Cryptogr. Commun. (2018) 10:177–193
DOI 10.1007/s12095-017-0250-9

POEx: A beyond-birthday-bound-secure on-line cipher

Christian Forler[1] · Eik List[2] · Stefan Lucks[2] · Jakob Wenzel[2]

Received: 29 November 2016 / Accepted: 30 June 2017 / Published online: 26 August 2017
© Springer Science+Business Media, LLC 2017

Abstract On-line ciphers are convenient building blocks for realizing efficient single- pass encryption. In particular, the trend to limit the consequences of nonce reuses rendered them popular in recent authenticated encryption schemes. While encryption schemes, such as POE, COPE, or the ciphers within ElmE/ElmD concentrated on efficiency, their security guarantees and that of all earlier on-line ciphers is limited by the birthday bound, and so are those of the AE schemes built upon them. This work proposes POEx, a beyond-birthday-bound-secure on-line cipher which employs one call to a tweakable block cipher and one call to a $2n$-bit universal hash function per message block. POEx builds upon the recently proposed XTX tweak extender by Iwata and Minematsu. We prove the security of our construction and discuss possible instantiations.

Keywords Symmetric cryptography · Provable security · On-line cipher · Universal hash function · Tweakable block cipher

Mathematics Subject Classification 11T71

This article is part of the Topical Collection on *Recent Trends in Cryptography*

✉ Eik List
Eik.List@uni-weimar.de

Christian Forler
cforler@beuth-hochschule.de

Stefan Lucks
Stefan.Lucks@uni-weimar.de

Jakob Wenzel
Jakob.Wenzel@uni-weimar.de

[1] Beuth Hochschule für Technik Berlin, Berlin, Germany

[2] Bauhaus-Universität Weimar, Weimar, Germany

1 Introduction

On-line Ciphers Restating the informal definition by Rogaway and Zhang [32], a cryptographic transform is called *on-line* if it can be realized by an algorithm that, for any allowed input, reads its input bytes one at a time in order and computes the corresponding output bytes one at a time in order, using only a constant-bound amount of memory and/or latency. On-line ciphers were introduced by Bellare et al. [7]: for some fixed n (which typically represents the block size of the underlying primitive), a message M is considered as a sequence of n-bit blocks $M_1, \ldots, M_m$, $|M_i| = n$, for $1 \leq i \leq m$. Then, an on-line cipher is a deterministic length-preserving permutation where the i-th output block C_i depends only on the first i input blocks $M_1, \ldots, M_i$ and the secret key. Boldyreva and Taesombut [13] later strengthened this definition to include the requirements of being computable with constant memory and latency.

Limitations Due to their nature, on-line ciphers cannot provide the usual security notion of general ciphers, i.e., indistinguishability from a PRP (pseudorandom permutation) or SPRP (strong PRP). This stems directly from the fact that a secure cipher must make every bit of the ciphertext depend on every bit of the plaintext and vice versa – a requirement which prohibits constant-bound memory and latency. So, the security notions of on-line ciphers are derived forms of the notions for ciphers in general. As the ideal primitive, Bellare et al. [7] proposed an *on-line permutation* (OPRP), that is a family of permutations $P = \{P_i | P_i : (\{0,1\}^n)^{i-1} \times \{0,1\}^n \rightarrow \{0,1\}^n\}$, for $i \geq 1$, such that every P_i takes the current message block M_i as input and all previous input blocks $M_1, \ldots, M_{i-1}$ as "tweak" that defines the permutation $P_i(M_1 \| \ldots \| M_{i-1}, \cdot)$. An on-line cipher is then called secure if it is infeasible to distinguish it from an OPRP (on-line pseudorandom permutation) or SOPRP (strong OPRP).

Applications Despite their limitations, on-line ciphers are highly valuable in practice, for they allow to securely encrypt messages within a single pass. This is particularly important in environments with demanding throughput or low-latency requirements. Moreover, various network APIs in practice are stream- oriented[1], which disallows to buffer the entire input. In authenticated encryption, the use of OPRP-secure on-line ciphers limit the consequences of nonce reuses which motivated their application in several CAESAR [10] candidates. Though, many renowned block-cipher modes, e.g., CBC, CTR, or the encryption procedures of GCM [25] and OCB [21] actually *are* on-line ciphers. What renders the latter secure against chosen-plaintext attacks is the dependency on the additional input such as an encrypted nonce. Without it, their standalone encryption would lack even the basic OPRP security due to the missing dependency between blocks. In the remainder, we focus on on-line ciphers that provide at least OPRP security without depending on auxiliary inputs.

Existing Block-Cipher-Based On-line Ciphers Due to their similarity to CBC, early on-line ciphers were inherently sequential. Bellare et al. [7] proposed HCBC1 and HCBC2, both of which use one call to the block cipher and one call to an almost-XOR-universal hash function. As major difference, the former construction employs a single multiplication and could provide only OPRP security, whereas the latter construction uses two

[1]For example, the OpenSSL `EVP_DecryptUpdate` interface [34].

multiplications to achieve SOPRP security. Boldyreva and Taesombut [13] later proposed a variant of HCBC2, called HPCBC, that prepended the encryption of a random IV in order to fit their strengthened notions. Nandi [29, 30] proposed two modified SOPRP-secure variants of HCBC1 and HCBC2, called MHCBC and MCBC. His MCBC construction could replace the additional call to a universal hash function by a second invocation of the block cipher. Rogaway and Zhang [32] could also eliminate any additional calls by employing a tweakable block cipher for their constructions TC1, TC2, and TC3. In parallel to them, Fleischmann et al. published the on-line authenticated encryption (AE) scheme McOE [17], also based on a tweakable block cipher. Their McOE-G variant was similar to TC3, but added the tag-splitting approach for handling arbitrary-length inputs. Yet, all of them were strictly sequential.

Recently proposed on-line ciphers targeted at improving efficiency. The on-line cipher POE (and the derived AE scheme POET) by Abed et al. [1] combines two calls to a universal hash function with one call to the block cipher per message block, which allows to *pipeline* the message processing. Andreeva et al. [4] proposed the first (almost) fully parallelizable OPRP-secure on-line cipher COPE (and the derived AE scheme COPA). Their design followed an Encrypt-Mix-Encrypt [18] approach at the cost of two block-cipher calls per block. Datta and Nandi adapted this strategy in their AE schemes ELmE and ELmD [14, 15] with a modified Mix layer; the CAESAR candidates ELmD and COPA have later been merged to COLM [2]. COBRA [5] employed a variant of the two-round Feistel network from OTR [27]; however, COBRA has been broken and withdrawn from the CAESAR competition. Last but not least, OleF [11] combined the Encrypt-Mix-Encrypt idea with the double-block approach and applied a four-round Feistel network to each for inverse-free decryption.

Beyond-Birthday-Bound Security While there has been significant progress on the efficiency, the security of all on-line ciphers and on-line AE schemes above is still limited by the birthday bound, i.e., at most $\ell \ll 2^{n/2}$ blocks can be encrypted under the same key for a block size of n bits. For on-line AE schemes, this bound matches the common expectations for privacy. Though, there is a significant gap to the optimal authenticity bound of $O(\ell/2^n)$, which was already criticized by Lu [24]. A birthday-bound limitation is also relevant in resource-constrained environments, where schemes would have to be instantiated with a lightweight block cipher. Using the de-facto standard block size of $n = 64$ bits for lightweight ciphers (e.g. for PRESENT [12, 19]), the security would have already vanished after encrypting about 2^{32} blocks (64 GiB); for example, if 1 MiB-messages were allowed (2^{17} blocks), the forgery probability would become about 1/16 already after 2^{13} messages had been processed under the same key. Nonetheless, one can imagine various further settings where beyond-birthday-bound (BBB) security is desirable for on-line ciphers.

Contribution This work proposes POEx, a BBB-secure family of on-line ciphers which combines the XTX approach by Iwata and Minematsu [28] with ideas from TCx [32], McOE [17], and POE [1]. While our proposal is similar to those three schemes, their security is limited by the birthday bound since the adversary has full control over n bits of the tweak, whereas it has not in POEx. Table 1 compares our approach to existing on-line ciphers and encryption procedures from on-line nonce-misuse resistant AE schemes. Since there exist well-known methods to transform a given on-line cipher into an AE-secure on-line AE scheme (e.g. [7, 17, 32]), this work proposes only a family of on-line ciphers and leaves the (non-trivial) task of extending it to a BBB-secure AE scheme for future work.

Table 1 Comparison of existing OPRP-secure on-line ciphers and on-line nonce-misuse-resistant block-cipher-based AE schemes with our proposal

		On-line ciphers											OAE schemes					
Aspect	**POEx**	COPE [4]	HCBC1 [7]	HCBC2 [7]	HPCBC [13]	MCBC [30]	MHCBC [30]	POE [1]	OLEF [11]	TC1 [32]	TC2 [32]	TC3 [32]	COBRA [5]	COLM [2]	ELMD [14]	ELME [15]	MCOE-G [17]	MCOE-X [17]
#(T)BC calls	m	$2m$	m	m	$m+1$	m	m	m	$2m$	m	m	m	m	$2m$	$2m$	$2m$	m	m
#Keys	2	1	2	2	2	1	2	2	1	1	1	1	1	1	1	1	2	1
SOPRP-secure	●	–	–	●	●	●	●	●	●	–	●	●	(NAE security)					
BBB-secure	●	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

● = provides feature, – = lacks feature. #(T)BC = (tweakable) block cipher calls, BBB = beyond-birthday-bound, NAE = nonce-based authenticated encryption

Outline What remains is structured as follows: after briefly reviewing the preliminaries, Section 3 describes the generic POEx. Next, Section 4 recalls the relevant security notions. Section 5 provides the results of our security analysis. Section 6 provides a discussion and conclusion.

2 Preliminaries

We use notions similar to those in [3]. We use lowercase letters x, y for indices and integers, uppercase letters X, Y for binary strings and functions, and calligraphic uppercase letters $\mathcal{X}, \mathcal{Y}$ for sets. We denote the concatenation of binary strings X and Y by $X \| Y$ and the result of their bitwise XOR by $X \oplus Y$. We indicate the length of X in bits by $|X|$, and write X_i for the i-th block. $X \twoheadleftarrow \mathcal{X}$ denotes that X is chosen uniformly at random from the set $\mathcal{X}$. We define three sets of particular interest: $\mathsf{Perm}(\mathcal{X})$ be the set of all permutations on X, $\mathsf{TPerm}(\mathcal{T}, \mathcal{X})$ the set of all tweaked permutations over $\mathcal{X}$ with associated non-empty tweak space $\mathcal{T}$, and OPerm_n the set of all n-bit on-line permutations. We define by $X_1, \ldots, X_j \overset{x}{\leftarrow} X$ an injective splitting of the string X into blocks such that $X = X_1 \| \cdots \| X_j$ with $|X_i| = x$ for $1 \le i \le j-1$, and $|X_j| \le x$. For an event E, we denote by $\Pr[E]$ the probability of E.

For a set $\mathcal{X}$, we denote an ℓ-element list by $X = (X_1, \ldots, \mathcal{X}^\ell)$ with $X \in \mathcal{X}^\ell$. We denote an i-element sub-list as $X_{1..i} := (X_1, \ldots, X_i)$. Given sets X and Y_i, for $1 \le i \le \ell$, and a mapping $H : \mathcal{X} \to \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_\ell$ with multiple outputs, we denote by $H(X)[i]$ the i-th output $Y_i \in Y_i$. For sets X and Y, let ℓ-element lists $X = (X_1, \ldots, \mathcal{X}^\ell)$, $X' = (X'_1, \ldots, X'_\ell)$, $Y = (Y_1, \ldots, Y_\ell)$, and $Y' = (Y'_1, \ldots, Y'_\ell)$ with $X, X' \in X^\ell$ and $Y, Y' \in \mathcal{Y}^\ell$. We call (X, Y) and (X', Y') element-wise disjoint and denote it by $(X, Y) \not\equiv (X', Y')$ iff it holds that $(X_{1..i}, Y_{1..i}) \neq (X'_{1..i}, Y'_{1..i})$, for all $1 \le i \le \ell$.

3 Generic definition of POEx

This section defines the generic POEx construction. Fix integers $n, \tau \ge 1$. Let $\mathcal{K}_1$ and $\mathcal{K}_2$ be non-empty key sets and $\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2$. Further, we define two non-empty tweak sets $\mathcal{T} = \{0,1\}^\tau$ and $\mathcal{T}' = (\{0,1\}^n)^2$. Let $\mathcal{M}, \mathcal{C} \subseteq (\{0,1\}^n)^*$ denote message and ciphertext space, respectively. Let $\mathcal{H} = \{H | H : \mathcal{T}' \to \{0,1\}^{n+\tau}\}$ be a family of ϵ-cpAXU hash functions, where a key $K_2 \in \mathcal{K}_2$ defines the instance $H \in \mathcal{H}$. We will explain this particular notion of universality in the following section.

Further, let $\widetilde{E} : \mathcal{K}_1 \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ denote a tweakable block cipher and let $\widetilde{D} : \mathcal{K}_1 \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ denote its inverse (since $\widetilde{E}^{-1,T}$ may be misleading). We will write $\widetilde{E}^T_K(\cdot)$ and $\widetilde{D}^T_K(\cdot)$ as short forms of $\widetilde{E}(K, T, \cdot)$ and $\widetilde{D}(K, T, \cdot)$, respectively. Moreover, we will use $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$ to also mean that W_i represents the first n bit of the output of H, and V_i the remaining bits. Next, we recall the definition of XTX briefly, which will simplify our later definition of POEx.

Definition 1 (XTX [28]) Let $K_1, K_2 \in \mathcal{K}$ be independent, let $\mathcal{T}$, $\widetilde{E}$, and $\mathcal{H}$ be defined as above, and let $\mathcal{T}'$ be a space. Let $H \in \mathcal{H}$ be defined by K_2. Then, $\mathrm{XTX}[\widetilde{E}, H] : \mathcal{K}_1 \times \mathcal{K}_2 \times \mathcal{T}' \times \{0,1\}^n \to \{0,1\}^n$ is defined as

$$\mathrm{XTX}[\widetilde{E}, H]^T_{K_1,K_2}(M) := \widetilde{E}^V_{K_1}(M \oplus W) \oplus W, \text{ where } (W, V) \leftarrow H_{K_2}(T).$$

For encryption, POEx fixes a pair of initial chaining values (X_0, Y_0) to constants: $X_0 = \text{const}_x$ and $Y_0 = \text{const}_y$, such that $(\text{const}_x \,\|\, \text{const}_y) \in \mathcal{T}'$. For the i-th input block, the values W_i and V_i are derived from X_{i-1} and Y_{i-1} from H_{K_2}. To compute the i-th ciphertext block C_i, V_i is used as tweak for $\widetilde{E}$, and W_i is XORed to the message block M_i to derive the next top-row chaining value $X_i \leftarrow M_i \oplus W_i$. The result of its encryption with $\widetilde{E}$ under key K_1 and tweak V_i yields the next bottom-row chaining value: $Y_i \leftarrow \widetilde{E}^{V_i}_{K_1}(X_i)$. C_i is then given by $C_i \leftarrow Y_i \oplus W_i$. The procedure is repeated for all message blocks and the ciphertext C results from the concatenation $C \leftarrow (C_1 \| \cdots \| C_m)$. A schematic illustration of the encryption process is shown in Fig. 1. The decryption works analogously. We define POEx$[\widetilde{E}, H] = (\mathcal{E}, \mathcal{D})$ with deterministic encryption algorithm $\mathcal{E} : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ and deterministic decryption algorithm $\mathcal{D} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$ as given in Algorithm 1. For all $K \in \mathcal{K}$, $M \in \mathcal{M}$, and $C \in \mathcal{C}$, it holds that $\mathcal{D}_K(\mathcal{E}_K(M)) = M$ and $\mathcal{E}_K(\mathcal{D}_K(C)) = C$.

Algorithm 1 Definition of the encryption and decryption algorithms of POEx

```
1: function $\mathcal{E}_{K_1,K_2}(M)$
2:   $X_0 \leftarrow \text{const}_x$; $Y_0 \leftarrow \text{const}_y$
3:   $m \leftarrow |M|/n$
4:   $(M_1, \ldots, M_m) \xleftarrow{n} M$
5:   for $i \leftarrow 1$ to $m$ do
6:     $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$
7:     $X_i \leftarrow M_i \oplus W_i$
8:     $Y_i \leftarrow \widetilde{E}^{V_i}_{K_1}(X_i)$
9:     $C_i \leftarrow Y_i \oplus W_i$
10:  return $C \leftarrow (C_1 \| \cdots \| C_m)$

11: function $\mathcal{D}_{K_1,K_2}(C)$
12:   $X_0 \leftarrow \text{const}_x$; $Y_0 \leftarrow \text{const}_y$
13:   $m \leftarrow |C|/n$
14:   $(C_1, \ldots, C_m) \xleftarrow{n} C$
15:   for $i \leftarrow 1$ to $m$ do
16:     $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$
17:     $Y_i \leftarrow C_i \oplus W_i$
18:     $X_i \leftarrow \widetilde{D}^{V_i}_{K_1}(Y_i)$
19:     $M_i \leftarrow X_i \oplus W_i$
20:  return $M \leftarrow (M_1 \| \cdots \| M_m)$
```

4 Security notions

4.1 Adversaries and Advantages

An adversary $\mathbf{A}$ is an efficient Turing machine that interacts with a given set of oracles that appear as black boxes to $\mathbf{A}$. We denote by $\mathbb{A}$ the class of all computationally bounded adversaries and $\mathbf{A}^{\mathcal{O}}$ for the output of $\mathbf{A}$ after interacting with an oracle $\mathcal{O}$. We write $\Delta_{\mathbf{A}} \mathcal{O}^L \mathcal{O}^R := |\Pr[\mathbf{A}^{\mathcal{O}^L} \Rightarrow 1] - \Pr[\mathbf{A}^{\mathcal{O}^R} \Rightarrow 1]|$ for the advantage of $\mathbf{A}$ to distinguish between

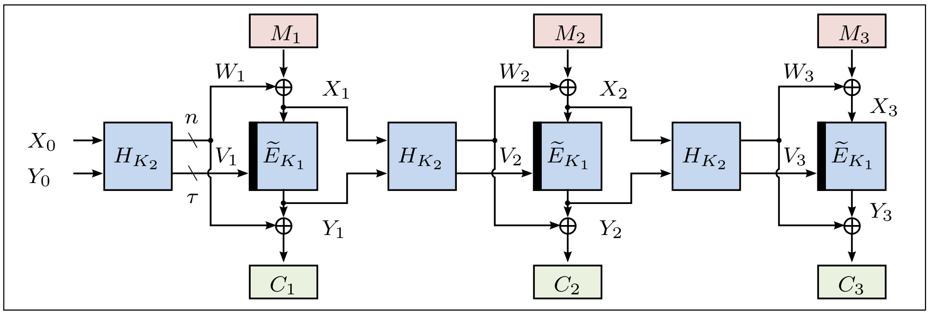

Fig. 1 Encryption of a three-block message $M = (M_1, M_2, M_3)$ with POEx$[\widetilde{E}, H]$. $\widetilde{E} : \mathcal{K}_1 \times \{0,1\}^\tau \times \{0,1\}^n \to \{0,1\}^n$ is a tweakable block cipher and $H : \{0,1\}^{2n} \to \{0,1\}^{n+\tau}$ a keyed universal hash function

oracles $\mathcal{O}^L$ and $\mathcal{O}^R$. All probabilities are defined over the random coins of the oracles and those of the adversary, if any. We say that $\mathbf{A}$ is a (q, σ, t)-X adversary if it asks at most q queries of at most σ blocks in total and runs in time at most t. We call $\mathbf{A}$ a (q, t)-X adversary if queries cannot contain multiple blocks. We write $\mathbf{Adv}_F^X(q, \sigma, t) := \max_{\mathbf{A} \in \mathbb{A}}\{\mathbf{Adv}_F^X(\mathbf{A})\}$ for the maximal advantage over all (q, σ, t)-X adversaries $\mathbf{A}$ on F and analogously for (q, t)-X adversaries. W.l.o.g., we assume that $\mathbf{A}$ never asks queries to which it already knows the answer.

We will provide pseudocode descriptions of the oracles, which will be referred to as games, according to the game-playing framework by Bellare and Rogaway [8]. Each game consists of a set of procedures. We define $\Pr[G(\mathbf{A}) \Rightarrow x]$ as the probability that the Game G outputs x when given $\mathbf{A}$ as input.

4.2 Security Definitions for Universal Hashing

Definition 2 (ϵ-Almost-(XOR-)Universal Hash Functions) Define two sets of bit strings $\mathcal{X}, \mathcal{Y} \subseteq \{0, 1\}^*$. Let $\mathcal{H} = \{H \mid H : \mathcal{X} \to \mathcal{Y}\}$ denote a family of hash functions. $\mathcal{H}$ is called ϵ-almost-universal (ϵ-AU) iff for all distinct elements $X, X' \in \mathcal{X}$, it holds that $\Pr_{H \twoheadleftarrow \mathcal{H}}\left[H(X) = H(X')\right] \leq \epsilon$. $\mathcal{H}$ is called *ϵ-almost-XOR-universal* (ϵ-AXU) iff for all distinct $X, X' \in \mathcal{X}$ and $Y \in \mathcal{Y}$, it holds that $\Pr_{H \twoheadleftarrow \mathcal{H}}[H(X) \oplus H(X') = Y] \leq \epsilon$.

Minematsu and Iwata [28] introduced the notion of partial ϵ-almost-XOR-universality, which will be useful for our later security analysis of POEx.

Definition 3 (ϵ-Partial-AXU Hash Functions) Let $n, m \geq 1$ be fixed and $\mathcal{X}$ be a non-empty set. Let $\mathcal{H} = \{H \mid H : \mathcal{X} \to \{0, 1\}^n \times \{0, 1\}^m\}$ be a family of hash functions. We say that $\mathcal{H}$ is (n, m, ϵ)-partial-AXU or short (n, m, ϵ)-pAXU iff for all distinct elements $X, X' \in \mathcal{X}$ and all $\Delta \in \{0, 1\}^n$, it holds that

$$\max_{X, X', \Delta} \Pr_{H \twoheadleftarrow \mathcal{H}}\left[H(X) \oplus H(X') = (\Delta, 0^m)\right] \leq \epsilon.$$

Informally spoken, the notion captures the probability that one part of a given pair of hashes $H(X)$ and $H(X')$ collides and the other part has a specific non-zero difference. Clearly, an ϵ-AXU hash function is also ϵ-pAXU. In the following, we introduce a notion of chained partial- almost-XOR universality to articulate the security requirements when the inputs depend on earlier outputs of the hash function.

Throughout the following, let integers $n, m \geq 1$ and let $\mathcal{H} = \{H \mid H : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}^n \times \{0, 1\}^m\}$ be a family of hash functions. From it, we derive a chained construction $\mathcal{H}^i = \{H^i \mid H^i : \{0, 1\}^n \times \{0, 1\}^n \times (\{0, 1\}^n)^i \times (\{0, 1\}^n)^i \to \{0, 1\}^n \times \{0, 1\}^m\}$ for a positive integer i.

Definition 4 (One-sided Chained Hash Functions) Let $H \in \mathcal{H}$ be given as above. We define the instances $H^i(X_0, Y_0, M_{1..i-1}, C_{1..i-1}) = (W_i, V_i)$ of $\mathcal{H}^i$ for all integers $i \geq 1$ and all inputs $X_0, Y_0 \in \{0, 1\}^n$ and $M, C \in (\{0, 1\}^n)^{i-1}$ recursively as

$$(W_i, V_i) := \begin{cases} H(X_0, Y_0) & \text{if } i = 1, \\ H(W_{i-1} \oplus M_{i-1}, W_{i-1} \oplus C_{i-1}) & \text{otherwise,} \end{cases}$$

where $W_{i-1} := H^{i-1}(X_0, Y_0, M_{1..i-2}, C_{1..i-2})[1]$ denotes the first output of H^{i-1}.

Definition 5 (ϵ-Chained-Partial-AXU Hash Functions) Let H and $\mathcal{H}^c$ be given as above for some positive integer c. For $1 \le i \le c-1$, we define $X_i = W_i \oplus M_i$ and $Y_i = W_i \oplus C_i$, and for $1 \le i \le c$, we define Boolean variables

$$E_i{:=}H^i\left(X_0, Y_0, M_{1..i-1}, C_{1..i-1}\right) \oplus H^i\left(X_0, Y_0, M'_{1..i-1}, C'_{1..i-1}\right) = \left(\Delta, 0^m\right).$$

We say that $\mathcal{H}^c$ is (n, m, c, ϵ)-chained-PAXU or short (n, m, c, ϵ)-CPAXU iff for all $\Delta \in \{0,1\}^n$, $X_0, Y_0 \in \{0,1\}^n$, and all $M, M', C, C' \in (\{0,1\}^n)^{c-1}$ s. t. for all intermediate values $(X_i, Y_i) \ne (X_j, Y_j)$ with $1 \le j < i \le c$, it holds theat $\Pr_{H \twoheadleftarrow \mathcal{H}}[E_i] \le \epsilon$.

Note that sampling the instance H from $\mathcal{H}$ also defines H^i. Further note that a (n, m, c, ϵ)-CPAXU hash function is also (n, m, ϵ)-PAXU since the partial-AXU notion represents the case $c = 1$; however, for $c > 1$, CPAXU poses a stronger requirement to the hash function than PAXU.

4.3 Security definitions for tweakable block ciphers

We briefly recall the security notions for tweakable block ciphers from [22, 32].

Definition 6 (TPRP/STPRP Advantage) Fix $n \ge 1$. Let $\mathcal{K}$ and $\mathcal{T}$ denote a non-empty key and tweak space, respectively. Let $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable block cipher and $\widetilde{D}$ its inverse. Further, let $\mathbf{A}$, $\mathbf{A}'$ be computationally bounded adversaries, where $\mathbf{A}$ has access to an oracle and $\mathbf{A}'$ has access to two oracles. Let $\widetilde{\pi} \twoheadleftarrow \mathsf{TPerm}(\mathcal{T}, \{0,1\}^n)$ and $K \twoheadleftarrow \mathcal{K}$. Then, the TPRP advantage of $\mathbf{A}$ wrt. $\widetilde{E}$ and the STPRP advantage of $\mathbf{A}'$ wrt. $\widetilde{E}$ and $\widetilde{D}$ are defined as $\mathbf{Adv}^{\mathrm{TPRP}}_{\widetilde{E}}(\mathbf{A}) := \underset{\mathbf{A}}{\Delta}(\widetilde{E}_K; \widetilde{\pi})$ and $\mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{E},\widetilde{D}}(\mathbf{A}') := \Delta_{\mathbf{A}'}(\widetilde{E}_K, \widetilde{D}_K; \widetilde{\pi}, \widetilde{\pi}^{-1})$.

4.4 Security definitions for on-line ciphers

A secure cipher should behave like a random permutation. It is easy to see that on-line ciphers cannot guarantee this property since the encryption of message block M_i must not depend on the subsequent blocks M_j, for $j > i$. The on- line behavior implies that two messages M, M' that share a p-block common prefix are always encrypted to two ciphertexts C, C' that also share a p-block common prefix. Hence, an on-line cipher Π is called secure iff no ciphertext reveals any further information about a plaintext than its length and the *longest common prefix* with previous messages.

Definition 7 (Length of Longest Common Prefix [17]) Fix $n \ge 1$ and let $\mathcal{M} \subseteq (\{0,1\}^n)^*$. Given $M, M' \in \mathcal{M}$, we define the length of their longest common prefix as $\mathrm{LLCP}_n(M, M') := \max_i \left\{1 \le j \le i : M_j = M'_j\right\}$. Given a set $\mathcal{Q}$ of messages $M' \in \mathcal{M}$, we define $\mathrm{LLCP}_n(M, \mathcal{Q}) := \max_{M' \in \mathcal{Q}} \left\{\mathrm{LLCP}_n(M, M')\right\}$.

Definition 8 (On-line Cipher) Fix $n \ge 1$. Let $\mathcal{M}, \mathcal{C} \subseteq (\{0,1\}^n)^*$ and let $\mathcal{K}$ be a non-empty space. Let $\mathcal{E} : \mathcal{K} \times (\{0,1\}^n)^* \to (\{0,1\}^n)^*$ be a keyed family of permutations that takes a key $K \in \mathcal{K}$ and a message $M \in \mathcal{M}$, and outputs a ciphertext $C \in \mathcal{C}$ such that $|C| = |M|$. We call $\Pi = (\mathcal{E}, \mathcal{D})$ an *on-line cipher* iff for all $i \in [1, |M|/n]$, C_i depends only on M_1 through M_i, and iff $\mathcal{D}$ is the decryption algorithm corresponding to $\mathcal{E}$.

Definition 9 (On-line Permutation) Define integers $i, j, m, n \geq 1$, and let $P_i : (\{0,1\}^n)^i \to \{0,1\}^n$ be a family of indexed n-bit permutations, i.e., for a fixed index $j \in (\{0,1\}^n)^{i-1}$, $P_i(j, \cdot)$ is a permutation. We define an n-bit on-line permutation $P : (\{0,1\}^n)^m \to (\{0,1\}^n)^m$ as a composition of m permutations $P_1 \cup P_2 \cup \cdots \cup P_m$. An m-block input $M = (M_1, \ldots, M_m)$ is mapped to an m-block output $C = (C_1, \ldots, C_m)$ by $C_i = P_i(M_1 \| \cdots \| M_{i-1}, M_i)$, for all $1 \leq i \leq m$.

For any two distinct m-block inputs M, M' that share an exactly p-block common prefix $M_1 \| \cdots \| M_p$, the corresponding outputs $C = P(M), C' = P(M')$ satisfy $C_i = C'_i$ for all $i \in [1, p]$ and $p \leq m$. However, it applies that $C_{p+1} \neq C'_{p+1}$; moreover, all further blocks C_i, C'_i, with $i \in [p+2, m]$, are pairwise different with high probability. We denote by OPerm_n the set of all n-bit on-line permutations. A random on-line permutation can be efficiently implemented by lazy sampling.

Definition 10 (OPRP/SOPRP Advantage) Let $\Pi = (\mathcal{E}, \mathcal{D})$ be an on-line cipher with block size $n \geq 1$ and $\mathcal{K}$ be a non-empty set. Let $K \twoheadleftarrow \mathcal{K}$ and $P \twoheadleftarrow \mathsf{OPerm}_n$. Let $\mathbf{A}$ and $\mathbf{A}'$ be computationally bounded adversaries, where $\mathbf{A}$ has access to an oracle $\mathcal{O}$, and $\mathbf{A}'$ has access to two oracles $\mathcal{O}_1$ and $\mathcal{O}_2$. Then, the OPRP advantage of $\mathbf{A}$ wrt. Π is defined as $\mathbf{Adv}_{\Pi}^{\mathrm{OPRP}}(\mathbf{A}) := \underset{\mathbf{A}}{\Delta}(\mathcal{E}_K; P)$ and the SOPRP advantage of $\mathbf{A}'$ wrt. Π and Pi^{-1} as $\mathbf{Adv}_{\Pi,\Pi^{-1}}^{\mathrm{SOPRP}}(\mathbf{A}') := \Delta_{\mathbf{A}'}(\mathcal{E}_K, \mathcal{D}_K; P, P^{-1})$.

5 Security analysis of POEx

Prior to the security analysis, we recall a theorem for XTX that will be used in the proof later in this section. Throughout the remainder, fix $\tau, n \geq 1$. Let $\mathcal{K}_1$, $\mathcal{K}_2$, $\mathcal{T} = \{0,1\}^\tau$, and $\mathcal{T}'$ denote spaces and define $\mathcal{K} = \mathcal{K}_1 \times \mathcal{K}_2$. Further, let $\widetilde{\pi} \twoheadleftarrow \mathsf{TPerm}(T, \{0,1\}^n)$ and $\mathcal{H} = \{H | H : \mathcal{T}' \to \{0,1\}^n \times \mathcal{T}\}$ be a family of hash functions.

Theorem 1 ([28]) *Let $\mathcal{H}$ be (n, τ, ϵ)-PAXU, where $H \in \mathcal{H}$ is defined by $K_2 \twoheadleftarrow \mathcal{K}_2$. Let $\mathbf{A}$ be a (q, t)-STPRP adversary on $\mathrm{XTX}[\widetilde{\pi}, H]$ with access to two oracles. Then,*

$$\mathbf{Adv}_{\mathrm{XTX}[\widetilde{\pi},H],\mathrm{XTX}[\widetilde{\pi}^{-1},H]^{-1}}^{\mathrm{STPRP}}(\mathbf{A}) \leq \epsilon \cdot q^2.$$

Theorem 2 (SOPRP Security of Generic POEx) *Let $K_1, K_2 \twoheadleftarrow \mathcal{K}$ and $\mathcal{T}' = \{0,1\}^n \times \{0,1\}^n$. Let $\widetilde{E} : \mathcal{K}_1 \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable block cipher and $\mathcal{H}$ be $(n, \tau, 2^{n-1}, \epsilon)$-CPAXU, where $H \in \mathcal{H}$ is defined by K_2. Let $\mathbf{A}$ be a (q, σ, t)-SOPRP adversary on $\mathrm{POEx}[\widetilde{E}, H]$ with access to two oracles and let $\sigma < 2^{n-1}$ blocks. Then,*

$$\mathbf{Adv}_{\mathrm{POEx}[\widetilde{E},H],\mathrm{POEx}[\widetilde{D},H]^{-1}}^{\mathrm{SOPRP}}(\mathbf{A}) \leq 2\sigma^2\epsilon\left(1 + \frac{2^\tau}{2^n}\right) + \frac{4\sigma}{2^n} + \mathbf{Adv}_{\widetilde{E},\widetilde{D}^{-1}}^{\mathrm{STPRP}}(\sigma, O(t)).$$

Proof Let $\mathbf{A}$ have access to two oracles that respond with either real encryptions (decryptions) using $\mathrm{POEx}[\widetilde{E}, H]$ (or $\mathrm{POEx}[\widetilde{D}, H]^{-1}$), under a uniformly at random sampled secret key (K_1, K_2) or with results from an ideal on-line permutation $P \twoheadleftarrow \mathsf{OPerm}_n$ (or P^{-1}). W.l.o.g., assume that $\mathbf{A}$ is deterministic and asks no queries to which it already knows the answer. We apply a common strategy for handling bad events from both worlds: in the real world, all secrets, i.e., the key, are revealed to the adversary $\mathbf{A}$ *after* it finished its interaction with the available oracles and has output its decision bit regarding which world it interacted with. Similarly, in the ideal world, the oracle samples the keys independently and uniformly at random and also reveals them to **A** *after* **A** finished its interaction and has output its decision bit. So, the internal variables can be computed by **A** in both worlds afterwards.

Proof Idea We bound $\Delta_{\mathbf{A}}(\mathcal{E}_K, \mathcal{D}_K; P, P^{-1})$ using a game-based approach. We define a sequence of games G_1 through G_4, for which the definitions of G_2 and G_3 are provided in Algorithm 3. We omit a separate definition of G_1 since its oracles $\mathcal{E}$ and $\mathcal{D}$ function identical to those in Algorithm 1, and we omit that of G_4 since it models the ideal on-line permutation. The second game G_2 will differ from G_1 in the sense that G_2 will keep track of the adversary's queries, and will replace the concrete used block cipher $\widetilde{E}$, $\widetilde{D}$ by a uniformly at random sampled tweakable permutation $\widetilde{\pi} \twoheadleftarrow \mathsf{TPerm}(\mathcal{T}, \{0,1\}^n)$ and $\widetilde{D}$ by $\widetilde{\pi}^{-1}$, where $\widetilde{\pi}$ and $\widetilde{\pi}^{-1}$ are defined by lazy sampling. Moreover, G_2 will define several bad events, although they will not affect the outputs.

Algorithm 2 Procedure INITIALIZE used in all games

```
1: procedure INITIALIZE
2:   $\mathcal{Q} \leftarrow \emptyset$
3:   $\mathsf{bad}_i \leftarrow \mathsf{false}$ for $1 \le i \le 4$
4:   $b \twoheadleftarrow \{0,1\}$
5:   if $b = 1$ then $(K_1, K_2) \twoheadleftarrow \mathcal{K}$
6:   else $P \twoheadleftarrow \mathsf{OPerm}_n$
7:   $X_0 \leftarrow \mathrm{const}_x$; $Y_0 \leftarrow \mathrm{const}_y$
8:   $\mathcal{B} \leftarrow \{(X_0, Y_0)\}$

21: function $\widetilde{\mathcal{E}}(M)$
22:   if $b = 1$ then
23:     return $\mathcal{E}_{K_1,K_2}(M)$
24:   return $P(M)$

31: function $\widetilde{\mathcal{D}}(C)$
32:   if $b = 1$ then
33:     return $\mathcal{D}_{K_1,K_2}(C)$
34:   return $P^{-1}(C)$
```

Subsequently, we will describe a third game G_3, which will behave differently from G_2 iff those bad events occur. Finally, we will describe a Game G_4, which models an ideal on-line permutation, i.e., the encryption and decryption oracles of G_4 are defined by $P \twoheadleftarrow \mathsf{OPerm}_n$ and P^{-1}, respectively. The function INITIALIZE provided in Algorithm 2 is identical for all games. Since it holds that

$$\underset{\mathbf{A}}{\Delta}(G_1; G_4) \le \underset{\mathbf{A}}{\Delta}(G_1; G_2) + \underset{\mathbf{A}}{\Delta}(G_2; G_3) + \underset{\mathbf{A}}{\Delta}(G_3; G_4),$$

we can upper bound the distance between G_1 and G_4 by successively bounding the distance between every pair of two subsequent games.

Upper bound of $\Delta_{\mathbf{A}}(G_1; G_2)$ G_2 keeps track of the adversary's queries by storing them in a set $\mathcal{Q}$. Moreover, G_2 stores the chaining values (X_i, Y_i) into a set $\mathcal{B}$. Storing them allows to invoke an oracle LLCP_n that returns the longest common prefix p, i.e., the maximum number of common starting blocks p that the current plaintext M shares with any plaintext of previous queries. Analogously, we define an oracle LLCP_n^C which, given the current ciphertext query C, always returns the longest common prefix p of the current ciphertext C with any ciphertext of previous queries.

We define a function domain which takes $\widetilde{\pi}[V_i]$ and returns the combined set of all previously occurred inputs to $\widetilde{\pi}[V_i]$ and all previously occurred outputs of $\widetilde{\pi}[V_i]^{-1}$ over all queried blocks by **A**. Similarly, range takes $\widetilde{\pi}[V_i]$ and returns the combined set of all previously occurred outputs of $\widetilde{\pi}[V_i]$ and all previously occurred inputs to $\widetilde{\pi}[V_i]^{-1}$ over all blocks of **A**'s queries. We further define two functions for codomain and corange:

$\overline{\mathsf{domain}}(\widetilde{\pi}[V_i]) := \{0,1\}^n \setminus \mathsf{domain}(\widetilde{\pi}[V_i])$, $\overline{\mathsf{range}}(\widetilde{\pi}[V_i]) := \{0,1\}^n \setminus \mathsf{range}(\widetilde{\pi}[V_i])$ for all $V_i \in \{0,1\}^\tau$. Moreover, recall that for a set $\mathcal{B} \subseteq \mathcal{A}$, the coset of $\mathcal{B}$ is defined as $\overline{\mathcal{B}} := \mathcal{A} \setminus \mathcal{B}$. Finally, we define a function $\mathsf{find} : (\{0,1\}^n)^2 \times \mathcal{T} \to (\{0,1\}^n)^2$

$$\mathsf{find}(\mathcal{B}, V_i) := \left(\overline{\mathsf{domain}}\,(\widetilde{\pi}[V_i]) \times \overline{\mathsf{range}}\,(\widetilde{\pi}[V_i])\right) \cap \overline{\mathcal{B}}.$$

Since for Game G_2, none of the sets or flags affects its outputs, it holds that

$$\underset{\mathbf{A}}{\Delta}(G_1; G_2) \leq \mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{E},\widetilde{D}^{-1}}(\sigma, O(t)).$$

Algorithm 3 Games G_2 and $\boxed{G_3}$ used in the SOPRP proof of POEx$[\widetilde{E}, H]$. The code in the boxes is only part of Game G_3

```
function $\mathcal{E}(M)$
  $p \leftarrow \mathrm{LLCP}_n(M, \mathcal{Q})$
  $m \leftarrow |M|/n$
  $(M_1, \ldots, M_m) \xleftarrow{n} M$
  for $i \leftarrow 1$ to $p$ do
    $(C_i, X_i, Y_i) \leftarrow \mathrm{Enc}(M_i, X_{i-1}, Y_{i-1})$
  for $i \leftarrow p+1$ to $m$ do
    $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$
    $X_i \leftarrow M_i \oplus W_i$
    if $X_i \in \mathsf{domain}\,(\widetilde{\pi}[V_i])$ then
      $\mathsf{bad}_1 \leftarrow \mathsf{true}$
      $\boxed{X_i \twoheadleftarrow \overline{\mathsf{domain}}\,(\widetilde{\pi}[V_i])}$
    $Y_i \twoheadleftarrow \overline{\mathsf{range}}\,(\widetilde{\pi}[V_i])$
    if $(X_i, Y_i) \in \mathcal{B}$ then
      $\mathsf{bad}_2 \leftarrow \mathsf{true}$
      $\boxed{(X_i, Y_i) \twoheadleftarrow \mathsf{find}(\mathcal{B}, V_i)}$
    $\widetilde{\pi}[V_i](X_i) \leftarrow Y_i$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(X_i, Y_i)\}$
    $C_i \leftarrow Y_i \oplus W_i$
  $C \leftarrow (C_1 \parallel \cdots \parallel C_m)$
  $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(M, C)\}$
  return $C$

function $\mathcal{D}(C)$
  $p \leftarrow \mathrm{LLCP}^C_n(C, \mathcal{Q})$
  $m \leftarrow |C|/n$
  $(C_1, \ldots, C_m) \xleftarrow{n} C$
  for $i \leftarrow 1$ to $p$ do
    $(M_i, X_i, Y_i) \leftarrow \mathrm{Dec}(C_i, X_{i-1}, Y_{i-1})$
  for $i \leftarrow p+1$ to $m$ do
    $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$
    $Y_i \leftarrow C_i \oplus W_i$
    if $Y_i \in \mathsf{range}\,(\widetilde{\pi}[V_i])$ then
      $\mathsf{bad}_3 \leftarrow \mathsf{true}$
      $\boxed{Y_i \twoheadleftarrow \overline{\mathsf{range}}\,(\widetilde{\pi}[V_i])}$
    $X_i \twoheadleftarrow \overline{\mathsf{domain}}\,(\widetilde{\pi}[V_i])$
    if $(X_i, Y_i) \in \mathcal{B}$ then
      $\mathsf{bad}_4 \leftarrow \mathsf{true}$
      $\boxed{(X_i, Y_i) \twoheadleftarrow \mathsf{find}(\mathcal{B}, V_i)}$
    $\widetilde{\pi}[V_i]^{-1}(Y_i) \leftarrow X_i$
    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(X_i, Y_i)\}$
    $M_i \leftarrow X_i \oplus W_i$
  $M \leftarrow (M_1 \parallel \cdots \parallel M_m)$
  $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{(M, C)\}$
  return $M$

function $\mathrm{Enc}(M_i, X_{i-1}, Y_{i-1})$
  $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$
  $X_i \leftarrow M_i \oplus W_i$
  $Y_i \leftarrow \widetilde{\pi}[V_i](X_i)$
  $C_i \leftarrow Y_i \oplus W_i$
  return $(C_i, X_i, Y_i)$

function $\mathrm{Dec}(C_i, X_{i-1}, Y_{i-1})$
  $(W_i, V_i) \leftarrow H_{K_2}(X_{i-1}, Y_{i-1})$
  $Y_i \leftarrow C_i \oplus W_i$
  $X_i \leftarrow \widetilde{\pi}[V_i]^{-1}(Y_i)$
  $M_i \leftarrow X_i \oplus W_i$
  return $(M_i, X_i, Y_i)$
```

Upper bound of $\Delta_{\mathbf{A}}(G_2; G_3)$ The third game G_3 differs from G_2 as follows: it resamples internal values if bad events occur in blocks beyond a common prefix:

- In the encryption oracle, bad_1 is set to true if both tweak and the cipher input for the current block repeat, i.e., there exist $i, j \geq 0$: $(V_i, X_i) = (V'_j, X'_j)$.
- In the encryption oracle, bad_2 is set to true if the pair of chaining values repeats, i.e., there exist $i, j \geq 0$: $(X_i, Y_i) = (X'_j, Y'_j)$.
- In the decryption oracle, bad_3 is set to true if both tweak and the cipher input for the current block repeat, i.e., there exist $i, j \geq 0$: $(V_i, Y_i) = (V'_j, Y'_j)$.
- In the decryption oracle, bad_4 is set to true if the pair of chaining values repeats, i.e., there exist $i, j \geq 0$: $(X_i, Y_i) = (X'_j, Y'_j)$.

The outputs of Game G_3 differ from those of Game G_2 only if a bad flag is set, i.e., they are identical until bad. We let $\mathbf{A}$ win if G_3 sets any of the bad flags. It follows from the fundamental lemma of game playing [9] that

$$\underset{\mathbf{A}}{\Delta}(G_2; G_3) \leq \Pr\left[G_3(\mathbf{A}) \text{ sets } \mathsf{bad}\right].$$

Thus, we bound the probability that Game G_3 sets bad. It holds that $\Pr[\mathsf{bad}] \leq \Pr[\mathsf{bad}_1 \vee \ldots \vee \mathsf{bad}_4]$. In the following, we consider two fixed blocks M_i, M'_j of two queries M, M'. First, we consider the probability that bad_1 gets set, that is the probability that both tweak V_i and chaining value X_i collide: $(V_i, X_i) = (V'_j, X'_j)$. We are interested in the probability

$$\begin{aligned}\Pr[\mathsf{bad}_1] &\leq \Pr\left[(V_i = V'_j) \wedge (X_i = X'_j)\right]\\ &= \Pr\left[(V_i \oplus V'_j = 0^\tau) \wedge (W_i \oplus W'_j = M_i \oplus M'_j)\right].\end{aligned}$$

If the i-th and j-th blocks would be located directly after the common prefix of M and M', it would automatically hold that $V_i = V'_j$; however, it would also hold that $W_i = W'_j$ and $M_i \neq M'_i$. Hence, it can never follow that $X_i = X'_j$. So, we consider the probability beyond being directly after a common prefix. This implies that $(i, j) \neq (1, 1)$, i.e., at least one of the indices must be greater than one; otherwise, the considered blocks M_i and M'_j would be directly after the (empty) common prefix. W.l.o.g., we assume $i > 1$. Then, Y_i is the result of $\widetilde{\pi}[V_{i-1}](\cdot)$ and is therefore sampled at random from a set of size at least $r \geq 2^n - \sigma \geq 2^{n-1}$. Since H is an $(n, \tau, 2^{n-1}, \epsilon)$-CPAXU family of hash functions, it holds for a fixed pair of blocks that

$$\Pr[\mathsf{bad}_1] = \Pr\left[(V_i \oplus V'_j = 0^\tau) \wedge (W_i \oplus W'_j = M_i \oplus M'_j)\right] \leq \epsilon,$$

and $\Pr[\mathsf{bad}_1] \leq \epsilon \cdot \sigma^2/2$ over all queries of at most σ blocks.

Next, we consider the probability that bad_2 gets set, that is the probability that a pair of chaining values repeats: $(X_i, Y_i) = (X'_j, Y'_j)$. If the tweaks $V_i = V'_j$ also collide, it is easy to see that the permutation $\widetilde{\pi}[V_i]$ maps equal inputs $X_i = X'_j$ to equal outputs $Y_i = Y'_j$. Though, this case is already covered by the probability that bad_1 gets set, and we can focus on the case that chaining values repeat for different tweaks $V_i \neq V'_j$, i.e., for two permutations. It holds that

$$\begin{aligned}\Pr[\mathsf{bad}_1 \vee \mathsf{bad}_2] &= \Pr[\mathsf{bad}_1] + \Pr[\mathsf{bad}_2] - \Pr[\mathsf{bad}_1 \wedge \mathsf{bad}_2]\\ &= \Pr[\mathsf{bad}_1] + \Pr[\neg\mathsf{bad}_1 \wedge \mathsf{bad}_2].\end{aligned}$$

We obtain

$$\Pr[\neg\mathsf{bad}_1 \wedge \mathsf{bad}_2] = \Pr\left[(V_i \neq V'_j) \wedge (X_i = X'_j) \wedge (Y_i = Y'_j)\right].$$

Again, it follows that $(i, j) \neq (1, 1)$ since otherwise, $V_i = V_j$ would hold. It can happen that one of the indices is 0; w.l.o.g., we assume $j = 0$, i.e., we consider a collision of the form $(X_i, Y_i) = (X_0, Y_0)$. Since Y_i is chosen by $\widetilde{\pi}[V_i](\cdot)$, the probability to hit Y_0 is upper bounded by $1/(2^n - \sigma)$ for a single block and $\sigma/(2^n - \sigma)$ over at most σ blocks. So, we can focus on $i, j > 0$ in the remainder, and assume w.l.o.g. $i > 1$.

Since we excluded events bad_1 to have occurred before, Y_{i-1} has been chosen by $\widetilde{\pi}[V_{i-1}](\cdot)$ randomly from a set of size at least 2^{n-1}. From the fact that H is an ϵ-CPAXU family of hash functions, it follows that

$$\Pr\left[(X_i = X'_j) \wedge (V_i \neq V'_j)\right] = \Pr\left[(W_i \oplus W'_j) = (M_i \oplus M'_j)\right] \leq 2^\tau \cdot \epsilon$$

since we allow all $2^\tau - 1$ non-zero differences $V_i \oplus V'_j$ from the assumption that bad_1 and bad_2 have not already occurred. It follows from $V_i \neq V'_j$ that $\widetilde{\pi}[V_i]$ and $\widetilde{\pi}[V'_j]$ are independent random permutations, and hence the probability that $Y_i = Y'_j$ holds is $1/(2^n - |\mathcal{B}|)$. The term $|\mathcal{B}|$ results from the fact that the game resamples (X_i, Y_i) uniformly at random from all possible chaining values which are not in $\mathcal{B}$ and which do not conflict with earlier values in the sets domain and range of $\widetilde{\pi}[V_i]$ and $\widetilde{\pi}[V'_j]$. There may be less than 2^n values to sample from, but always at least $2^n - (\sigma + 1)$ values. The '1' represents the fact that (X_0, Y_0) is already contained in $\mathcal{B}$ at the initialization. So, we can upper bound

$$\Pr[\neg\mathsf{bad}_1 \wedge \mathsf{bad}_2] \leq \frac{2^\tau \cdot \epsilon}{2^n - (\sigma + 1)}$$

for fixed query and blocks, and, over all queries of σ blocks follows that

$$\Pr[\neg\mathsf{bad}_1 \wedge \mathsf{bad}_2] \leq \frac{\sigma^2}{2} \cdot \frac{2^\tau \cdot \epsilon}{2^n - (\sigma + 1)} + \frac{\sigma}{2^n - \sigma}.$$

Since bad_3 and bad_4 represent events similar to bad_1 and bad_2 (only in decryption direction), it also follows that $\Pr[\mathsf{bad}_3] \leq \epsilon \cdot \sigma^2/2$ and $\Pr[\neg\mathsf{bad}_3 \wedge \mathsf{bad}_4] \leq (\sigma^2/2) \cdot 2^\tau\epsilon/(2^n - (\sigma + 1)) + \sigma/(2^n - \sigma)$. Hence, it holds that

$$\begin{aligned}\Pr[\mathsf{bad}] &\leq 2\left(\frac{\sigma^2}{2}\epsilon + \frac{\sigma^2}{2}\epsilon \cdot \frac{2^\tau}{2^n - (\sigma + 1)} + \frac{\sigma}{2^n - \sigma}\right)\\ &\leq \sigma^2\epsilon\left(1 + \frac{2^\tau}{2^n - (\sigma + 1)}\right) + \frac{2\sigma}{2^n - \sigma}.\end{aligned}$$

Upper bound of $\mathbf{\Delta_A(G_3; G_4)}$ We let the adversary already win if G_3 has set bad. It remains to consider the setting where G_3 does not set bad for the current query. We consider a pair of two distinct queries $Q = (M, C)$, $Q' = (M', C') \in \mathcal{Q}$ with $m = |M|/n$ and $m' = |M'|/n$. W.l.o.g., we assume $m \geq m'$ and denote by $p = \mathrm{LLCP}_n(M, M')$ the length of their common prefix. We study the difference in the outputs of Game G_3 and an ideal on-line permutation of Game G_4 for three cases: (C1) for output blocks in a common prefix, $1 \leq i \leq p$, (C2) for the diverging block, i.e., $i = p + 1$, and (C3) for output blocks after the diverging block, i.e., $i > p + 1$.

Case C1: Behavior in the Common Prefix In Game G_4, P is a deterministic on-line cipher. In Game G_3, the output blocks in the common prefix are chosen by an ideal tweaked permutation $\widetilde{\pi}[V_i](X_i)$ (or $\widetilde{\pi}[V_i]^{-1}(Y_i)$). For the first block, the tweak V_1 is always fixed. Hence, if $M_1 = M'_1$ holds for two queries, it follows from the definition of POEx that $C_1 = C'_1$. Clearly, the chaining values for the next block, (X_1, Y_1) and (X'_1, Y'_1) are also identical for both queries; since H is deterministic, the tweaks for the next block must match, i.e., $(W_2, V_2) = (W'_2, V'_2)$. In general, for all $i \leq p$, the inputs $M_i = M'_i$ (or the outputs $C_i = C'_i$) are identical in the prefix, and the tweaks and therefore the inputs to $\widetilde{\pi}[V_i]$ are also identical since $X_{i-1} = X'_{i-1}$ and $Y_{i-1} = Y'_{i-1}$. Hence, the advantage of **A** to distinguish both games is zero in this case. A similar argument holds for the decryption.

Case C2: Behavior Directly after the Common Prefix Since $M_i = M'_i$ for all $1 \leq i \leq p$, it follows that $V_{p+1} = V'_{p+1}$ and $W_{p+1} = W'_{p+1}$. So, it follows from $M_{p+1} \neq M'_{p+1}$ that

$X_{p+1} \neq X'_{p+1}$ and therefore $Y_{p+1} \neq Y'_{p+1}$ since $\widetilde{\pi}[V_p]$ is a permutation. Consequently, it follows that

$$C_{p+1} = (W_{p+1} \oplus Y_{p+1}) \neq (W'_{p+1} \oplus Y'_{p+1}) = C'_{p+1}.$$

So, C_{p+1} and C'_{p+1} always differ in both games. A similar argument holds when concerning decryption. Again, the advantage of **A** to distinguish both games is zero in this case.

Case C3: Behavior after the $(p+1)$-th Block In Game G_4, each output block after a common prefix is chosen uniformly at random from $\{0, 1\}^n$. Let $\mathcal{V}_i$ denote a set that collects all chaining-value pairs (X_i, Y_i) for each occurring tweak $V_i \in \{0, 1\}^\tau$. In the real world, each output block is chosen uniformly at random from $\{0, 1\}^n \setminus \mathcal{V}_i$ for the respective current tweak V_i. We concern two subcases: (3.1) blocks, for which all the tweaks $V_i \neq V'_j$ are distinct, and (3.2) blocks, for which the corresponding tweaks collide, i.e., $V_i = V'_j$ for two message blocks M_i and M'_j. Recall that the interest of this subcase limits to blocks that are not part of a common prefix nor follow directly after a common prefix, plus it assumes that no bad flags are set, i.e., no simultaneous collision in tweaks $V_i = V'_j$ *and* inputs $X_i = X'_j$ (or $Y_i = Y'_j$ in decryption direction) occurs.

Subcase 3.1 When the tweaks $V_i \neq V'_j$ are distinct, $\widetilde{\pi}[V_i]$ and $\widetilde{\pi}[V'_j]$ define independent permutations. Therefore, there is no difference in the behavior between both games, i.e., the distinguishing advantage of **A** is zero in this subcase.

Subcase 3.2 If Game G_3 does not set bad, the tuples (X_i, Y_i) are always distinct for blocks that are not part of a longest common prefix. In the remainder of this subcase, we make the adversary **A** stronger than it is by giving it control over the pairs (X_{i-1}, Y_{i-1}) with the restriction that $(X_{i-1}, Y_{i-1}) \neq (X'_{j-1}, Y'_{j-1})$ always holds. So, **A** is able to choose M_i, X_{i-1}, and Y_{i-1}, and must distinguish POEx with XTX$[\widetilde{\pi}, H]$ from P only by observing the outputs C_i (and similarly, chooses C_i, X_{i-1}, and Y_{i-1}, and observes M_i in decryption direction). It is easy to see that, due to Theorem 1, we can upper bound its advantage in this subcase by

$$\mathbf{Adv}^{\text{STPRP}}_{\text{XTX}[\widetilde{\pi},H],\text{XTX}[\widetilde{\pi}^{-1},H]^{-1}}(\sigma, O(t)) \leq \epsilon \cdot \sigma^2.$$

The bound in Theorem 2 follows then from adding up all individual terms. □

Algorithm 4 Birthday Attack on TC2 when instantiated with XTX.

```
1: procedure
2:   Choose $\widehat{M} \in \{0,1\}^n$ arbitrarily
3:   Construct $M = (M_1, \ldots, M_m)$ where $M_i = \widehat{M}$ for all $i \in 1, \ldots, m$
4:   Ask its corresponding ciphertext $C = (C_1, \ldots, C_m)$
5:   if for some $i, j \in \{1, \ldots, m\}$, $i \neq j$: $(C_i, C_{i+1}) = (C_j, C_{j+1})$ then
6:     return $b' = 1$
7:   return $b' = 0$
```

6 Discussion and conclusion

Efficient tweakable block ciphers The TWEAKEY [20] framework is an efficient key-scheduling approach for substitution-permutation networks where tweak and key words are treated equally, and are mixed into a so-called tweakey. Three tweakable block ciphers have been published in [20] alongside TWEAKEY: JOLTIK-BC, a lightweight cipher with 64-bit state and tweakey sizes of 128 and 192 bits, DEOXYS-BC, is AES-like with a 128-bit state, and tweakey sizes of 256 and 384 bits, and KIASU-BC, a 128-bit cipher with 64-bit tweak that does not follow the TWEAKEY approach, but is identical to the AES if the tweak is zeroed. Recently, Beierle et al. proposed SKINNY [6], another AES-like tweakable block ciphers with a lightweight linear layer, also treating tweak and key as combined tweakey. The specification of SKINNY proposes state sizes of $n \in \{64, 128\}$ bits, and tweakey sizes of n, $2n$, and $3n$ bits. Hence, DEOXYS-BC-256 or SKINNY-$n/2n$ could appear as an appropriate choice for instantiation. Though, the question which tweakable block cipher is suited depends on the application setting. Less restricted environments may also allow wide-block tweakable ciphers such as ThreeFish [16]. Note that POEX can also be instantiated with a tweakable block cipher based on a classical block cipher, e.g., the well-known LRW/XEX designs [22, 31]. Though, the term $\mathbf{Adv}^{\mathrm{STPRP}}_{\widetilde{E},\widetilde{D}}(\sigma, O(t))$ limits their security to the birthday bound, whereas designs such as, e.g., [26, 33] can achieve beyond-birthday-bound security.

The bound A bound with $2^\tau/2^n$ might look bad for tweakable block ciphers with larger tweak length than state length, i.e., $\tau \gg n$, as it is the case for, e.g., DEOXYS-BC-384 and SKINNY-64/192. However, we stress that desirable universal hash functions should provide a CPAXU bound of $\epsilon \leq c/2^{\tau+n}$, for some small c. Thus, a bound of $2\sigma^2\epsilon \cdot \left(1 + 2^{\tau-n}\right)$ will then become $(2c\sigma^2)(1/2^{\tau+n} + 1/2^{2n})$, and provide security for up to $O(2^{(\tau+n)/2})$ message blocks encrypted under the same key.

Comparison to TC2 POEX seems similar to TC2 by Rogaway and Zhang [32] if TC2 would be instantiated with the XTX [28] tweak-domain extender by Minematsu and Iwata as black box. However, such a generic instantiation could yield only birthday-bound security given a simple OPRP attack as in Algorithm 4. Assume, the adversary chooses the same value for all blocks in a long message. Then, if some pair of ciphertext blocks $C_j = C'_{j'}$ collides, the collision will continue to the subsequent blocks $C_{j+1} = C'_{j'+1}$ etc. since C_{j+1} depends only on (M_j, C_j, M_{j+1}) and $C'_{j'+1}$ only on $(M'_{j'}, C'_{j'}, M'_{j'+1})$ which is equal to (M_j, C_j, M_{j+1}). Since the probability of such a collision is about $\sigma^2/2^{n+1}$, it limits the OPRP security to the birthday bound.

In contrast to TC2, POEX avoids to directly use the plain- and ciphertext blocks as chaining values, but only their XOR with two pseudorandom internal values that the adversary cannot fully control. Hence, two n-bit values have to match to get a collision in the chaining values that would lead to a similar distinguisher as above.

Extension to on-line authenticated encryption The proposed on-line cipher can be extended to an on-line authenticated encryption scheme, when combined with a MAC that achieves beyond-birthday-bound security, e.g., PMAC2X [23], which can yield a $2n$-bit output that can be used as initial value for POEX. Since it is desirable to use only a single key for the tweakable block cipher, a common approach is to introduce domains into the tweak such that the tweakable block cipher defines domain-separated distinct permutations for encrypting full message blocks, processing a partial final message block, the associated data, the nonce, as well as for computing the authentication tag. However, the details of defining those operations have to be handled with care since they still need to provide beyond-birthday-bound security.

Conclusion This work proposed POEX, an on-line cipher with beyond-birthday-bound security that represents a chained construction of Iwata and Minematsu's XTX domain extender for tweakable block ciphers. Our construction appears well-suited for environments where parallelism plays a subordinate role but where security beyond the birthday bound is essential due to small cipher block sizes and/or rare key changes. Moreover, POEX may serve as a starting point towards more secure on-line nonce-misuse-robust AE schemes. In this domain, all known existing schemes provide authenticity only up to the birthday bound, which differs significantly from the expected security of standardized schemes such as OCB and GCM. While there exist straight-forward approaches to transform an on-line cipher into such a scheme (e.g. [17]), it remains an interesting future work to design a beyond-birthday-secure on-line AE scheme from POEX which needs efficient complementing algorithms for processing the associated data and deriving the tag. We defined POEX in a generic manner to allow the choice of an appropriate family of hash functions and a tweakable block cipher.

Acknowledgements The authors would like to thank the anonymous reviewers of the ArcticCrypt 2016 and of the journal for Cryptography and Communications for their very helpful comments and suggestions. We would thank in particular Mridul Nandi and Ashwin Jha for pointing out that subsequent inputs to the hash function are dependent, which helped to improve our work.

References

1. Abed, F., Forler, C., McGrew, D., List, E., Fluhrer, S., Lucks, S., Wenzel, J.: Pipelineable on-line encryption. In: Cid, C., Rechberger, C. (eds.) FSE, volume 8540 of Lecture Notes in Computer Science, pp. 205–223. Springer (2014)
2. Andreeva, E., Bogdanov, A., Datta, N., Luykx, A., Mennink, B., Nandi, M., Tischhauser, E., Yasuda, K.: COLM v1. http://competitions.cr.yp.to/caesar-submissions.html (2016)
3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: How to securely release unverified plaintext in authenticated encryption. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT (1), volume 8873 of LNCS, pp. 105–125. Springer (2014)
4. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K., Sarkar, P.: Parallelizable and authenticated online ciphers. In: Sako, K. (ed.) ASIACRYPT (1), vol. 8269, pp. 424–443. Springer (2013)
5. Andreeva, E., Luykx, A., Mennink, B., Yasuda, K.: COBRA: A parallelizable authenticated online cipher without block cipher inverse. In: Cid, C., Rechberger, C. (eds.) FSE, volume 8540 of LNCS, pp. 187–204. Springer (2014)
6. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO II, volume 9815 of LNCS, pp. 123–153. Springer (2016)
7. Bellare, M., Boldyreva, A., Knudsen, L.R., Namprempre, C.: Online ciphers and the Hash-CBC construction. In: Kilian, J. (ed.) CRYPTO, volume 2139 of Lecture Notes in Computer Science, pp. 292–309. Springer (2001)
8. Bellare, M., Rogaway, P.: Code-based game-playing proofs and the security of triple encryption. IACR Cryptol ePrint Archive **2004**, 331 (2004)
9. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT, volume 4004 of LNCS, pp. 409–426. Springer (2006)
10. Bernstein, D.: Caesar: Competition for authenticated encryption: Security, applicability, and robustness. https://competitions.cr.yp.to/caesar.html, Version2016.08.15
11. Bhaumik, R., Mridul, N.: OleF: An inverse-free online cipher. Trans Symmetric Cryptol Issue **2016**(2), 30–51 (2016)

12. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y.annick., Vikkelsoe, C.: PRESENT: An ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES, volume 4727 of LNCS, pp. 450–466. Springer (2007)
13. Boldyreva, A., Taesombut, N.: Online encryption schemes: New security notions and constructions. In: Okamoto, T. (ed.) CT-RSA, volume 2964 of Lecture Notes in Computer Science, pp. 1–14. Springer (2004)
14. Datta, N., Nandi, M.: ELmD. http://competitions.cr.yp.to/caesar-submissions.html (2014)
15. Datta, N., Nandi, M., Susilo, W., Mu, Y.: ELmE: A misuse resistant parallel authenticated encryption. In: ACISP, volume 8544 of Lecture Notes in Computer Science, pp. 306–321. Springer (2014)
16. Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The skein hash function family. Submission to NIST (Round 3) (2010)
17. Fleischmann, E., Forler, C., Lucks, S.: McOE: A family of almost foolproof on-line authenticated encryption schemes. In: Canteaut, A. (ed.) FSE, volume 7549 of LNCS, pp. 196–215. Springer (2012)
18. Halevi, S., Rogaway, P.: A parallelizable enciphering mode. In: Okamoto, T. (ed.), pp. 292–304. Springer (2004)
19. ISO/IEC. ISO/IEC 29192-2:2012, Information technology ? Security techniques ? Lightweight cryptography ? Part 2: Block ciphers, 2012
20. Jean, J., Nikolic, I., Peyrin, T., Sarkar, P., Iwata, T.: Tweaks and keys for block ciphers: The TWEAKEY framework. In: ASIACRYPT (2), volume 8874 of Lecture Notes in Computer Science, pp. 274?-288 (2014)
21. Krovetz, T., Rogaway, P.: The software performance of authenticated-encryption modes. In: Joux, A. (ed.) FSE, volume 6733 of Lecture Notes in Computer Science, pp. 306–327. Springer (2011)
22. Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. In: Yung, M. (ed.) CRYPTO, volume 2442 of Lecture Notes in Computer Science, pp. 31–46. Springer (2002)
23. List, E., Nandi, M.: Revisiting full-PRF-secure PMAC and using it for beyond-birthday authenticated encryption. In: Handschuh, H. (ed.) CT-RSA, volume 10159 of LNCS, pp. 258–274. Springer (2017)
24. Lu, J.: On the security of the COPA and marble authenticated encryption algorithms against (almost) universal forgery attack. IACR Cryptology ePrint Archive **2015**, 79 (2015)
25. McGrew, D., Viega, J.: The Galois/Counter Mode of Operation (GCM). Submission to NIST. http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-spec.pdf (2004)
26. Mennink, B.: Optimally secure tweakable blockciphers. In: Leander, G. (ed.) FSE, volume 9054 of Lecture Notes in Computer Science, pp. 428–448. Springer (2015)
27. Minematsu, K.: Parallelizable rate-1 authenticated encryption from pseudorandom functions. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT, volume 8441 of Lecture Notes in Computer Science, pp. 275–292. Springer (2014)
28. Minematsu, K., Iwata, T.: Tweak-length extension for tweakable blockciphers. In: Groth, J. (ed.) IMA International Conference, volume 9496 of Lecture Notes in Computer Science, pp. 77–93. Springer (2015)
29. Nandi, M.: A simple security analysis of hash-cbc and a new efficient one-key online cipher. Cryptology ePrint Archive, Report 2007/158 (2007)
30. Nandi, M.: Two new efficient CCA-secure online ciphers: MHCBC and MCBC. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT, volume 5365 of LNCS, pp. 350–362. Springer (2008)
31. Rogaway, P.: Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In: ASIACRYPT, volume 3329 of Lecture Notes in Computer Science, pp. 16–31. Springer (2004)
32. Rogaway, P., Zhang, H.: Online ciphers from tweakable blockciphers. In: CT-RSA, volume 6558 of Lecture Notes in Computer Science, pp. 237–249. Springer (2011)
33. Wang, L., Guo, J., Zhang, G., Zhao, J., Gu, D.: How to build fully secure tweakable blockciphers from classical blockciphers. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT (1), volume 10031 of LNCS, pp. 455?-483 (2016)
34. Young, E.A., Hudson, T.J.: OpenSSL: The Open Source toolkit for SSL/TLS. http://www.openssl.org/ (2011)