A Modular Framework for Building Variable-Input-Length Tweakable Ciphers

Thomas Shrimpton and R. Seth Terashima

Dept. of Computer Science, Portland State University
{teshrim,seth}@cs.pdx.edu

Abstract. We present the Protected-IV construction (PIV) a simple, modular method for building variable-input-length tweakable ciphers. At our level of abstraction, many interesting design opportunities surface. For example, an obvious pathway to building beyond birthday-bound secure tweakable ciphers with performance competitive with existing birthday-bound-limited constructions. As part of our design space exploration, we give two fully instantiated PIV constructions, TCT_1 and TCT_2; the latter is fast and has beyond birthday-bound security, the former is faster and has birthday-bound security. Finally, we consider a generic method for turning a VIL tweakable cipher (like PIV) into an authenticated encryption scheme that admits associated data, can withstand nonce-misuse, and allows for multiple decryption error messages. Thus, the method offers robustness even in the face of certain sidechannels, and common implementation mistakes.

Keywords: tweakable blockciphers, beyond-birthday-bound security, authenticated encryption, associated data, full-disk encryption

1 Introduction

The main contribution of this paper is the Protected-IV construction (PIV), see Figure 1. PIV offers a simple, modular method for building length-preserving, tweakable ciphers that:

(1) may take plaintext inputs of essentially any length;
(2) provably achieves the strongest possible security property for this type of primitive, that of being a strong, tweakable-PRP (STPRP);
(3) admit instantiations from n-bit primitives that are STPRP-secure well beyond the birthday-bound of $2^{n/2}$ invocations.

Moreover, by some measures of efficiency, beyond-birthday secure instantiations of PIV are competitive with existing constructions that are only secure to the birthday bound. (See Table 1.) We will give a concrete instantiation of PIV that has beyond birthday-bound security and, when compared to EME [16], the overhead is a few extra modular arithmetic operations for each n-bit block of input.

Tweakable ciphers with beyond birthday-bound security may have important implications for cryptographic practice. For example, in large-scale data-at-rest settings, where the amount of data that must be protected by a single key is typically greater than in settings where keys can be easily renegotiated.

At least two important applications have already made tweakable ciphers their tool-of-choice, namely full-disk encryption (FDE) and format-preserving encryption (FPE). Our work provides interesting new results for both FDE and FPE.

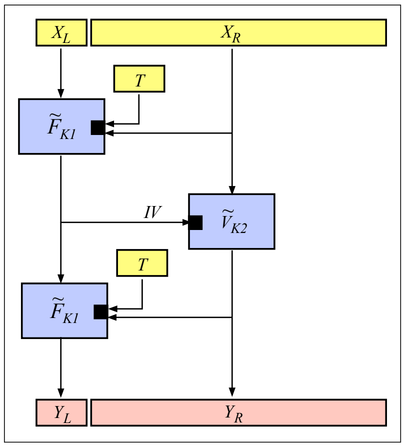


Fig. 1: The $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]$ tweakable cipher. Input T is the tweak, and $X = X_L \parallel X_R$ is a bit string, where $|X_L| = N$ and X_R is any length accepted by $\widetilde{V}$. The filled-in box is the tweak input.

We also show that tweakable ciphers enable a simple mechanism for building authenticated encryption schemes with associated data (AEAD), via an extension of the encode-then-encipher approach of Bellare and Rogaway [4]. This approach has some practical benefits, for example, it securely handles the reporting of multiple types of decryption errors. It can also eliminate ciphertext expansion by exploiting any existing nonces, randomness, or redundancies appearing in either the plaintext or associated data inputs. Combined with our other results, encode-then-encipher over PIV gives a new way to build AEAD schemes with beyond birthday-bound security.

Background. Tweakable blockciphers (TBCs) were introduced and formalized by Liskov, Rivest and Wagner [20]. An n-bit TBC $\widetilde{E}$ is a family of permutations over $\{0,1\}^n$, each permutation named by specifying a key and a *tweak*. In typical usage, the key is secret and fixed across many calls, while the tweak is not secret, and may change from call to call; this allows variability in the behavior of the primitive, even though the key is fixed. A tweakable cipher[1] is the natural extension of a tweakable blockcipher to the variable-input-length (VIL) setting, forming a family of length-preserving permutations.

Since the initial work of Liskov, Rivest and Wagner, there has been substantial work on building tweakable ciphers. Examples capable of handling long inputs (required for FDE) include CMC [15], EME [16], HEH [30], HCH [10], and HCTR [33]. Loosely speaking, the common approach has been to build up the VIL primitive from an underlying n-bit blockcipher, sometimes in concert with one or more hashing operations. The security guaranteed by each of these constructions become vacuous after about $2^{n/2}$ bits have been enciphered.

[1] Sometimes called a “tweakable enciphering scheme”, or even a “large-block cipher”.

One of our main goals is to break through this birthday bound, i.e., to build a tweakable cipher that remains secure long after $2^{n/2}$ bits have been enciphered.

The PIV *construction.* To this end, we begin by adopting a top-down, compositional viewpoint on the design of tweakable ciphers, our PIV construction. It is a type of three-round, unbalanced Feistel network, where the left "half" of the input is of a fixed bit length N, and the right "half" has variable length. The first and third round-functions are an N-bit tweakable blockcipher ($\widetilde{F}$), where N is a parameter of the construction, e.g. $N = 128$ or $N = 256$. The middle round-function ($\widetilde{V}$) is itself a VIL tweakable cipher, whose tweak is the output of first round.

It may seem as though little has been accomplished, since we need a VIL tweakable cipher $\widetilde{V}$ in order to build our VIL tweakable cipher $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]$. However, we require substantially less of $\widetilde{V}$ than we do of $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]$. In particular, the target security property for PIV is that of being a strong tweakable pseudorandom permutation. Informally, being STPRP-secure means withstanding chosen-ciphertext attacks in which the attacker also has full control over all inputs. The attacker can, for example, repeat a tweak an arbitrary number of times. Our PIV security theorem (Theorem 1) says the following: given (1) a TBC $\widetilde{F}$ that is STPRP-secure over a domain of N-bit strings, and (2) a tweakable cipher $\widetilde{V}$ that is secure against attacks *that never repeat a tweak*, then the tweakable cipher $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]$ is STPRP-secure. Thus, qualitatively, the PIV construction promotes security (over a large domain) against a restricted kind of attacker, into security against arbitrary chosen-ciphertext attacks.

Quantitatively, the PIV security bound contains an additive term $q^2/2^N$, where q is the number of times PIV is queried. Now, N might be the blocksize n of some underlying blockcipher; in this case the PIV composition delivers a bound comparable to those achieved by existing constructions. But $N = 2n$ presents the possibility of using an n-bit primitive to instantiate $\widetilde{F}$ and $\widetilde{V}$, and yet deliver a tweakable cipher with security well beyond beyond-birthday of $2^{n/2}$ queries.

As a small, additional benefit, the PIV proof of STPRP-security is short and easy to verify.

Impacts of modularity on instantiations. Adopting this modular viewpoint allows us to explore constructions of $\widetilde{F}$ and $\widetilde{V}$ independently. This is particularly beneficial, since building efficient and secure instantiations of VIL tweakable ciphers ($\widetilde{V}$) is relatively easy, when tweaks can be assumed not to repeat. The more difficult design task, of building a tweakable blockcipher ($\widetilde{F}$) that remains secure when tweaks may be repeated, is also made easier, by restricting to plaintext inputs of a fixed bit length N. In practice, when (say) $N = 128$ or 256, inefficiencies incurred by $\widetilde{F}$ can be offset by efficiency gains in $\widetilde{V}$.

To make thing concrete, we give two fully-specified PIV tweakable ciphers, each underlain by n-bit blockciphers. The first, TCT_1, provides birthday-bound security. It requires only one blockcipher invocation and some arithmetic, modulo a power of two, per n-bit block of input. In contrast, previous modes either

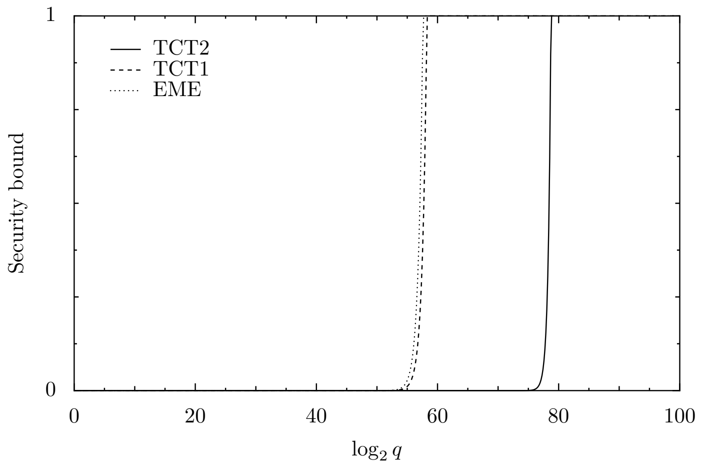


Fig. 2: Security bounds for TCT_1, EME and TCT_2, all using an underlying 128-bit primitive and 4096-byte inputs, typical for FDE. The EME curve is representative of other prior constructions.

require two blockcipher invocations per n-bit block, or require per-block finite field operations.

The second, TCT_2, delivers security beyond the birthday-bound. When compared to existing VIL tweakable ciphers with only birthday-bound security, like EME* construction, TCT_2 incurs only some additional, simple arithmetic operations per n bit block of input. Again, this arithmetic is performed modulo powers of two, rather than in a finite field.

In both TCT_1 and TCT_2, the VIL component is instantiated using counter-mode encryption, but over a TBC instead of a blockcipher. The additional tweak input of the TBC allows us to consider various 'tweak-scheduling' approaches, e.g. fixing a single per-message tweak across all blocks, or changing the tweak each message block.[2] We will see that the latter approach of re-tweaking on a block-by-block basis leads to a beyond birthday-bound secure PIV construction that admits strings of any length at least N.

AEAD via encode-then-(tweakable)encipher. The ability to construct beyond birthday-bound secure tweakable ciphers with large and flexible domains motivates us to consider their use for traditional encryption. Specifically, we build upon the "encode-then-encipher" results of Bellare and Rogaway [4]. They show that messages endowed with randomness (or nonces) and redundancy do not need to be processed by a authenticated encryption (AE) scheme in order to enjoy privacy and authenticity guarantees; a VIL strong-PRP suffices. This is

[2] There is a natural connection between changing the tweak of a TBC, and changing the key of a blockcipher. Both can be used to boost security, but the former is cleaner because tweaks do not need to be secret.

valuable when typical messages are short, as there is no need to waste bandwidth upon transmitting an AE scheme's IV and a dedicated authenticity tag.

We find that the tweakable setting gives additional advantages to the encode-then-encipher approach. An obvious one is that the tweak empowers support for associated data. More interesting, one can explore the effects of randomness, state or redundancy present in the message *and* tweak inputs. We find that randomness and state can be shifted to from the message to the tweak without loss of security, potentially reducing the number of bits that must be processed cryptographically.

We also find that AEAD schemes are built this way, via encode-then-encipher over a tweakable cipher, can accommodate multiple decryption error messages. Multiple, descriptive error messages can be quite useful in practice, but have often empowered damaging attacks (e.g. padding-oracle attacks [32, 7, 27, 1, 12]). These attacks don't work against our AEAD schemes because, loosely, changing any bit of a ciphertext will randomize every bit of the decrypted string.

Our work in this direction suggests useful implications for FPE [3, 5], and for layered-encryption schemes, for example the onion-encryption scheme used by Tor [23].

Due to space limitations, we refer the reader to the full version of this paper for our results on AEAD, and a discussion of their potential impacts.

Related work. Here we give a much abbreviated discussion of other related work. Please refer to Table 1 for a summary comparison of $\mathsf{TCT}_1, \mathsf{TCT}_2$ with other constructions. A more complete discussion will appear in the full version.

Researchers have developed three general approach for constructing tweakable ciphers from n-bit blockciphers. Each approach has yielded a series of increasingly refined algorithms. The first, Encrypt-Mask-Encrypt, places a lightweight "masking" layer between two encryption layers; examples include CMC [15] and EME* [13]. The second, Hash-ECB-Hash, sandwiches ECB-mode encryption between two invertible hashes. PEP [9], TET [14], and HEH [30, 31] are examples. Finally, Hash-CTR-Hash uses non-invertible hashes with CTR-mode encryption. Both HCH [10] and HCTR [33] use this approach. Mancillas-Lópeze et al. [22] report on the hardware performance of most of these modes. Chakraborty et al. [8] discuss implementations of the more recent HEH [30] construction and its refinement [31], which halves the number of finite field multiplications.

We contribute a new, top-down approach that leads us to the first beyond-birthday-bound secure tweakable cipher suitable for encrypting long inputs (i.e., longer than the blocksize of an underlying blockcipher). Table 1 and Figure 2 compare some of these algorithms with our new TCT_1 and TCT_2 constructions in terms of computational cost and security, respectively. Note that the finite field operations counted in Table 1 take hundreds of cycles in software [21, 2], whereas their cost relative to an AES blockcipher invocation is much lower in hardware [22]. TCT_1 is the first tweakable cipher to require only a single blockcipher invocation and no extra finite field multiplications for each additional n bits of input, while TCT_2 is the first to provide beyond-birthday-bound security (and still gets away with a fixed number of finite field multiplications).

	Cost				
Cipher	[BC]	[$\mathbb{F}_{2^n}\times$]	[$\mathbb{Z}_w+$]	[$\mathbb{Z}_{2^w}$]	Ref.
HCTR	ℓ	$2\ell+2$	–	–	[33]
CMC	$2\ell+1$	–	–	–	[15]
EME	$2\ell+1$	–	–	–	[16]
EME*	$2\ell+3$	–	–	–	[13]
PEP	$\ell+5$	$4\ell-6$	–	–	[9]
HCH	$\ell+3$	$2\ell-2$	–	–	[10]
TET	ℓ	2ℓ	–	–	[14]
HEH	$\ell+1$	$\ell+2$	–	–	[30, 31]
TCT_1	$\ell+1$	5	$2\ell\left(\frac{n}{w}\right)^2$	$2\ell\left(\frac{n}{w}\right)^2$	–
TCT_2	$2\ell+8$	32	$4\ell\left(\frac{n}{w}\right)^2$	$4\ell\left(\frac{n}{w}\right)^2$	–

Table 1: Tweakable ciphers and their computational costs for ℓn-bit inputs. Costs measured in n-bit blockcipher calls [BC], finite field multiplications [$\mathbb{F}_{2^n}\times$], and ring operations [$\mathbb{Z}_w+$] and [$\mathbb{Z}_{2^w}$], for some word size w. Typically, $\ell = 32$ for FDE, and we anticipate $n = 128$, $w = 64$.

We mention the LargeBlock constructions due to Minematsu and Iwata [25], since they provide ciphers with beyond-birthday-bound security. These do not support tweaking, but it seems plausible that they could without significant degradation of performance or security. These constructions overcome the birthday bound by using $2n$-bit blockciphers as primitives, which are in turn constructed from an n-bit TBC. To our knowledge, CLRW2 [19] is the most efficient n-bit TBC with beyond-birthday-bound security that supports the necessary tweakspace (Minematsu's TBC [24] limits tweak lengths to fewer than $n/2$ bits). Compared to TCT_2, instantiating the LargeBlock constructions with this primitive ultimately requires an extra six finite field multiplications for each n bits of input. Thus, we suspect the LargeBlock designs would be impractical even if adding tweak support proves feasible.

A construction due to Coron, et al. [11], which we refer to as CDMS (after the authors), builds a $2n$-bit TBC from an n-bit TBC, providing beyond-birthday-bound security in n. Like PIV, CDMS uses three rounds of a Feistel-like structure. However, our middle round uses a VIL tweakable cipher, and we require a weaker security property from the round. This allows PIV to efficiently process long inputs. That said, CDMS provides an excellent way to implement a highly-secure $2n$-bit TBC, and we will use it for this purpose inside of TCT_2 to build $\widetilde{F}$. (Nesting CDMS constructions could create $(2^m n)$-bit tweakable blockciphers for any $m > 1$, but again, this would not be practical). We note that Coron, et al. were primarily concerned with constructions indifferentiable from an ideal cipher, a goal quite different from ours.

The Thorp shuffle [26] and its successor, swap-or-not [17], are highly-secure ciphers targeting very small domains (e.g., $\{0,1\}^n$ for $n \leq 64$). Swap-or-not could almost certainly become a VIL tweakable cipher, without changing the security bounds, by using domain separation for each input length and tweak in the underlying PRF. Essentially, one would make an input-length parameterized family of (tweakable) swap-or-not ciphers, with independent round-keys for each length. While still offering reasonable performance and unmatched security for very small inputs, the result would be wildly impractical for the large domains we are considering: swap-or-not's PRF needs to be invoked at least $6b$ times to securely encipher a b-bit input (below that, the bound becomes vacuous against even $q = 1$ query), and disk sectors are often 4096 bytes. Also, to match TCT_2's security, the PRF itself would need to be secure beyond the birthday bound (with respect to n).

Finally, we note that Rogaway and Shrimpton [29] considered some forms of tweakable encode-then-encipher in the context of deterministic AE ("keywrapping"), and our work generalizes theirs.

2 Tweakable Primitives

Preliminary notation. Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ be the set of non-negative integers. For $n \in \mathbb{N}$, $\{0,1\}^n$ denotes the set of all n-bit binary strings, and $\{0,1\}^*$ denotes the set of all (finite) binary strings. We write ε for the empty string. Let $s, t \in \{0,1\}^*$. Then $|s|$ is the length of s in bits, and $|(s,t)| = |s \,\|\, t|$, where $s \,\|\, t$ denotes the string formed by concatenating s and t. If $s \in \{0,1\}^{nm}$ for some $m \in \mathbb{N}$, $s_1 s_2 \cdots s_m \xleftarrow{n} s$ indicates that each s_i should be defined so that $|s_i| = n$ and $s = s_1 s_2 \cdots s_m$. When n is implicit from context, it will be omitted from the notation. If $s = b_1 b_2 \cdots b_n$ is an n-bit string (each $b_i \in \{0,1\}$), then $s[i..j] = b_i b_{i+1} \cdots b_j$, $s[i..] = s[i..n]$, and $s[..j] = s[1..j]$. The string $s \oplus t$ is the bitwise xor of s and t; if, for example, $|s| < |t|$, then $s \oplus t$ is the bitwise xor of s and $t[..|s|]$. Given $R \subseteq \mathbb{N}$ and $n \in \mathbb{N}$ with $n \le \min(R)$, $\{0,1\}^R = \bigcup_{i \in R} \{0,1\}^i$, and by abuse of notation, $\{0,1\}^{R-n} = \bigcup_{i \in R} \{0,1\}^{i-n}$. Given a finite set $\mathcal{X}$, we write $X \xleftarrow{\$} \mathcal{X}$ to indicate that the random variable X is sampled uniformly at random from $\mathcal{X}$. Throughout, the distinguished symbol $\bot$ is assumed not to be part of any set except $\{\bot\}$. Given an integer n known to be in some range, $\langle n \rangle$ denotes some fixed-length (e.g., 64-bit) encoding of n.

Let $H : \mathcal{K} \times \mathcal{D} \to \mathcal{R} \subseteq \{0,1\}^*$ be a function. Writing its first argument as a subscripted key, H is ϵ-almost universal (ϵ-AU) if for all distinct $X, Y \in \mathcal{D}$, $\Pr[\, H_K(X) = H_K(Y) \,] \le \epsilon$ (where the probability is over $K \xleftarrow{\$} \mathcal{K}$). Similarly, H is ϵ-almost 2-XOR universal if for all distinct $X, Y \in \mathcal{D}$ and $C \in \mathcal{R}$, $\Pr[\, H_K(X) \oplus H_K(Y) = C \,] \le \epsilon$.

An adversary is an algorithm taking zero or more oracles as inputs, which it queries in a black-box manner before returning some output. Adversaries may be random. The notation $A^f \Rightarrow b$ denotes the event that an adversary A outputs b after running with oracle f as its input.

Syntax. Let $\mathcal{K}$ be a non-empty set, and let $\mathcal{T}, \mathcal{X} \subseteq \{0,1\}^*$. A *tweakable cipher* is a mapping $\widetilde{E} : \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$ with the property that, for all $(K, T) \in \mathcal{K} \times \mathcal{T}$, $\widetilde{E}(K, T, \cdot)$ is a permutation on $\mathcal{X}$. We typically write the first argument (the key) as a subscript, so that $\widetilde{E}_K(T, X) = \widetilde{E}(K, T, X)$. As $\widetilde{E}_K(T, \cdot)$ is invertible, we let $\widetilde{E}_K^{-1}(T,\cdot)$ denote this mapping. We refer to $\mathcal{K}$ as the *key space*, $\mathcal{T}$ as the *tweak space*, and $\mathcal{X}$ as the *message space*. We say that a tweakable cipher $\widetilde{E}$ is *length preserving* if $|\widetilde{E}_K(T,X)| = |X|$ for all $X \in \mathcal{X}$, $T \in \mathcal{T}$, and $K \in \mathcal{K}$. All tweakable ciphers in this paper will be length preserving. Restricting the tweak or message spaces of a tweakable cipher gives rise to other objects. When $\mathcal{X} = \{0,1\}^n$ for some $n > 0$, then $\widetilde{E}$ is a *tweakable blockcipher* with blocksize n. When $|\mathcal{T}| = 1$, we make the tweak implicit, giving a *cipher* $E\colon \mathcal{K} \times \mathcal{X} \to \mathcal{X}$, where $E_K(\cdot)$ is a (length-preserving) permutation over $\mathcal{X}$ and E_K^{-1} is its inverse. Finally, when $\mathcal{X} = \{0,1\}^n$ and $|\mathcal{T}| = 1$, we have a conventional *blockcipher* $E : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$.

Security notions. Let $\mathrm{Perm}\,(\mathcal{X})$ denote the set of all permutations on $\mathcal{X}$. Similarly, we define $\mathrm{BC}(\mathcal{K},\mathcal{X})$ be the set of all ciphers with keyspace $\mathcal{K}$ and message space $\mathcal{X}$. When $\mathcal{X},\mathcal{X}'$ are sets, we define $\mathrm{Func}(\mathcal{X},\mathcal{X}')$ to be the set of all functions $f\colon \mathcal{X} \to \mathcal{X}'$.

Fix a tweakable cipher $\widetilde{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$. We define the strong, tweakable pseudorandom-permutation (STPRP) advantage measure as $\mathbf{Adv}_{\widetilde{E}}^{\widetilde{\mathrm{sprp}}}(A) = \Pr\left[\, K \xleftarrow{\$} \mathcal{K} :\, A^{\widetilde{E}_K(\cdot,\cdot),\widetilde{E}_K^{-1}(\cdot,\cdot)} \Rightarrow 1 \,\right] - \Pr\left[\, \mathit{\Pi} \xleftarrow{\$} \mathrm{BC}(\mathcal{T},\mathcal{X}) :\, A^{\mathit{\Pi}(\cdot,\cdot),\mathit{\Pi}^{-1}(\cdot,\cdot)} \Rightarrow 1 \,\right]$. The TPRP advantage measure is defined analogously, by dropping the $\widetilde{E}_K^{-1}$ oracle from the first probability, and the $\mathit{\Pi}^{-1}$ oracle from the second. We assume that A never makes *pointless* queries. By this we mean that for the (S)TPRPexperiments, the adversary never repeats a query to an oracle. For the STPRP advantage measure, this also means that if A queries (T,X) to its leftmost oracle and receives Y in return, then it never queries (T,Y) to its rightmost oracle, and vice versa. These assumptions are without loss of generality.

The strong, indistinguishable-from-random-bits (SRND) advtantage is defined as $\mathbf{Adv}_{\widetilde{E}}^{\widetilde{\mathrm{srnd}}}(A) = \Pr\left[\, K \xleftarrow{\$} \mathcal{K} :\, A^{\widetilde{E}_K(\cdot,\cdot),\widetilde{E}_K^{-1}(\cdot,\cdot)} \Rightarrow 1 \,\right] - \Pr\left[\, A^{\$(\cdot,\cdot),\$(\cdot,\cdot)} \Rightarrow 1 \,\right]$, where the $\$(\cdot,\cdot)$ oracle always outputs a random string equal in length to its second input: $|\$(T,X)| = |X|$ for all T and X. As before, we assume that A never makes a pointless query. Here, these assumptions are not without loss of generality, but instead prevent trivial wins. Adversaries for the (S)TPRP and SRND advantages are *nonce-respecting* if the transcript of their oracle queries $(T_1,X_1),\ldots,(T_q,X_q)$ does not include $T_i = T_j$ for any $i \neq j$.

For a cipher $E\colon \mathcal{K} \times \mathcal{X} \to \mathcal{X}$, we define the strong, pseudorandom permutation (SPRP) advantage as $\mathbf{Adv}_{E}^{\mathrm{sprp}}(A) = \Pr\left[\, K \xleftarrow{\$} \mathcal{K} :\, A^{E_K(\cdot),E_K^{-1}(\cdot)} \Rightarrow 1 \,\right] - \Pr\left[\, \pi \xleftarrow{\$} \mathrm{Perm}\,(\mathcal{X}) :\, A^{\pi(\cdot),\pi^{-1}(\cdot)} \Rightarrow 1 \,\right]$. As above, the PRP advantage is defined analogously, by dropping the E_K^{-1} oracle from the first probability, and the π^{-1} oracle from the second. We again assume (without loss of generality) that the adversary does not make pointless queries.

For all security notions in this paper, we track three adversarial resources: the time complexity t, the number of oracle queries q, and the total length of these queries μ. The time complexity of A is defined to include the complexity of its enveloping probability experiment (including sampling of keys, oracle computations, etc.), and we define the parameter t to be the maximum time complexity of A, taken over both experiments in the advantage measure.[3]

3 The PIV Construction

We begin by introducing our high-level abstraction, PIV, shown in Figure 1. Let $\mathcal{T} = \{0,1\}^t$ for some $t \geq 0$, and let $\mathcal{Y} \subseteq \{0,1\}^*$ be such that if $Y \in \mathcal{Y}$, then $\{0,1\}^{|Y|} \subseteq \mathcal{Y}$. Define $\mathcal{T}' = \mathcal{T} \times \mathcal{Y}$. Fix an integer $N > 0$. Let $\widetilde{F}\colon \mathcal{K}' \times \mathcal{T}' \times \{0,1\}^N \to \{0,1\}^N$ be a tweakable blockcipher and let $\widetilde{V}\colon \mathcal{K} \times \{0,1\}^N \times \mathcal{Y} \to \mathcal{Y}$ be a tweakable cipher. From these, we produce a new tweakable cipher $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]\colon (\mathcal{K}' \times \mathcal{K}) \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$, where $\mathcal{X} = \{0,1\}^N \times \mathcal{Y}$. As shown in Figure 1, the PIV composition of $\widetilde{F}, \widetilde{V}$ is a three-round Feistel construction, working as follows. On input (T, X), let $X = X_L \parallel X_R$ where $|X_L| = N$. First, create an N-bit string $IV = \widetilde{F}_{K'}(T \parallel X_R, X_L)$. Next, use this IV to encipher X_R, creating a string $Y_R = \widetilde{V}_K(IV, X_R)$. Now create an N-bit string $Y_L = \widetilde{F}_{K'}(T \parallel Y_R, IV)$, and return $Y_L \parallel Y_R$ as the value of $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]_{K',K}(T, X)$. The inverse $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]^{-1}_{K',K}(T, Y)$ is computed in the obvious manner.

At first glance, it seems that nothing interesting has been accomplished: we took an N-bit TBC and a tweakable cipher, and produced a tweakable cipher with a slightly larger domain. However, the following theorem statement begins to surface what our abstraction delivers.

Theorem 1. *Let sets $\mathcal{T}, \mathcal{Y}, \mathcal{T}', \mathcal{X}$ and integer N be as above. Let $\widetilde{F}\colon \mathcal{K}' \times \mathcal{T}' \times \{0,1\}^N \to \{0,1\}^N$ be a tweakable blockcipher, and let $\widetilde{V}\colon \mathcal{K} \times \{0,1\}^N \times \mathcal{Y} \to \mathcal{Y}$ be a tweakable cipher. Let $\mathsf{PIV}[\widetilde{F}, \widetilde{V}]$ be as just described. Let A be an adversary making $q < 2^N/4$ queries totaling μ bits and running in time t. Then there exist adversaries B and C, making q and $2q$ queries, respectively, and both running in $O(t)$ time such that $\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\mathsf{PIV}[\widetilde{F},\widetilde{V}]}(A) \leq \mathbf{Adv}^{\widetilde{\mathrm{srnd}}}_{\widetilde{V}}(B) + \mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{F}}(C) + \frac{4q^2}{2^N}$, where B is* nonce-respecting *and whose queries total $\mu - qN$ bits in length.*

The first thing to notice is that the VIL portion of the PIV composition, $\widetilde{V}$, need be SRND-secure against *nonce-respecting* adversaries only. As we will see in the next section, it is easy to build efficient schemes meeting this requirement. Only the FIL portion, $\widetilde{F}$, needs to be secure against STPRP adversaries that can use arbitrary querying strategies. Thus the PIV composition promotes nonce-respecting security over a large domain into full STPRP security over a slightly larger domain.

The intuition for why this should work is made clear by the picture. Namely, if $\widetilde{F}$ is a good STPRP, then if any part of T or X is "fresh", then the string IV should be random. Hence it is unlikely that an IV value is repeated, and

[3] We do this simply to make our theorem statements easier to read. A more explicit accounting of time resources in reductions, e.g. separating the running time of A from the time to run cryptographic objects "locally", would not significantly alter any of our results.

so nonce-respecting security of the VIL component is enough. Likewise when deciphering, if any part of T, Y is “fresh”.

The term $4q^2/2^N$ accounts for collisions in IV and the difference between $\widetilde{F}$ and a random function. This is a birthday-bound term in N, the blocksize of $\widetilde{F}$. Since most TBC designs employ (one or more) underlying blockciphers, we have deliberately chosen the notation N, rather than n, to stress that the blocksize of $\widetilde{F}$ can be larger than that of some underlying blockcipher upon which it might be built. Indeed, we’ll see in the next section that, given an n-bit blockcipher (and a hash function), we can build $\widetilde{F}$ with $N = 2n$. This gives us hope of building beyond birthday-bound secure VIL STPRPs in a modular fashion; we will do so, and with relatively efficient constructions, too.

It will come as no surprise that, if one does away with the lower $\widetilde{F}$ invocation and returns $IV \parallel Y_R$, the resulting composition does not generically deliver a secure STPRP. On the other hand, it *is* secure as a TPRP (just not a *strong* TPRP). This can be seen through a straight-forward modification of the PIV security proof.

4 Concrete Instantiations of PIV

Instantiating a PIV composition requires two objects, a (fixed-input-length) tweakable blockcipher $\widetilde{F}$ with an N-bit blocksize, and a variable-input-length tweakable cipher $\widetilde{V}$. In this section we explore various ways to instantiate these two objects, under the guidance of Theorem 1 and practical concerns.

Theorem 1 suggests setting N to be as large as possible, so that the final term is vanishingly small for any realistic number of queries. But for this to be useful, one must already know how to build a TBC $\widetilde{F}$ with domain $\{0,1\}^N$ for a large N, and for which $\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{F}}(C)$ approaches $q^2/2^N$. To our knowledge, there are no efficient constructions that permit $\mathbf{Adv}^{\widetilde{\mathrm{sprp}}}_{\widetilde{F}}(C)$ to be smaller than $\mathcal{O}(q^3/2^{2n})$ when using an n-bit blockcipher as a starting point. (A recent result by Lampe and Seurin [18] shows how to beat this security bound, but at a substantial performance cost.) A construction by Coron, et al., which will be discussed in more detail shortly, does meet this bound[4] while providing $N = 2n$.

So we restrict our attention to building TBC $\widetilde{F}$ with small N. In particular, we follow the common approach of building TBCs out of blockciphers. Letting n be the blockcipher blocksize, we will consider $N = n$, and $N = 2n$. In the former case, Theorem 1 only promises us security up to roughly $q = 2^{n/2}$, which is the birthday bound with respect to the blockcipher. With this security bound in mind, we can use simple and efficient constructions of both $\widetilde{F}$ and the VIL tweakable cipher $\widetilde{V}$. On the other hand, when $N = 2n$, Theorem 1 lets us hope for security to roughly $q = 2^n$ queries. To realize this hope we will need a bit more from both $\widetilde{F}$ and $\widetilde{V}$, but we will still find reasonably efficient constructions delivering beyond birthday bound security.

[4] However, nesting this construction to provide a VIL tweakable cipher is prohibitively inefficient.

In what follows, we will sometimes refer to objects constructed in other works. These are summarized for convenience in Figure 5, found in Appendix A.

An efficient VIL tweakable cipher. We will start by considering general methods for constructing the VIL tweakable cipher, $\widetilde{V}$. Recall that $\widetilde{V}$ need only be secure against adversaries that never repeat a tweak. In Figure 3, we see an analogue of conventional counter-mode encryption, but over an n-bit TBC $\widetilde{E}$ instead of a blockcipher. Within a call (T, X) to TCTR, each n-bit block X_i of the input X is

procedure $\mathrm{TCTR}[\widetilde{E}]_K(T, X)$:	**procedure** $\mathrm{TCTR}[\widetilde{E}]_K^{-1}(T, Y)$:
$X_1, X_2, \ldots, X_\nu \xleftarrow{n} X$	$Y_1, Y_2 \ldots, Y_\nu \xleftarrow{n} Y$
for $i = 1$ to ν	for $i = 1$ to ν
$\quad T_i \leftarrow g(T, i)$; $Z_i \leftarrow \langle i \rangle$	$\quad T_i \leftarrow g(T, i)$; $Z_i \leftarrow \langle i \rangle$
$\quad Y_i \leftarrow \widetilde{E}_K(T_i, Z_i) \oplus X_i$	$\quad X_i \leftarrow Y_i \oplus \widetilde{E}_K(T_i, Z_i)$
Return $Y_1, Y_2, \ldots, Y_\nu$	Return $X_1, \ldots, X_\nu$

Fig. 3: The TCTR VIL tweakable cipher.

processed using a per-block tweak T_i, this being determined by a function $g\colon \mathcal{T}' \times \mathbb{N} \to \mathcal{T}$ of the input tweak T and the block index i.

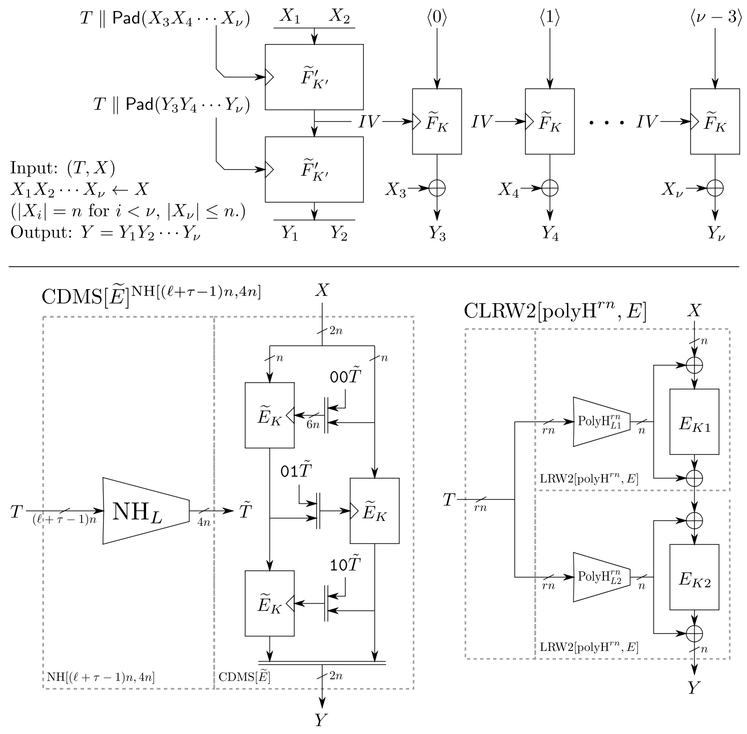
Consider the behavior of TCTR when $g(T, i) = T$. The following result is easily obtained using standard techniques.

Theorem 2. *Let $\widetilde{E}\colon \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher, and let $\mathrm{TCTR}[\widetilde{E}]_K$ and $\mathrm{TCTR}[\widetilde{E}]_K^{-1}$ be defined as above, with $g(T, i) = T \in \mathcal{T}$. Let A be a nonce-respecting adversary that runs in time t, and asks q queries, each of length at most ℓn bits (so, $\mu \le q\ell n$). Then for some adversary B making at most $q\ell$ queries and running in time $\mathcal{O}(t)$, $\mathbf{Adv}^{\widetilde{\mathrm{srnd}}}_{\mathrm{TCTR}[\widetilde{E}]}(A) \le \mathbf{Adv}^{\widetilde{\mathrm{prp}}}_{\widetilde{E}}(B) + 0.5q\ell^2/2^n$.*

We note that the bound displays birthday-type behavior when $\ell = o(\sqrt{q})$, and is tightest when ℓ is a small constant. An important application with small, constant ℓ is full-disk encryption. Here plaintexts X would typically be 4096 bytes long, so if the underlying TBC has blocksize $n = 128$, we get $\ell = 256$ blocks.[5]

Extending tweakspaces. In PIV, the TBC $\widetilde{F}$ will need to handle long tweaks. Fortunately, a result by Coron, et al. [11] shows that one can compress tweaks using an ϵ-AU hash function at the cost of adding a $q^2\epsilon$ term to the tweakable cipher's TPRP security bound. In particular, we will use (a slight specialization

[5] Actually, slightly less than this when used in the PIV composition, since the first N bits are enciphered by $\widetilde{F}$.



$T \parallel \mathsf{Pad}(X_3X_4\cdots X_\nu)$
X_1 X_2
$\widetilde{F}'_{K'}$
$T \parallel \mathsf{Pad}(Y_3Y_4\cdots Y_\nu)$
$\widetilde{F}'_{K'}$
Y_1 Y_2
$\langle 0 \rangle$
$\langle 1 \rangle$
$\langle \nu - 3 \rangle$
IV
$\widetilde{F}_K$
X_3 X_4 X_ν
Y_3 Y_4 Y_ν
Input: (T, X)
$X_1X_2\cdots X_\nu \leftarrow X$
$(|X_i| = n$ for $i < \nu$, $|X_\nu| \le n.)$
Output: $Y = Y_1Y_2\cdots Y_\nu$
$\mathrm{CDMS}[\widetilde{E}]^{\mathrm{NH}[(\ell+\tau-1)n,4n]}$
X
$2n$
n
$00\widetilde{T}$
$\widetilde{E}_K$
$6n$
$01\widetilde{T}$
T
$(\ell+\tau-1)n$
NH_L
$4n$
$\widetilde{T}$
$10\widetilde{T}$
$\mathrm{NH}[(\ell+\tau-1)n, 4n]$
$\mathrm{CDMS}[\widetilde{E}]$
Y
$\mathrm{CLRW2}[\mathrm{polyH}^{rn}, E]$
rn
PolyH^{rn}_{L1}
E_{K1}
$\mathrm{LRW2}[\mathrm{polyH}^{rn}, E]$
PolyH^{rn}_{L2}
E_{K2}

Fig. 4: The TCT_2 construction (top). TCT_2 takes τn-bit tweaks, and the input length is between $2n$ and ℓn bits, inclusive. Here, $\widetilde{F}$ is implemented using the $2n$-bit CDMS construction coupled with the NH hash function (bottom left). Both $\widetilde{V}$ and the TBC $\widetilde{E}$ used inside of CDMS are implemented using CLRW2[polyH^{rn}, E] (bottom right), with $r = 6$ and $r = 2$, respectively. The function Pad maps s to $s \parallel 10^{(\ell+1)n-1-|s|}$. In the diagram for CDMS, the strings $00\widetilde{T}$, $01\widetilde{T}$, and $10\widetilde{T}$ are padded with 0s to length $5n$ before being used.

of) the NH hash, defined by Black, et al. [6]; NH$[r, s]_L$ takes r-bit keys ($|L| = r$), maps r-bit strings to s-bit strings, and is $2^{s/2}$-AU. Please see Table 5 for the description. Given a TBC $\widetilde{E}$, $\widetilde{E}^{\mathrm{NH}}$ denotes the resulting TBC, whose tweakspace is now the domain of NH, rather than its range.

4.1 Targeting efficiency at birthday-type security: TCT_1

Let us begin with the case of $N = n$. To instantiate the n-bit TBC $\widetilde{F}$ in PIV we refer to the pioneering TBC work of Liskov, Rivest and Wagner [20], from which we draw the LRW2 TBC; please refer to Figure 5 for a description.

Before we give the TCT_1 construction, a few notes. In Figure 5 we see that in addition to a blockcipher E, LRW2$[H, E]$ uses an ϵ-AXU$_2$ hash function, H, and so, in theory, it could natively accommodate large tweaks. But for practical purposes, it will be more efficient to implement LRW2 with a small tweakspace, and then extend this using a fast ϵ-AU hash function.[6] For the ϵ-AXU$_2$ hash function itself, we use the polynomial hash polyH (also described in Table 5).

Now are ready to give our TCT_1 construction, which is birthday-bound secure for applications with small plaintext messages (e.g. FDE).

The TCT_1 Construction. Fix $k, n > 0$, and let $N = n$. Let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, and let polyH^{mn}, and NH be as defined in Table 5. Then define $\mathsf{TCT}_1 = \mathsf{PIV}[\widetilde{F}, \widetilde{V}]$, where to obtain a τn-bit tweakspace and domain $\{0,1\}^{\{n,n+1,\ldots,\ell n\}}$ we set:

1. n-bit TBC $\widetilde{F} = \text{LRW2}[\text{polyH}^{2n}, E]^{\text{NH}[(\ell+\tau)n,2n]}$, i.e. LRW2 with its tweakspace extended using NH. The keyspace for $\widetilde{F}$ is $\{0,1\}^k \times \{0,1\}^{2n} \times \{0,1\}^{(\ell+\tau)n}$, with key K' partitioning into keys for E, polyH^{2n}, and NH$[(\ell+\tau)n, 2n]$. (Since NH supports only fixed length inputs, we implicitly pad NH inputs with a 1 and then as many 0s as are required to reach a total length of $(\ell+\tau)n$ bits.) The tweakspace for $\widetilde{F}$ is $\{0,1\}^{\{0,1,2,\ldots,(\ell+\tau-1)n\}}$.
2. VIL tweakable cipher $\widetilde{V} = \text{TCTR}\,[\text{LRW2}[\text{polyH}^n, E]]$ with the TCTR function $g\colon \{0,1\}^n \times \mathbb{N} \to \{0,1\}^n$ as $g(T,i) = T$. The keyspace for $\widetilde{V}$ is $\{0,1\}^k \times \{0,1\}^n$, with key K partitioning into keys for E and polyH^n. The tweakspace for $\widetilde{V}$ is $\{0,1\}^n$, and its domain is $\{0,1\}^{\{0,1,\ldots,(\ell-1)n\}}$.

Putting together Theorems 1,2, and results from previous works [6, 20], we have the following security bound.

Theorem 3 (STPRP-security of TCT_1). *Define TCT_1 as above, and let A be an adversary making $q < 2^n/4$ queries and running in time t. Then there exist adversaries B and C, both running in time $\mathcal{O}(t)$ and making $(\ell-1)q$ and $2q$ queries, respectively, such that* $\mathbf{Adv}^{\widetilde{\text{sprp}}}_{\mathsf{TCT}_1[E]}(A) \le \mathbf{Adv}^{\text{prp}}_{E}(B) + \mathbf{Adv}^{\text{sprp}}_{E}(C) + \frac{32q^2}{2^n} + \frac{4q^2(\ell-1)^2}{2^n}$.

The proof appears in the full version. This algorithm requires $2k + (3+\tau+\ell)n$ bits of key material, including two keys for $\widetilde{E}$. As we show at the end of this section, we can get away with a single key for E with no significant damage to our security bound, although this improvement is motivated primarily by performance concerns.

Thus TCT_1 retains the security of previous constructions (see Figure 2 for a visual comparison), uses arithmetic in rings with powers-of-two moduli, rather than in a finite field. This may potentially improve performance in some architectures.

[6] Indeed, one can show composing an ϵ-AU hash function with an ϵ'-AXU$_2$ hash function yields an $(\epsilon+\epsilon')$-AXU$_2$ hash function; however, we prefer to work on a higher level of abstraction.

4.2 Aiming for beyond birthday-bound security: TCT_2

Now let us consider the PIV composition with $N = 2n$. For the FIL component, we can use Coron et al.'s [11] CDMS construction to get a $2n$-bit TBC from an n-bit TBC, and implement the latter using the CLRW2, a recent beyond-birthday-bound secure construction by Landecker, Shrimpton, and Terashima [19]. Table 5 describes both constructions.[7] We again extend the tweakspace using NH. (To stay above the birthday bound, we set the range of NH to $\{0,1\}^{2n}$). Ultimately, setting $\widetilde{F} = \text{CDMS}[\text{CLRW2}]^{\text{NH}}$ is secure against up to around $2^{2n/3}$ queries.

CLRW2 also gives us a way to realize a beyond birthday-bound secure VIL component, namely $\widetilde{V} = \text{TCTR}[\text{CLRW2}[E, H]$, at least for $\ell = o(q^{1/4})$. (We'll see how to avoid this restriction, if desired, in a moment.)

We are now ready to give our second fully concrete PIV composition, TCT_2, targeted at applications that would benefit from beyond birthday-bound security. This algorithm requires us to nest four layers of other constructions, so we provide an illustration in Figure 4. Again we emphasize that the (admittedly significant) cost of $\widetilde{F}$ can be amortized.

TCT_2 supports τn-bit tweaks and has domain $\{0,1\}^{\{2n, 2n+1, \ldots, \ell n\}}$.

The TCT_2 Construction. Fix $k, \ell, n, \tau > 0$, and let $N = 2n$. Let $E\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, and let $\text{polyH}^{\ell n}$, and NH be as defined in Table 5. Then define $\mathsf{TCT}_2 = \mathsf{PIV}[\widetilde{F}, \widetilde{V}]$, where:

1. $\widetilde{F} = \text{CDMS}\left[\text{CLRW2}[\text{polyH}^{6n}, E]\right]^{\text{NH}[(\ell+\tau-1)n, 4n]}$, that is, the $2n$-bit TBC $\text{CDMS}\left[\text{CLRW2}[\text{polyH}^{6n}, E]\right]$ with its tweakspace extended using NH. The keyspace for $\widetilde{F}$ is $\{0,1\}^{2k} \times \{0,1\}^{12n} \times \{0,1\}^{(\ell+\tau-1)n}$, with key K' partitioning into two keys for E, two keys for polyH^{6n}, and a key for $\text{NH}[\ell n, 4n]$. The tweakspace for $\widetilde{F}$ is $\{0,1\}^{\tau n}$.
2. $\widetilde{V} = \text{TCTR}\left[\text{CLRW2}[\text{polyH}^{2n}, E]\right]$, with the TCTR function $g\colon \{0,1\}^n \times \mathbb{N} \to \{0,1\}^n$ as $g(T, i) = T$. The keyspace for $\widetilde{V}$ is $\{0,1\}^{2k} \times \{0,1\}^{4n}$ with key K partitioning into two keys for E and two keys for polyH^{2n}. The tweakspace for $\widetilde{V}$ is $\{0,1\}^{2n}$, and its domain is $\{0,1\}^{\{0,1,2,\ldots,(\ell-2)n\}}$.

TCT_2 requires $4k + (\ell+\tau+15)n$ bits of key material. Putting together Theorems 1, 5, and results from previous works [6, 11, 19], we have the following security result.

Theorem 4 (STPRP-security of TCT_2). *Define TCT_2 as above, and let A be an adversary making q queries and running in time t, where $6q, \ell q < 2^{2n}/4$. Then there exist adversaries B and C, both running in $\mathcal{O}(t)$ time and making $(\ell-1)q$ and $6q$ queries, respectively, such that* $\mathbf{Adv}^{\widetilde{\text{sprp}}}_{\mathsf{TCT}_2}(A) \le 2\mathbf{Adv}^{\text{prp}}_{E}(B) + 2\mathbf{Adv}^{\text{sprp}}_{E}(C) + \frac{12q^2}{2^{2n}} + \frac{q(\ell-1)^2}{2^n} + \frac{6\ell^3 q^3}{2^{2n-2} - \ell^3 q^3} + \frac{6^4 q^3}{2^{2n-2} - 6^3 q^3}$.

[7] We note that for $\text{CDMS}[\widetilde{E}]$, we enforce domain separation via $\widetilde{E}$'s tweak, whereas the authors of [11] use multiple keys for $\widetilde{E}$. The proof of our construction follows easily from that of the original.

Again, the proof appears in the full version. Some of the constants in this bound are rather significant. However, as Figure 2 shows, TCT_2 nevertheless provides substantially better security bounds than TCT_1 and previous constructions.

4.3 Additional practical considerations

Several variations and optimizations on TCT_1 and TCT_2 are possible. We highlight a few of them here. None of these changes significantly impact the above security bounds, unless otherwise noted.

Reducing the number of blockcipher keys. In the case of TCT_1, we can use a single key for both LRW2 instances provided we enforce domain separation through the tweak. This allows us to use a single key for the underlying blockcipher, which in some situations may allow for significant implementation benefits (for example, by allowing a single AES pipeline). One method that accomplishes this is to replace $\mathrm{LRW2}[\mathrm{polyH}^{2n}, E]^{\mathrm{NH}[(\ell+1)n,2n]}$ with $\mathrm{LRW2}[\mathrm{polyH}^{3n}, E]^{f(\varepsilon,\cdot)}$ and $\mathrm{LRW2}[\mathrm{polyH}^{n}, E]$ with $\mathrm{LRW2}[\mathrm{polyH}^{3n}, E]^{f(\cdot,\varepsilon)}$. Here, f is a 2^{-n}-AU function with keyspace $\{0,1\}^{3n} \times \{0,1\}^{\ell n}$, taking inputs of the form (X, ε) (for some $X \in \{0,1\}^n$) or (ε, Y) (for some $Y \in \{0,1\}^{\{0,1,\ldots,\ell n\}}$), and outputting a $3n$-bit string. Let $f_L(X,\varepsilon) = 0^{2n} \parallel X$ and $f_L(\varepsilon, Y) = 1^n \parallel \mathrm{NH}[(\ell+1)n, 2n]_L(Y)$. The function f described here is a mathematical convenience to unify the signatures of the two LRW2 instances, thereby bringing tweak-based domain separation into scope; in practice, we imagine the two instances would be implemented independently, save for a shared blockcipher key. We note that TCT_2 can be modified in a similar manner to require only two blockcipher keys.

Performance optimizations. If we need only a tweakable (FIL) blockcipher, we can use $\mathrm{NH}[\ell n, 2n]$ in place of $\mathrm{NH}[(\ell+1)n, 2n]$ by adjusting our padding scheme appropriately. We emphasize that in the TCTR portion, the polyH functions only need to be computed once, since each LRW2 invocation uses the same tweak. The corresponding optimizations apply to TCT_2, as well.

A naïve implementation of TCT_2 would make a total 72 finite field multiplications during the two FIL phases (a result of evaluating polyH^{6n} twelve times). We can cache an intermediate value of the polyH^{6n} hash used inside of CDMS (four n-bit tweak blocks are constant per invocation), and this saves 32 finite field multiplications. Precomputing the terms of the polynomial hash corresponding to the domain-separation constants eliminates 12 more multiplications, leaving 28 in total. Four more are required during the VIL phase, giving the count of 32 reported in Table 1.

Handling large message spaces. Both TCT_1 and TCT_2 are designed with FDE applications in mind. In particular, they require ℓ to be fixed ahead of time, and require more than ℓn bits of key material.

These limitations are a consequence of using the NH hash function; however, a simple extension to NH (described by the original authors [6]) accommodates arbitrarily long strings. Fix a positive integer r and define $\mathrm{NH}^*_L(M_1M_2\cdots M_\nu) = \mathrm{NH}_L(M_1) \parallel \mathrm{NH}_L(M_2) \parallel \cdots \parallel \mathrm{NH}_L(M_\nu) \parallel \langle |M| \bmod rn\rangle$, where $|M_i| = rn$ for $i < \nu$, $|M_\nu| \le rn$, and NH_L abbreviates $\mathrm{NH}_L[rn, 2N]$. Thus defined, NH^* is 2^{-N}-almost universal, has domain $\{0,1\}^*$, and requires rn bits of key material. This modification shifts some of the weight to the polyH hash; we now require eight extra finite field multiplications for each additional rn bits of input. As long as $r > 4$, however, we require fewer of these multiplications when compared to previous hash-ECB-hash or hash-CTR-hash constructions.

With these modifications, the final two terms in TCT_1's security bound (Theorem 3) would become $8q^2/2^n + 600q^2\ell^2/r^22^n + 4q^2(\ell-1)^2/2^n$, where ℓn is now the length of the adversary's longest query, $\ell > 2.5r$, and the remaining terms measure the (S)PRP security of the underlying blockcipher. We also assume $2^n \ge rn$, so that $|M| \bmod rn$ can be encoded within a single n-bit block. Although the constant of 600 is large, we note that setting $r = 16$, for example, reduces it to a more comfortable size — in this case to less than three. The bound for TCT_2 changes in a similar manner. (Note that if $2^{n-2} \ge rn$, we can use a single n-bit block for both the tweak domain-separation constants and $\langle |M| \bmod rn\rangle$.)

Beyond birthday-bound security for long messages. When ℓ is not bounded to some small or moderate value, TCT_2 no longer provides beyond-birthday-bound security. The problematic term in the security bound is $q(\ell-1)^2/2^n$. To address this, we return to TCTR (Figure 3) and consider a different per-block tweak function.

In particular, $g(T, i) = T \parallel \langle i\rangle$. In the nonce-respecting case, the underlying TBC $\widetilde{E}$ is then retweaked with a never-before-seen value on each message block. Again, think about what happens when $\widetilde{E}$ is replaced by an ideal cipher $\mathit{\Pi}$: in the nonce-respecting case, every *block* of plaintext is masked by the output of a fresh random permutation.[8] In other words, every block returned will be uniformly random. Thus we expect a tight bound, in this case. Formalizing this logic yields the following theorem.

Theorem 5. *Let $\widetilde{E}\colon \{0,1\}^k \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher, and let $\mathrm{TCTR}[\widetilde{E}]_K$ and $\mathrm{TCTR}[\widetilde{E}]^{-1}_K$ be defined as above, with $g\colon \mathcal{T}' \times \mathbb{N} \to \mathcal{T}$ an arbitrary injective mapping. Let A be a nonce-respecting adversary that runs in time t, and asks q queries of total length at most $\mu = \sigma n$ bits. Then there exists some adversary B making at most σ queries and running in time $\mathcal{O}(t)$ such that $\mathbf{Adv}^{\widetilde{\mathrm{srnd}}}_{\mathrm{TCTR}[\widetilde{E}]}(A) \le \mathbf{Adv}^{\widetilde{\mathrm{prp}}}_{\widetilde{E}}(B)$.*

Consequently, using this variation of TCTR in Theorems 3 and 4 would remove the $q(\ell-1)^2$ term from the bounds, thereby lifting message length concerns. Note that if this change is made, $g(T, i)$ needs to be computed up to ℓ times per invocation, rather than just once. This problem may be mitigated by using the

[8] Notice that one could use (say) $Z_i \leftarrow 0^n$ and the same would be true. We present it as $Z_i \leftarrow \langle i\rangle$ for expositional purposes.

XEX [28] TBC in place of LRW2, which makes incrementing the tweak extremely fast without significantly changing our security bound.

When the above change are made, TCT_1 and TCT_2 offer efficient tweakable ciphers on an unbounded domain, losing security guarantees only after $\mathcal{O}(2^{n/2})$ (resp., $\mathcal{O}(2^{2n/3})$) bits have been enciphered. Finally, we note that one can use a conventional blockcipher mode of operation to build the VIL component. We report on this in the full version.

5 Acknowledgements

Portions of this work were carried out while Terashima was visiting Voltage Security. We thank them, especially Terence Spies, for their support. We also thank the CRYPTO and Asiacrypt 2013 reviewers for their diligence and useful feedback. Both Terashima and Shrimpton were supported by NSF grants CNS-0845610 and CNS-1319061.

References

1. N AlFardan and KG Paterson. Lucky 13: Breaking the TLS and DTLS record protocols. In *IEEE Symposium on Security and Privacy*, 2013.
2. D. Aranha, J. López, and D. Hankerson. Efficient software implementation of binary field arithmetic using vector instruction sets. *Progress in Cryptology–LATINCRYPT 2010*, pages 144–161, 2010.
3. Mihir Bellare, Thomas Ristenpart, Phillip Rogaway, and Till Stegers. Format-preserving encryption. In *Selected Areas in Cryptography*, pages 295–312. Springer, 2009.
4. Mihir Bellare and Phillip Rogaway. Encode-then-encipher encryption: How to exploit nonces or redundancy in plaintexts for efficient cryptography. In *ASIACRYPT*, pages 317–330, 2000.
5. Mihir Bellare, Phillip Rogaway, and Terence Spies. The FFX mode of operation for format-preserving encryption. Unpublished NIST proposal, 2010.
6. John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, and Phillip Rogaway. UMAC: Fast and secure message authentication. In *CRYPTO*, pages 216–233, 1999.
7. Brice Canvel, Alain Hiltgen, Serge Vaudenay, and Martin Vuagnoux. Password interception in a SSL/TLS channel. *Advances in Cryptology–CRYPTO 2003*, pages 583–599, 2003.
8. D. Chakraborty, C. Mancillas-López, F. Rodríguez-Henríquez, and P. Sarkar. Efficient hardware implementations of brw polynomials and tweakable enciphering schemes. *Computers, IEEE Transactions on*, 62(2):279–294, 2013.
9. Debrup Chakraborty and Palash Sarkar. A new mode of encryption providing a tweakable strong pseudo-random permutation. In Matthew Robshaw, editor, *Fast Software Encryption*, volume 4047 of *Lecture Notes in Computer Science*, pages 293–309. Springer Berlin / Heidelberg, 2006.
10. Debrup Chakraborty and Palash Sarkar. HCH: A new tweakable enciphering scheme using the hash-counter-hash approach. *IACR Cryptology ePrint Archive*, 2007:28, 2007.

11. Jean-Sébastien Coron, Yevgeniy Dodis, Avradip Mandal, and Yannick Seurin. A domain extender for the ideal cipher. In *TCC*, pages 273–289, 2010.
12. Jean Paul Degabriele and Kenneth G Paterson. On the (in)security of IPsec in MAC-then-encrypt configurations. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 493–504. ACM, 2010.
13. Shai Halevi. EME*: Extending EME to handle arbitrary-length messages with associated data. In *INDOCRYPT*, pages 315–327, 2004.
14. Shai Halevi. Invertible universal hashing and the TET encryption mode. In *CRYPTO*, pages 412–429, 2007.
15. Shai Halevi and Phillip Rogaway. A tweakable enciphering mode. In *Advances in Cryptology — CRYPTO 2003*, pages 482–499, 2003.
16. Shai Halevi and Phillip Rogaway. A parallelizable enciphering mode. In *CT-RSA*, pages 292–304, 2004.
17. Viet Tung Hoang, Ben Morris, and Phillip Rogaway. An enciphering scheme based on a card shuffle. In *Advances in Cryptology–CRYPTO 2012*, pages 1–13. Springer, 2012.
18. Rodolphe Lampe and Yannick Seurin. Tweakable blockciphers with asymptotically optimal security. In *FSE*, 2013.
19. Will Landecker, Thomas Shrimpton, and R. Terashima. Tweakable blockciphers with beyond birthday-bound security. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology — CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 14–30. Springer Berlin / Heidelberg, 2012.
20. Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In *Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '02, pages 31–46, London, UK, UK, 2002. Springer-Verlag.
21. J. Luo, K.D. Bowers, A. Oprea, and L. Xu. Efficient software implementations of large finite fields GF(2^n) for secure storage applications. *ACM Transactions on Storage (TOS)*, 8(1):2, 2012.
22. Cuauhtemoc Mancillas-Lopez, Debrup Chakraborty, and Francisco Rodriguez-Henriquez. Reconfigurable hardware impementations of tweakable enciphering schemes. *IEEE Transactions on Computers*, 59:1547–1561, 2010.
23. Nick Mathewson. Cryptographic challenges in and around Tor. Talk given at the Workshop on Real-World Cryptography, Jan 2013.
24. Kazuhiko Minematsu. Beyond-birthday-bound security based on tweakable block cipher. In *FSE*, pages 308–326, 2009.
25. Kazuhiko Minematsu and Tetsu Iwata. Building blockcipher from tweakable blockcipher: Extending FSE 2009 proposal. In *IMA International Conference*, pages 391–412, 2011.
26. Ben Morris, Phillip Rogaway, and Till Stegers. How to encipher messages on a small domain. In Shai Halevi, editor, *Advances in Cryptology–CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 286–302. Springer Berlin Heidelberg, 2009.
27. Kenneth Paterson and Arnold Yau. Padding oracle attacks on the ISO CBC mode encryption standard. *Topics in Cryptology–CT-RSA 2004*, pages 1995–1995, 2004.
28. P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. *Advances in Cryptology–ASIACRYPT 2004*, pages 55–73, 2004.
29. Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In *EUROCRYPT*, pages 373–390, 2006.

30. Palash Sarkar. Improving upon the TET mode of operation. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *Information Security and Cryptology - ICISC 2007*, volume 4817 of *Lecture Notes in Computer Science*, pages 180–192. Springer Berlin / Heidelberg, 2007.
31. Palash Sarkar. Efficient tweakable enciphering schemes from (block-wise) universal hash functions. *IEEE Trans. Inf. Theor.*, 55(10):4749–4760, October 2009.
32. Serge Vaudenay. Security flaws induced by CBC paddingapplications to SSL, IPSEC, WTLS.... In *Advances in Cryptology–EUROCRYPT 2002*, pages 534–545. Springer, 2002.
33. P. Wang, D. Feng, and W. Wu. HCTR: A variable-input-length enciphering mode. In *Information Security and Cryptology*, pages 175–188. Springer, 2005.
34. M.N. Wegman and J.L. Carter. New hash functions and their use in authentication and set equality. *Journal of computer and system sciences*, 22(3):265–279, 1981.

A Components for TCT1 and TCT2

LRW2 [20]: Birthday-bound TBC. Needs blockcipher E, ϵ-AXU$_2$ function H.

$$\mathrm{LRW2}[H,E]_{(K,L)}(T,X) = E_K(X \oplus H_L(T)) \oplus H_L(T)$$

CLRW2[19]: TBC with beyond-birthday-bound security. Requires blockcipher E and ϵ-AXU$_2$ function H.

$$\mathrm{CLRW2}[H,E]_{(K_1,K_2,L_1,L_2)}(T,X) = \mathrm{LRW2}[H,E]_{(K_2,L_2)}(T, \mathrm{LRW2}[H,E]_{(K_1,L_1)}(T,X))$$

polyH^{mn} [34]: ϵ-AXU$_2$ function with domain $(\{0,1\}^n)^m$ and $\epsilon = m/2^n$. All operations in $\mathbb{F}_{2^n}$.

$$\mathrm{polyH}_L^{mn}(T_1 T_2 \cdots T_m) = \bigoplus_{i=1}^{m} T_i \otimes L^i,$$

NH$(\nu w, 2tw)$ [6]: ϵ-AU hash function with $\epsilon = 1/2^{tw}$. Inputs are νw bits, where ν is even and $w > 0$ is fixed.

$$\mathrm{NH}[\nu,t]_{K_1 \,\|\, \cdots \,\|\, K_{\nu+2(t-1)}}(M) = H_{K_1 \cdots K_\nu}(M) \,\|\, H_{K_3 \cdots K_{\nu+2}}(M) \,\|\, \cdots \,\|\, H_{K_{2t-1} \cdots K_{\nu+2t-2}}(M)$$

where $H_{K_1 \,\|\, \cdots \,\|\, K_\nu}(X_1 \cdots X_\nu) = \sum_{i=1}^{\nu/2} (K_{2i-1} +_w X_{2i-1}) \cdot (K_{2i} +_w X_{2i}) \bmod 2^{2w}$.

CDMS [11]: Feistel-like domain extender for TBC $\widetilde{E}$.

$$\mathrm{CDMS}[\widetilde{E}]_K(T, L \,\|\, R) = \widetilde{E}_K(10 \,\|\, T \,\|\, R', L') \,\|\, R'$$

where $R' = \widetilde{E}_K(01 \,\|\, T \,\|\, L', R)$ and $L' = \widetilde{E}_K(00 \,\|\, T \,\|\, R, L)$.

Fig. 5: TCT_1 and TCT_2 use these constructions as components.