathcal{K} \times \mathcal{P} \times \mathcal{A} \to \mathcal{IS} & \mathcal{D}.init &: \mathcal{K} \times \mathcal{P} \times \mathcal{A} \to \mathcal{IS} \\ \mathcal{E}.smn &: \mathcal{IS} \times \mathcal{S} \to \{0,1\}^* \times \mathcal{IS} & \mathcal{D}.smn &: \mathcal{IS} \times \mathcal{S} \to (\{0,1\}^* \times \mathcal{IS}) \\ \mathcal{E}.msg &: \mathcal{IS} \times \{0,1\}^{**} \to \{0,1\}^{**} & \mathcal{D}.msg &: \mathcal{IS} \times \{0,1\}^{**} \to \{0,1\}^{**} \cup \perp . \end{aligned}$$

The encryption algorithm (with authentication) operates on SMN and segmented plaintext $\boldsymbol{M} = (M_1, M_2, \ldots, M_m) \in \{0,1\}^{**}$ and produces the segmented ciphertext $\boldsymbol{C} = (C_0, C_1, \ldots, C_m) = \mathcal{E}(K, PMN, AD, SMN, \boldsymbol{M})$. There is a corresponding decryption algorithm $\mathcal{D}$ such as $\mathcal{D}(K, PMN, AD, \boldsymbol{C}) = (SMN, \boldsymbol{M})$ or $\perp$ if decryption algorithm together with the verification fail on some of the segments. The algorithms are defined in Figure 2.

Algorithm 1: $\mathcal{E}(K, PMN, AD, SMN, \boldsymbol{M})$

```
1: m ← |M|
2: if m = 0 then
3:    return λ
4: end if
5: (M_1, M_2, ..., M_m) ← M
6: CIS ← E.init(K, PMN, AD)
7: (C_0, CIS') ← E.smn(CIS, SMN)
8: for i ← 1 to m do
9:    C_i ← E.msg(CIS', M_i)
10: end for
11: return (C_0, C_1, ..., C_m)
```

Algorithm 2: $\mathcal{D}(K, PMN, AD, \boldsymbol{C})$

```
1: m ← |C| − 1
2: if m = 0 then
3:    return λ
4: end if
5: (C_0, C_1, ..., C_m) ← C
6: CIS ← D.init(K, PMN, AD)
7: (SMN, CIS') ← D.smn(CIS, C_0)
8: for i ← 1 to m do
9:    if D.msg(CIS', C_i) =⊥ then
10:      if m = 1 then
11:         return λ
12:      else
13:         return (M_1, ..., M_{i−1})
14:      end if
15:   else
16:      M_i ← D.msg(CIS', C_i)
17:   end if
18: end for
19: return (M_1, ..., M_m)
```

Figure 2: Two algorithms for encryption and decryption of segmented AEAD scheme with SMN - π-Cipher

4 Security Notions

The security proof of this mode of π-Cipher is based on the detailed proof for the sponge based authenticated ciphers and is given by Jovanovic, Luykx and Mennink in the ASIACRYPT 2014 paper [16]. In this paper, the authors have shown that several candidates of CAESAR may achieve the significantly higher bound than the traditional one of $min\{2^{c/2}, 2^k\}$. They have left out π-Cipher for some future analysis because of its structural difference in the way how it maintains the states.

In this paper, in order to prove the security of the mode of operation with intermediate tags of π-Cipher, we made a model with two games both being by choosing a random key K from the space $\mathcal{K}$. In the real game an oracle $\mathcal{O}$ asks a query to $\mathcal{E}$ as it is defined in Definition 3 and is answered with real values. On the other hand, in the ideal game an oracle asks a query to \$ which is an ideal permutation model of $\mathcal{E}$ and is answered with uniformly random bit strings. The main role in these games plays an adversary $\mathcal{A}$, which is a probabilistic algorithm that has access to some of the oracles $\mathcal{O}$. By $\mathcal{A}^{\mathcal{O}} \Rightarrow 1$ we denote the event that adversary $\mathcal{A}$ running with its oracle $\mathcal{O}$, outputs 1. Moreover, the adversary has unbounded computational power, and its complexity is measured by the number of queries made to the oracles.

For this security proof, we consider an adversary $\mathcal{A}$ that makes q_p permutation queries and q_ε encryption queries of total length λ_ε blocks. By $\mathcal{E}_K$ we denote an encryption query, consisting of a associated data blocks, and m message blocks. According to the design of π-Cipher we will define a state value s as an input to exactly one call to the permutation function, π-function. Note that s^{init} is a state value for the first call to the π-function in the initialization phase. After that there are a parallel threads in the associated data phase. We denote with $s^{AD}_{l,0}$ the first input to the permutation function, and with $s^{AD}_{l,1}$ the second call to each of the branches $l = \overline{1,a}$. s^{CIS} is a state value for updating the common internal state, that is done at the end of the AD phase. The next state values are two inputs to the π-function into the SMN phase, s^{SMN}_0 and s^{SMN}_1. At the end, in the message phase the state values are distributed in the same way as in the AD phase, so they are denoted as $s^M_{l,0}$ and $s^M_{l,1}$ for $l = \overline{1,m}$. Hence, the corresponding state values can be represented as:

$$\left(s^{init}; \quad \begin{bmatrix} s^{AD}_{1,0} & s^{AD}_{1,1} \\ \vdots & \vdots \\ s^{AD}_{a,0} & s^{AD}_{a,1} \end{bmatrix}; \quad s^{CIS}; \quad s^{SMN}_0; \quad s^{SMN}_1; \quad \begin{bmatrix} s^{M}_{1,0} & s^{M}_{1,1} \\ \vdots & \vdots \\ s^{M}_{m,0} & s^{M}_{m,1} \end{bmatrix} \right).$$

Here, if the j-th encryption query is of length $a+m$ blocks, then the number of state values $\sigma_{\epsilon,j}$ is $2a + 2m + 4$. So, the total number of π-function evaluations via the encryption queries is:

$$\sigma_\varepsilon := \sum_{j=1}^{q_\varepsilon} \sigma_{\varepsilon,j} \leqslant q_\varepsilon(2a + 2m + 4) = 2\lambda_\varepsilon + 4q_\varepsilon. \tag{1}$$

Also for decryption queries the number is the same and is denoted as $\sigma_{\mathcal{D}}$ and $\sigma_{\mathcal{D},j}$ analogously.

5 Privacy of π-Cipher With Intermediate Tags

Theorem 1. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the proposed π-Cipher with intermediate tags, where the permutation function π is replaced with an ideal permutation p which operates on b bits. Then,*

$$\boldsymbol{Adv}_{\Pi}^{priv}(\mathcal{A}) \leqslant \frac{2(q_p + \sigma_\varepsilon)^2}{2^b} + \frac{q_p + \sigma_\varepsilon}{2^k} + \frac{q_p r}{2^c} + \frac{q_\varepsilon a}{2^{b/2}} + \sqrt{\frac{8e\sigma_\varepsilon q_p}{2^b}},$$

where σ_ε is defined in (1), q_p is the maximum number of calls to ideal permutation $p^{\pm}$, q_ε is the maximum number of encryption queries of total length of λ_ε blocks and a is the maximum number of associated data blocks.

Proof. For the privacy proof we need to obtain an upper bound for the advantage of an adversary who can distinguish the output of the proposed scheme with a random oracle in the ideal permutation model.

$$\mathbf{Adv}_{\Pi}^{priv}(\mathcal{A}) = |\mathbf{Pr}(\mathcal{A}^{p^{\pm},\mathcal{E}_K} \Rightarrow 1) - \mathbf{Pr}(\mathcal{A}^{p^{\pm},\$} \Rightarrow 1)| = \Delta_{\mathcal{A}}(p^{\pm}, \mathcal{E}_K; p^{\pm}, \$). \quad (2)$$

With replacing the permutation function p with two-directional function f from $\{0,1\}^b$ to $\{0,1\}^b$ and using the PRP/PRF Switching Lemma [23], we have:

$$\begin{aligned}\Delta_{\mathcal{A}}(p^{\pm}, \mathcal{E}_K; p^{\pm}, \$) - \Delta_{\mathcal{A}}(f^{\pm}, \mathcal{E}_K; f^{\pm}, \$) &\leqslant \frac{(q_p + \sigma_\varepsilon)(q_p + \sigma_\varepsilon - 1)}{2^{b+1}} \\ &\leqslant \frac{(q_p + \sigma_\varepsilon)^2}{2^b}. \quad (3)\end{aligned}$$

Now we will restrict our attention to an adversary with oracle access to $(f^{\pm}, \{\mathcal{E}_K, \$\})$. The function $f^{\pm}$ maintains a list $\mathcal{F}$ of all query/response tuples (x, y). This list initially is empty. For the forward query $f(x)$, if $(x, y) \in \mathcal{F}$, the corresponding value $y = f(x)$ is returned. For a new forward query $f(x)$, and y randomly drown from $\{0,1\}^b$, if (x, y) is in the list $\mathcal{F}$ then the primitive aborts, otherwise the tuple (x, y) is added to it. The description for the query $f^{-1}(y)$ is similar.

Also, for this proof we assume that the adversary is nonce respecting and only queries full blocks where no padding rules are involved.

To aim the goal we have two main collision events, **guess** and **hit**. The event **guess** is a way of how a primitive call in an encryption query hits a direct primitive query, or the opposite. On the other side, the event **hit** is set when two independent states collide in the encryption query. For two states we say that they are independent if they come from different previous state values. So event is set to bad, if some of the events **guess** or **hit** occur:

$$\Delta_{\mathcal{A}}(f^{\pm}, \mathcal{E}_K; f^{\pm}, \$) \leqslant \mathbf{Pr}(\mathcal{A}^{f^{\pm},\mathcal{E}_K} \text{ sets event}). \quad (4)$$

The probability of *event is set* according to [16] is as follows:

$$\mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit}) = \mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit} | \neg(\mathbf{key} \vee \mathbf{multi})) + \mathbf{Pr}(\mathbf{key} \vee \mathbf{multi}). \quad (5)$$

Note that there are two more additional events, **key** and **multi**. The event **key** corresponds to all primitive queries hitting the key. The event **multi** is used to bound the number of states that collide in the rate part. Let $\rho \geqslant 1$ be any threshold, for which the following is satisfied:

$$max_{\alpha \in \{0,1\}^r} |\{j' \leqslant j, 1 < k' \leqslant k : \alpha\{[s_{j',k'}]^r, [f(s_{j',k'})]^r\}\}| > \rho,$$

for $j \in \{1, \ldots, q_\varepsilon\}$ and $k \in \{1, \ldots, \sigma_{\varepsilon,j}\}$.

Event guess. This event may occur in the i-th primitive query (for $i = 1, \cdots, q_p$) or in any state evaluation of the j-th encryption query (for $j = 1, \cdots, q_\epsilon$). Denote that the state values of the j-th encryption query are as follows:

$$\left(s_j^{init}; \quad \begin{bmatrix} s_{j,1,0}^{AD} & s_{j,1,1}^{AD} \\ \vdots & \vdots \\ s_{j,a,0}^{AD} & s_{j,a,1}^{AD} \end{bmatrix}; \quad s_j^{CIS}; \quad s_{j,0}^{SMN}; \quad s_{j,1}^{SMN}; \quad \begin{bmatrix} s_{j,1,0}^{M} & s_{j,1,1}^{M} \\ \vdots & \vdots \\ s_{j,m,0}^{M} & s_{j,m,1}^{M} \end{bmatrix} \right). \quad (6)$$

We assume that event (**guess** $\vee$ **hit** $\vee$ **key** $\vee$ **multi**) has not been set before and also event (**key** $\vee$ **multi**) has not been set by this query before. From the further analysis we will exclude s_j^{init} from **guess** event because that case belongs to the **key** event. For $i = 1, \cdots, q_p$ let $j_i \in \{1, \cdots, q_\varepsilon\}$ be the number of encryption queries made before the i-th permutation query. And for $j = 1, \cdots, q_\varepsilon$ let $i_j \in \{1, \cdots, q_p\}$ be the number of permutation queries made before the j-th encryption query. Here we have two games, the first one is when we play with the permutation queries, and the other with encryption queries.

- In the first one the bad event occurs when input to the $f^{\pm}$ (x_i, y_i) collide with the elements in set $\mathcal{F}$. So for the forward query x_i there are at most ρ state values that have the same rate part, thus the capacity is unknown to the adversary and the bad event occurs with probability at most $\rho/2^c$. For the inverse query the situation is more complicated. We take y_i from the set of all encryption queries made before the i-th permutation query. Therefore the probability that guess is set via a direct query to the primitive is at most $\frac{q_p \rho}{2^c} + \sum_{i=1}^{q_p} \sum_{j=1}^{j_i} \frac{\sigma_{\epsilon,j}}{2^b}$.

- In the second one, bad event is set in the j-th encryption query for $j \in 1, \cdots, q_\varepsilon$. In this case we consider the probability that some of the states from (6) sets guess, assuming that it has not been set before. Thus we have 3 subcases here.

    1. The state value $s_{j,l,0}^{AD}$ for $l = \{1, \ldots, a\}$, equals $f(s_j^{init}) \oplus (ctr + l)$ where ctr is some secret value not determined by the adversarial input. By assumption that $f(s_j^{init})$ is randomly drawn from $\{0,1\}^b$, the

probability to guess the state is at most $\frac{ai_j}{2^b}$ (where i_j is the number of permutation queries made before the j-th encryption query). The state value $s^{AD}_{j,l,1}$ for $l = 1, \cdots, a$, equals $f(s^{AD}_{j,l,0}) \oplus AD_l$ where AD_l is a value determined by the adversarial input. The probability to guess the state is at most $\frac{ai_j}{2^b}$.

2. $s^{CIS}_j = f(s^{init}_j) \oplus T'$ and is guessed with probability at most $i_j/2^b$
3. The same is situation in $s^{SMN}_{j,0}$ and $s^{SMN}_{j,1}$, probability is bound to $i_j/2^b$.
4. For the last branching (processing the message) we have the same situation as in the associated data part.

Concluding, the j-th encryption query sets **guess** with probability at most $\frac{ai_j}{2^b} + \frac{ai_j}{2^b} + 3\frac{i_j}{2^b} + \frac{mi_j}{2^b} + \frac{mi_j}{2^b}$. Or summing over all q_ε encryption queries, we get

$$\sum_{j=1}^{q_\varepsilon} \frac{ai_j}{2^b} + \frac{ai_j}{2^b} + 3\frac{i_j}{2^b} + \frac{mi_j}{2^b} + \frac{mi_j}{2^b} = \sum_{j=1}^{q_\varepsilon} \frac{i_j(2a+2m+3)}{2^b} \leqslant \sum_{j=1}^{q_\varepsilon} \frac{i_j \sigma_{\varepsilon,j}}{2^b}.$$

Here we use that $\sum_{i=1}^{q_p} \sum_{j=1}^{j_i} \sigma_{\epsilon,j} \leqslant q_p \sigma_\epsilon$ as j_i always has its maximum value q_ϵ and $\sum_{j=1}^{q_\epsilon} i_j \sigma_{\epsilon,j} = \sum_{j=1}^{q_\epsilon} \sum_{k=1}^{\sigma_{\epsilon,j}} i_j \leqslant q_p \sigma_\epsilon$ as i_j always has its maximum value q_p.

Concluding,

$$\begin{aligned} \mathbf{Pr}(\mathbf{guess} | \neg(\mathbf{key} \vee \mathbf{multi})) &\leqslant \frac{q_p \rho}{2^c} + \sum_{i=1}^{q_p} \sum_{j=1}^{j_i} \frac{\sigma_{\varepsilon,j}}{2^b} + \sum_{j=1}^{q_\varepsilon} \frac{i_j \sigma_{\varepsilon,j}}{2^b} \\ &\leqslant \frac{q_p \rho}{2^c} + \frac{2 q_p \sigma_\varepsilon}{2^b}. \end{aligned} \tag{7}$$

Event hit. We need to find whether the event hit is set, or whether two independent states (with different parents) collide in the encryption queries. It is clear that for the initialization state s^{init} stands: $s^{init}_j \neq s^{init}_{j'}$ because of the uniqueness of the nonce. So here any state value $s_{j,l}$ hits an initial state value $s_{j',1}$ only if $[s_{j,l}]^k = K$ which happens with probability $\sigma_\varepsilon/2^k$. In the other states $(\sigma_\varepsilon - q_\epsilon)$ for any two states $s_{j,l}, s_{j',l'}$ we have the following:

- $s^{AD}_{j,l,0} = f(s^{init}_j) \oplus (ctr + l)$, $l = \{1, \ldots, a\}$. Because of the fact that s^{init}_j is always fresh, collision between $s^{AD}_{j,l,0}$ and some older state will happen with probability no more than $a/2^b$ for all l's. Note that $s^{AD}_{j,l,0}$ can never collide with a state from the same query because of the incremental counter's value. If $s^{AD}_{j,l,0}$ is a new state, then also it is a new input to f and $s^{AD}_{j,l,1}$ is a new one too. It hits a certain older state with probability $1/2^b$. If $s^{AD}_{j,l,1}$ is new for all l's, then the output of the function f is random, and the tag T' generated from the associated data phase is random too.

- $s_j^{CIS} = f(s_j^{init}) \oplus T'$ hits a state from an older query with probability at most $1/2^b$. Here we have a special case. The state s_j^{CIS} can collide with some of the states $s_{j,l,0}^{AD}$ from this query if and only if $[(ctr+l)||0^*]^r$ collide with T' (they have the same parent state s_j^{init})). Probability for this is no more than $a/2^r$.
- $s_{j,0}^{SMN} = f(s_j^{CIS}) \oplus (ctr + a + 1)$. This state is new, so it can hit some older state with probability $1/2^b$, and the same is for $s_{j,1}^{SMN}$.
- $s_{j,l,0}^{M} = f(s_{j,1}^{SMN}) \oplus (ctr + a + 1 + l)$, so it is a new state for all parallel instances l and can collide with some other state from an older query with probability at most $m/2^b$. However, if $s_{j,l,0}^{M}$ is a new state, also new states are $s_{j,l,1}^{M}$ for all l's and the output of the function f is random. This proofs that the intermediate tags t_j are random values, and also the generated final tag is a random value too.

In total, the j-th encryption query sets event **hit** with probability at most: $\frac{(2a+3+2m)}{2^b}(\sigma_{\varepsilon,1} + \sigma_{\varepsilon,2} + \ldots + \sigma_{\varepsilon,j-1}) + \frac{a}{2^r}$.

Summing over all queries,

$$\begin{aligned}\mathbf{Pr}(\mathbf{hit}|\neg(\mathbf{key} \vee \mathbf{multi})) &\leqslant \frac{\sigma_\varepsilon}{2^k} + \sum_{j=1}^{q_\varepsilon} \frac{(2a+3+2m)}{2^b}(\sigma_{\varepsilon,1} + \ldots + \sigma_{\varepsilon,j-1}) + \frac{a}{2^r} \\ &\leqslant \frac{\sigma_\varepsilon}{2^k} + \frac{q_\varepsilon a}{2^r} + \frac{\binom{\sigma_\varepsilon - q_\varepsilon}{2}}{2^b} \\ &\leqslant \frac{\sigma_\varepsilon}{2^k} + \frac{q_\varepsilon a}{2^r} + \frac{(\sigma_\varepsilon - q_\varepsilon)^2}{2^{b+1}}. \end{aligned} \tag{8}$$

Event key. This event is used to bound the collision between the key K and the k leftmost bits of x_i from the i-th primitive query where $i \in \{1, \ldots q_p\}$. An adversary makes at most q_p attempts, and hence $\mathbf{Pr}(\mathbf{key}) \leqslant q_p/2^k$.

Event multi. This event is used to bound the number of states that collide in the rate part. So, consider a new state value $s_{j,l-1}$; then for a fixed state value $x \in \{0,1\}^b$ it satisfies $f(s_{j,l-1}) = x$ or $s_{j,l} = f(s_{j,l-1}) \oplus w = x$ for some predetermined value w with probability $2/2^b$. Now, let $\alpha \in \{0,1\}^r$, more than ρ state values hit α with probability at most $\binom{\sigma_\epsilon}{\rho}(2/2^r)^\rho$. According to the Stirling's approximation for factorials ($n! \sim \sqrt{2\pi n}(\frac{n}{e})^n \geqslant (\frac{n}{e})^n$), we have $\binom{\sigma_\epsilon}{\rho}(2/2^r)^\rho \leqslant (\frac{2e\sigma_\epsilon}{\rho 2^r})^\rho$. Considering any possible choice of α we obtain that

$$\mathbf{Pr}(\mathbf{multi}) = 2^r \left(\frac{2e\sigma_\epsilon}{\rho 2^r}\right)^\rho. \tag{9}$$

So at the end, to complete the proof we need to make a sum of the previously

computed four bounds as follows:

$$\begin{aligned}\mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit}) &= \mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit}|\neg(\mathbf{key} \vee \mathbf{multi})) + \mathbf{Pr}(\mathbf{key} \vee \mathbf{multi}) \\ &= \mathbf{Pr}(\mathbf{guess}|\neg(\mathbf{key} \vee \mathbf{multi})) + \mathbf{Pr}(\mathbf{hit}|\neg(\mathbf{key} \vee \mathbf{multi})) \\ &+ \mathbf{Pr}(\mathbf{key}) + \mathbf{Pr}(\mathbf{multi}) \\ &\leqslant \frac{q_p + \sigma_\varepsilon}{2^k} + \frac{(\sigma_\varepsilon - q_\varepsilon)^2}{2^{b+1}} + \frac{q_p\rho}{2^c} + \frac{q_\varepsilon a}{2^r} + \frac{2q_p\sigma_\varepsilon}{2^b} + 2^r\left(\frac{2e\sigma_\epsilon}{\rho 2^r}\right)^\rho. \end{aligned} \tag{10}$$

To simplify previous evaluation and also substitute ρ with some known values, we put $\rho \geqslant max\{r, \sqrt{\frac{2e\sigma_\varepsilon 2^c}{q_p 2^r}}\}$, so simply we put $\rho = r + \sqrt{\frac{2e\sigma_\varepsilon 2^c}{q_p 2^r}}$ and get the following:

$$\mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit}) \leqslant \frac{q_p + \sigma_\varepsilon}{2^k} + \frac{(\sigma_\varepsilon - q_\varepsilon)^2}{2^{b+1}} + \frac{q_p r}{2^c} + \frac{q_\varepsilon a}{2^r} + \frac{2q_p\sigma_\varepsilon}{2^b} + \sqrt{\frac{8e\sigma_\varepsilon q_p}{2^b}}. \tag{11}$$

At the end, we have the final bound for privacy of π-Cipher:

$$\mathbf{Adv}^{priv}_{\Pi}(\mathcal{A}) \leqslant \frac{(q_p + \sigma_\varepsilon)^2}{2^b} + \frac{2q_p\sigma_\varepsilon}{2^b} + \frac{q_p + \sigma_\varepsilon}{2^k} + \frac{(\sigma_\varepsilon - q_\varepsilon)^2}{2^{b+1}} + \frac{q_p r}{2^c} + \frac{q_\varepsilon a}{2^r} + \sqrt{\frac{8e\sigma_\varepsilon q_p}{2^b}}. \tag{12}$$

We assume that $\frac{2q_p\sigma_\varepsilon + (\sigma_\varepsilon - q_\varepsilon)^2/2}{2^b} \leqslant \frac{(q_p+\sigma_\varepsilon)^2}{2^b}$. Also, $\frac{aq_\varepsilon}{2^r} = \frac{aq_\varepsilon}{2^{b/2}}$ according to the design of the permutation function and get the following bound for the privacy of π-Cipher with intermediate tags, that completes the proof:

$$\mathbf{Adv}^{priv}_{\Pi}(\mathcal{A}) \leqslant \frac{2(q_p + \sigma_\varepsilon)^2}{2^b} + \frac{q_p + \sigma_\varepsilon}{2^k} + \frac{q_p r}{2^c} + \frac{q_\varepsilon a}{2^{b/2}} + \sqrt{\frac{8e\sigma_\varepsilon q_p}{2^b}}.$$

■

6 Authenticity of π-Cipher With Intermediate Tags

Theorem 2. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be the proposed π-Cipher with intermediate tags, where the permutation function π is replaced with an ideal permutation p that operates on b bits. Then,*

$$\begin{aligned}\boldsymbol{Adv}^{auth}_{\Pi}(\mathcal{A}) \leqslant &\frac{(q_p + \sigma_\varepsilon + \sigma_\mathcal{D})^2}{2^{b+1}} + \frac{\sigma_\mathcal{D}(q_p + \sigma_\varepsilon + \sigma_\mathcal{D})}{2^c} + \frac{q_p r}{2^c} + \\ &\frac{a(q_\varepsilon + q_\mathcal{D})}{2^{b/2}} + \frac{q_p + \sigma_\varepsilon + \sigma_\mathcal{D}}{2^k} + \frac{m^2 q_\mathcal{D}^2 + m q_\mathcal{D}}{2^k} + \sqrt{\frac{8e\sigma_\varepsilon q_p}{2^b}},\end{aligned}$$

where σ_ε and $\sigma_\mathcal{D}$ are defined in (1), q_p is the maximum number of calls to ideal permutation $p^{\pm}$, q_ε is the maximum number of encryption queries of total length of λ_ε blocks, $q_\mathcal{D}$ is the maximum number of decryption queries of total length of $\lambda_\mathcal{D}$ blocks and a is the maximum number of associated data blocks.

Proof. A forgery of an AEAD scheme is defined as the ability of an adversary $\mathcal{A}$ to generate a valid (N, AD, C, T) tuple, without directly querying it to the encryption oracle. Stated differently, an adversary attempts to make a decryption query that does not result in $\perp$.

Let $\Pi = \{\mathcal{K}, \mathcal{E}, \mathcal{D}\}$ be π-Cipher defined with Definition 3 with ideal permutation (π-function) which operates on $b = (r + c)$ bits. The adversary $\mathcal{A}$ is given access to an encryption oracle $\mathcal{E}_K$, decryption oracle $\mathcal{D}_K$, ideal permutation function p and its inverse p^{-1} function. Our goal is to bound the adversary's advantage to forge the scheme $\Pi = \{\mathcal{K}, \mathcal{E}, \mathcal{D}\}$:

$$Adv_{\Pi}^{Auth}(\mathcal{A}) = \mathbf{Pr}(\mathcal{A}^{p^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ forges}). \tag{13}$$

By replacing the ideal permutation function p with two-directional function $f : \{0,1\}^b \rightarrow \{0,1\}^b$ and using the PRP/PRF Switching Lemma [23], also knowing that adversary can ask no more than $q_p + \sigma_\varepsilon + \sigma_\mathcal{D}$ evaluations to the function f, we have the following bound:

$$\begin{aligned}\mathbf{Pr}(\mathcal{A}^{p^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ forges}) - \mathbf{Pr}(\mathcal{A}^{f^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ forges}) &\leqslant \frac{(q_p + \sigma_\varepsilon + \sigma_\mathcal{D})(q_p + \sigma_\varepsilon + \sigma_\mathcal{D} - 1)}{2^{b+1}} \\ &\leqslant \frac{(q_p + \sigma_\varepsilon + \sigma_\mathcal{D})^2}{2^{b+1}}.\end{aligned}$$

We will focus on adversary $\mathcal{A}$ to have an oracle access to $(f^{\pm}, \mathcal{E}_K, \mathcal{D}_K)$ and only to make full-block queries. Also we assume that adversary $\mathcal{A}$ is nonce-respecting, which means that it never makes two queries to $\mathcal{E}_K$ with the same nonce, but it is allowed to repeat nonces in decryption queries.

For this proof we need the same setting as in the privacy proof, about the **guess** and **hit** events, but here extended with new $\mathcal{D}$-related collision events $\mathcal{D}_{\mathbf{guess}}$ and $\mathcal{D}_{\mathbf{hit}}$. The state values are the same as in (6) with a δ appended to the subscript, where $\delta \in \{\mathcal{E}, \mathcal{D}\}$.

We observe that:

$$\begin{aligned}\mathbf{Pr}(\mathcal{A}^{f^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ forges}) \leqslant \mathbf{Pr}(\mathcal{A}^{f^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ forges} | \neg\text{event}) + \\ \mathbf{Pr}(\mathcal{A}^{f^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ sets event}).\end{aligned} \tag{14}$$

A bound on the probability that $\mathcal{A}$ forges when event does not happen is the same with the case where $\mathcal{A}$ can guess some of the intermediate tags $t_{i,j}$ for some decryption query j. In this case, the j-th forgery attempt is successful with probability at most $m/2^\tau$. Summing over all decryption queries $q_\mathcal{D}$ we get:

$$\mathbf{Pr}(\mathcal{A}^{f^{\pm}, \mathcal{E}_K, \mathcal{D}_K} \text{ forges} | \neg\text{event}) \leqslant \frac{mq_\mathcal{D}}{2^\tau}.$$

The length of the intermediate tags is the same as the key length as we stated in Section 3, so freely we can write that probability of the adversary to forge the scheme when event does not happen is $\frac{mq_\mathcal{D}}{2^k}$.

Next we need to explain what is the bound on the probability when adversary sets event. Here, event = $\mathbf{guess} \vee \mathbf{hit} \vee \mathcal{D}_{\mathbf{guess}} \vee \mathcal{D}_{\mathbf{hit}}$ and

$$\mathbf{Pr}(\mathcal{A}^{f^{\pm},\mathcal{E}_K,\mathcal{D}_K} \text{ sets event}) \leqslant \mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit} \vee \mathcal{D}_{\mathbf{guess}} \vee \mathcal{D}_{\mathbf{hit}}) \leqslant$$
$$\mathbf{Pr}(\mathbf{guess} \vee \mathbf{hit} \vee \mathcal{D}_{\mathbf{guess}} \vee \mathcal{D}_{\mathbf{hit}} | \neg(\mathbf{key} \vee \mathbf{multi})) + \mathbf{Pr}(\mathbf{key} \vee \mathbf{multi}). \quad (15)$$

Event $\mathcal{D}_{\mathbf{guess}}$. Note that the adversary may freely choose the rate part in decryption queries and primitive queries (the ciphertext and intermediate tags are known for the adversary). $\mathcal{D}_{\mathbf{guess}}$ sets bad as soon as there is a primitive state and a decryption state whose capacity parts are equal. We write this as, $\mathcal{D}_{\mathbf{guess}}(i; j, k) \equiv x_i = s_{\delta,j,k}$, where $\delta = \mathcal{D}$ which means that an adversary $\mathcal{A}$ is not able to ask a query $\mathcal{E}$. This happens with probability at most $q_p\sigma_{\mathcal{D}}/2^c$,

$$\mathbf{Pr}(\mathcal{D}_{\mathbf{guess}} | \neg(\mathbf{key} \vee \mathbf{multi})) \leqslant q_p\sigma_{\mathcal{D}}/2^c.$$

Event $\mathcal{D}_{\mathbf{hit}}$. In this case an adversary has an ability to reuse nonce in the decryption queries. Note that just the public part of the nonce can be reused, PMN. In this case SMN is still private and unknown to the adversary. Any decryption state can hit the initial one (where just the key is unknown) with probability at most $\sigma_{\mathcal{D}}/2^k$. For the rest we have several sub-cases:

1. $(PMN; AD, C) = (PMN_{\delta,j}; AD_{\delta,j}, C_{\delta,j})$ but $T \neq T_{\delta,j}$. This case is infeasible and probability is 0, because intermediate tags and also the final tag are the same only if PMN, AD, SMN and plaintext pairs up to that are the same too.

2. $(PMN; AD) = (PMN_{\delta,j}; AD_{\delta,j})$ but $C \neq C_{\delta,j}$. Let ciphertexts $C \neq C_{\delta,j}$ are different in all their blocks, then $s^{SMN}_{j,0} = s^{SMN}_{\delta,j,0}$ and $s^{SMN}_{j,1} = C_0||[s^{SMN}_{\delta,j,1}]_c \neq s^{SMN}_{\delta,j,1}$. This means that the state is fresh and can be hit with some older state with probability $1/2^c$. In total, the j-th decryption query sets event *hit* with probability at most: $\frac{\sigma_{\mathcal{E}}\sigma_{\mathcal{D},j}+\sigma_{\mathcal{D},j}(\sigma_{\mathcal{D},1}+\ldots+\sigma_{\mathcal{D},j-1})}{2^c}$.

   Here we have one subcase, or if SMN is the same. This means that $C_0 = C_{\delta,j,0}$ and CIS for the message phase is the same $f(s^{SMN}_{j,1}) = f(s^{SMN}_{\delta,j,1})$. So the reasoning carries over the case where the rest of the ciphertext is different or it has longest common prefix.

   Let we say that $C_{\delta,j}$ shares the longest common prefix l with C and $l < m$. In this case $s^C_{j,l,0} = s^C_{\delta,j,l,0}$ and $s^C_{j,l,1} = C_l||[s^C_{\delta,j,l,1}]_c \neq s^C_{\delta,j,l,1}$, thus $s^C_{j,l,1}$ is a new state and new input to f. It can hit a certain older state with probability $1/2^c$, and the same bound holds as previous.

   Because we are working with intermediate tags, where after every block message, the tag is released, we have the following scenario. Let ciphertexts $C \neq C_{\delta,j}$ are different in all their blocks, then $s^C_{j,i,1} \neq s^C_{\delta,j,i,1}$, and $[f(s^C_{j,i,1})]^\tau = t_{j,i}$. The probability to hit the tag $t_{j,i}$ with some older state $[f(s^C_{\delta,j,i,1})]^\tau$-bits can be done with probability $1/2^\tau$. In total for all decryption queries *event* is set with probability at most $\binom{mq_{\mathcal{D}}}{2}/2^\tau$. The length of the intermediate tags is the same as the key length, so we can write for the probability: $\binom{mq_{\mathcal{D}}}{2}/2^k$.

3. $PMN = PMN_{\delta,j}$ but $AD \neq AD_{\delta,j}$. The analysis is the same as in the ciphertext case, except the case where inner collision can be done between the state where CIS is updated and states from the associated data ($a/2^r$). In the rest the reasoning carries over for all new future states.

4. $PMN \neq PMN_{\delta,j}$. The nonce is new so always s^{init} is fresh by construction and all future states in this query will be new.

Summing over all queries we get:

$$\begin{aligned}\Pr(\mathcal{D}_{\mathbf{hit}}|\neg(\mathbf{key} \vee \mathbf{multi})) &\leqslant \sum_{j=1}^{q_{\mathcal{D}}} \frac{\sigma_{\mathcal{E}}\sigma_{\mathcal{D},j} + \sigma_{\mathcal{D},j}(\sigma_{\mathcal{D},1} + \ldots + \sigma_{\mathcal{D},j-1})}{2^c} \\ &+ \frac{\binom{mq_{\mathcal{D}}}{2}}{2^k} + \frac{q_{\mathcal{D}}a}{2^r} + \frac{\sigma_{\mathcal{D}}}{2^k} \\ &\leqslant \frac{\sigma_{\mathcal{D}}\sigma_{\varepsilon} + \sigma_{\mathcal{D}}^2/2}{2^c} + \frac{q_{\mathcal{D}}a}{2^r} + \frac{\sigma_{\mathcal{D}} + m^2q_{\mathcal{D}}^2}{2^k}.\end{aligned}$$

Together with all the bounds from the privacy proof via (15) we get:

$$\begin{aligned}\mathbf{Pr}(\text{event}) \leqslant &\frac{q_p + \sigma_{\varepsilon}}{2^k} + \frac{(q_p + \sigma_{\varepsilon})^2}{2^b} + \frac{q_p r}{2^c} + \frac{q_{\varepsilon}a}{2^r} + \sqrt{\frac{8e\sigma_{\varepsilon}q_p}{2^b}} + \\ &\frac{q_p\sigma_{\mathcal{D}}}{2^c} + \frac{\sigma_{\mathcal{D}}\sigma_{\varepsilon} + \sigma_{\mathcal{D}}^2/2}{2^c} + \frac{q_{\mathcal{D}}a}{2^{b/2}} + \frac{\sigma_{\mathcal{D}} + m^2q_{\mathcal{D}}^2}{2^k}. \qquad (16)\end{aligned}$$

At the end, we have the final bound for integrity of π-Cipher with intermediate tags, that completes the proof:

$$\begin{aligned}\mathbf{Adv}_{\Pi}^{Auth}(\mathcal{A}) \leqslant &\frac{(q_p + \sigma_{\varepsilon} + \sigma_{\mathcal{D}})^2}{2^{b+1}} + \frac{\sigma_{\mathcal{D}}(q_p + \sigma_{\varepsilon} + \sigma_{\mathcal{D}})}{2^c} + \frac{q_p r}{2^c} + \frac{a(q_{\varepsilon} + q_{\mathcal{D}})}{2^{b/2}} + \\ &\frac{q_p + \sigma_{\varepsilon} + \sigma_{\mathcal{D}}}{2^k} + \frac{m^2q_{\mathcal{D}}^2 + mq_{\mathcal{D}}}{2^k} + \sqrt{\frac{8e\sigma_{\varepsilon}q_p}{2^b}}.\end{aligned}$$

■

7 Conclusion

In this paper we propose a new mode of operation of the second round candidate of the CAESAR competition, π-Cipher. It supports intermediate tags. The main idea behind this mode is to achieve an online property even in the phase of decryption and verification. Also with it, the cipher can be used on devices with limited memory storage without releasing unverified plaintext.

We proved that the privacy and authenticity of π-Cipher with intermediate tags are approximately $min\{2^k, 2^c, 2^{b/2}\}$ and with this π-Cipher can be put among the other CAESAR candidates and sponge based designs as ICEPOLE, Keyak, NORX and PRIMATEs [16].

References

[1] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Nicky Mouha, and Kan Yasuda. *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*, chapter How to Securely Release Unverified Plaintext in Authenticated Encryption, pages 105–125. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[2] Mihir Bellare and Chanathip Namprempre. *Advances in Cryptology — ASIACRYPT 2000: 6th International Conference on the Theory and Application of Cryptology and Information Security Kyoto, Japan, December 3–7, 2000 Proceedings*, chapter Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm, pages 531–545. Springer Berlin Heidelberg, Berlin, Heidelberg, 2000.

[3] Mihir Bellare, Phillip Rogaway, and David Wagner. *Fast Software Encryption: 11th International Workshop, FSE 2004, Delhi, India, February 5-7, 2004. Revised Papers*, chapter The EAX Mode of Operation, pages 389–407. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[4] D. J. Bernstein. Caesar: Competition for authenticated encryption: Security, applicability, and robustness. CAESAR web page, 2013. `http://competitions.cr.yp.to/index.html`.

[5] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Assche. *Selected Areas in Cryptography: 18th International Workshop, SAC 2011, Toronto, ON, Canada, August 11-12, 2011, Revised Selected Papers*, chapter Duplexing the Sponge: Single-Pass Authenticated Encryption and Other Applications, pages 320–337. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[6] Chanathip Namprempre, Phillip Rogaway, and Thomas Shrimpton. AE5 Security Notions. Definitions Implicit in the CAESAR Call. Cryptology ePrint Archive, 2013. `https://eprint.iacr.org/2013/242.pdf`.

[7] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schlaffer. Ascon v1.1. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/asconv11.pdf`.

[8] Danilo Gligoroski, Hristina Mihajloska, Simona Samardjiska, Hakon Jacobsen, Mohamed El-Hadedy, Rune Erlend Jensen, and Daniel Otte. π-Cipher v2.0. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/picipherv20.pdf`.

[9] Morris J. Dworkin. Sp 800-38c. recommendation for block cipher modes of operation: The ccm mode for authentication and confidentiality. Technical report, Gaithersburg, MD, United States, 2004.

[10] Elena Andreeva, Andrey Bogdanov, Atul Luykx, Bart Mennink, Elmar Tischhauser, and Kan Yasuda. AES-COPA v2.0. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/aescopav2.pdf`.

[11] Elena Andreeva, Begul Bilgin, Andrey Bogdanov, Atul Luykx, Florian Mendel, Bart Mennink, Nicky Mouha, Qingju Wang, and Kan Yasuda. PRIMATEs v1.02. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/primatesv102.pdf`.

[12] Abed Farzaneh, Scott Fluhrer, John Foley, Christian Forler, Eik List, Stefan Lucks, David McGrew, and Jakob Wenzel. The POET Family of On-Line Authenticated Encryption Schemes v2.0. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/poetv20.pdf`.

[13] Guido Bertoni, Joan Daemen, Michael Peeters, Gilles Van Assche, and Ronny Van Keer. Keyak v2. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/keyakv2.pdf`.

[14] Viet Tung Hoang, Reza Reyhanitabar, Phillip Rogaway, and Damian Vizár. *Advances in Cryptology – CRYPTO 2015: 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, chapter Online Authenticated-Encryption and its Nonce-Reuse Misuse-Resistance, pages 493–517. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[15] Jean-Philippe Aumasson, Philipp Jovanovic, and Samuel Neves. NORX v2.0. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/norxv20.pdf`.

[16] Philipp Jovanovic, Atul Luykx, and Bart Mennink. Beyond 2c2 security in sponge-based authenticated encryption modes. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology ASIACRYPT 2014*, volume 8873 of *Lecture Notes in Computer Science*, pages 85–104. Springer Berlin Heidelberg, 2014.

[17] Markku-Juhani O. Saarinen and Billy B. Brumley. STRIBOBr2: WHIRLBOB. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/stribobr2.pdf`.

[18] David A. McGrew and John Viega. *Progress in Cryptology - INDOCRYPT 2004: 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004. Proceedings*, chapter The Security and Performance of the Galois/Counter Mode (GCM) of Operation, pages 343–355. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[19] Nilanjan Datta and Mridul Nandi. ELmD v2.0. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/elmdv20.pdf`.

[20] Pawel Morawiecki, Kris Gaj, Ekawat Homsirikamol, Krystian Matusiewicz, Josef Pieprzyk, Marcin Rogawski, Marian Srebrny, and Marcin Wojcik. ICPOLE v2. CAESAR web page, 2015. `https://competitions.cr.yp.to/round2/icepolev2.pdf`.

[21] Phillip Rogaway. *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*, chapter Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC, pages 16–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[22] Phillip Rogaway and Thomas Shrimpton. *Advances in Cryptology - EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006. Proceedings*, chapter A Provable-Security Treatment of the Key-Wrap Problem, pages 373–390. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[23] Phillip Rogaway and Thomas Shrimpton. *Advances in Cryptology - EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006. Proceedings*, chapter A Provable-Security Treatment of the Key-Wrap Problem, pages 373–390. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.