How to Build Fully Secure Tweakable Blockciphers from Classical Blockciphers

Lei Wang[1,4(✉)], Jian Guo[2], Guoyan Zhang[3], Jingyuan Zhao[4], and Dawu Gu[1]

[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
wanglei_hb@sjtu.edu.cn, dwgu@sjtu.edu.cn
[2] Nanyang Technological University, Singapore, Singapore
guojian@ntu.edu.sg
[3] School of Computer Science and Technology, Shandong University, Jinan, China
guoyanzhang@sdu.edu.cn
[4] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
jingyuanzhao@live.com

Abstract. This paper focuses on building a tweakable blockcipher from a classical blockcipher whose input and output wires all have a size of n bits. The main goal is to achieve full 2^n security. Such a tweakable blockcipher was proposed by Mennink at FSE'15, and it is also the only tweakable blockcipher so far that claimed full 2^n security to our best knowledge. However, we find a key-recovery attack on Mennink's proposal (in the proceeding version) with a complexity of about $2^{n/2}$ adversarial queries. The attack well demonstrates that Mennink's proposal has at most $2^{n/2}$ security, and therefore invalidates its security claim. In this paper, we study a construction of tweakable blockciphers denoted as $\widetilde{\mathbb{E}}[s]$ that is built on s invocations of a blockcipher and additional simple XOR operations. As proven in previous work, at least two invocations of blockcipher with linear mixing are necessary to possibly bypass the birthday-bound barrier of $2^{n/2}$ security, we carry out an investigation on the instances of $\widetilde{\mathbb{E}}[s]$ with $s \geq 2$, and find 32 highly efficient tweakable blockciphers $\widetilde{E}1$, $\widetilde{E}2$, ..., $\widetilde{E}32$ that achieve 2^n provable security. Each of these tweakable blockciphers uses two invocations of a blockcipher, one of which uses a tweak-dependent key generated by XORing the tweak to the key (or to a secret subkey derived from the key). We point out the provable security of these tweakable blockciphers is obtained in the ideal blockcipher model due to the usage of the tweak-dependent key.

Keywords: Tweakable blockcipher · Full security · Ideal blockcipher · Tweak-dependent key

1 Introduction

Tweakable blockcipher, formalized by Liskov *et al.* [34,35], introduces an additional parameter called *tweak* to the classical blockcipher. More formally,

J.H. Cheon and T. Takagi (Eds.): ASIACRYPT 2016, Part I, LNCS 10031, pp. 455–483, 2016.
DOI: 10.1007/978-3-662-53887-6_17

a classical blockcipher $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ is a family of permutations on $\mathcal{M}$ indexed by a secret key $k \in \mathcal{K}$. A tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ is a family of permutations on $\mathcal{M}$, indexed by two functionally distinct parameters: a key $k \in \mathcal{K}$ that is secret and used to provide the security, and a tweak $t \in \mathcal{T}$ that is public and used to provide the variability. The tweak is assumed to be known or even controlled by the adversary. $\widetilde{E}$ is considered secure if it with a secret key k uniformly chosen from the key space $\mathcal{K}$ is indistinguishable from an ideal tweakable blockcipher $\widetilde{P} : \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$ that is a family of random permutations on $\mathcal{M}$ indexed by a public tweak $t \in \mathcal{T}$. As a more natural primitive for building modes of operation, tweakable blockcipher has found wide applications. Examples include encryption schemes [7,16,23,43,49,53], authenticated encryption [1,34,47,48], and disk encryption [24,25]. Moreover, many candidates of the ongoing cryptographic competition CAESAR [5] on authenticated encryption are based on tweakable blockciphers, e.g., Deoxys [29], Joltik [30], Scream [22], SHELL [51], etc.

There are mainly three approaches to design a tweakable blockcipher. The first one is from the scratch, including Hasty Pudding Cipher [50], Mercy [12] and Threefish (used in the hash function SKEIN [19]). Such designs usually have a drawback of lacking a security proof.

The second approach is to introduce the additional parameter tweak to generic constructions of blockcipher, including tweaking Luby-Rackoff cipher or Feistel cipher [20], tweaking Generalized Feistel cipher [44] and tweaking key-alternating cipher or (iterated) Even-Mansour [9–11,18,21,28,39]. These tweakable blockciphers except TWEAKEY framework in [28] are provably secure. In details, the designs in [11,18,20,39,44] have a provable security up to $2^{n/2}$ adversarial queries, often referred to as the *birthday-bound* security with respect to the n-bit block size of the underlying blockcipher (that is, the message space $\mathcal{M} = \{0,1\}^n$). To bypass the birthday-bound barrier and to achieve a higher security bound, Jean *et al.* proposed TWEAKEY framework [28] to construct ad-hoc tweakable blockciphers from key-alternating ciphers, and specified several TWEAKEY instances which are conjectured fully 2^n secure but lack formal security proofs. After that, Cogliati *et al.* designed several tweakable blockciphers[1] by tweaking Even-Mansour ciphers in [9,10], and these proposals are provably secure up to $2^{2n/3}$ adversarial queries.

The last and the most common approach is to start from a classical blockcipher and to use it as a black box to build a tweakable blockcipher, including LRW1 [34], LRW2 [34], variants and extensions of LRW2 such as XEX and CLRW2 [6,31,32,40,46,47], Minematsu's design [41] and Mennink's design [36]. Early proposals LRW1, LRW2, XEX and their variants [6,34,40,47] are limited to the birthday-bound security. After that, cryptographers considered the cascade of LRW2 in order to design tweakable blockciphers achieving beyond-birthday-bound security. One evaluation of LRW2 contains one invocation of a blockcipher, one invocation of a universal hash function, and each evaluation

[1] These tweakable blockciphers can be regarded as instances of TWEAKEY framework.

of LRW2 in the cascade construction requires an independent secret key. Landecker *et al.* proposed CLRW2 [32] that makes two evaluations of LRW2 (that is, two calls to a blockcipher, two invocations of a universal hash function, and two secret keys), and is proven secure up to $2^{2n/3}$ adversarial queries.[2] Lampe and Seurin analyzed the general case of the cascade of LRW2 [31]. For such a tweakable blockcipher making s evaluations of LRW2 (that is, s invocations of the underlying blockcipher and universal hash function, and s secret keys), they proved that it has a security up to $2^{sn/(s+2)}$ queries (against adaptive chosen-ciphertext adversaries), and also conjectured that its security bound can be improved to $2^{sn/(s+1)}$ queries. Therefore by increasing the integer s, these tweakable blockciphers *asymptotically* approach full 2^n security, but meanwhile the efficiency gets worse as the necessary number of blockcipher invocations, universal hash function invocations, and the necessary key size linearly increase with s. Another direction to design a tweakable blockcipher achieving beyond-birthday-bound security is to use so-called *tweak-dependent key*. Roughly speaking, a tweak-dependent key is a key of an invocation of blockcipher in a tweakable blockcipher that is generated depending on the tweak. Liskov *et al.* suggested in [34] that changing the tweak should be less costly than changing the key from the efficiency concerns. Following it, early proposals of tweakable blockcipher avoided the usage of the tweak-dependent key. However, recently Jean *et al.* [28] pointed out that this suggestion is somewhat counter-intuitive from the security concern, because the adversary has full control on the tweak, but has very limited control on the key. They suggested that the tweak and the key should be treated comparably. In fact even before Jean *et al.*'s work, Minematsu [41] proposed a tweakable blockcipher built on two invocations of blockcipher, one of which uses a tweak-dependent key. His design is proven secure up to $\max\{2^{n/2}, 2^{n-|t|}\}$ adversarial queries, where $|t|$ is the bit size of the tweak (that is, the tweak space $\mathcal{T} = \{0,1\}^{|t|}$). Hence Minematsu's design is beyond-birthday-bound secure as long as the tweak is shorter than $n/2$ bits. A scheme XTX has been proposed to extend the tweak-length of any black-box tweakable blockcipher by using a universal hash function [42]. Recently Mennink [36] proposed two tweakable blockciphers $\widetilde{F}[1]$ and $\widetilde{F}[2]$ with the usage of the tweak-dependent key. $\widetilde{F}[1]$ consists of one invocation of blockcipher and one finite-field multiplication, and is proven secure up to $2^{2n/3}$ adversarial queries. $\widetilde{F}[2]$ makes two calls to blockcipher, and is *surprisingly* proven secure up to 2^n adversarial queries, that is achieving full security with very high efficiency. On the other hand, the security proof of Mennink's designs [36] are in the ideal blockcipher (information-theoretic) model, while other proposals [6,31,32,34,40,41,47] have security proofs in the standard (complexity-theoretic) model of assuming the underlying blockcipher as a pseudorandom permutation.

Our Contributions. In this paper, we focus on constructing tweakable blockciphers that achieve full 2^n security. This is mainly motivated by the scenarios where the blockciphers only have 32-, 48- or 64-bit block size, e.g., Simon and

[2] A flaw in the original proof was found and fixed by Procter [46].

Speck family of blockciphers [3] (refer to Sect. 4.2 for more discussions). As summarized above, so far there is only one tweakable blockcipher $\widetilde{F}[2]$ designed by Mennink [36] that claims full security. As a first contribution, we present a key-recovery attack on $\widetilde{F}[2]$ with a complexity of around $2^{n/2}$ adversarial queries, which invalidates the designer's security claim in [36]. Our attack has been verified by the designer [38]. Accordingly Mennink proposed a patch [37] to $\widetilde{F}[2]$ of the proceeding version, which can resist our key-recovery attack.

This paper designs tweakable blockciphers from classical blockciphers in the black-box way, that is following the above third design approach. We focus on a construction of tweakable blockcipher (see Fig. 2 as an example) denoted as $\widetilde{\mathbb{E}}[s] : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \rightarrow \mathcal{M}$, which consists of s invocations of a blockcipher $E : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{M}$ and extra simple XOR operations. As a second and main contribution, we carry out a heuristic search to investigate the instances of $\widetilde{\mathbb{E}}[s]$, and successfully find 32 highly efficient tweakable blockciphers $\widetilde{E1}$, $\widetilde{E2}$, ..., and $\widetilde{E32}$ that achieve full 2^n security. Each of these tweakable blockcipher (see Figs. 6 and 7) makes two calls to the blockcipher E. In details, the first blockcipher call is to derive a secret subkey y from the key k such that $y = E(k, k)$, $y = E(k, 0)$ or $y = E(0, k)$. The second blockcipher call encrypts a plaintext p (or decrypts a ciphertext c) with a tweak-dependent key, which is generated by XORing the tweak t to the key k, the subkey y, or $k \oplus y$. In particular, we stress that by pre-computing and storing the subkey y, our tweakable blockciphers just need to make one blockcipher call for encrypting (t, p) or decrypting (t, c).

A comparison with previous tweakable blockciphers is detailed in Table 1. The main advantage of our designs is optimal 2^n provable security and high efficiency. From the security view, previous tweakable blockciphers except LRW2$[s]$(with $s \rightarrow \infty$) and the patched $\widetilde{F}[2]$ (in ePrint version) have (at most) $2^{2n/3}$ provable security. From the efficiency view, LRW2$[s]$ requires s blockcipher calls, and s universal hash function invocations, and hence the efficiency is significantly worse. Our designs also have an efficiency advantage compared with the patched $\widetilde{F}[2]$, as our designs require just one blockcipher call for encrypting a plaintext or decrypting a ciphertext when the subkey is pre-computed and stored.

Organization. The rest of the paper is organized as follows. Section 2 gives notations and definitions. Section 3 describes a key-recovery attack on Mennink's proposal. Section 4 presents the target construction, design goal and search strategy. We then write the search procedure and the found constructions in Sect. 5, and provide security proofs in Sect. 6. Finally we conclude the paper in Sect. 7.

2 Preliminaries

2.1 Notations

$\{0,1\}^n$ denotes the set of all n-bit strings. For $a, b \in \{0,1\}^n$, $a \oplus b$ denotes their bitwise exclusive-OR (XOR). For $a \in \{0,1\}$ and $b \in \{0,1\}^b$, $a \cdot b$ denotes the multiplication of a and b, that is equal to b if $a = 1$, and equal to 0 if

Table 1. Comparison of our designs with previous tweakable blockciphers: if we precompute and store the subkey, $\widetilde{E1}, \ldots, \widetilde{E32}$ require just one blockcipher call for encrypting a plaintext or decrypting a ciphertext.

tweakable blockciphers	key size	security ($\log_2$)	cost E	cost $\otimes/h$	tdk	reference
LRW1	n	$n/2$	1	0	N	[34]
LRW2	$2n$	$n/2$	1	2	N	[34]
XEX	n	$n/2$	1	0	N	[47]
LRW2[2]	$4n$	$2n/3$	2	2	N	[32]
LRW2[s]	$2sn$	$sn/(s+2)$	s	s	N	[31]
Min	n	$\max\{n/2, n-\lvert t\rvert\}$	2	0	Y	[41]
$\widetilde{F}$[1]	n	$2n/3$	1	1	Y	[36]
$\widetilde{F}$[2]	n	$n/2$	2	0	Y	[36]
patched $\widetilde{F}$[2]	n	n	2	0	Y	[37]
$\widetilde{E1}, \ldots, \widetilde{E32}$	n	n	2 (1)	0	Y	Sect. 5

– $\otimes/h$ stands for multiplications or universal hashes;
– tdk stands for the tweak-dependent key. 'N' refers to not using tdk, and 'Y' refers to using tdk;
– $|t|$ stands for the bit length of the tweak;

$a = 0$. For a finite set $\mathcal{X}$, $x \xleftarrow{\$} \mathcal{X}$ denotes that an element x is selected from $\mathcal{X}$ uniformly at random. $|\mathcal{X}|$ denotes the number of the elements in $\mathcal{X}$. Blockcipher is commonly denoted as $E : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$, and tweakable blockcipher as $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$, where $\mathcal{K}$ is the key space, $\mathcal{T}$ is the tweak space, and $\mathcal{M}$ is the message space. Throughout this paper, we fix $\mathcal{K} = \mathcal{T} = \mathcal{M} = \{0,1\}^n$. Let $E(k,\cdot)$ and $E^{-1}(k,\cdot)$ be the encryption and the decryption of blockcipher E with a key $k \in \mathcal{K}$ respectively. Let $E^{\pm}(k,\cdot)$ consist of both $E(k,\cdot)$ and $E^{-1}(k,\cdot)$. Sometimes we denote $E(k,\cdot)$, $E^{-1}(k,\cdot)$ and $E^{\pm}(k,\cdot)$ as $E_k(\cdot)$, $E_k^{-1}(\cdot)$ and $E_k^{\pm}(\cdot)$ respectively. Similarly we define notations $\widetilde{E}(k,\cdot,\cdot)$, $\widetilde{E}^{-1}(k,\cdot,\cdot)$, and $\widetilde{E}^{\pm}(k,\cdot,\cdot)$ for tweakable blockcipher $\widetilde{E}$, which can also be denoted as $\widetilde{E}_k(\cdot,\cdot)$, $\widetilde{E}_k^{-1}(\cdot,\cdot)$ and $\widetilde{E}_k^{\pm}(\cdot,\cdot)$, respectively. An input-output tuple of E is commonly denoted as (l,u,w) such that $w = E(l,u)$. An input-output tuple of $\widetilde{E}_k$ with $k \xleftarrow{\$} \mathcal{K}$ is denoted as (t,p,c) such that $\widetilde{E}_k(t,p) = c$. Let $\texttt{Bloc}$ be the set of all blockciphers with key space $\mathcal{K}$ and message space $\mathcal{M}$. A blockcipher E is said to be an ideal blockcipher if it is selected from $\texttt{Bloc}$ uniformly at random, that is $E \xleftarrow{\$} \texttt{Bloc}$. Let $\widetilde{\texttt{Perm}}$ be the set of all functions $\widetilde{P} : \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ such that for each $t \in \mathcal{T}$, $\widetilde{P}(t,\cdot)$ is a permutation on $\mathcal{M}$. A function $\widetilde{P}$ is said to be an ideal tweakable blockcipher if it is selected from $\widetilde{\texttt{Perm}}$ at random, that is $\widetilde{P} \xleftarrow{\$} \widetilde{\texttt{Perm}}$. Similarly we define notations $\widetilde{P}(\cdot,\cdot)$, $\widetilde{P}^{-1}(\cdot,\cdot)$ and $\widetilde{P}^{\pm}(\cdot,\cdot)$.

2.2 Tweakable Blockcipher and Security Definition

A distinguisher $\mathcal{D}$ is an algorithm that is given query access to one (or more) oracle of being either $\mathcal{O}$ or $\mathcal{Q}$, and outputs one bit. Its advantage in distinguishing these two primitives $\mathcal{O}$ and $\mathcal{Q}$ is defined as

$$\mathbf{Adv}(\mathcal{D}) = \left|\Pr\left[\mathcal{D}^{\mathcal{O}} \Rightarrow 1\right] - \Pr\left[\mathcal{D}^{\mathcal{Q}} \Rightarrow 1\right]\right|$$

A tweakable blockcipher with key space $\mathcal{K}$, tweak space $\mathcal{T}$ and message space $\mathcal{M}$ is a mapping $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ such that for any key $k \in \mathcal{K}$ and any tweak $t \in \mathcal{T}$, $\widetilde{E}(k, t, \cdot)$ is a permutation over $\mathcal{M}$. The security of a tweakable blockcipher is defined via upper bounding the advantage of distinguisher $\mathcal{D}$ in the following game. $\mathcal{D}$ is given query access to oracles $(\mathcal{O}_1, E^{\pm})$: $\mathcal{O}_1$ is either $\widetilde{E}_k^{\pm}(\cdot,\cdot)$ with $k \xleftarrow{\$} \mathcal{K}$ or an ideal tweakable blockcipher $\widetilde{P}(\cdot,\cdot) \xleftarrow{\$} \widetilde{\texttt{Perm}}$; $E^{\pm}$ is an ideal blockcipher (that is $E \xleftarrow{\$} \texttt{Bloc}$) which is used as the underlying blockcipher of $\widetilde{E}$. The advantage of $\mathcal{D}$ in distinguishing $\widetilde{E}$ and $\widetilde{P}$ is defined as

$$\mathbf{Adv}_{\widetilde{E}}^{\widetilde{\text{sprp}}}(\mathcal{D}) = \left|\Pr\left[\mathcal{D}^{\widetilde{E}_k^{\pm}(\cdot,\cdot),E^{\pm}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\mathcal{D}^{\widetilde{P}^{\pm}(\cdot,\cdot),E^{\pm}(\cdot,\cdot)} \Rightarrow 1\right]\right|,$$

where the probabilities are taken over the choices of $k \xleftarrow{\$} \mathcal{K}$, $E \xleftarrow{\$} \texttt{Bloc}$, $\widetilde{P} \xleftarrow{\$} \widetilde{\texttt{Perm}}$, and $\mathcal{D}$'s coin (if any).

Throughout the paper, we consider information-theoretic distinguisher $\mathcal{D}$ such that $\mathcal{D}$ is computationally unbounded, but sorely limited by the number of queries to its oracles. We write

$$\mathbf{Adv}_{\widetilde{E}}^{\widetilde{\text{sprp}}}(q) = \max_{\mathcal{D}}\{\mathbf{Adv}_{\widetilde{E}}^{\widetilde{\text{sprp}}}(\mathcal{D})\},$$

where the maximum is taken over all distinguisher $\mathcal{D}$ that makes at most q queries to its oracles.

A tweak-dependent key of a tweakable blockcipher is a key of an invocation of blockcipher which is generated depending on the tweak. In other words, changing the value of tweak leads to re-keying that blockcipher call. Liskov *et al.* suggested in [34] that changing the tweak should be less costly than changing the key. However, Jean *et al.* [28] pointed out that this suggestion is counter-intuitive, because the adversary has full control on the tweak, but has very limited control on the key. Indeed the tweak and the key should be treated comparably.

2.3 The H-Coefficient Technique

Our proof adopts the H-coefficient Technique [8,45], which is briefly introduced as follows. This paper considers information-theoretic distinguisher $\mathcal{D}$ that is computationally unbounded. Hence without loss of generality, we always assume $\mathcal{D}$ is deterministic. Suppose $\mathcal{D}$ interacts with $\mathcal{O}$ and $\mathcal{Q}$, and its advantage is defined in Sect. 2.2. A view v is the query-response tuples that $\mathcal{D}$ receives when interacting with $\mathcal{O}$ or $\mathcal{Q}$. Let X be the probability distribution of the view when

$\mathcal{D}$ interacts with $\mathcal{O}$, and Y be the probability distribution of the view when $\mathcal{D}$ interacts with $\mathcal{Q}$. $\mathcal{V}$ is defined as the set of all attainable views v while $\mathcal{D}$ interacting with $\mathcal{Q}$, that is $\mathcal{V} = \{v \mid \Pr[Y = v] > 0\}$.

The H-coefficient technique evaluates the upper bound of $\mathbf{Adv}(\mathcal{D})$ as follows. Firstly, partition $\mathcal{V}$ to two disjoint subsets $\mathcal{V}_{\text{good}}$ and $\mathcal{V}_{\text{bad}}$ such that $\mathcal{V} = \mathcal{V}_{\text{good}} \cup \mathcal{V}_{\text{bad}}$. Secondly, estimate a real value $\epsilon_{v_{\text{good}}}$ with $0 \leq \epsilon_{v_{\text{good}}} \leq 1$ such that for each view $v \in \mathcal{V}_{\text{good}}$, it has that

$$\frac{\Pr[X = v]}{\Pr[Y = v]} \geq 1 - \epsilon_{v_{\text{good}}}.$$

Moreover, compute the probability of $\mathcal{D}$ receiving a view from $\mathcal{V}_{\text{bad}}$ when interacting with $\mathcal{Q}$, that is $\Pr[Y \in \mathcal{V}_{\text{bad}}]$. Finally, conclude that the advantage of $\mathcal{D}$ is upper bounded as

$$\mathbf{Adv}(\mathcal{D}) \leq \epsilon_{v_{\text{good}}} + \Pr[Y \in \mathcal{V}_{\text{bad}}].$$

3 Key-Recovery Attack on Mennink's Design [36]

This section presents a key-recovery attack on Mennink's design in [36], which is depicted in Fig. 1. Let $\widetilde{E}_k : \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ to denote Mennink's tweakable blockcipher with a secret key $k \in \mathcal{K}$ and $E : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ to denote its underlying blockcipher. The key-recovery attacker has query access to $\widetilde{E}_k^{\pm}(\cdot, \cdot)$ and $E^{\pm}(\cdot, \cdot)$. The attack procedure is detailed below.

At first step, the attacker recovers the value of $E(k, 0)$ by sending one query $(0, 0)$ to $\widetilde{E}_k^{-1}(\cdot, \cdot)$ to receive a plaintext p such that $E(k, 0) = p$ holds. This is based on an observation for the case of tweak $t = 0$ and ciphertext $c = 0$.

- tweak $t = 0$ implies that the two blockcipher calls in $\widetilde{E}_k$ shares the same key value, and hence are identical permutation.
- ciphertext $c = 0$ implies that the outputs of two blockcipher calls in $\widetilde{E}_k$ are equal from $c = y_1 \oplus y_2 = 0$, that is $y_1 = y_2$.

When querying $(t = 0, c = 0)$ to $\widetilde{E}_k^{-1}(\cdot, \cdot)$, it has that $x_2 = t = 0$, and in turn the received plaintext $p = y_1 \oplus x_2 = y_1$, where y_1 is computed as $y_1 = E(k, 0)$. Hence the attacker gets the value of $E(k, 0)$ by sending one query $(0, 0)$ to $\widetilde{E}_k^{-1}(\cdot, \cdot)$.

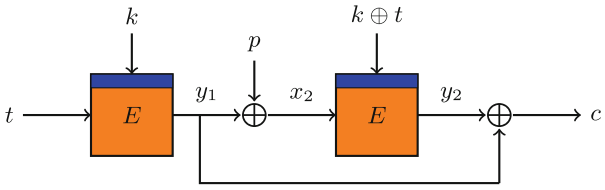

Fig. 1. Tweakable blockcipher in [36]

At second step, the attacker collects and stores a set of $E(k \oplus t, \mathsf{const})$, where const is a fixed constant, for $2^{n/2}$ distinct tweak values t by making $2^{n/2+1}$ queries to $\widetilde{E}_k(\cdot,\cdot)$. In details, for each tweak t, the attacker starts with recovering the value of $E(k,t)$ by sending one query $(0, E(k,0) \oplus t)$ to $\widetilde{E}_k(\cdot,\cdot)$ to receive ciphertext c and computing $E(k,t) = c \oplus E(k,0)$. The reason is as follows. Note $y_1 = E(k,0)$. It has $x_2 = (E(k,0) \oplus t) \oplus y_1 = t$, which implies $y_2 = E(k,t)$. Also from $c = y_1 \oplus y_2$, it has that $y_2 = c \oplus y_1 = c \oplus E(k,0)$. Hence $E(k,t)$ is equal to $c \oplus E(k,0)$. Next and with a similar reason, the attacker recovers the value of $E(k \oplus t, \mathsf{const})$ by sending one query $(t, E(k,t) \oplus \mathsf{const})$ to $\widetilde{E}_k(\cdot,\cdot)$, and computing $E(k \oplus t, \mathsf{const}) = c \oplus E(k,t)$. Overall, the attacker is able to recover the value of $E(k \oplus t, \mathsf{const})$ for any tweak t, by sending two queries to $\widetilde{E}_k(\cdot,\cdot)$.

At third and the last step, the attacker selects $2^{n/2}$ distinct values l, queries (l, const) to $E(\cdot,\cdot)$ to receive $E(l, \mathsf{const})$, and matches it to the set $\{E(k \oplus t, \mathsf{const})\}$ stored at second step. If a match is found that is $E(k \oplus t, \mathsf{const}) = E(l, \mathsf{const})$, the attacker recovers the secret key k as $k = l \oplus t$.

Now we evaluate the complexity and the success probability. The first step requires one query, the second step requires $2^{n/2+1}$ queries and the last step requires $2^{n/2}$ queries. Summing up, the total complexity is less than $2^{n/2+2}$ queries. Since there are $2^{n/2}$ distinct tweak values t and $2^{n/2}$ distinct values l, the probability of existing a value of t and a value of l such that $t \oplus l = k$ is trivially computed as $1 - (1 - 2^{-n})^{2^n} \approx 1 - 1/e \approx 0.63$. Hence the success probability of recovering the key is about 0.63. Overall, the tweakable blockcipher designed by Mennink in [36] has at most around $2^{n/2}$ security, in other words, birthday-bound security, which is exponentially far lower than the designer's claim of full 2^n security.

On proof flaw in [36]. In the proof, under the condition that the attacker cannot guess the key correctly (that is, (12a) defined in [36] is not set), it claimed that the distribution of output variable of the first blockcipher call, $y_1 = E(k,t)$, is independent from the second blockcipher call $y_2 = E(k \oplus t, x_2)$. This is a wrong claim. When tweak $t = 0$, both the two blockcipher calls share the same key, and therefore the distribution of their outputs are highly related.

4 Target Construction, Design Goal and Search Strategy

4.1 Tweakable Blockcipher $\widetilde{\mathbb{E}}[s]$

In this paper, we study a construction of tweakable blockcipher consisting of blockcipher calls and linear transformations. Furthermore, we restrict linear transformations to be just simple XOR operations for efficiency benefits. For a more generic construction of tweakable blockcipher from a classical blockcipher, we refer interested readers to [36].

We denote the target tweakable blockcipher as $\widetilde{\mathbb{E}}[s]$, which is built on s blockcipher calls. Let E denote its underlying blockcipher with n-bit block size and n-bit key size. Let k, t, p and c denote its key, tweak, plaintext and ciphertext, respectively, which are all n-bit long. Let $a_{i,j}$ and $b_{i,j}$ for $1 \le i \le s+1$ and

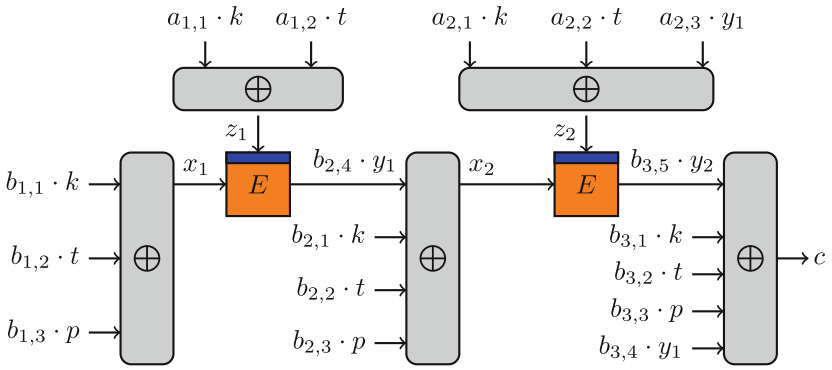

Fig. 2. Graphical view of $\widetilde{\mathbb{E}}[2]$ with key k, tweak t, plaintext p, ciphertext c, $a_{i,j} \in \{0,1\}$ and $b_{i,j} \in \{0,1\}$

Algorithm 1. Encryption of $\widetilde{E}[s](\cdot,\cdot,\cdot)$: '+' stands for addition operation in $\mathrm{GF}(2^n)$, that is XOR operation.

Input: key k, plaintext p, tweak t, blockcipher $E(\cdot,\cdot)$, one-bit variables $a_{i,j}$'s and $b_{i,j}$'s

Output: ciphertext c

1. $x_1 = b_{1,1} \cdot k + b_{1,2} \cdot t + b_{1,3} \cdot p$
2. $z_1 = a_{1,1} \cdot k + a_{1,2} \cdot t$
3. **for** $i = 1$ **to** $s - 1$, **do**
4. $\quad y_i = E(z_i, x_i)$
5. $\quad x_{i+1} = b_{i+1,1} \cdot k + b_{i+1,2} \cdot t + b_{i+1,3} \cdot p + \sum_{j=4}^{i+3} b_{i+1,j} \cdot y_{j-3}$
6. $\quad z_{i+1} = a_{i+1,1} \cdot k + a_{i+1,2} \cdot t + \sum_{j=3}^{i+2} a_{i+1,j} \cdot y_{j-2}$
7.
8. **endfor**
9. $y_s = E(z_s, x_s)$
10. $c = b_{s+1,1} \cdot k + b_{s+1,2} \cdot t + b_{s+1,3} \cdot p + \sum_{j=4}^{s+3} b_{s+1,j} \cdot y_{j-3}$
11. **return** ciphertext c

$1 \leq j \leq i+2$ be one-bit variables of being 0 or 1. The encryption procedure of $\widetilde{\mathbb{E}}[s]$ is provided in Algorithm 1. Each concrete instantiation of $\widetilde{\mathbb{E}}[s]$ is to determine the values of $a_{i,j}$'s and $b_{i,j}$'s. Moreover, a graphical view of $\widetilde{\mathbb{E}}[2]$ is depicted in Fig. 2 as an example, which is also useful for next sections. Throughout this paper, we always assume that all the s blockcipher calls are indeed involved in the computation of the ciphertext c from the key k, the tweak t and the plaintext p for $\widetilde{\mathbb{E}}[s]$.

A tweakable blockcipher must be invertible, namely plaintext p should be efficiently decrypted from key k, tweak t and ciphertext c. Such a requirement

sets a few constraints on the above construction $\widetilde{\mathbb{E}}[s]$. Firstly, plaintext p should be involved in exactly one linear transformation.

Constraint 1. *For $\widetilde{\mathbb{E}}[s]$ to be invertible, there should exist an integer $i \in \{1, 2, \ldots, s+1\}$ such that $b_{i,3} = 1$ and $b_{j,3} = 0$ for all $j \in \{1, 2, \ldots, s+1\}$ and $j \neq i$.*

Secondly, suppose plaintext p is involved in the linear transformation that outputs x_i, then the values of both x_i and y_i depend on plaintext p in the encryption process. We will call such x_i and y_i *plaintext-dependent* variables. Moreover, if y_i is used to compute some variable x_j $(j > i)$, then both x_j and y_j are also called plaintext-dependent variables. Iteratively, we have the definition below.

Definition 1. *For our target construction $\widetilde{\mathbb{E}}[s]$, internal variables x_i and y_i are said to be plaintext-dependent, if x_i is computed depending on plaintext p or a plaintext-dependent variable y_j in the encryption process. Also we include plaintext p as a plaintext-dependent variable.*

A plaintext-dependent variable cannot be used to produce any key value z_j.[3] Otherwise, the construction is not (efficiently) invertible, since one cannot compute z_j without the knowledge of plaintext p.

Constraint 2. *For $\widetilde{\mathbb{E}}[s]$ to be invertible, if an internal state y_i with $1 \leq i \leq s$ is a plaintext-dependent variable, the values of $a_{j,i+2}$'s for all $j \in \{i+1, i+2, \ldots, s\}$ must be 0.*

Moreover, the linear transformation to produce any internal state x_i with $1 \leq i \leq s$ should have at most one input plaintext-dependent variable. Otherwise, one cannot efficiently inverse such a linear transformation in the decryption, because there are more than one unknown input variable.

Constraint 3. *For $\widetilde{\mathbb{E}}[s]$ to be invertible, the linear transformations to produce internal states x_i's for all $i \in \{1, 2, \ldots, s+1\}$ must have at most one input variable that is plaintext-dependent.*

Summarizing up, an instantiation of $\widetilde{\mathbb{E}}[s]$ is efficiently invertible and therefore a valid tweakable blockcipher, as long as it satisfies the above three constraints. Nevertheless, additional conditions might be necessary from the concerns of security and efficiency. For example, it is important that all s blockcipher invocations of $\widetilde{\mathbb{E}}[s]$ are indeed involved for computing ciphertext c from the key k, the tweak t and plaintext p. Here we omit such discussions for the general case, but leave them in next sections for specific case, *e.g.*, the instances of $\widetilde{\mathbb{E}}[2]$.

[3] Recall that all blockcipher calls are indeed involved in the computation of ciphertext c from the key k, the tweak t and plaintext p.

Remarks. It is interesting to note that many tweakable blockciphers proposed previously are instances of our target construction $\widetilde{\mathbb{E}}[2]$ in Fig. 2. For example, LRW1 construction designed by Liskov *et al.* in [34] is the instance with $b_{1,3} = a_{1,1} = b_{2,4} = b_{2,2} = a_{2,1} = b_{3,5} = 1$ and 0 for the other $a_{i,j}$'s and $b_{i,j}$'s. Minematsu's construction in [41] is the instance with $b_{1,2} = a_{1,1} = b_{2,3} = a_{2,3} = b_{3,5} = 1$ and 0 for the other $a_{i,j}$'s and $b_{i,j}$'s. Mennink's construction in [36] is the instance with $b_{1,2} = a_{1,1} = b_{2,4} = b_{2,3} = a_{2,1} = a_{2,2} = b_{3,5} = b_{3,4} = 1$ and 0 for the other $a_{i,j}$'s and $b_{i,j}$'s.

4.2 Design Goal

Our first and top-priority goal is *full* 2^n *provable security*, which has both theoretical and practical interests. A typical blockcipher nowadays such as AES [14] and SIMON [3] has a block size of 128 bits or 64 bits. In some constrained environment, the block size of lightweight blockciphers can be even shorter, *e.g.*, SIMON-48 [3]. Hence tweakable blockcipher constructions with merely a birthday-bound security may not be suited for various applications. Consequently other constructions providing higher security is definitely necessary. Particularly, designing tweakable blockciphers with optimal 2^n provable security is indeed a very interesting research topic.

Our second goal is *the minimum number of blockcipher calls*, which obviously comes from the efficiency concern. For our target construction, a blockcipher call is much more time-consuming than linear transformations which are merely XOR operations. Therefore the number of blockcipher calls dominates the overall efficiency of tweakable blockcipher. Besides, we also aim to optimize the efficiency of linear transformations under the condition of no security sacrifice, *i.e.*, erasing unnecessary input variables. In fact this is also the reason that we have limited the linear transformations to simple XORing variables when choosing the target construction $\widetilde{\mathbb{E}}[s]$.

Our third goal is *(comparably) high efficiency of changing a tweak*, which in particular should be more efficient than changing a key. It is motivated by the fact that tweak is changed more frequently than the key in applications. For instance, in most modes of operation such as OCB [48], tweak is changed for every plaintext block, while the secret key can be kept the same for up to birthday-bound number of plaintext blocks. Such a criteria of designing tweakable blockcipher has been suggested by Liskov *et al.* [35] and followed by several constructions in [6,31,32,40,46,47]. However, differently from those constructions, we allow to use tweak-dependent keys, in other words, changing a tweak leads to re-keying blockcipher. This is due to the above goals of security and efficiency. Indeed as shown in [31], without using tweak-dependent keys, an (almost) optimal secure tweakable blockcipher requires an unrestrained increase of blockcipher calls and the number of keys.

4.3 Search Strategy

In order to achieve the design goals listed in Sect. 4.2, we adopt a heuristic approach to search among the instances of $\widetilde{\mathbb{E}}[s]$.

- For the goal of full 2^n security, we should investigate the instances of $\widetilde{\mathbb{E}}[s]$ with $s \geq 2$. The reason is that Mennink in [36] proved any instance of $\widetilde{\mathbb{E}}[1]$ (that is with linear mixing) has at most $2^{n/2}$ security. It implies that at least 2 blockcipher calls are necessary to possibly bypass birthday-bound barrier and to reach full 2^n security.
- For the goal of minimum number of blockcipher calls, we start with analyzing the instances of $\widetilde{\mathbb{E}}[2]$. Moreover, we will not move to investigate the instances of $\widetilde{\mathbb{E}}[s+1]$, unless we have examined all the instances of $\widetilde{\mathbb{E}}[s]$ and none of them can achieve 2^n security. Once some instance of $\widetilde{\mathbb{E}}[s]$ is found with 2^n security, it is not needed to investigate the instances of $\widetilde{\mathbb{E}}[s']$ where $s' > s$.
- For the goal of high efficiency of changing a tweak, we should use the minimum number of tweak-dependent keys. Let i denote the number of tweak-dependent keys. While searching among the instances of $\widetilde{\mathbb{E}}[s]$, we start with those with one tweak-dependent key. Moreover, we will not move to investigate the instances with $i+1$ tweak-dependent keys, unless we have examined all the instances with i tweak-dependent keys and none of them can achieve 2^n security. Once some instance of $\widetilde{\mathbb{E}}[s]$ with i tweak-dependent keys is found with 2^n security, it is not needed to investigate the instances of $\widetilde{\mathbb{E}}[s]$ with i' tweak-dependent keys, where $i' > i$.

Following the above search strategy, we start with investigating the instances of $\widetilde{\mathbb{E}}[2]$ with one tweak-dependent key, and find 32 such instances achieving full 2^n provable security. The search process is detailed in next section.

5 Search Among Instances of $\widetilde{\mathbb{E}}[2]$ with One Tweak-Dependent Key

To start with, we provide an observation that is used during the search: XORing tweak t to plaintext p and ciphertext c does not have any impact to the security of tweakable blockcipher.

Observation 1. *For a tweakable blockcipher $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$, define a set of tweakable blockcipher $\widetilde{E}[b_p, b_c] : \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ with $b_p, b_c \in \{0,1\}$ as*

$$\widetilde{E}[b_p, b_c](k, t, p) := \widetilde{E}(k, t, p \oplus (b_p \cdot t)) \oplus (b_c \cdot t),$$

for all $k \in \mathcal{K}$, $t \in \mathcal{T}$ and $p \in \mathcal{M}$. Each tweakable blockcipher $\widetilde{E}[b_p, b_c]$ provides the same security level as $\widetilde{E}$, that is $\mathbf{Adv}^{\widetilde{\text{sprp}}}_{\widetilde{E}[b_p,b_c]}(q) = \mathbf{Adv}^{\widetilde{\text{sprp}}}_{\widetilde{E}}(q)$. Thus, we do not use XORing tweak t to plaintext p and ciphertext c for (slight) efficiency benefit.

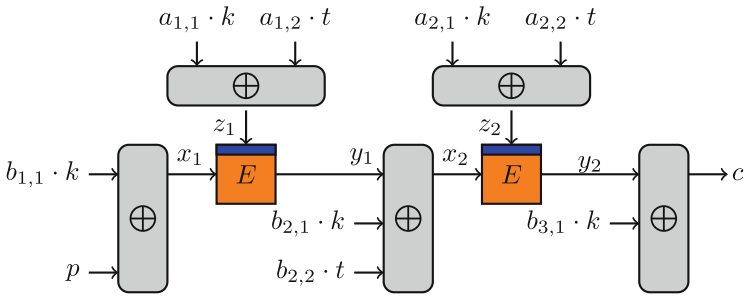

Fig. 3. Type I Constructions of $\widetilde{\mathbb{E}}[2]$

The proof of this observation is rather straightforward, and provided in full version of this paper [52].

Next, according to Constraint 1, we divide the instances of $\widetilde{\mathbb{E}}[2]$ into three types with respect to the place where the plaintext p is injected.

Type I: p is XORed to compute x_1, which sets $b_{1,3} = 1$, $b_{2,3} = 0$ and $b_{3,3} = 0$;
Type II: p is XORed to compute x_2, which sets $b_{1,3} = 0$, $b_{2,3} = 1$ and $b_{3,3} = 0$;
Type III: p is XORed to compute x_3, which sets $b_{1,3} = 0$, $b_{2,3} = 0$ and $b_{3,3} = 1$.

We search the instances of these types independently.

5.1 On the Instances of Type I

Constraint 2 sets $a_{2,3} = 0$, since y_1 is plaintext-dependent. Observation 1 sets $b_{1,2} = 0$ and $b_{3,2} = 0$. We set $b_{3,5} = 1$ such that the second blockcipher call is involved in $\widetilde{\mathbb{E}}[2]$.[4] Moreover, we set $b_{2,4} = 1$ in order to avoid overlap between the instances of Type I and of Type II, because if $b_{2,4} = 0$, the two blockcipher calls are parallel and indeed those instances are included in Type II. In turn, it implies that x_2 and y_2 are plaintext-dependent variables. Then Constraint 3 sets $b_{3,4} = 0$, because y_2 as a plaintext-dependent variable is already used to compute c. Putting all these together, the (simplified) construction of Type I is depicted in Fig. 3.

We investigate all the instances of Type I with one tweak-dependent key, which are divided into two cases depending on the position of the tweak-dependent key. More precisely, it depends on the values of $a_{1,2}$ and $a_{2,2}$.

Case (1): $\boldsymbol{a_{1,2} = 1}$ **and** $\boldsymbol{a_{2,2} = 0}$**.** z_1 is the tweak-dependent key. For these instances, the computation from internal variable x_2 to ciphertext c is that

$$c = E(a_{2,1} \cdot k, x_2) \oplus b_{3,1} \cdot k.$$

[4] Otherwise, only one blockcipher call is actually involved, and such instances have at most $2^{n/2}$ security [36].

Hence, for any plaintext-ciphertext pair (t, p, c) and (t', p', c') with $(t, p) \neq (t', p')$, it has that

$$c = c' \quad \Longrightarrow \quad x_2 = x'_2.$$

Exploiting this property, the attacker mainly focuses on the first blockcipher call and peels off the second blockcipher call, by using pairs of (t, p, c) and (t', p', c') with $c = c'$. Note that such a plaintext-ciphertext pair can be easily obtained by sending a query (t, p) to $\widetilde{\mathbb{E}}[2]_k(\cdot, \cdot)$ to receive c, and then sending a query (t', c) to $\widetilde{\mathbb{E}}[2]_k^{-1}(\cdot, \cdot)$ to receive p'.[5] Thanks to such plaintext-ciphertext pairs, the attacker gets to know and even control the internal difference $\Delta y_1 = b_{2,2} \cdot (t \oplus t')$. As a result, he can succeed to recover the key k or to distinguish $\widetilde{\mathbb{E}}[2]$ from a random tweakable blockcipher $\widetilde{P}$ with a complexity of at most $O(2^{n/2})$ adversarial queries. The attack procedure is slightly different depending on the values of $a_{1,1}$ and $b_{1,1}$. Therefore, we further divide this case into four subcases, and describe the procedure for each subcase separately.

Subcase (1.1): $a_{1,1} = 0$ *and* $b_{1,1} = 0$. The key k is not used in the first blockcipher $y_1 = E(t, p)$. Hence the attacker can get the value of y_1 by querying (t, p) to $E(\cdot, \cdot)$. A distinguisher $\mathcal{D}$ is launched as follows. Firstly, $\mathcal{D}$ obtain a plaintext-ciphertext pair (t, p, c) and (t', p', c') with $c = c'$, and computes $\Delta y_1 = b_{2,2} \cdot (t \oplus t')$. Secondly, $\mathcal{D}$ queries (t, p) and (t', p') to $E(\cdot, \cdot)$ to receive w and w' respectively, and computes $\Delta w = w \oplus w'$. Finally, $\mathcal{D}$ outputs 1 if $\Delta y_1 = \Delta w$, and outputs 0 otherwise. The probability of $\mathcal{D}$ outputting 1 is 1 when interacting $\widetilde{\mathbb{E}}[2]$, and is 2^{-n} when interacting with $\widetilde{P}$. Thus, the advantage of $\mathcal{D}$ is $1 - 2^{-n}$. The complexity of $\mathcal{D}$ is 4 queries.

Subcase (1.2): $a_{1,1} = 0$ *and* $b_{1,1} = 1$. The first blockcipher call is $y_1 = E(t, p \oplus k)$. Its key z_1 is the tweak t, and can be controlled by the attacker. A key-recovery attack $\mathcal{A}$ is launched as follows. Firstly, $\mathcal{A}$ fixes a tweak value t and a non-zero value Δ. Secondly, $\mathcal{A}$ collects plaintext-ciphertext pairs (t, p, c) and (t', p', c') such that $t' = t \oplus \Delta$ and $c' = c$. Each pair has that

$$p \oplus p' = x_1 \oplus x'_1 = E^{-1}(t, y_1) \oplus E^{-1}(t \oplus \Delta, y_1 \oplus b_{2,2} \cdot \Delta).$$

$\mathcal{A}$ stores $\{(p, p \oplus p')\}$ for $2^{n/2}$ distinct values of p, whose corresponding values of y_1 are also distinct. This needs $2^{n/2+1}$ queries. Thirdly, $\mathcal{A}$ selects $2^{n/2}$ distinct values w. For each w, he queries (t, w) and $(t \oplus \Delta, w \oplus b_{2,2} \cdot \Delta)$ to $E^{-1}(\cdot, \cdot)$ to receive u and u' respectively, which has that

$$u \oplus u' = E^{-1}(t, w) \oplus E^{-1}(t \oplus \Delta, w \oplus b_{2,2} \cdot \Delta).$$

$\mathcal{A}$ matches $u \oplus u'$ to previously stored $p \oplus p'$. If a matched is found that implies $x_1 = p \oplus k = u$, the attacker computes the key k as $k = u \oplus p$. The complexity

[5] Of course one may directly query (t, c) and $(t, c' = c)$ to $\widetilde{\mathbb{E}}[2]_k^{-1}(\cdot, \cdot)$ to obtain such a pair. But the above approach allows the attacker to control the plaintext p, which is necessary in our attacks.

of $\mathcal{A}$ is around $2^{n/2+2}$ adversarial queries, and its success probability can be trivially computed as $1-(1-2^{-n})^{2^n} \approx 1-1/e \approx 0.63$, since there are $2^{n/2}$ distinct values of y_1 and $2^{n/2}$ distinct values of w.

Subcase (1.3): $a_{1,1}=1$ *and* $b_{1,1}=0$. The first blockcipher call is $y_1 = E(t \oplus k, p)$. Its input x_1 is plaintext p, and can be controlled by the attacker. A key-recovery attack $\mathcal{A}$ is launched as follows. Firstly, $\mathcal{A}$ fixes a plaintext value p and a non-zero value Δ. Secondly, $\mathcal{A}$ collects plaintext-ciphertext pairs (t, p, c) and (t', p', c') such that $t' = t \oplus \Delta$ and $c' = c$. Each pair has that

$$p' = E^{-1}(t \oplus \Delta \oplus k, E(t \oplus k, p) \oplus b_{2,2} \cdot \Delta).$$

$\mathcal{A}$ stores $\{(t, p')\}$ for $2^{n/2}$ distinct values of t, which needs $2^{n/2+1}$ queries. Thirdly, $\mathcal{A}$ selects $2^{n/2}$ distinct values l. For each l, he queries (l, p) to $E(\cdot,\cdot)$, receives w, and then queries $(l \oplus \Delta, w \oplus b_{2,2} \cdot \Delta)$ to $E^{-1}(\cdot,\cdot)$ to receive u', which have that

$$u' = E^{-1}(l \oplus \Delta, E(l, p) \oplus b_{2,2} \cdot \Delta)$$

$\mathcal{A}$ matches u' to previously stored p'. If a matched is found that implies $l = t \oplus k$, $\mathcal{A}$ computes the key k as $k = l \oplus t$. The complexity of $\mathcal{A}$ is around $2^{n/2+2}$ adversarial queries. Similarly with the above subcases, its success probability can be computed as 0.63.

Subcase (1.4): $a_{1,1}=1$ *and* $b_{1,1}=1$. The first blockcipher call is $y_1 = E(t \oplus k, p \oplus k)$. XORing its inputs x_1 and z_1 is $x_1 \oplus z_1 = p \oplus t$, which can be controlled by the attacker. A key-recovery attack $\mathcal{A}$ is launched as follows. Firstly, $\mathcal{A}$ fixes a plaintext p and a non-zero value Δ. Secondly, $\mathcal{A}$ collects plaintext-ciphertext pairs $(t, p \oplus t, c)$ and (t', p', c') with $t' = t \oplus \Delta$ and $c' = c$. Each pair has that

$$p' \oplus t = E^{-1}(t \oplus \Delta \oplus k, E(t \oplus k, p \oplus t \oplus k) \oplus b_{2,2} \cdot \Delta) \oplus k \oplus t$$

$\mathcal{A}$ stores $\{(t, p' \oplus t)\}$ for $2^{n/2}$ distinct values of t, which needs $2^{n/2+1}$ queries. Thirdly, $\mathcal{A}$ selects $2^{n/2}$ distinct values l. For each l, he queries $(l, p \oplus l)$ to $E(\cdot,\cdot)$, receives w, and then queries $(l \oplus \Delta, w \oplus b_{2,2} \cdot \Delta)$ to $E^{-1}(\cdot,\cdot)$ to receive u', which have that

$$u' \oplus l = E^{-1}(l \oplus \Delta, E(l, p \oplus l) \oplus b_{2,2} \cdot \Delta) \oplus l$$

$\mathcal{A}$ matches $u' \oplus l$ to previously stored $p' \oplus t$. f a matched is found that implies $l = t \oplus k$, $\mathcal{A}$ computes the key k as $k = l \oplus t$. The complexity of $\mathcal{A}$ is around $2^{n/2+2}$ adversarial queries, and its success probability can be trivially computed as 0.63 similarly with the above subcases.

Overall, we conclude that all the instances of Case (1) using one tweak-dependent key have at most around $2^{n/2}$ security.

Case (2): $\boldsymbol{a_{1,2}=0}$ **and** $\boldsymbol{a_{2,2}=1}$**.** z_2 is the tweak-dependent key. The analysis is highly similar with Case (1), which is written in full version of this paper [52]. In a high level, Case (2) can be regarded as the inverse of Case (1) by analyzing the decryption oracle $\widetilde{E}^{-1}$. Here we just provide the conclusion: all the instances of Case (2) using one tweak-dependent key have at most around $2^{n/2}$ security.

5.2 On the Instances of Type II

Observation 1 sets $b_{2,2} = 0$ and $b_{3,2} = 0$. We set $b_{3,5} = 1$ such that the second blockcipher call is involved in $\widetilde{\mathbb{E}}[2]$. The construction of Type II is depicted in Fig. 4. Similarly we also divide the instances of Type II into two cases depending on the position of the tweak-dependent key. More precisely, it depends on the values of $a_{1,2}$, $a_{2,2}$, and $a_{2,3}$ if y_1 is computed related to tweak t.

Case (1): $a_{1,2} = 1$, $a_{2,2} = 0$, $a_{2,3} = 0$. z_1 is the tweak-dependent key. The reason of setting $a_{2,3} = 0$ is that y_1 is computed depending on t as

$$y_1 = E(a_{1,1} \cdot k \oplus t, b_{1,1} \cdot k \oplus b_{1,2} \cdot t).$$

We find the instances of this case have at most $2^{n/2}$ security based on the following observation. The computation from internal variable y_1 to ciphertext c is that

$$c = E(a_{2,1} \cdot k, p \oplus b_{2,4} \cdot y_1 \oplus b_{2,1} \cdot k) \oplus b_{3,1} \cdot k \oplus b_{3,4} \cdot y_1,$$

which is not related to the tweak value. Therefore, for two distinct tweaks t and t' colliding on y_1 that is

$$E(a_{1,1} \cdot k \oplus t, b_{1,1} \cdot k \oplus b_{1,2} \cdot t) = E(a_{1,1} \cdot k \oplus t', b_{1,1} \cdot k' \oplus b_{1,2} \cdot t'),$$

it leads to the same ciphertext for any plaintext, more precisely,

$$\widetilde{\mathbb{E}}[2]_k(t, p) = \widetilde{\mathbb{E}}[2]_k(t', p), \quad \text{for} \quad \forall p \in \mathcal{M}.$$

Such a pair of tweaks can be found after trying $2^{n/2}$ distinct tweaks. Putting all together, a distinguisher $\mathcal{D}$ can be launched as follows. Firstly, $\mathcal{D}$ fixes a plaintext p. Secondly, he selects $2^{n/2}$ distinct tweak values t, queries (t, p) to $\widetilde{\mathbb{E}}[2]_k(\cdot, \cdot)$ to search a collision among received ciphertexts. Let t and t' denote the corresponding tweaks for the colliding ciphertexts. Thirdly, $\mathcal{D}$ selects another plaintext p' with $p' \neq p$, and queries (t, p') and (t', p') to $\widetilde{\mathbb{E}}[2]_k(\cdot, \cdot)$ and receives

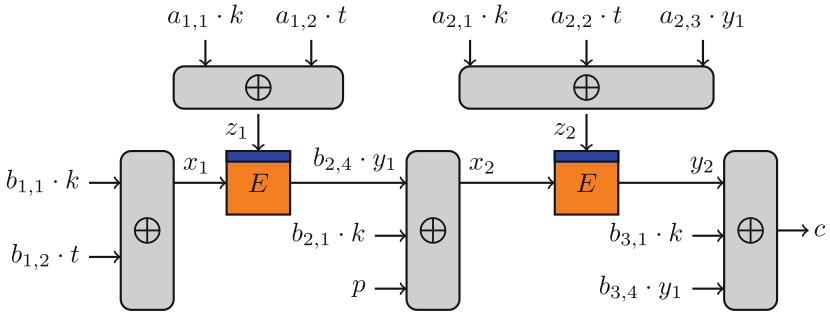

Fig. 4. Type II Construction of $\widetilde{\mathbb{E}}[2]$

ciphertexts c' and c'' respectively. Finally, $\mathcal{D}$ outputs 1 if $c' \neq c''$, and outputs 0 otherwise. The complexity of $\mathcal{D}$ is around $2^{n/2}$ queries. When interacting with $\widetilde{\mathbb{E}}[2]$, $\mathcal{D}$ outputs 1, as long as he succeeds to find the colliding ciphertexts at second step, which has a probability of $1-(1-2^{-n})^{2^{n-1}} \approx 0.4$. When interacting with a random tweakable blockcipher, the probability of $\mathcal{D}$ outputting 1 is obviously 2^{-n}. Therefore, the advantage of $\mathcal{D}$ is computed as $0.4 - 2^{-n} \approx 0.4$.

Case (2): $a_{1,2} = 0$. We need to further set the values of $a_{2,2}$ and $a_{2,3}$ such that z_2 is a tweak-dependent key. There are two possible setting depending on the value of $b_{1,2}$. More precisely, if $b_{1,2} = 0$, then y_1 is computed unrelated to tweak t, and therefore $a_{2,2}$ must be 1. Otherwise, as long as one of $a_{2,2}$ and $a_{2,3}$ is not zero, z_2 is a tweak-dependent key. Accordingly we divide Case (2) to two subcases.

Subcase (2.1): $b_{1,2} = 0$, $a_{2,2} = 1$. A graphical view is provided in Fig. 5. Notably internal variable y_1 is computed as $y_1 = E(a_{1,1} \cdot k, b_{1,1} \cdot k)$, which is unrelated to tweak t. We refer to y_1 as a subkey derived from the key k for those instances with $(a_{1,1}, b_{1,1}) \neq (0,0)$. Moreover, the computation from p to x_2 is $x_2 = p \oplus b_{2,1} \cdot k \oplus b_{2,4} \cdot y_1$, and hence $\Delta x_2 = \Delta p$ always holds. Similarly, $\Delta y_2 = \Delta c$ always holds. In other words, for any plaintext-ciphertext pair (t, p, c) and (t', p', c'), the internal variable differences Δx_2 and Δy_2 is known to the attacker. Due to these properties, we find several conditions on the instances of this subcase in order to possibly have a security beyond the birthday bound.

- $(a_{1,1}, b_{1,1}) \neq (0,0)$
  If $a_{1,1}, b_{1,1} = (0,0)$, it has that $y_1 = E(0,0)$. Then an attacker can query $(0,0)$ to $E(\cdot,\cdot)$, receive the value of y_1, and then peel off the first blockcipher call. As a result, the instances become essentially based on one blockcipher call in the view of the attacker. As proven in [36], the attacker can distinguish such instances from a random tweakable blockcipher with a complexity of at most $2^{n/2}$ adversarial queries.
- $(a_{2,1}, a_{2,3}) \neq (0,0)$
  If $(a_{2,1}, a_{2,3}) = (0,0)$, an attacker can fix the tweak t to a constant and

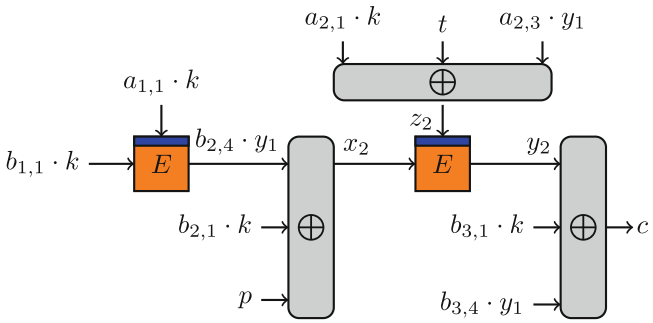

Fig. 5. Subcase (2.1) of Type II of $\widetilde{\mathbb{E}}[2]$

regard $b_{2,1} \cdot k \oplus b_{2,4} \cdot y_1$ and $b_{3,1} \cdot k \oplus b_{3,4} \cdot y_1$ as the pre- and post-whitening keys respectively. As a result, the instances become essentially one-step Even-Mansour blockcipher [17], and several attack procedures with a complexity of $2^{n/2}$ queries have been presented in [4,13,15].

- $(b_{2,1}, b_{2,4}) \neq (0,0)$ and $(b_{3,1}, b_{3,4}) \neq (0,0)$
  If $(b_{2,1}, b_{2,4}) = (0,0)$, it has $x_2 = p$. Then an attacker gets to know and control the value of x_2. A distinguisher $\mathcal{D}$ is launched as follows. Firstly, $\mathcal{D}$ fixes two distinct plaintexts p and p'. Secondly, he selects $2^{n/2}$ distinct tweaks t. For each t, $\mathcal{D}$ queries (t,p) and (t,p') to $\widetilde{\mathbb{E}}[2]_k(\cdot,\cdot)$, receives ciphertexts c and c' respectively, and stores $(t, c \oplus c')$. Thirdly, $\mathcal{D}$ selects $2^{n/2}$ distinct values l. For each l, he queries (l,p) and (l,p') to $E(\cdot,\cdot)$, receives w and w' respectively, and matches $w \oplus w'$ to previously stored $c \oplus c'$ at second step. Once a matched is found, that is

$$E(a_{2,1} \cdot k \oplus t \oplus a_{2,3} \cdot y_1, p) \oplus E(a_{2,1} \cdot k \oplus t \oplus a_{2,3} \cdot y_1, p') = E(l,p) \oplus E(l,p'),$$

  $\mathcal{D}$ recovers $a_{2,1} \cdot k \oplus b_{2,3} \cdot y_1 = t \oplus l$. Finally, for any plaintext-ciphertext pair of (t,p,c) and (t',p',c'), $\mathcal{D}$ can compute internal variables z_2 and z_2', and query (z_2,p) and (z_2',p') to $E(\cdot,\cdot)$ to recover y_2 and y_2', respectively. $\mathcal{D}$ outputs 1 if $c \oplus c' = y_2 \oplus y_2'$, and outputs 0 otherwise. The complexity of $\mathcal{D}$ is around $2^{n/2+2}$ queries. When interacting with $\widetilde{\mathbb{E}}[2]$, $\mathcal{D}$ outputs 1 as long as he recovers $a_{2,1} \cdot k \oplus b_{2,3} \cdot y_1$, which succeeds with a probability $1-(1-2^{-n})^{2^n} \approx 1-1/e \approx 0.63$. When interacting with a random tweakable blockcipher, $\mathcal{D}$ outputs 1 with a probability 2^{-n}. Therefore the advantage of $\mathcal{D}$ is $0.63 - 2^{-n} \approx 0.63$.
  $(b_{3,1}, b_{3,4}) \neq (0,0)$ is observed after a very similar analysis. Just the attacker gets to know and control the value of y_2. Accordingly, he fixes two ciphertexts c and c', and queries (t,c) and (t,c') to $\widetilde{\mathbb{E}}[2]_k^{-1}(\cdot,\cdot)$ for distinct tweaks t. We omit the details.
- $(b_{2,1}, b_{2,4}) \neq (a_{2,1}, a_{2,3})$ and $(b_{3,1}, b_{3,4}) \neq (a_{2,1}, a_{2,3})$
  If $(b_{2,1}, b_{2,4}) = (a_{2,1}, a_{2,3})$, it has $b_{2,1} \cdot k \oplus b_{2,4} \cdot y_1 = a_{2,1} \cdot k \oplus a_{2,3} \cdot y_1$, which is denoted as g. Then $x_2 \oplus z_2 = g \oplus p \oplus g \oplus t = p \oplus t$. Hence an attacker gets to know and control $x_2 \oplus z_2$. A distinguisher $\mathcal{D}$ can be launched. Firstly, $\mathcal{D}$ fixes a non-zero Δ. Secondly, he selects $2^{n/2}$ distinct tweaks t, queries $(t, p = t)$ and $(t, p' = t \oplus \Delta)$ to $\widetilde{\mathbb{E}}[2]_k(\cdot,\cdot)$, receives c and c' respectively, and stores $(t, c \oplus c')$. Thirdly, $\mathcal{D}$ selects $2^{n/2}$ distinct values l, queries (l,l) and $(l, l \oplus \Delta)$ to $E(\cdot,\cdot)$ to receive w and w' respectively, and matches $w \oplus w'$ to previously stored $c \oplus c'$. If a matched is found, that is

$$E(g \oplus t, g \oplus t) \oplus E(g \oplus t, g \oplus t \oplus \Delta) = E(l,l) \oplus E(l, l \oplus \Delta),$$

  $\mathcal{D}$ recovers g as $g = t \oplus l$. Therefore $\mathcal{D}$ is able to compute x_2 and z_2 for any plaintext-ciphertext, and gets y_2 by querying $E(\cdot,\cdot)$. After that, similarly with the above analysis, $\mathcal{D}$ just needs to make several additional queries. Overall, the complexity of $\mathcal{D}$ is around $2^{n/2}$ queries, and has an advantage of 0.63.
  $(b_{3,1}, b_{3,4}) \neq (a_{2,1}, a_{2,3})$ is observed after a very similar analysis. Here we omit the details.

Putting all these conditions together, there are 32 instances of this subcase left, which are denoted as $\widetilde{E1}$, $\widetilde{E2}, \ldots, \widetilde{E32}$ and have been depicted in Figs. 6 and 7. After further investigation, we find that these constructions achieve full 2^n provable security. The proof is presented in Sect. 6.

Subcase (2.2): $b_{1,2} = 1$, $(a_{2,2}, a_{2,3}) \neq (0, 0)$. Interestingly, we notice that the instances of Subcase (2.1) has an efficiency advantage over the instances of Subcase (2.2). More precisely, if one pre-computes and stores internal variable y_1 as a subkey, an instance of Subcase (2.1) requires just one block-cipher call for encrypting (t, p) or decrypting (t, c), while the instances of Subcase (2.2) always need two blockcipher calls. Since we have found instances of Subcase (2.1) achieving full 2^n security, it is unnecessary to search among instances of Subcase (2.2). Nevertheless, we did investigate the instances of Subcase (2.2), and found 24 instances achieving full 2^n provable security. Here we omit the discussion on this subcase due to the limited space.

5.3 On the Instances of Type III

Clearly, plaintext and ciphertext are linearly related in this type of construction, and can be trivially distinguished by making two queries to $\widetilde{\mathbb{E}}[2]_k(\cdot, \cdot)$ with a fixed difference in plaintexts, *e.g.*, (t, p) and $(t, p \oplus \Delta)$, and verifying $\Delta c = \Delta$.

6 Security Proof of $\widetilde{E1}, \ldots, \widetilde{E32}$

Let $\widetilde{E}$ be any tweakable blockcipher of $\widetilde{E1}, \widetilde{E2}, \ldots, \widetilde{E32}$, and E denotes its underlying blockcipher. Let $\widetilde{P}$ be a random tweakable blockcipher that is $\widetilde{P} \xleftarrow{\$} \widetilde{\text{Perm}}$. Let $(\mathcal{O}_1, \mathcal{O}_2)$ be either $(\widetilde{E}_k^{\pm}(\cdot, \cdot), E^{\pm}(\cdot, \cdot))$ with $k \xleftarrow{\$} \mathcal{K}$ or $(\widetilde{P}^{\pm}(\cdot, \cdot), E^{\pm}(\cdot, \cdot))$. Let $\mathcal{D}$ be a distinguisher interacting with $(\mathcal{O}_1, \mathcal{O}_2)$ that makes (at most) q queries. We denote the number of $\mathcal{D}$'s queries to $\mathcal{O}_1$ and to $\mathcal{O}_2$ as q_1 and q_2 respectively: $q = q_1 + q_2$. Without loss of generality, we assume that $\mathcal{D}$ does not make duplicated queries to $\mathcal{O}_1$ or $\mathcal{O}_2$. We use views $v_1 = \{(t_1, p_1, c_1), \ldots, (t_{q_1}, p_{q_1}, c_{q_1})\}$ and $v_2 = \{(l_1, u_1, w_1), \ldots, (l_{q_2}, u_{q_2}, w_{q_2})\}$ to denote the transcripts, which are lists of query-responses, created by $\mathcal{D}$ interacting with $\mathcal{O}_1$ and $\mathcal{O}_2$, respectively. At the end of the interaction with $(\mathcal{O}_1, \mathcal{O}_2)$, the distinguisher $\mathcal{D}$ obtains a view $v = (v_1, v_2)$ before determining the output bit. Since $\mathcal{D}$ is computationally unbounded, without loss of generality we assume that $\mathcal{D}$ is deterministic. Therefore $\mathcal{D}$ computes its decision bit deterministically based on the view v. Accordingly, the probability distribution of the decision bit of $\mathcal{D}$ solely depends on the probability distribution of the view v.

Our proof adopts the H-coefficient technique [8,45], which has been introduced in Sect. 2.3. We use X and Y to denote the probability distribution on views when $\mathcal{D}$ interacts with $(\widetilde{E}_k^{\pm}(\cdot, \cdot), E^{\pm}(\cdot, \cdot))$ and interacts with $(\widetilde{P}^{\pm}(\cdot, \cdot), E^{\pm}(\cdot, \cdot))$, respectively. We use $\mathcal{V}$ to denote the set of attainable views v when $\mathcal{D}$ interacts with $(\widetilde{P}^{\pm}(\cdot, \cdot), E^{\pm}(\cdot, \cdot))$, that is $\mathcal{V} = \{v \mid \Pr[Y = v] > 0\}$. Next,

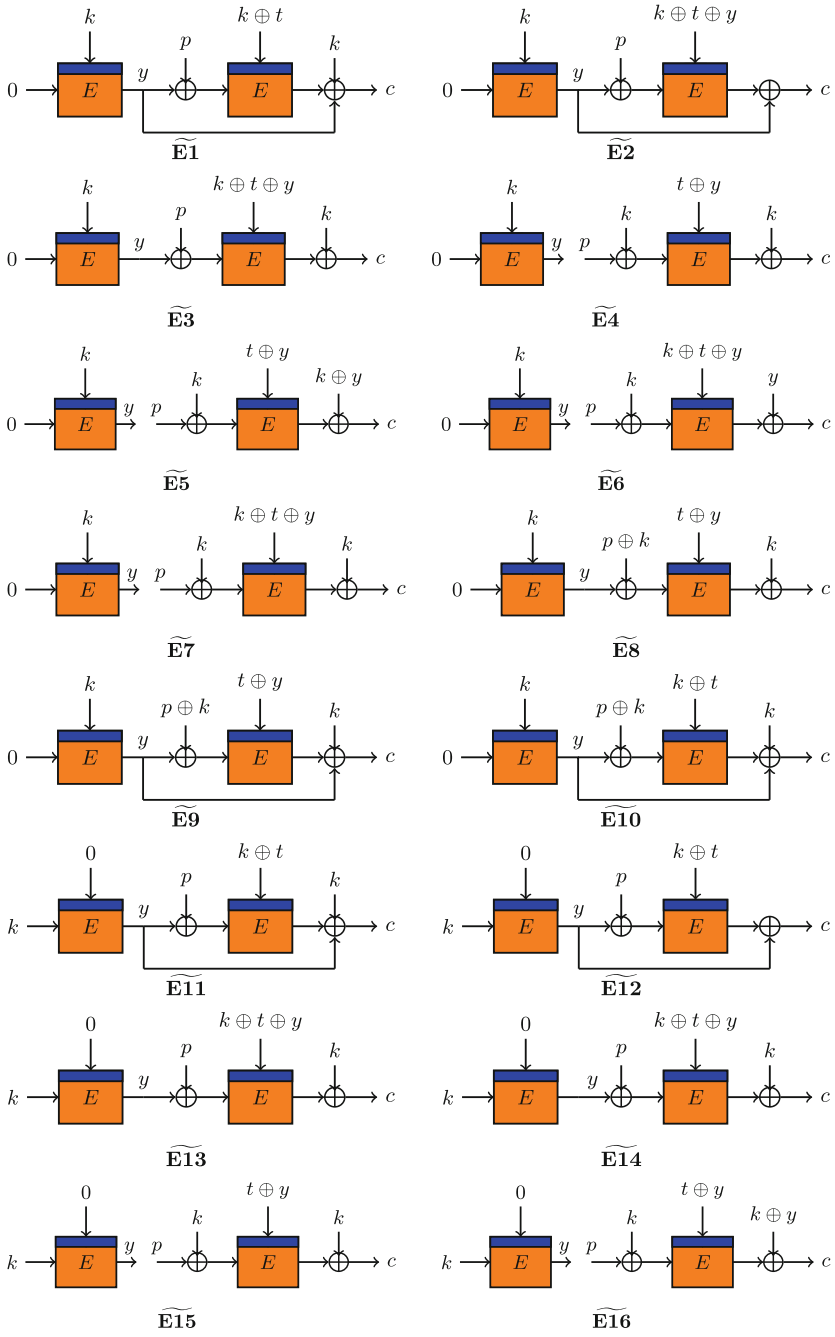

Fig. 6. $\widetilde{E1}$ to $\widetilde{E16}$ of the 32 efficient constructions: the internal variable y is referred to as the subkey for these constructions.

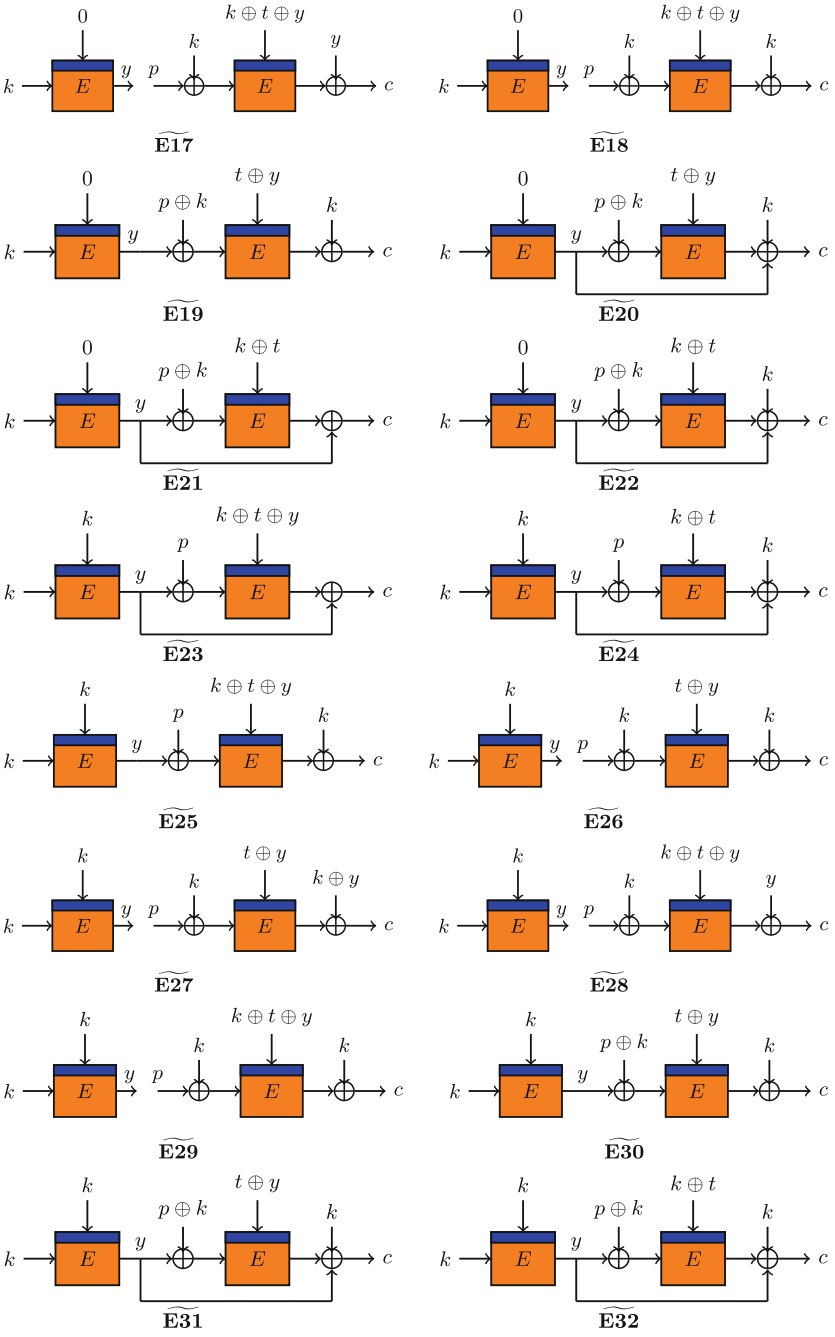

Fig. 7. $\widetilde{E17}$ to $\widetilde{E32}$ of the 32 efficient constructions: the internal variable y is referred to as the subkey for these constructions.

we partition $\mathcal{V}$ to disjoint subsets $\mathcal{V}_{\text{bad}}$ and $\mathcal{V}_{\text{good}}$ such that $\mathcal{V} = \mathcal{V}_{\text{good}} \bigcup \mathcal{V}_{\text{bad}}$, and evaluate upper bound of $\epsilon_{v_{\text{good}}}$ (defined in Sect. 2.3) for the views $v \in \mathcal{V}_{\text{good}}$ and upper bound of $\Pr[Y \in \mathcal{V}_{\text{bad}}]$.

6.1 Partition of $\mathcal{V}$

In our proof, we disclose the values of the secret key k and the subkey y to $\mathcal{D}$, after he finishes the interaction with $(\mathcal{O}_1, \mathcal{O}_2)$ and before he determines the output bit. In the case of $(\widetilde{P}^{\pm}(\cdot,\cdot), E^{\pm}(\cdot,\cdot))$ as $(\mathcal{O}_1, \mathcal{O}_2)$, we choose the value of k at random, namely $k \xleftarrow{\$} \mathcal{K}$, and get the corresponding subkey y by querying $E^{\pm}$. This is without loss of generality since it will only increase the advantage of $\mathcal{D}$. With the knowledge of k and y, $\mathcal{D}$ can easily derive the query-responses (l, u, w)'s of invocations of $E^{\pm}(\cdot,\cdot)$ for each query-response (t_i, p_i, c_i) in view v_1. Therefore $\mathcal{D}$ gets all query-responses of blockcipher E during the interaction with $(\mathcal{O}_1, \mathcal{O}_2)$.

For each view $v = (v_1, v_2) \in \mathcal{V}$, we divide the query-responses of blockcipher E, derived from it thanks to the disclosed values of k and y, into three subsets, and store them separately in different tables. The first subset consists of a single query-response of E that generates the subkey y, and is stored in a table $\mathcal{T}^1 = \{(l_1^1, u_1^1, w_1^1 = y)\}$. The second subset consists of the other query-responses of E derived from v_1, and is stored in a table $\mathcal{T}^2 = \{(l_1^2, u_1^2, w_1^2), (l_2^2, u_2^2, w_2^2), \ldots, (l_{q_1}^2, u_{q_1}^2, w_{q_1}^2)\}$. The last subset consists of all query-responses of E derived from v_2, and is stored in a table $\mathcal{T}^3 = \{(l_1^3, u_1^3, w_1^3), (l_2^3, u_2^3, w_2^3), \ldots, (l_{q_2}^3, u_{q_2}^3, w_{q_2}^3)\}$.

Definition of $\mathcal{V}_{\text{bad}}$. We define that $\mathcal{V}_{\text{bad}}$ is the set of views which causes the following bad event, and accordingly define $\mathcal{V}_{\text{good}}$ as $\mathcal{V}_{\text{good}} = \mathcal{V} \backslash \mathcal{V}_{\text{bad}}$.

- *Bad event*: for a view $v \in \mathcal{V}$, if there exist (l_j^i, u_j^i, w_j^i) in Table $\mathcal{T}^i$ and $(l_{j'}^{i'}, u_{j'}^{i'}, w_{j'}^{i'})$ in Table $\mathcal{T}^{i'}$ such that $(l_j^i, u_j^i) = (l_{j'}^{i'}, u_{j'}^{i'})$ or $(l_j^i, w_j^i) = (l_{j'}^{i'}, w_{j'}^{i'})$, where $1 \le i, i' \le 3$ and $i \ne i'$, we say v causes a bad event.

The reasoning of the above definition of bad views is to ensure that for any view $v \in \mathcal{V}_{\text{good}}$, every query-response of $\mathcal{D}$ interacting with $\mathcal{O}_1$ leads to one unique query-response of blockcipher E, which is essentially helpful to evaluate the upper bound of $\epsilon_{v_{\text{good}}}$.

6.2 Upper Bound of $\epsilon_{v_{\text{good}}}$

Firstly, we deal with $\Pr[X = v]$. The random variable X is defined on the probability space of all possible secret key k and all possible underlying blockcipher E. We denote by all_X the probability space of X, and its cardinality $|\text{all}_X|$ is $2^n \cdot (2^n!)^{2^n}$, that is the number of keys times the number of blockciphers. We write an element π in all_X compatible with v if π produces exactly the same responses for all queries in v. We denote by $\text{comp}_X(v)$ all the elements in all_X

compatible with view v. Since k is chosen uniformly at random and E is an ideal blockcipher, we have that

$$\Pr[X = v] = \frac{|\mathrm{comp}_X(v)|}{|\mathrm{all}_X|}.$$

Similarly, Y is defined on the probability space of the key k, tweakable blockcipher $\widetilde{\mathcal{P}}$ and blockcipher E. Define $\mathrm{comp}_Y(v)$ and all_Y accordingly, and then we have that

$$\Pr[Y = v] = \frac{|\mathrm{comp}_Y(v)|}{|\mathrm{all}_Y|}.$$

all_Y is $2^n \cdot (2^n!)^{2^n} \cdot (2^n!)^{2^n}$, that is the number of keys times the number of tweakable blockciphers times the number of blockciphers.

Next is to compute $|\mathrm{comp}_X(v)|$ and $|\mathrm{comp}_Y(v)|$. Recall that the view v contains the value of k, which is disclosed to $\mathcal{D}$ at the end of interaction, and then a set of input-outputs of underlying blockcipher E are derived and separately stored in tables $\mathcal{T}^1$, $\mathcal{T}^2$ and $\mathcal{T}^3$. Let α_i and β_i denote the number of input-outputs (l, u, w)'s of E with the value i as the key value (that is $l = i$) in $\mathcal{T}^2$ and $\mathcal{T}^3$, respectively, for $0 \le i \le 2^n - 1$. Denote i_t as the tweak value that produces i as the key value (that is z_2 in Fig. 5) for the second blockcipher call in $\mathcal{O}_1$, and denote γ_{i_t} the number of queries to $\mathcal{O}_1$ with tweak values as i_t. Since v is a good view, there is no element collision between any two tables. Moreover, $\mathcal{D}$ does not make duplicate queries. Hence all input-outputs of E in $\mathcal{T}^1$, $\mathcal{T}^2$ and $\mathcal{T}^3$ are distinct. Therefore, it implies that $\gamma_{i_t} = \alpha_i$. The query-response (l_1^1, u_1^1, w_1^1) of E in $\mathcal{T}^1$ has $l_1^1 = k$ or $l_1^1 = 0$.[6] Without loss of generality, we assume $l_1^1 = k$. Then, we get that

$$|\mathrm{comp}_X(v)| = (2^n - \alpha_k - \beta_k - 1)! \cdot \prod_{i=0}^{k-1} (2^n - \alpha_i - \beta_i)! \cdot \prod_{i=k+1}^{2^n-1} (2^n - \alpha_i - \beta_i)!,$$

and

$$\begin{aligned}
|\mathrm{comp}_Y(v)| &= \prod_{i=0}^{2^n-1} (2^n - \gamma_{i_t})! \cdot \left((2^n - \beta_k - 1)! \cdot \prod_{i=0}^{k-1} (2^n - \beta_i)! \cdot \prod_{i=k+1}^{2^n-1} (2^n - \beta_i)! \right) \\
&= \prod_{i=0}^{2^n-1} (2^n - \alpha_i)! \cdot \left((2^n - \beta_k - 1)! \cdot \prod_{i=0}^{k-1} (2^n - \beta_i)! \cdot \prod_{i=k+1}^{2^n-1} (2^n - \beta_i)! \right) \\
&= (2^n - \alpha_k)! \cdot (2^n - \beta_k - 1)! \cdot \prod_{i=0}^{k-1} ((2^n - \alpha_i)! \cdot (2^n - \beta_i)!) \cdot \prod_{i=k+1}^{2^n-1} ((2^n - \alpha_i)! \cdot (2^n - \beta_i)!).
\end{aligned}$$

From $(2^n - \alpha)! \cdot (2^n - \beta)! \le (2^n - \alpha - \beta)! \cdot 2^n!$, we have that

$$|\mathrm{comp}_Y(v)| \le (2^n - \alpha_k - \beta_k - 1)! \cdot (2^n!)^{2^n} \cdot \prod_{i=0}^{k-1} (2^n - \alpha_i - \beta_i)! \cdot \prod_{i=k+1}^{2^n-1} (2^n - \alpha_i - \beta_i)!.$$

[6] More precisely, $\widetilde{E1}, \ldots, \widetilde{E10}, \widetilde{E23}, \ldots, \widetilde{E32}$ have $l_1^1 = k$, and the other tweakable blockciphers have $l_1^1 = 0$.

Then we compute

$$\frac{|\mathrm{comp}_X(v)|}{|\mathrm{comp}_Y(v)|} \geq \frac{(2^n-\alpha_k-\beta_k-1)!\cdot\prod_{i=0}^{k-1}(2^n-\alpha_i-\beta_i)!\cdot\prod_{i=k+1}^{2^n-1}(2^n-\alpha_i-\beta_i)!}{(2^n-\alpha_k-\beta_k-1)!\cdot(2^n!)^{2^n}\cdot\prod_{i=0}^{k-1}(2^n-\alpha_i-\beta_i)!\cdot\prod_{i=k+1}^{2^n-1}(2^n-\alpha_i-\beta_i)!}$$
$$= \frac{1}{(2^n!)^{2^n}}$$

Finally, we compute

$$\frac{\Pr[X=v]}{\Pr[Y=v]} = \frac{|\mathrm{comp}_X(v)|}{|\mathrm{comp}_Y(v)|}\times\frac{|\mathrm{all}_Y|}{|\mathrm{all}_X|}$$
$$\geq \frac{1}{(2^n!)^{2^n}}\times\frac{2^n\cdot(2^n!)^{2^n}\cdot(2^n!)^{2^n}}{2^n\cdot(2^n!)^{2^n}} = 1$$

which give that $\epsilon_{v_{\mathrm{good}}} = 0$.

Note. We highlight that this upper bound of $\epsilon_{v_{\mathrm{good}}} = 0$ is indeed shared by all these 32 constructions $\widetilde{E1},\ldots,\widetilde{E32}$. Moreover, as long as every view in $\mathcal{V}_{\mathrm{good}}$ does not cause the above bad event defined in Sect. 6.1, it always has that $\epsilon_{v_{\mathrm{good}}} = 0$. Therefore, the advantage of all distinguishers making at most q queries is upper bounded as

$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{E}}(q) \leq \Pr[Y\in\mathcal{V}_{\mathrm{bad}}].$$

Thus, the remaining work is to evaluate $\Pr[Y \in \mathcal{V}_{\mathrm{bad}}]$ for each construction of $\widetilde{E1},\ldots,\widetilde{E32}$, separately.

6.3 Upper Bound of $\Pr[Y \in \mathcal{V}_{\mathrm{bad}}]$

For each construction of $\widetilde{E1}$ to $\widetilde{E32}$, we give the exact definition of $\mathcal{V}_{\mathrm{bad}}$ according to the specification, which also defines $\mathcal{V}_{\mathrm{good}} = \mathcal{V}\backslash\mathcal{V}_{\mathrm{bad}}$. We must ensure that every view $v \in \mathcal{V}_{\mathrm{good}}$ does not cause the bad event defined in Sect. 6.1, such that the probability $\Pr[Y \in \mathcal{V}_{\mathrm{bad}}]$ is upper bound of $\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{E}}(q)$. Due to the limited space, in this section we use $\widetilde{E1}$ as an example, and write the definitions of $\mathcal{V}_{\mathrm{bad}}$ for the other constructions in full version of this paper [52].

$\mathcal{V}_{\mathrm{bad}}$ *of* $\widetilde{E1}$ is defined as the set of views $v = (v_1, v_2)$ such that (at least) one of the following events occur:

(1a). $\exists\ (l_j, u_j, w_j) \in v_2$ such that $l_j = k$;
(1b). $\exists\ (t_i = 0, p_i, c_i) \in v_1$ such that $p_i = y$ or $c_i = k$;
(1c). $\exists\ (t_i, p_i, c_i) \in v_1$ and $(l_j, u_j, w_j) \in v_2$ such that $(l_j = k \oplus t_i, u_j = p_i \oplus y)$ or $(l_j = k \oplus t_i, w_j = c_i \oplus y \oplus k)$.

Since both k and y are selected uniformly at random from a set of size at least $2^n - q - 1$, we have that

$$\Pr[(1a)] \leq q/(2^n - q - 1);$$
$$\Pr[(1b)] \leq 2q/(2^n - q - 1);$$
$$\Pr[(1c)] \leq 2q^2/(2^n - q - 1)^2.$$

Therefore, we get that

$$\begin{aligned}\Pr[Y \in \mathcal{V}_{\mathrm{bad}}] &\leq \Pr[(1a)] + \Pr[(1b)] + \Pr[(1c)] \\ &\leq \frac{3q}{2^n - q - 1} + \frac{2q^2}{(2^n - q - 1)^2}\end{aligned}$$

Supposing $q < 2^{n-1}$, we have that

$$\Pr[Y \in \mathcal{V}_{\mathrm{bad}}] \leq \frac{3q}{2^{n-1}} + \frac{2q^2}{(2^{n-1})^2} \leq \frac{5q}{2^{n-1}}.$$

Next, we look into the views in $\mathcal{V}_{\mathrm{good}}$. A view in $\mathcal{V}_{\mathrm{good}}$ implies that nonce of the three events $(1a)$, $(1b)$ and $(1c)$ occur. Then we have that

- (1a) does not occur $\Longrightarrow$ the tuple elements in $\mathcal{T}^1$ and in $\mathcal{T}^3$ do not collide;
- (1b) does not occur $\Longrightarrow$ the tuple elements in $\mathcal{T}^1$ and in $\mathcal{T}^2$ do not collide;
- (1c) does not occur $\Longrightarrow$ the tuple elements in $\mathcal{T}^2$ and in $\mathcal{T}^3$ do not collide;

where the notations $\mathcal{T}^1$, $\mathcal{T}^2$ and $\mathcal{T}^3$ are defined in Sect. 6.1. Combining them together, we can conclude that every view in $\mathcal{V}_{\mathrm{good}}$ does not cause the bad event in Sect. 6.1. Hence $\epsilon_{v_{\mathrm{good}}} = 0$ holds. Therefore it has that

$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{E1}}(q) \leq \frac{10q}{2^n}$$

6.4 Provable Security

Putting all together, we obtain the following theorem on the provable security of $\widetilde{E1}, \ldots, \widetilde{E32}$.

Theorem 1. *Let $\widetilde{E}$ be any tweakable blockcipher construction from the set of $\widetilde{E1}, \ldots, \widetilde{E32}$ depicted in Figs. 6 and 7. Let q be an integer such that $q < 2^{n-1}$. Then the following bound holds.*

$$\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{E}}(q) \leq \frac{10q}{2^n}.$$

7 Conclusions and Discussions

This paper has proposed 32 tweakable blockcipher constructions that achieve full provable security via a minimum number of blockcipher calls, in the ideal blockcipher model. A direction of future work would be to investigate if such fully secure tweakable blockciphers can be constructed in the standard pseudo-random-permutation model with a constant number of blockcipher calls.

On Key Check Value. As highlighted in [27], ANSI X9.24-1 [2] suggests the use of the key check value KCV for the integrity verification of the blockcipher key, which may cause security loss for cryptographic primitives. In details, ANSI X9.24-1 suggests $\mathsf{KCV} = E_k(0)$.[7] Moreover, KCV is a public value, and will be transmitted, sent or stored in clear. In other words, an attacker has chance to learn the value of KCV. It has a serious security impact to our constructions $\widetilde{E1}, \widetilde{E2}, \ldots, \widetilde{E10}$, whose subkey y is computed as $y = E(k, 0)$. As we can see, $\mathsf{KCV} = y$ holds, and hence an attacker can get the value of the subkey, and then is able to recover the key k with a complexity of $2^{n/2}$ queries. We propose alternatives to these tweakable blockciphers when KCV is used: replace 0 by a non-zero constant `const`, and derive the subkey y from the key k as $y = E(k, \texttt{const})$. On other hand, the usage of KCV has negligible impact to the security of the other tweakable blockcipher constructions $\widetilde{E11}, \ldots, \widetilde{E32}$.

Acknowledgements. Lei Wang and Dawu Gu are sponsored by the Natural Science Foundation of Shanghai (16ZR1416400), Major State Basic Research Development Program (973 Plan), the National Natural Science Foundation of China (61472250), and Innovation Plan of Science and Technology of Shanghai (14511100300). Guoyan Zhang is sponsored by National Natural Science Foundation of China (61602276). Jingyuan Zhao is sponsored by the National Science Foundation of China (no. 61379139) and the Strategic Priority Research Program of the Chinese Academy of Sciences (no. XDA06100701).

References

1. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Tischhauser, E., Yasuda, K.: Parallelizable and authenticated online ciphers. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8269, pp. 424–443. Springer, Heidelberg (2013). doi:10.1007/978-3-642-42033-7_22
2. ANSI: Retail Financial Services Symmetric Key Management Part 1: Using Symmetric Techniques. ANSI X9.24-1: 2009 (2009)
3. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: SIMON and SPECK: Block Ciphers for the Internet of Things. Cryptology ePrint Archive, Report 2015/585 (2015). http://eprint.iacr.org/
4. Biryukov, A., Wagner, D.: Advanced slide attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 589–606. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_41
5. CAESAR Competition. http://competitions.cr.yp.to/caesar.html
6. Chakraborty, D., Sarkar, P.: A General construction of tweakable block ciphers and different modes of operations. In: Lipmaa, H., Yung, M., Lin, D. (eds.) Inscrypt 2006. LNCS, vol. 4318, pp. 88–102. Springer, Heidelberg (2006). doi:10.1007/11937807_8
7. Chakraborty, D., Sarkar, P.: HCH: A new tweakable enciphering scheme using the hash-counter-hash approach. IEEE Trans. Inf. Theory **54**(4), 1683–1699 (2008)

[7] More precisely, ANSI X9.24-1 suggests to use a few most significant bits of $E_k(0)$ as KCV.

8. Chen, S., Steinberger, J.: Tight security bounds for key-alternating ciphers. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 327–350. Springer, Heidelberg (2014). doi:10.1007/978-3-642-55220-5_19
9. Cogliati, B., Lampe, R., Seurin, Y.: Tweaking even-mansour ciphers. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 189–208. Springer, Heidelberg (2015). doi:10.1007/978-3-662-47989-6_9
10. Cogliati, B., Seurin, Y.: Beyond-birthday-bound security for tweakable even-mansour ciphers with linear tweak and key mixing. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9453, pp. 134–158. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48800-3_6
11. Cogliati, B., Seurin, Y.: On the provable security of the iterated even-mansour cipher against related-key and chosen-key attacks. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 584–613. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46800-5_23
12. Crowley, P.: Mercy: A fast large block cipher for disk sector encryption. In: Goos, G., Hartmanis, J., Leeuwen, J., Schneier, B. (eds.) FSE 2000. LNCS, vol. 1978, pp. 49–63. Springer, Heidelberg (2001). doi:10.1007/3-540-44706-7_4
13. Daemen, J.: Limitations of the even-mansour construction. In: [26], pp. 495–498
14. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer, Heidelberg (2002)
15. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in cryptography: the even-mansour scheme revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 336–354. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29011-4_21
16. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Condentiality on Storage Devices. NIST Special Publication 800–38E (2010)
17. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. In: [26], pp. 210–224
18. Farshim, P., Procter, G.: The related-key security of iterated even-mansour ciphers. In: [33], pp. 342–363
19. Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The SKEIN Hash Function Family. NIST SHA-3 Competition (2008)
20. Goldenberg, D., Hohenberger, S., Liskov, M., Schwartz, E.C., Seyalioglu, H.: On tweaking luby-rackoff blockciphers. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 342–356. Springer, Heidelberg (2007). doi:10.1007/978-3-540-76900-2_21
21. Granger, R., Jovanovic, P., Mennink, B., Neves, S.: Improved masking for tweakable blockciphers with applications to authenticated encryption. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 263–293. Springer, Heidelberg (2016). doi:10.1007/978-3-662-49890-3_11
22. Grosso, V., Leurent, G., Standaert, F., Varici, K., Journault, A., Durvaux, F., Gaspar, L., Kerckhof, S.: SCREAM Side-Channel Resistant Authenticated Encryption with Masking V3. CAESAR Competition Candidate (2015). http://competitions.cr.yp.to/round2/screamv3.pdf
23. Halevi, S.: EME*: Extending EME to handle arbitrary-length messages with associated data. In: Canteaut, A., Viswanathan, K. (eds.) INDOCRYPT 2004. LNCS, vol. 3348, pp. 315–327. Springer, Heidelberg (2004). doi:10.1007/978-3-540-30556-9_25

24. Halevi, S., Rogaway, P.: A tweakable enciphering mode. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 482–499. Springer, Heidelberg (2003). doi:10.1007/978-3-540-45146-4_28
25. Halevi, S., Rogaway, P.: A parallelizable enciphering mode. In: Okamoto, T. (ed.) CT-RSA 2004. LNCS, vol. 2964, pp. 292–304. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24660-2_23
26. Imai, H., Rivest, R.L., Matsumoto, T. (eds.): ASIACRYPT 1991. LNCS, vol. 739. Springer, Heidelberg (1993). doi:10.1007/3-540-57332-1_17
27. Iwata, T., Wang, L.: Impact of ANSI X9.24-1:2009 key check value on ISO/IEC 9797-1:2011 MACs. In: Cid, C., Rechberger, C. (eds.) FSE 2014. LNCS, vol. 8540, pp. 303–322. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46706-0_16
28. Jean, J., Nikolić, I., Peyrin, T.: Tweaks and keys for block ciphers: the `TWEAKEY` framework. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8874, pp. 274–288. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45608-8_15
29. Jean, J., Nikolic, I., Peyrin, T.: Deoxys v1.3. CAESAR Competition Candidate (2015). http://competitions.cr.yp.to/round2/deoxysv13.pdf
30. Jean, J., Nikolic, I., Peyrin, T.: Joltik v1.3. CAESAR Competition Candidate (2015). http://competitions.cr.yp.to/round2/joltikv13.pdf
31. Lampe, R., Seurin, Y.: Tweakable blockciphers with asymptotically optimal security. In: Moriai, S. (ed.) FSE 2013. LNCS, vol. 8424, pp. 133–151. Springer, Heidelberg (2014). doi:10.1007/978-3-662-43933-3_8
32. Landecker, W., Shrimpton, T., Terashima, R.S.: Tweakable blockciphers with beyond birthday-bound security. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 14–30. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32009-5_2
33. Leander, G. (ed.): FSE 2015. LNCS, vol. 9054, pp. 428–448. Springer, Heidelberg (2015). doi:10.1007/978-3-662-48116-5_21
34. Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 31–46. Springer, Heidelberg (2002). doi:10.1007/3-540-45708-9_3
35. Liskov, M., Rivest, R.L., Wagner, D.: Tweakable block ciphers. J. Cryptol. **24**(3), 588–613 (2011)
36. Mennink, B.: Optimally secure tweakable blockciphers. In: [33], pp. 428–448
37. Mennink, B.: Optimally Secure Tweakable Blockciphers. IACR Cryptology ePrint Archive 2015 363 (2015). http://eprint.iacr.org/2015/363
38. Mennink, B.: Private communication (2015)
39. Mennink, B.: XPX: Generalized tweakable even-mansour with improved security guarantees. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9814, pp. 64–94. Springer, Heidelberg (2016). doi:10.1007/978-3-662-53018-4_3
40. Minematsu, K.: Improved security analysis of XEX and LRW modes. In: Biham, E., Youssef, A.M. (eds.) SAC 2006. LNCS, vol. 4356, pp. 96–113. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74462-7_8
41. Minematsu, K.: Beyond-birthday-bound security based on tweakable block cipher. In: Dunkelman, O. (ed.) FSE 2009. LNCS, vol. 5665, pp. 308–326. Springer, Heidelberg (2009). doi:10.1007/978-3-642-03317-9_19
42. Minematsu, K., Iwata, T.: Tweak-length extension for tweakable blockciphers. In: Groth, J. (ed.) IMACC 2015. LNCS, vol. 9496, pp. 77–93. Springer, Heidelberg (2015). doi:10.1007/978-3-319-27239-9_5

43. Minematsu, K., Matsushima, T.: Tweakable enciphering schemes from hash-sum-expansion. In: Srinathan, K., Rangan, C.P., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 252–267. Springer, Heidelberg (2007). doi:10.1007/978-3-540-77026-8_19
44. Mitsuda, A., Iwata, T.: Tweakable pseudorandom permutation from generalized feistel structure. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) ProvSec 2008. LNCS, vol. 5324, pp. 22–37. Springer, Heidelberg (2008). doi:10.1007/978-3-540-88733-1_2
45. Patarin, J.: A proof of security in $O(2^n)$ for the Xor of two random permutations. In: Safavi-Naini, R. (ed.) ICITS 2008. LNCS, vol. 5155, pp. 232–248. Springer, Heidelberg (2008). doi:10.1007/978-3-540-85093-9_22
46. Procter, G.: A Note on the CLRW2 Tweakable Block Cipher Construction. Cryptology ePrint Archive, Report 2014/111 (2014). http://eprint.iacr.org/2014/111
47. Rogaway, P.: Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 16–31. Springer, Heidelberg (2004). doi:10.1007/978-3-540-30539-2_2
48. Rogaway, P., Bellare, M., Black, J., Krovetz, T.: OCB: a block-cipher mode of operation for efficient authenticated encryption. In: Reiter, M.K., Samarati, P., (eds.) ACM CCS 2001, pp. 196–205. ACM (2001)
49. Sarkar, P.: Efficient tweakable enciphering schemes from (block-wise) universal hash functions. IEEE Trans. Inf. Theory **55**(10), 4749–4760 (2009)
50. Schroeppel, R.: The Hasty Pudding Cipher. NIST AES Proposal (1998)
51. Wang, L.: SHELL v2.0. CAESAR Competition Candidate (2015). http://competitions.cr.yp.to/round2/shellv20.pdf
52. Wang, L., Guo, J., Zhang, G., Zhao, J., Gu, D.: How to Build Fully Secure Tweakable Blockciphers from Classical Blockciphers. Cryptology ePrint Archive, Report 2016/876 (2016). http://eprint.iacr.org/2016/876
53. Wang, P., Feng, D., Wu, W.: HCTR: A variable-input-length enciphering mode. In: Feng, D., Lin, D., Yung, M. (eds.) CISC 2005. LNCS, vol. 3822, pp. 175–188. Springer, Heidelberg (2005). doi:10.1007/11599548_15