Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC

Phillip Rogaway *

September 24, 2004

Abstract

We describe highly efficient constructions, XE and XEX, that turn a blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ into a tweakable blockcipher $\widetilde{E}\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ having tweak space $\mathcal{T} = \{0,1\}^n \times \mathbb{I}$ where $\mathbb{I}$ is a set of tuples of integers such as $\mathbb{I} = [1 \mathinner{..} 2^{n/2}] \times [0 \mathinner{..} 10]$. When tweak T is obtained from tweak S by incrementing one if its numerical components, the cost to compute $\widetilde{E}_K^T(M)$ having already computed some $\widetilde{E}_K^S(M')$ is one blockcipher call plus a small and constant number of elementary machine operations. Our constructions work by associating to the i^{th} coordinate of $\mathbb{I}$ an element $\alpha_i \in \mathbb{F}_{2^n}^*$ and multiplying by α_i when one increments that component of the tweak. We illustrate the use of this approach by refining the authenticated-encryption scheme OCB and the message authentication code PMAC, yielding variants of these algorithms that are simpler and faster than the original schemes, and yet have simpler proofs. Our results bolster the thesis of Liskov, Rivest, and Wagner [11] that a desirable approach for designing modes of operation is to start from a tweakable blockcipher. We elaborate on their idea, suggesting the kind of tweak space, usage-discipline, and blockcipher-based instantiations that give rise to simple and efficient modes.

Key words: modes of operation, provable security, tweakable blockciphers.

*Dept. of Computer Science, Kemper Hall of Engineering, One Shields Ave., University of California, Davis, California, 95616, USA; and Dept. of Computer Science, Faculty of Science, Chiang Mai University, Chiang Mai 50200 Thailand. E-mail: `rogaway@cs.ucdavis.edu` WWW: `www.cs.ucdavis.edu/~rogaway/`

Contents

1 Introduction 1

2 Preliminaries 2

3 The XE and XEX Constructions 3

4 Parameter Sets Yielding Unique Representations 5

5 Security of XE 7

6 Security of XEX 7

7 An Almost-Free Alternative to Key Separation 7

8 Combining XE and XEX 8

9 Security of the Combined Construction 9

10 The OCB1 Authenticated-Encryption Scheme 9

11 The PMAC1 Message Authentication Code 12

12 Comments 14

Acknowledgments 14

References 14

A Tweakable Blockciphers Implicit in Prior Work 16

B Proof of Theorem 7 — Security of XE 16

C Proof of Theorem 11 — Security of XEX* 18

D Proof of Theorem 12 — Security of OCB1 26

E Proof of Theorem 15 — Security of PMAC1 27

1 Introduction

Liskov, Rivest and Wagner [11] defined the notion of a tweakable blockcipher and put forward the thesis that these objects make a good starting point for doing blockcipher-based cryptographic design. In this paper we describe a good way to build a tweakable blockcipher $\widetilde{E}$ out of an ordinary blockcipher E. Used as intended, our constructions, XE and XEX, add just a few machine instructions to the cost of computing E. We illustrate the use of these constructions by improving on the authenticated-encryption scheme OCB [16] and the message authentication code PMAC [4].

Tweakable blockciphers. Schroeppel [17] designed a blockcipher, Hasty Pudding, wherein the user supplies a non-secret *spice* and changing this spice produces a completely different permutation. Liskov, Rivest, and Wagner [11] formally defined the syntax and security measures for such a *tweakable* blockcipher, and they suggested that this abstraction makes a desirable starting point to design modes of operation and prove them secure. They suggested ways to build a tweakable blockcipher $\widetilde{E}$ out of a standard blockcipher E, as well as ways to modify existing blockcipher designs to incorporate a tweak. They illustrated the use of these objects. Formally, a tweakable blockcipher is a map $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ where each $\widetilde{E}^T_K(\cdot) = \widetilde{E}(K,T,\cdot)$ is a permutation and $\mathcal{T}$ is the set of *tweaks*.

Our contributions. We propose efficient ways to turn a blockcipher $E\colon \mathcal{K}\times\{0,1\}^n \to \{0,1\}^n$ into a tweakable blockcipher $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$. (See Appendix A for the best constructions formerly known.) Our *powering-up* constructions, XE and XEX, preserve the key space and blocksize of E but endow $\widetilde{E}$ with a tweak space $\mathcal{T} = \{0,1\}^n \times \mathbb{I}$ where $\mathbb{I}$ is a set of tuples of integers, like $\mathbb{I} = [1\,..\,2^{n/2}] \times [0\,..\,10]$. The XE construction turns a CPA-secure blockcipher into a CPA-secure tweakable blockcipher, while XEX turns a CCA-secure blockcipher into a CCA-secure tweakable blockcipher. (CPA stands for chosen-plaintext attack and CCA for chosen-ciphertext attack.) The methods are highly efficient when tweaks arise in sequence, with most tweaks $(N,\mathbf{i})$ being identical to the prior tweak $(N,\mathbf{i}')$ except for incrementing a component of $\mathbf{i}$.

As an illustrative and useful example, consider turning a conventional blockcipher $E\colon \mathcal{K}\times\{0,1\}^n \to \{0,1\}^n$ into a tweakable blockcipher $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ by defining $\widetilde{E}^{N\,i\,j}_K(M) = E_K(M\oplus\Delta)\oplus\Delta$ where offset $\Delta = 2^i 3^j\,\mathsf{N}$ and $\mathsf{N} = E_K(N)$. Arithmetic is done in the finite field $\mathbb{F}_{2^n}$. For concreteness, assume $n = 128$ and a tweak space of $\mathcal{T} = \{0,1\}^n \times [1\,..\,2^{64}] \times [0\,..\,10]$. We show that $\widetilde{E}$ is secure (as a strong, tweakable PRP) as long as E is secure (as a strong, untweakable PRP). Computing $\widetilde{E}^{N\,i\,j}_K(X)$ will usually cost about 1 shift, 1 conditional, and 3–4 xors more than computing $E_K(X)$.

We illustrate how the use of tweakable blockciphers during mode design, followed by the instantiation of the tweakable blockcipher with an ordinary blockcipher using one of our constructions, can give rise to modes that are simpler, faster, and easier to prove correct than what designing directly from a blockcipher has delivered. We do this by refining two already-optimized modes, OCB [16] and PMAC [4], yielding new modes, OCB1 and PMAC1, that are are easier to understand, easier to implement, and faster. Computing offsets in the new modes does not involve Gray-code sequence or counting the number of trailing zero bits in successive integers. OCB1 eliminates the utility of preprocessing, saving a blockcipher call.

Intuition. The idea behind the powering-up constructions can be explained like this. Apart from Gray-code reordering, PMAC authenticates an m-block message using a sequence of offsets $L, 2L, 3L, \ldots, (m-1)L$, where multiplication is in the finite field $\mathbb{F}_{2^n}$ and $L = E_K(0^n)$ is a variant of the underlying key K. When a special kind of offset is needed, a value $huge \cdot L$ is added (xored) into the current offset, where $huge$ is so large that it could never be among $\{1, 2, \ldots, m-1\}$. What we now do instead is to use the easier-to-compute sequence of offsets $2^1L, 2^2L, \ldots, 2^{m-1}L$. We insist that our field be represented using a primitive polynomial instead of merely an irreducible one, which ensures that $2^1, 2^2, 2^3, \ldots, 2^{2^n-1}$ will all be distinct. When a special offset is needed we can no longer add to the current offset some huge constant times L and expect this never to land on a point in $2^1L, 2^2L, \ldots, 2^{m-1}L$. Instead, we multiply the current offset by 3 instead of 2. If the index of 3 (in $\mathbb{F}^*_{2^n}$) is enormous relative to the base 2 then multiplying by 3 is equivalent to multiplying by 2^{huge} and $2^i 3\,L$ won't be among of $2^1L, 2^2L, \ldots, 2^{m-1}L$ for any reasonable value of m. The current paper will make all of the ideas of this paragraph precise.

Further related work. Halevi and Rogaway [8] used the sequence of offsets $2L, 2^2L, 2^3L, \ldots,$ in their EME mode. They give no general results about this construction, and EME did not use tweakable blockciphers, yet this offset ordering was our starting point.

2 Preliminaries

The field with 2^n points. Let $\mathbb{F}_{2^n}$ denote the field with 2^n points and let $\mathbb{F}^*_{2^n}$ denote the multiplicative subgroup of this field (which contains $2^n - 1$ points). We interchangeably think of a point a in $\mathbb{F}_{2^n}$ in any of the following ways: (i) as an abstract point in the field; (ii) as an n-bit string $a_{n-1} \ldots a_1 a_0 \in \{0,1\}^n$; (iii) as a formal polynomial $a(\mathtt{x}) = a_{n-1}\mathtt{x}^{n-1} + \cdots + a_1\mathtt{x} + a_0$ with binary coefficients; (iv) as an integer between 0 and $2^n - 1$, where the string $a \in \{0,1\}^n$ corresponds to the number $\sum_{i=0}^{n-1} a_i 2^i$. For example, one can regard the string $a = 0^{125}101$ as a 128-bit string, as the number 5, as the polynomial $\mathtt{x}^2 + 1$, or as an abstract point in $\mathbb{F}_{2^{128}}$.

To add two points in $\mathbb{F}_{2^n}$, take their bitwise xor. We denote this operation by $a \oplus b$. To multiply two points, fix a primitive polynomial p_n having binary coefficients and degree n; for concreteness, select the lexicographically first primitive polynomial among the primitive degree n polynomials having a minimum number of nonzero coefficients. For $n = 128$, the indicated polynomial is $p_{128}(\mathtt{x}) = \mathtt{x}^{128} + \mathtt{x}^7 + \mathtt{x}^2 + \mathtt{x} + 1$. For $n = 64$ we have $p_{64} = \mathtt{x}^{64} + \mathtt{x}^4 + \mathtt{x}^3 + \mathtt{x} + 1$. Saying that $p_n(\mathtt{x})$ is primitive means that it is irreducible over $\mathbb{F}_2$ and $\mathtt{x} = 2$ generates all of $\mathbb{F}^*_{2^n}$ (so the point 2 has order $2^n - 1$). To multiply $a, b \in \mathbb{F}_{2^n}$, which we denote ab, regard a and b as polynomials $a(\mathtt{x}) = a_{n-1}\mathtt{x}^{n-1} + \cdots + a_1\mathtt{x} + a_0$ and $b(\mathtt{x}) = b_{n-1}\mathtt{x}^{n-1} + \cdots + b_1\mathtt{x} + b_0$, form their product $c(\mathtt{x})$ over $\mathbb{F}_2$, and take the remainder one gets when dividing $c(\mathtt{x})$ by $p_n(\mathtt{x})$.

It is computationally simple to multiply $a \in \{0,1\}^n$ by 2 (to "double" a). We illustrate the method for $n = 128$, in which case multiplying $a = a_{n-1} \cdots a_1 a_0$ by $\mathtt{x} = 2$ yields $a_{n-1}\mathtt{x}^n + a_{n-2}\mathtt{x}^{n-1} + a_1\mathtt{x}^2 + a_0\mathtt{x}$. Thus, if the first bit of a is 0, then $2a = a \ll 1$, the left-shift of a by one bit. If the first bit of a is 1 then we must add $\mathtt{x}^{128}$ to $a \ll 1$. Since $p_{128} = \mathtt{x}^{128} + \mathtt{x}^7 + \mathtt{x}^2 + \mathtt{x} + 1 = 0$ we know that $\mathtt{x}^{128} = \mathtt{x}^7 + \mathtt{x}^2 + \mathtt{x} + 1$, so adding $\mathtt{x}^{128}$ means to xor by $0^{120}10^41^3$. In summary, when $n = 128$ we have that $2a = a \ll 1$ if $\mathsf{firstbit}(a) = 0$ and $2a = (a \ll 1) \oplus 0^{120}10^41^3$ if $\mathsf{firstbit}(a) = 1$.

One can easily multiply by other small constants as well: $3a = 2a \oplus a$ and $5a = 2(2a) \oplus a$ and $7a = 2(2a) \oplus 2a \oplus a$ and so forth.

Blockciphers and tweakable blockciphers. We review the standard definitions for blockciphers and their security [2] and the extension of these notions to tweakable blockciphers [11]. A *blockcipher* is a function $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ where $n \geq 1$ is a number and $\mathcal{K}$ is a finite nonempty set and $E(K, \cdot) = E_K(\cdot)$ is a permutation for all $K \in \mathcal{K}$. A *tweakable blockcipher* is a function $\widetilde{E}\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ where n and $\mathcal{K}$ are as above and $\mathcal{T}$ is a nonempty set and $\widetilde{E}(K, T, \cdot) = \widetilde{E}^T_K(\cdot)$ is a permutation for all $K \in \mathcal{K}$ and $T \in \mathcal{T}$. For blockciphers and tweakable blockciphers we call n the *blocksize* and $\mathcal{K}$ the *key space*. For tweakable blockciphers we call $\mathcal{T}$ the *tweak space*.

Let $\mathrm{Perm}(n)$ be the set of all permutations on n bits. Let $\mathrm{Perm}(\mathcal{T}, n)$ be the set of all mappings from $\mathcal{T}$ to permutations on n bits. In writing $\pi \xleftarrow{\$} \mathrm{Perm}(n)$ we are choosing a random permutation $\pi(\cdot)$ on $\{0,1\}^n$. In writing $\pi \xleftarrow{\$} \mathrm{Perm}(\mathcal{T}, n)$ we are choosing a random permutation $\pi(T, \cdot) = \pi_T(\cdot)$ on $\{0,1\}^n$ for each $T \in \mathcal{T}$. If E: $\mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a blockcipher then its inverse is the blockcipher $D = E^{-1}$ where D: $\mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is defined by $D(K, Y) = D_K(Y)$ being the unique point X such that $E_K(X) = Y$. If $\widetilde{E}$: $\mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is a tweakable blockcipher then its inverse is the tweakable blockcipher $\widetilde{D} = \widetilde{E}^{-1}$ where $\widetilde{D}$: $\mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is defined by $\widetilde{D}(K, T, Y) = \widetilde{D}_K^T(Y)$ being the unique point X such that $\widetilde{E}_K^T(X) = Y$.

An adversary is a probabilistic algorithm with access to zero or more oracles. Without loss of generality, adversaries never ask a query for which the answer is trivially known: an adversary does not repeat a query, does not ask $D_K(Y)$ after receiving Y in response to a query $E_K(X)$, and so forth. Oracles will have an implicit domain of valid queries and, for convenience, we assume that all adversarial queries lie within that domain. This is not a significant restriction because membership can be easily tested for all domains of interest to us.

Definition 1 [Blockcipher and tweakable-blockcipher security] *Let E: $\mathcal{K}\times\{0,1\}^n \to \{0,1\}^n$ be a blockcipher and let $\widetilde{E}$: $\mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher. Let A be an adversary. Then:*

$$\begin{aligned}
\mathbf{Adv}_{E}^{\mathrm{prp}}(A) &= \Pr[K \xleftarrow{\$} \mathcal{K} :\ A^{E_K(\cdot)} \Rightarrow 1] - \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(n) :\ A^{\pi(\cdot)} \Rightarrow 1] \\
\mathbf{Adv}_{E}^{\pm\mathrm{prp}}(A) &= \Pr[K \xleftarrow{\$} \mathcal{K} :\ A^{E_K(\cdot)\, D_K(\cdot)} \Rightarrow 1] - \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(n) :\ A^{\pi(\cdot)\, \pi^{-1}(\cdot)} \Rightarrow 1] \\
\mathbf{Adv}_{\widetilde{E}}^{\widetilde{\mathrm{prp}}}(A) &= \Pr[K \xleftarrow{\$} \mathcal{K} :\ A^{\widetilde{E}_K(\cdot,\cdot)} \Rightarrow 1] - \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(\mathcal{T}, n) :\ A^{\pi(\cdot,\cdot)} \Rightarrow 1] \\
\mathbf{Adv}_{\widetilde{E}}^{\pm\widetilde{\mathrm{prp}}}(A) &= \Pr[K \xleftarrow{\$} \mathcal{K} :\ A^{\widetilde{E}_K(\cdot,\cdot)\, \widetilde{D}_K(\cdot,\cdot)} \Rightarrow 1] - \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(\mathcal{T}, n) :\ A^{\pi(\cdot,\cdot)\, \pi^{-1}(\cdot,\cdot)} \Rightarrow 1] \quad \square
\end{aligned}$$

Of course D and $\widetilde{D}$ denote the inverses of blockciphers E and $\widetilde{E}$. By Pr[experiment : event] we mean the probability of the specified event after performing the specified experiment. In writing $A \Rightarrow 1$ we are referring to the event that the adversary A outputs the bit 1.

In the usual way we lift advantage measures that depend on an adversary to advantage measures that depend on named resources: $\mathbf{Adv}_{\Pi}^{\mathrm{xxx}}(\mathcal{R}) = \max_A\{\mathbf{Adv}_{\Pi}^{\mathrm{xxx}}(A)\}$ over all adversaries A that use resources at most $\mathcal{R}$. The resources of interest to us are the total number of oracle queries q and the total length of those queries σ and the running time t. For convenience, the total length of queries will be measured in n-bit blocks, for some understood value of n, so a query X contributes $|X|_n$ to the total, where $|X|_n$ means $\max\{|X|/n, 1\}$. Running time, by convention, includes the description size of the algorithm relative to some standard encoding. When we speak of authenticity, the block length of the adversary's output is included in σ.

3 The XE and XEX Constructions

Goals. We want to support tweak sets that look like $\mathcal{T} = \{0,1\}^n \times \mathbb{I}$ where $\mathbb{I}$ is a set of tuples of integers. In particular, we want to be able to make $\mathbb{I}$ the cross product of a large subrange of integers, like $[1 \mathinner{..} 2^{n/2}]$, by the cross product of small ranges of integers, like $[0 \mathinner{..} 10]\times[0 \mathinner{..} 10]$. Thus an example tweak space is $\mathcal{T} = \{0,1\}^n \times [1 \mathinner{..} 2^{n/2}] \times [0 \mathinner{..} 10] \times [0 \mathinner{..} 10]$. Tweaks arise in some sequence $T_1, T_2, \ldots$ and we will obtain impressive efficiency only to the extent that most tweaks are an increment of the immediately prior one. When we say that tweak $T = (N, i_1, \ldots, i_k)$ is an increment of another tweak we mean that one of $i_1, \ldots, i_k$ got incremented and everything else stayed the same. The second component of tweak $(N, i_1, \ldots, i_k)$, meaning i_1, is the component that we expect to get incremented most often. We want there to be a simple, constant-time procedure to increment a tweak at any given component of $\mathbb{I}$. To increment a tweak it shouldn't be necessary to go to memory, consult a table, or examine which number tweak this in in sequence. Incrementing tweaks should be endian-independent and avoid extended-precision arithmetic. Efficiently incrementing tweaks shouldn't require precomputation. Tweaks that are not the increment of a prior tweak will also arise, and they will typically look like $(N, 1, 0 \ldots, 0)$. Constructions should be reasonably efficient in dealing with such tweaks.

We emphasize that the efficiency measure we are focusing on is not the cost of computing $\widetilde{E}_K^T(X)$ from scratch—by that measure our constructions will not be particularly good. Instead, we are interested in the cost of computing $\widetilde{E}_K^T(X)$ given that one has just computed $\widetilde{E}_K^S(X')$ and T is obtained by incrementing S at some component. Most often that component will have been the second component of S. It is a thesis underlying our work, supported by the design of OCB1 and PMAC1, that one will often be able to arrange that most tweaks are an increment to the prior one.

Tweaking with $\Delta = 2^i\, \mathsf{N}$. Recall that we have chosen to represent points in $\mathbb{F}_{2^n}$ using a primitive polynomial, not just an irreducible one. This means that the point 2 is a generator of $\mathbb{F}_{2^n}$: the points $1, 2, 2^2, 2^3, \ldots, 2^{2^n-2}$ are all distinct. This property turns out to be the crucial one that lets us construct from a blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ a tweakable blockcipher $\widetilde{E}\colon \mathcal{K} \times (\{0,1\}^n \times [1 \mathinner{..} 2^n - 2]) \times \{0,1\}^n \to \{0,1\}^n$ by way of

$$\widetilde{E}_K^{N\,i}(M) \;=\; E_K(M \oplus \Delta) \oplus \Delta \quad \text{where} \quad \Delta = 2^i\, \mathsf{N} \quad \text{and} \quad \mathsf{N} = E_K(N).$$

The tweak set is $\mathcal{T} = \{0,1\}^n \times \mathbb{I}$ where $\mathbb{I} = [1 \mathinner{..} 2^n - 2]$ and the tweakable blockcipher just described is denoted $\widetilde{E} = \mathrm{XEX}[E, 2^{\mathbb{I}}]$. When computing the sequence of values $\widetilde{E}_K^{N\,1}(M_1), \ldots, \widetilde{E}_K^{N\,m-1}(M_{m-1})$ each $\widetilde{E}_K^{N\,i}(M_i)$ computation but the first uses one blockcipher call and one doubling operation. Doubling takes a shift followed by a conditional xor. We call the construction above, and all the subsequent constructions of this section, *powering-up* constructions.

Tweaking by $\Delta = 2^i 3^j\, \mathsf{N}$. To facilitate mode design we may want tweaks that look like (N, i, j) where $N \in \{0,1\}^n$ and i is an integer from a large set $\mathbb{I}$, like $\mathbb{I} = [1 \mathinner{..} 2^{n/2}]$, and j is an integer from some small set $\mathbb{J}$, like $\mathbb{J} = \{0, 1\}$. To get the "diversity" associated to the various j-values we just multiply by 3 instead of 2. That is, we construct from a blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ a tweakable blockcipher $\widetilde{E}\colon \mathcal{K} \times (\{0,1\}^n \times \mathbb{I} \times \mathbb{J}) \times \{0,1\}^n \to \{0,1\}^n$ by way of

$$\widetilde{E}_K^{N\,i\,j}(M) \;=\; E_K(M \oplus \Delta) \oplus \Delta \quad \text{where} \quad \Delta = 2^i 3^j\, \mathsf{N} \quad \text{and} \quad \mathsf{N} = E_K(N).$$

The tweakable blockcipher just described is denoted $\widetilde{E} = \mathrm{XEX}[E, 2^{\mathbb{I}} 3^{\mathbb{J}}]$. Incrementing the tweak at component i is done by doubling, while incrementing the tweak at component j is done by tripling.

The XEX construction. Generalizing the two examples above, we have the following definition. See Figure 1.

Definition 2 [XEX construction] *Let $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher, let $\alpha_1, \ldots, \alpha_k \in \mathbb{F}_{2^n}^*$, and let $\mathbb{I}_1, \ldots, \mathbb{I}_k \subseteq \mathbb{Z}$. Then $\widetilde{E} = \mathsf{XEX}[E, \alpha_1^{\mathbb{I}_1} \cdots \alpha_k^{\mathbb{I}_k}]$ is the tweakable blockcipher $\widetilde{E}\colon \mathcal{K} \times (\{0,1\}^n \times \mathbb{I}_1 \times \cdots \times \mathbb{I}_k) \times \{0,1\}^n \to \{0,1\}^n$ defined by $\widetilde{E}_K^{N i_1 \ldots i_k}(M) = E_K(M \oplus \Delta) \oplus \Delta$ where $\Delta = \alpha_1^{i_1} \alpha_2^{i_2} \cdots \alpha_k^{i_k}\, \mathsf{N}$ and $\mathsf{N} = E_K(N)$.* $\square$

The XE construction. As made clear in the work of Liskov, Rivest, and Wagner [11], constructions of the form $\widetilde{E}_K^T(M) = E_K(M \oplus \Delta) \oplus \Delta$ aim for chosen-ciphertext attack (CCA) security, while for chosen-plaintext attack (CPA) security one can omit the outer xor. Thus we consider

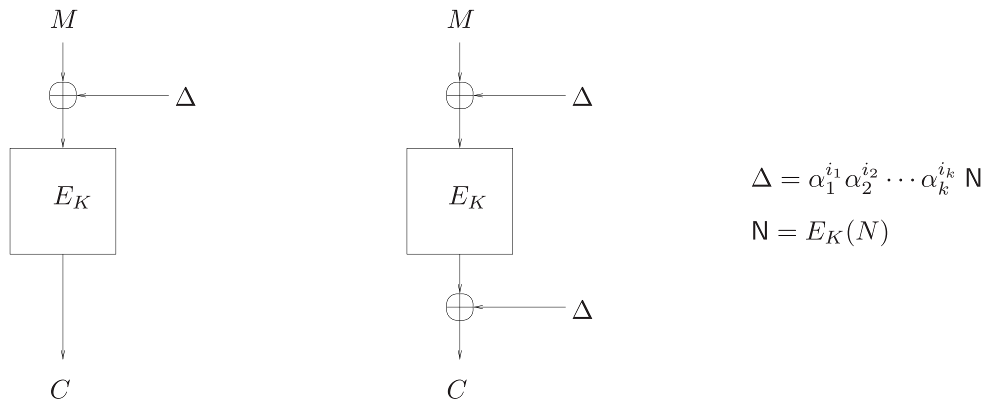


Figure 1: The XE and XEX constructions. Starting with a blockcipher $E\colon \mathcal{K}\times\{0,1\}^n \to \{0,1\}^n$ and points $\alpha_1,\ldots,\alpha_k \in \mathbb{F}_{2^n}^*$ and sets $\mathbb{I}_1,\ldots,\mathbb{I}_k \subseteq \mathbb{Z}$ we construct the blockcipher $\widetilde{E} = \mathrm{XE}[E, \alpha_1^{\mathbb{I}_1}\cdots\alpha_k^{\mathbb{I}_k}]$ (shown on the left) and $\widetilde{\boldsymbol{E}} = \mathrm{XEX}[E, \alpha_1^{\mathbb{I}_1}\cdots\alpha_k^{\mathbb{I}_k}]$ (shown on the right). Multiplication (to form Δ) is in $\mathbb{F}_{2^n}$.

the construction $E_K(M\oplus\Delta)$. This is slightly more efficient than XEX, saving one xor. Again see Figure 1.

Definition 3 [XE construction] *Let $E\colon \mathcal{K}\times\{0,1\}^n \to \{0,1\}^n$ be a blockcipher, $\alpha_1,\ldots,\alpha_k \in \mathbb{F}_{2^n}^*$, and $\mathbb{I}_1,\ldots,\mathbb{I}_k \subseteq \mathbb{Z}$. Then $\widetilde{E} = \mathsf{XE}[E, \alpha_1^{\mathbb{I}_1}\cdots\alpha_k^{\mathbb{I}_k}]$ is the tweakable blockcipher $\widetilde{E}\colon \mathcal{K}\times(\{0,1\}^n\times\mathbb{I}_1\times\cdots\times\mathbb{I}_k)\times\{0,1\}^n \to \{0,1\}^n$ defined by $\widetilde{E}_K^{N i_1\ldots i_k}(M) = E_K(M\oplus\Delta)$ where $\Delta = \alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k}\,\mathsf{N}$ and $\mathsf{N} = E_K(N)$.* □

4 Parameter Sets Yielding Unique Representations

It is easy to see that the XE and XEX constructions can only "work" if $\alpha_1^{i_1}\cdots\alpha_k^{i_k}$ are distinct throughout $(i_1,\ldots,i_k)\in\mathbb{I}_1\times\cdots\times\mathbb{I}_k$. This motivates the following definition.

Definition 4 [Unique representations] *Fix a group G. A* ***choice of parameters*** *is a list $\alpha_1,\ldots,\alpha_k \in G$ of* ***bases*** *and a set $\mathbb{I}_1\times\cdots\times\mathbb{I}_k \subseteq \mathbb{Z}^k$ of* ***allowed indices****. We say that the choice of parameters provides* ***unique representations*** *if for every $(i_1,\ldots,i_k)$, $(j_1,\ldots,j_k)\in\mathbb{I}_1\times\cdots\times\mathbb{I}_k$ we have that $\alpha_1^{i_1}\cdots\alpha_k^{i_k} = \alpha_1^{j_1}\cdots\alpha_k^{j_k}$ implies $(i_1,\ldots,i_k) = (j_1,\ldots,j_k)$.* □

In other words, representable points are uniquely representable: any group element $\alpha_1^{i_1}\cdots\alpha_k^{i_k}$ that can be represented using allowed indices can be represented in only one way (using allowed indices).

For tweak spaces of practical interest, discrete-log calculations within $\mathbb{F}_{2^n}^*$ can be used to help choose and verify that a given choice of parameters provides unique representations. The following result gives examples for $\mathbb{F}_{2^{128}}^*$.

Proposition 5 [Can use 2, 3, 7 when $n = 128$] *In the group $\mathbb{F}_{2^{128}}^*$ the following choices for parameters provide unique representations:*

(1) $\alpha_1 = 2$ *and* $\mathbb{I}_1 = [-2^{126} .. 2^{126}]$.

(2) $\alpha_1,\alpha_2 = 2,3$ *and* $\mathbb{I}_1\times\mathbb{I}_2 = [-2^{115} .. 2^{115}]\times[-2^{10} .. 2^{10}]$.

(3) $\alpha_1,\alpha_2,\alpha_3 = 2,3,7$ *and* $\mathbb{I}_1\times\mathbb{I}_t\times\mathbb{I}_3 = [-2^{108} .. 2^{108}]\times[-2^7 .. 2^7]\times[-2^7 .. 2^7]$. □

We comment that the above result does depend on the choice of representations for the group; recall that we fixed a representation of $\mathbb{F}^*_{2^n}$ using the lexicographically first primitive polynomial.

Proof: For statement (1) recall that 2 is a generator of the group (by our choice of irreducible polynomial) and the order of the group $\mathbb{F}^*_{2^{128}}$ is $2^{128}-1$ and so $2^i = 2^j$ iff $i = j \pmod{2^{128}-1}$ and so any contiguous range of $2^{128}-1$ or fewer integers will provide unique representations with respect to base 2.

To prove statement (2) we need to compute $\log_2 3$ in the group $\mathbb{F}^*_{2^{128}}$:

$$\log_2 3 \quad = \quad 338793687469689340204974836150077311399 \quad \text{(decimal)}$$

This and subsequent discrete logs were computed using a Maple-implementation combining the Pohlig-Hellman [12] and Pollard-rho [13] algorithms. (A naive implementation computes discrete logs in $\mathbb{F}^*_{2^{128}}$ in a few hours.) Now note that $2^a3^b = 2^{a'}3^{b'}$ iff $2^a 2^{b\log_2 3} = 2^{a'}2^{b'\log_2 3}$ iff $2^{a+b\log_2 3} = 2^{a'+b'\log_2 3}$ iff $a + b\log_2 3 = a' + b'\log_2 3 \pmod{2^{128}-1}$ because 2 is a generator of the group $\mathbb{F}^*_{2^{128}}$. Thus $2^a3^b = 2^{a'}3^{b'}$ iff $a - a' = (b'-b)\log_2 3 \pmod{2^{128}-1}$. If $b, b' \in [-2^{10} \mathinner{..} 2^{10}]$ then $\Delta_b = b'-b \in [-2^{11} \mathinner{..} 2^{11}]$ and computer-assisted calculation then shows that the smallest value of $\Delta_b \log_2 3 \pmod{2^{128}-1}$ for $\Delta_b \in [-2^{11}..2^{11}]$ and $\Delta_b \neq 0$ is $1600\log_2 3 = \texttt{00113a0ce508326c006763c0b80c59f9}$ (in hexadecimal) which is about $2^{116.1}$. (By "smallest" we refer to the distance from 0, modulo $2^{128}-1$, so 2^{100} and $(2^{128}-1)-2^{100}$ are equally small, for example.) Thus if a, a' are restricted to $[-2^{115} \mathinner{..} 2^{115}]$ and b, b' are restricted to $[-2^{10} \mathinner{..} 2^{10}]$ then $\Delta_a = a - a' \leq 2^{116}$ can never equal $\Delta_b \log_2 3 \pmod{2^{128}-1} > 2^{116}$ unless $\Delta_b = 0$. This means that the only solution to $2^a3^b = 2^{a'}3^{b'}$ within the specified range is $a = a'$ and $b = b'$.

To prove statement (3) is similar. First we need the value

$$\log_2 7 \quad = \quad 305046802472688182329780655685899195396 \quad \text{(decimal)}$$

Now $2^a3^b7^c = 2^{a'}3^{b'}7^{c'}$ iff $a - a' = (b'-b)\log_2 3 + (c'-c)\log_2 7 \pmod{2^{128}-1}$. The smallest value for $\Delta_b \log_2 3 + \Delta_c \log_2 7 \pmod{2^{128}-1}$ when $\Delta_b, \Delta_c \in [-2^8 \mathinner{..} 2^8]$ and at least one of these is non-zero is $-48\log_2 3 + 31\log_2 7 \pmod{2^{128}-1} = \texttt{00003bfabac91e02b278b7e69a379d18}$ (hexadecimal) which is about $2^{109.9}$. So restricting the index for base-2 to $[-2^{108} \mathinner{..} 2^{108}]$ ensures that $a - a' \leq 2^{109}$ while $(b'-b)\log_2 3 + (c'-c)\log_2 7 > 2^{109}$ unless $b = b'$ and $c = c'$ and $a = a'$. ∎

We emphasize that not just any list of bases will work. Notice, for example, that $3^2 = 5$ in $\mathbb{F}^*_{2^n}$ (because $3^2 = (\mathtt{x}+1)^2 = \mathtt{x}^2+1 = 5$) so the list of bases $2, 3, 5$ does *not* give unique representations, even for a tiny list of allowed indices like $\mathbb{I}_1 \times \mathbb{I}_2 \times \mathbb{I}_3 = \{0,1,2\}^3$.

Similar calculations can be done in other groups; here we state the analogous result for $\mathbb{F}^*_{2^{64}}$.

Proposition 6 [Can use 2, 3, 11 when $n = 64$] *In the group $\mathbb{F}^*_{2^{64}}$ the following choices for parameters provide unique representations:*

(1) $\alpha_1 = 2$ *and* $[-2^{62} \mathinner{..} 2^{62}]$.

(2) $\alpha_1, \alpha_2 = 2, 3$ *and* $[-2^{51} \mathinner{..} 2^{51}] \times [-2^{10} \mathinner{..} 2^{10}]$.

(3) $\alpha_1, \alpha_2, \alpha_3 = 2, 3, 11$ *and* $[-2^{44} \mathinner{..} 2^{44}] \times [-2^7 \mathinner{..} 2^7] \times [-2^7 \mathinner{..} 2^7]$. □

This time 2, 3, 7 does *not* work as a list of bases, even with a small set of allowed indices like $[1 \mathinner{..} 64] \times \{0,1,2\} \times \{0,1,2\}$, due to the fact that $2^{64} = 3^2 \cdot 7$ in this group. Machine-assisted verification seems essential here; a relation like that just given is found immediately when computing the possible values for $\Delta_b \log_2 3 + \Delta_c \log_2 7 \pmod{2^{64}-1}$ but it might not otherwise be anticipated.

We comment that one has the freedom to switch the irreducible polynomial used to represent the field if one is dissatisfied with the associated parameter choices that provide unique representations.

5 Security of XE

The following result quantifies the security of the XE construction.

Theorem 7 [Security of XE] *Fix $n \geq 1$ and let $\alpha_1, \ldots, \alpha_k \in \mathbb{F}_{2^n}^*$ be base elements and let $\mathbb{I}_1 \times \cdots \times \mathbb{I}_k$ be allowed indices such that these parameters provide unique representations. Fix a blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and let $\widetilde{E} = \mathsf{XE}[E, \alpha_1^{\mathbb{I}_1} \cdots \alpha_k^{\mathbb{I}_k}]$. Then*

$$\mathbf{Adv}^{\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t, q) \quad \leq \quad \mathbf{Adv}^{\mathrm{prp}}_{E}(t', 2q) + \frac{4.5\, q^2}{2^n}$$

where $t' = t + ckn(q+1)$ for some absolute constant c. □

In English, the XE construction promotes a CPA-secure blockcipher to a CPA-secure tweakable blockcipher, assuming that the chosen base elements and range of allowed indices provide unique representations. The proof of Theorem 7 is given in Appendix B.

6 Security of XEX

Some added care is needed to address the security of XEX. Suppose, to be concrete, that we are looking at $\mathsf{XEX}[E, 2^{\mathbb{I}}]$ and $\mathbb{I} = [0 \,..\, 2^{n-2}]$. Let the adversary ask a deciphering query with ciphertext $C = 0^n$ and tweak $(0^n, 0)$. If the adversary has a construction-based deciphering oracle then it will get a response of $M = \widetilde{D}_K^{0^n\, 0}(0^n) = D_K(\Delta) \oplus \Delta = D_K(\mathsf{N}) \oplus \mathsf{N} = 0^n \oplus \mathsf{N} = \mathsf{N}$, where $\mathsf{N} = E_K(0^n) = \Delta$. This allows the adversary to defeat the CCA-security. For example, enciphering $2M = 2\mathsf{N}$ with a tweak of $(0^n, 1)$ and enciphering $4M = 4\mathsf{N}$ with a tweak of $(0^n, 2)$ will give identical results (if the adversary has the construction-based enciphering oracle). Corresponding to this attack we exclude any tweak $(N, i_1, \ldots, i_k)$ for which $(i_1, \ldots, i_k)$ is a representative of 1—that is, any tweak $(N, i_1, \ldots, i_k)$ for which $\alpha_1^{i_1} \ldots \alpha_k^{i_k} = 1$. In particular, this condition excludes any tweak $(N, 0, \ldots, 0)$. The proof of the following is omitted, as Theorem 11 will be more general.

Theorem 8 [Security of XEX] *Fix $n \geq 1$ and let $\alpha_1, \ldots, \alpha_k \in \mathbb{F}_{2^n}^*$ be base elements and let $\mathbb{I}_1 \times \cdots \times \mathbb{I}_k$ be allowed indices such that these parameters provide unique representations. Assume $\alpha_1^{i_1} \cdots \alpha_k^{i_k} \neq 1$ for all $(i_1, \ldots, i_k) \in \mathbb{I}_1 \times \cdots \times \mathbb{I}_k$. Fix a blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and let $\widetilde{E} = \mathsf{XEX}[E, \alpha_1^{\mathbb{I}_1} \cdots \alpha_k^{\mathbb{I}_k}]$. Then*

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t, q) \quad \leq \quad \mathbf{Adv}^{\pm\mathrm{prp}}_{E}(t', 2q) + \frac{9.5\, q^2}{2^n}$$

where $t' = t + ckn(q+1)$ for some absolute constant c. □

7 An Almost-Free Alternative to Key Separation

When combining two blockcipher-based cryptographic mechanisms into a composite mechanism, it is, in general, essential to use two different keys. Either these two keys together comprise the key for the joint mechanism, or else each key is obtained from an underlying one by a key-derivation technique. The first possibility increases the key length in the composite mechanism while the second involves extra computation at key setup. Both possibilities incur the inefficiency of blockcipher re-keying when the combined mode runs. For all of these reasons, some new "composite" modes of operation have gone to considerable trouble in order to make do (for their particular context) with a *single* blockcipher key. Examples include EAX, CCM, and O$\dot{\text{C}}$B [3, 14, 18]. Using a single key complicates proofs—when the mechanism works at all—because one can no longer reason about generically combining lower-level mechanisms.

Tweakable blockciphers open up a different possibility: the same underlying key is used across the different mechanisms that are being combined, but one arranges that the tweaks are disjoint across different mechanisms. In this way one retains the modularity of design and analysis associated to using separate keys—one reasons in terms of generic composition—yet one can instantiate in a way that avoids having extra key material or doing extra key setups. Because the tweak space for XE and XEX is a Cartesian product of ranges of integers, it is easy, for these constructions, to separate the different tweaks.

8 Combining XE and XEX

Some blockcipher-based constructions need CCA-security in some places and CPA-security in other places. One could assume CCA-security throughout, later instantiating all blockcipher calls with a CCA-secure construction, but it might be better to use a CPA-secure construction where sufficient and a CCA-secure one where necessary. Regardless of subsequent instantiation, it is good to be able to talk, formally, about *where* in a construction one needs *what* assumption.

To formalize where in a construction one is demanding what, we *tag* each blockcipher call with an extra bit. We say that a tweakable blockcipher $\widetilde{E}\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ is *tagged* if $\mathcal{T} = \{0,1\} \times \mathcal{T}^*$ for some nonempty set $\mathcal{T}^*$. Think of $\mathcal{T}^*$, the *effective tweak space*, as the tweak space actually used by the mode. The extra bit indicates what is demanded for each tweak. A first bit of 0 indicates a demand of CPA security, and 1 indicates a demand for CCA security. For a given $T \in \mathcal{T}$ one should be asking for one or the other.

An adversary A launching an attack on a tagged blockcipher is given two oracles, $e(\cdot,\cdot)$ and $d(\cdot,\cdot)$, where the second oracle computes the inverse of the first (meaning $d(T,Y)$ is the unique X such that $e(T,X) = Y$). The adversary must respect the semantics of the tags, meaning that the adversary may not make any query $d(T,Y)$ where the first component of T is 0, and if the adversary makes an oracle query with a tweak $(b,\, T^*)$ then it may make no subsequent query with a tweak $(1-b,\, T^*)$. As always, we insist that there be no pointless queries: an adversary may not repeat an $e(T,X)$ query or a $d(T,Y)$ query, and it may not ask $d(T,Y)$ after having learned $Y = e(T,X)$, nor ask $e(T,X)$ after having learned $X = d(T,Y)$. The definition for security is now as follows.

Definition 9 [Security of a tagged, tweakable blockcipher] *Let* $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ *be a tagged, tweakable blockcipher and let* A *be an adversary. Then* $\mathbf{Adv}^{[\pm]\widetilde{\mathrm{prp}}}_{\widetilde{E}}(A)$ *is defined as* $\Pr[K \xleftarrow{\$} \mathcal{K}:\ A^{\widetilde{E}_K(\cdot,\cdot)\,\widetilde{D}_K(\cdot,\cdot)} \Rightarrow 1] - \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(\mathcal{T},n):\ A^{\pi(\cdot,\cdot)\,\pi^{-1}(\cdot,\cdot)} \Rightarrow 1]$ □

Naturally $\widetilde{D}$, above, is the inverse of $\widetilde{E}$. Security in the $\widetilde{\mathrm{prp}}$-sense and security in the $\pm\widetilde{\mathrm{prp}}$-sense are special cases of security in the $[\pm]\widetilde{\mathrm{prp}}$ sense (but for the enlarged tweak space).

If we combine XE and XEX using our tagging convention we get the tagged, tweakable blockcipher XEX*.

Definition 10 [XEX* construction] *Let* $E\colon \mathcal{K}\times\{0,1\}^n \to \{0,1\}^n$ *be a blockcipher, let* $\alpha_1,\ldots,\alpha_k \in \mathbb{F}^*_{2^n}$, *and let* $\mathbb{I}_1,\ldots,\mathbb{I}_k \subseteq \mathbb{Z}$. *Then* $\widetilde{E} = \mathrm{XEX}^*[E,\ \alpha_1^{\mathbb{I}_1}\cdots\alpha_k^{\mathbb{I}_k}]$ *is the tweakable blockcipher* $\widetilde{E}\colon \mathcal{K} \times (\{0,1\} \times \{0,1\}^n \times \mathbb{I}_1 \cdots \times \mathbb{I}_k) \times \{0,1\}^n \to \{0,1\}^n$ *defined by* $\widetilde{E}_K^{0\,N\,i_1\ldots i_k}(M) = E_K(M \oplus \Delta)$ *and* $\widetilde{E}_K^{1\,N\,i_1\ldots i_k}(M) = E_K(M \oplus \Delta) \oplus \Delta$ *where* $\Delta = \alpha_1^{i_1}\alpha_2^{i_2}\cdots\alpha_k^{i_k}\,\mathsf{N}$ *and* $\mathsf{N} = E_K(N)$. □

9 Security of the Combined Construction

We now specify the security of the XEX* construction. The result encompasses that XE is $\widetilde{\text{prp}}$-secure and XEX is $\pm\widetilde{\text{prp}}$-secure. But the theorem says more, facilitating designs where one intermixes XE-modified blockciphers and XEX-modified blockciphers, both with the same key. The proof is given in Appendix C.

Theorem 11 [Security of XEX*] *Fix $n \geq 1$ and let $\alpha_1, \ldots, \alpha_k \in \mathbb{F}_{2^n}^*$ be base elements and let $\mathbb{I}_1 \times \cdots \times \mathbb{I}_k$ be allowed indices such that these parameters provide unique representations and such that $\alpha_1^{i_1} \cdots \alpha_k^{i_k} \neq 1$ for all $(i_1, \ldots, i_k) \in \mathbb{I}_1 \times \cdots \times \mathbb{I}_k$. Fix a blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and let $\widetilde{E} = \mathit{XEX}^*[E, \alpha_1^{\mathbb{I}_1} \cdots \alpha_k^{\mathbb{I}_k}]$. Then*

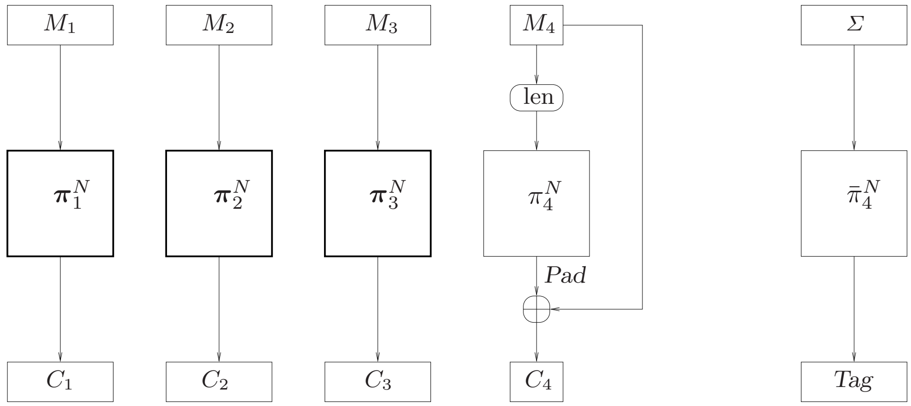
$$\mathbf{Adv}_{\widetilde{E}}^{[\pm]\widetilde{\text{prp}}}(t, q) \quad \leq \quad \mathbf{Adv}_{E}^{\pm\text{prp}}(t', 2q) + \frac{9.5\, q^2}{2^n}$$

where $t' = t + ckn(q+1)$ for some absolute constant c. □

10 The OCB1 Authenticated-Encryption Scheme

We recast OCB [16] to use a tweakable blockcipher instead of a conventional blockcipher. Liskov, Rivest, and Wagner first did this [11], but our formulation is different from theirs. First, guided by what we have done so far, we choose a tweak space of $\mathcal{T} = \{0,1\} \times \{0,1\}^n \times [1 \mathinner{..} 2^{n/2}] \times \{0,1\}$. The first bit of the tweak is the tag; the effective tweak space is $\mathcal{T}^* = \{0,1\}^n \times [1 \mathinner{..} 2^{n/2}] \times \{0,1\}$. Second, we want tweaks to increase monotonically, and so we switch the "special" processing done in OCB from the penultimate block to the final block. The resulting algorithm is shown in Figure 2. Algorithm OCB1 is parameterized by a tweakable blockcipher $\widetilde{E}\colon K \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ and a number $\tau \in [0 \mathinner{..} n]$. For clarity, we write $\boldsymbol{\pi}_i^N$ for $\widetilde{E}_K^{1\,N\,i\,0}$ and π_i^N for $\widetilde{E}_K^{0\,N\,i\,0}$ and $\bar{\pi}_i^N$ for $\widetilde{E}_K^{0\,N\,i\,1}$. By "Partition M into $M[1] \cdots M[m]$" we mean to set $m \leftarrow \max\{\lceil |M|/n \rceil, 1\}$ and define $M[1], \ldots, M[m]$ such that $M[1] \cdots M[m] = M$ and $|M[1]| = \cdots = |M[m-1]| = n$. By "Partition $\mathcal{C}$ into $C[1] \cdots C[m]\, T$" we mean to return Invalid if $|\mathcal{C}| \leq \tau$. Otherwise, we let C be the first $|\mathcal{C}| - \tau$ bits of $\mathcal{C}$ and we let T be the last τ bits of $\mathcal{C}$. Then we set $m \leftarrow \max\{\lceil |C|/n \rceil, 1\}$ and define $C[1], \ldots, C[m]$ such that $C[1] \cdots C[m] = C$ and $|C[1]| = \cdots = |C[m-1]| = n$. By $C[m]0^*$ we mean to append to $C[m]$ enough 0-bits so as to make the resulting string n-bits long. By $X \oplus \mathit{Pad}$ we mean $X[1 \mathinner{..} |\mathit{Pad}|] \oplus \mathit{Pad}$, the xor of Pad with the matching-length prefix of X.

The security of OCB1[Perm$(\mathcal{T}, n)$] is much simpler to prove than the security of OCB[Perm(n)]. (Liskov, Rivest, and Wagner [11] had made the same point for their tweakable-blockcipher variant of OCB.) To state the result we give a couple of definitions from [16]. For privacy of a nonce-based encryption scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ we use the notion of indistinguishability-from-random-strings, which defines $\mathbf{Adv}_{\Pi}^{\text{priv}}(A)$ as $\Pr[K \stackrel{\$}{\leftarrow} \mathcal{K}\colon\ A^{\mathcal{E}_K(\cdot,\cdot)} \Rightarrow 1] - \Pr[A^{\$(\cdot,\cdot)} \Rightarrow 1]$. Here $\$(\cdot,\cdot)$ is an oracle that, on input (N, M), returns $|M|$ random bits. The adversary is not allowed to repeat a nonce N. For authenticity we use the nonce-based notion of integrity of ciphertexts: the adversary is given an encryption oracle $\mathcal{E}_K(\cdot,\cdot)$ and is said to *forge* if it outputs an $(N, \mathcal{C})$ that is valid and $\mathcal{C}$ was not the result of any prior (N, M) query. The adversary is not allowed to repeat a nonce N while it queries its encryption oracle. We write $\mathbf{Adv}_{\Pi}^{\text{auth}}(A)$ for $\Pr[K \stackrel{\$}{\leftarrow} \mathcal{K}\colon\ A^{\mathcal{E}_K(\cdot,\cdot)} \text{ forges }]$. We have the following theorem for the information-theoretic security of OCB1. The proof is in Appendix D.



Algorithm $\text{OCB1.Encrypt}_K^N\,(M)$	**Algorithm** $\text{OCB1.Decrypt}_K^N\,(\mathcal{C})$
Partition M into $M[1]\cdots M[m]$	Partition $\mathcal{C}$ into $C[1]\cdots C[m]\,T$
for $i \in [1\,..\,m-1]$ **do** $C[i] \leftarrow \boldsymbol{\pi}_i^N(M[i])$	**for** $i \in [1\,..\,m-1]$ **do** $M[i] \leftarrow \left(\boldsymbol{\pi}_i^N\right)^{-1}(C[i])$
$\mathit{Pad} \leftarrow \pi_m^N(\text{len}(M[m]))$	$\mathit{Pad} \leftarrow \pi_m^N(\text{len}(C[m]))$
$C[m] \leftarrow M[m] \oplus \mathit{Pad}$	$M[m] \leftarrow C[m] \oplus \mathit{Pad}$
$C \leftarrow C[1]\cdots C[m]$	$M \leftarrow M[1]\cdots M[m]$
$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus C[m]0^* \oplus \mathit{Pad}$	$\Sigma \leftarrow M[1] \oplus \cdots \oplus M[m-1] \oplus C[m]0^* \oplus \mathit{Pad}$
$\mathit{Tag} \leftarrow \bar{\pi}_m^N(\Sigma)$	$\mathit{Tag} \leftarrow \pi_m^N(\Sigma)$
$T \leftarrow \mathit{Tag}\,[\text{first } \tau \text{ bits}]$	$T' \leftarrow \mathit{Tag}\,[\text{first } \tau \text{ bits}]$
return $\mathcal{C} \leftarrow C \,\\|\, T$	**if** $T = T'$ **then return** M **else return** INVALID

Figure 2: $\text{OCB1}[\widetilde{E},\tau]$ with a tweakable blockcipher $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ and tweak space $\mathcal{T} = \{0,1\}\times\{0,1\}^n\times[1\,..\,2^{n/2}]\times\{0,1\}$ and tag length $\tau\in[0\,..\,n]$. We write $\boldsymbol{\pi}_i^N$ and π_i^N and $\bar{\pi}_i^N$ for $\widetilde{E}_K^{1\,N\,i\,0}$ and $\widetilde{E}_K^{0\,N\,i\,0}$ and $\widetilde{E}_K^{0\,N\,i\,1}$.

Theorem 12 [OCB1 with an ideal tweakable blockcipher] *Fix $n \geq 1$, $\tau \in [0\,..\,n]$, and $\mathcal{T} = \{0,1\}\times\{0,1\}^n\times[1\,..\,2^{n/2}]\times\{0,1\}$. Let A be an adversary. Then*

$$\begin{aligned}\mathbf{Adv}^{\text{priv}}_{\text{OCB1}[\text{Perm}(\mathcal{T},n),\tau]}(A) &= 0 \quad \textit{and} \\ \mathbf{Adv}^{\text{auth}}_{\text{OCB1}[\text{Perm}(\mathcal{T},n),\tau]}(A) &\leq \frac{2^{n-\tau}}{2^n-1}\,.\end{aligned}$$

$\square$

Note that the authenticity bound is close to $2^{-\tau}$; in particular, $2^{n-\tau}/(2^n-1) \leq 1/(2^\tau-1)$ for all $\tau \geq 2$. The bounds do not degrade with the number of queries asked by the adversary, the length of these queries, or the time the adversary runs. For the complexity-theoretic analog we have the following.

Corollary 13 [Security of OCB1 with a tweakable blockcipher] *Fix $n \geq 1$, $\tau \in [0\,..\,n]$, $\mathcal{T} = \{0,1\}\times\{0,1\}^n\times[1\,..\,2^{n/2}]\times\{0,1\}$, and $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ a tagged, tweakable blockcipher. Then $\mathbf{Adv}^{\text{priv}}_{\text{OCB1}[\widetilde{E},\tau]}(t,\sigma) \leq \mathbf{Adv}^{\widetilde{\text{prp}}}_{\widetilde{E}}(t',\sigma)$ and $\mathbf{Adv}^{\text{auth}}_{\text{OCB1}[\widetilde{E},\tau]}(t,\sigma) \leq \mathbf{Adv}^{[\pm]\widetilde{\text{prp}}}_{\widetilde{E}}(t',\sigma) + 2^{n-\tau}/(2^n-1)$, where $t' = t + cn\sigma$ for some absolute constant c.* $\square$

The proof requires CPA-security for privacy but authenticity uses the notion that combines CPA- and CCA-security (Definition 9). It is here that one has formalized the intuition that the first $m-1$

Algorithm $\text{OCB1.Encrypt}_K^N\,(M)$	**Algorithm** $\text{OCB1.Decrypt}_K^N\,(\mathcal{C})$
Partition M into $M[1]\cdots M[m]$	Partition $\mathcal{C}$ into $C[1]\cdots C[m]\,T$
$\Delta \leftarrow 2\,E_K(N)$	$\Delta \leftarrow 2\,E_K(N)$
$\Sigma \leftarrow 0^n$	$\Sigma \leftarrow 0^n$
for $i \in [1\,..\,m-1]$ **do**	**for** $i \in [1\,..\,m-1]$ **do**
$C[i] \leftarrow E_K(M[i] \oplus \Delta)\ \oplus\ \Delta$	$M[i] \leftarrow E_K^{-1}(C[i] \oplus \Delta)\ \oplus\ \Delta$
$\Delta \leftarrow 2\Delta$	$\Delta \leftarrow 2\Delta$
$\Sigma \leftarrow \Sigma \oplus M[i]$	$\Sigma \leftarrow \Sigma \oplus M[i]$
$Pad \leftarrow E_K(\text{len}(M[m]) \oplus \Delta)$	$Pad \leftarrow E_K(\text{len}(C[m]) \oplus \Delta)$
$C[m] \leftarrow M[m] \oplus Pad$	$M[m] \leftarrow C[m] \oplus Pad$
$C \leftarrow C[1]\cdots C[m]$	$M \leftarrow M[1]\cdots M[m]$
$\Sigma \leftarrow \Sigma \oplus C[m]0^* \oplus Pad$	$\Sigma \leftarrow \Sigma \oplus C[m]0^* \oplus Pad$
$\Delta \leftarrow 3\Delta$	$\Delta \leftarrow 3\Delta$
$Tag \leftarrow E_K(\Sigma \oplus \Delta)$	$Tag \leftarrow E_K(\Sigma \oplus \Delta)$
$T \leftarrow Tag\,[\text{first } \tau \text{ bits}]$	$T' \leftarrow Tag\,[\text{first } \tau \text{ bits}]$
return $\mathcal{C} \leftarrow C \,\|\, T$	**if** $T = T'$ **then return** M **else return** INVALID

Figure 3: $\text{OCB1}[E,\tau]$ with a conventional blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a tag length $\tau \in [0\,..\,n]$. This coincides with $\text{OCB1}[\widetilde{E},\tau]$ where $\widetilde{E} = \text{XEX}[E, 2^{[1\,..\,2^{n/2}]}3^{\{0,1\}}]$.

tweakable-blockcipher calls to OCB1 need to be CCA-secure but the last two calls need only be CPA-secure.

To realize OCB1 with a conventional blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$, use XEX*, instantiating $\text{OCB1}[\widetilde{E},\tau]$ by way of $\widetilde{E} = \text{XEX}^*[E,\ 2^{\mathbb{I}}3^{\mathbb{J}}]$ where $\mathbb{I} = [1\,..\,2^{n/2}]$ and $\mathbb{J} = \{0,1\}$. Overloading the notation, we write this scheme as $\text{OCB1}[E,\tau]$. The method is rewritten in Figure 3. Security of the blockcipher-based OCB1 follows from Theorem 11 and Corollary 13.

Corollary 14 [OCB1 with a blockcipher] *Fix $n \geq 1$ and $\tau \in [0\,..\,n]$. Assume that $2, 3$ provide unique representations on $[1\,..\,2^{n/2}] \times \{0,1\}$ and $2^i 3^j \neq 1$ for all $(i,j) \in [1\,..\,2^{n/2}] \times \{0,1\}$. Let $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher. Then*

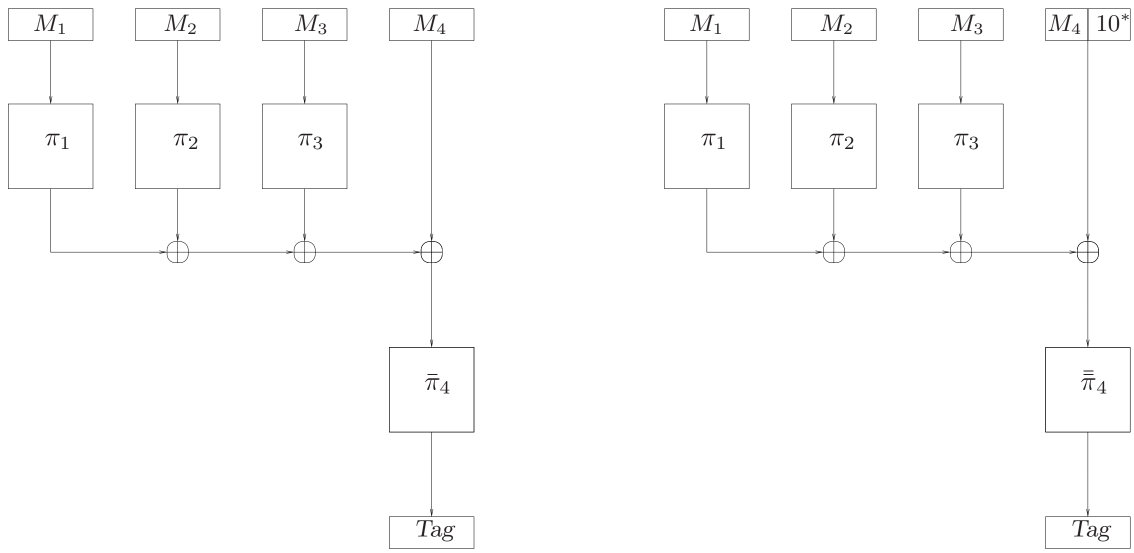
- $\mathbf{Adv}^{\text{priv}}_{\text{OCB1}[E,\tau]}(t,\sigma) \ \leq\ \mathbf{Adv}^{\text{prp}}_{E}(t', 2\sigma) + \dfrac{4.5\sigma^2}{2^n}$ *and*
- $\mathbf{Adv}^{\text{auth}}_{\text{OCB1}[E,\tau]}(t,\sigma) \ \leq\ \mathbf{Adv}^{\pm\text{prp}}_{E}(t', 2\sigma) + \dfrac{9.5\sigma^2}{2^n} + \dfrac{2^{n-\tau}}{2^n - 1}$

where $t' = t + cn\sigma$ for some absolute constant c. □

Propositions 5 and 6 establish that $n = 128$ and $n = 64$ satisfy the requirement for unique representations. They also guarantee that there is no representative of 1 within $[1\,..\,2^{n/2}] \times \{0,1\}$. To see this, note that the propositions imply that $(0,0)$ is the only representative for 1 within a space $\mathbb{I}_1 \times \mathbb{I}_2$ that includes $[1\,..\,2^{n/2}] \times \{0,1\}$, and so there can be *no* representative of 1 within a subspace of $\mathbb{I}_1 \times \mathbb{I}_2$ that excludes $(0,0)$.

Blockcipher-based OCB1 is more efficient than OCB. With OCB one expects to use preprocessing to compute a value $L = E_K(0^n)$ and a collection of $2^i L$-values. This is gone in OCB1; preprocessing is not useful there beyond setting up the underlying blockcipher key. Beyond this, with OCB processing the j^{th} block involved xoring into the current offset a value $L(i) = 2^i L$ where $i = \mathsf{ntz}(j)$ was the number of trailing zero-bits in the index j. In the absence of preprocessing, offset-calculations were not constant time. This too is gone.

The previous paragraph notwithstanding, the time difference or chip-area difference between optimized implementations of OCB and OCB1 will be small, since the overhead of OCB over a



Algorithm $\mathrm{PMAC1}_K\ (M)$ *//Based on a* ***tweakable*** *blockcipher*

Partition M into $M[1]\cdots M[m]$
for $i \in [1\,..\,m-1]$ **do** $Y[i] \leftarrow \pi_i(M[i])$
if $|M[m]| = n$ **then** $\mathit{Tag} \leftarrow \bar{\pi}_m(Y[1] \oplus \cdots \oplus Y[m-1] \oplus M_m)$
            else $\mathit{Tag} \leftarrow \bar{\bar{\pi}}_m(Y[1] \oplus \cdots \oplus Y[m-1] \oplus M_m 10^*)$
$T \leftarrow \mathit{Tag}$ [first τ bits]
return T

Figure 4: $\mathrm{PMAC1}[\widetilde{E},\tau]$ using a tweakable blockcipher $\widetilde{E}$ and a tag length $\tau \in [0\,..\,n]$ where $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ and $\mathcal{T} = \{0,1\}^n \times [1\,..\,2^{n/2}] \times [2\,..\,4]$. We write π_i and $\bar{\pi}_i$ and $\bar{\bar{\pi}}_i$ for $\widetilde{E}_K^{0^n\,i\,2}$ and $\widetilde{E}_K^{0^n\,i\,3}$ and $\widetilde{E}_K^{0^n\,i\,4}$.

mode like CBC was already small. The larger gain is that the mode is simpler to understand, implement, and prove correct.

11 The PMAC1 Message Authentication Code

As with OCB, we recast PMAC [4] to use a tweakable blockcipher. The resulting algorithm is shown in Figure 4. Algorithm $\mathrm{PMAC1}[\widetilde{E},\tau]$ depends on a tweakable blockcipher $\widetilde{E}\colon \mathcal{K}\times\mathcal{T}\times\{0,1\}^n \to \{0,1\}^n$ where $\mathcal{T} = \{0,1\}^n \times [1\,..\,2^{n/2}] \times [2\,..\,4]$, and on a number $\tau \in [1\,..\,n]$. Letting K be implicit in the notation, we write π_i for $\widetilde{E}_K^{0^n\,i\,2}$ and $\bar{\pi}_i$ for $\widetilde{E}_K^{0^n\,i\,3}$ and $\bar{\bar{\pi}}_i$ for $\widetilde{E}_K^{0^n\,i\,4}$ in our description of PMAC1. By "Partition M into $M[1]\cdots M[m]$" we mean to set $m \leftarrow \max\{\lceil |M|/n \rceil, 1\}$ and define $M[1],\ldots,M[m]$ such that $M[1]\cdots M[m] = M$ and $|M[1]| = \cdots = |M[m-1]| = n$. By $X10^*$ we mean $X10^i$ where i is the smallest number such that $|X10^i| = n$.

We have been intentionally unfaithful in abstracting the original PMAC algorithm. In particular, in PMAC1 the tweak for the final blockcipher call depends on the message length m. This is because, when using the powering-up constructions, it is desirable for tweaks to increase monotonically.

We have selected the final component of PMAC1's tweak as being in $[2\,..\,4]$, rather than the more natural range of $[0\,..\,2]$. This is done to separate the tweakable-blockcipher calls used by PMAC1 from those used by OCB1. Arranging that the tweaks be non-overlapping across these

Algorithm $\mathrm{PMAC1}_K\ (M)$ *//Based on a* ***conventional*** *blockcipher*
Partition M into $M[1]\cdots M[m]$
$\Theta \leftarrow 10\, E_K(0^n)$
$\Sigma \leftarrow 0^n$
for $i \leftarrow 1$ **to** $m-1$ **do**
    $Y \leftarrow E_K(M[i] \oplus \Theta)$
    $\Sigma \leftarrow \Sigma \oplus Y$
    $\Theta \leftarrow 2\Theta$
if $\lvert M[m]\rvert = n$ **then** $\Theta \leftarrow 3\Theta,\ \Sigma \leftarrow \Sigma \oplus M[m]$
                else $\Theta \leftarrow 5\Theta,\ \Sigma \leftarrow \Sigma \oplus M[m]10^*$
$\mathit{Tag} \leftarrow E_K(\Sigma \oplus \Theta)$
$T \leftarrow \mathit{Tag}$ [first τ bits]
return T

Figure 5: $\mathbf{PMAC1}[\boldsymbol{E}, \boldsymbol{\tau}]$ using a conventional blockcipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ and a number $\tau \in [0\mathinner{..}n]$. The algorithm coincides with $\mathrm{PMAC1}[\widetilde{E}, \tau]$ where $\widetilde{E} = \mathrm{XE}[E, 2^{[1\mathinner{..}2^{n/2}]}3^{[2\mathinner{..}4]}]$.

two modes facilitates the specification of an authenticated-encryption scheme, AEM, that combines OCB1 and PMAC1 in order to handle associated-data. AEM encryption is defined in Section 12.

PMAC1 is not only secure as a MAC, but as a pseudorandom function (PRF) from $\{0,1\}^*$ to $\{0,1\}^\tau$. For a PRF with such a signature recall that $\mathbf{Adv}^{\mathrm{prf}}_F(A) = \Pr[K \xleftarrow{\$} \mathcal{K}\colon\ A^{F_K(\cdot)} \Rightarrow 1] - \Pr[\rho \xleftarrow{\$} \mathrm{Rand}(\{0,1\}^*, \tau)\colon\ A^{\rho(\cdot)} \Rightarrow 1]$ where $\mathrm{Rand}(\{0,1\}^*, \tau)$ denotes the set of all functions from $\{0,1\}^*$ to $\{0,1\}^\tau$. The definition is extended in the usual way to give resource-parameterized advantage measures, with σ measuring the total number of n-bit blocks asked in all adversary queries.

Theorem 15 [Security of PMAC1 with an ideal tweakable blockcipher] *Fix $n \ge 1$ and $\tau \in [1\mathinner{..}\tau]$. Let $\mathcal{T} = \{0,1\}^n \times [1\mathinner{..}2^{n/2}] \times [2\mathinner{..}4]$. Then*

$$\mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Perm}(\mathcal{T},n),\tau]}(\sigma) \le \frac{\sigma^2}{2^n}\ .$$

□

The proof for the theorem above is given in Appendix E. The following corollary is then obtained in the customary way.

Corollary 16 [Security of PMAC1 with a tweakable blockcipher] *Fix $n \ge 1$ and $\tau \in [1\mathinner{..}n]$. Let $\mathcal{T} = \{0,1\}^n \times [1\mathinner{..}2^{n/2}] \times [2\mathinner{..}4]$ and let $\widetilde{E}\colon \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^n$ be a tweakable blockcipher. Then $\mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\widetilde{E},\tau]}(t, \sigma+1) \le \mathbf{Adv}^{\widetilde{\mathrm{prp}}}_{\widetilde{E}}(t', \sigma) + \sigma^2/2^n$ where $t' = t + cn\sigma$ for some absolute constant c.* □

To realize PMAC1 starting from a conventional blockcipher, use the XE construction, defining $\widetilde{E} = \mathrm{XE}[E, 2^{\mathbb{I}}3^{\mathbb{J}}]$ with $\mathbb{I} = [1\mathinner{..}2^{n/2}]$ and $\mathbb{J} = [2\mathinner{..}4]$. Overloading the notation, when E is an n-bit blockcipher we write $\mathrm{PMAC1}[E, \tau]$. The algorithm is shown in Figure 5. The constant 10 used in the initialization of offset Θ is $10 = 2 \cdot 3^2$ (in $\mathbb{F}_{2^n}$). Security is given below, obtained by combining Corollary 16 and Theorem 7.

Corollary 17 [Security of PMAC1 with a blockcipher] *Fix $n \ge 1$ and $\tau \in [1\mathinner{..}n]$. Assume that $2, 3$ provide unique representations on $[1\mathinner{..}2^{n/2}] \times [2\mathinner{..}4]$. Let $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a* *blockcipher. Then* $\mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[E,\tau]}(t,\sigma) \le \mathbf{Adv}^{\mathrm{prp}}_{E}(t',\sigma+1) + \frac{5.5\,\sigma^2}{2^n}$ *where* $t' = t + cn\sigma$ *for some absolute constant* c. $\square$

Algorithm PMAC1 has advantages over PMAC. With PMAC one had to xor into the current offset a value $L(i) = 2^i L$ where i was the number of trailing zero-bits in the current block index j. Nothing like this exists in PMAC, simplifying offset calculations. Writing an optimized PMAC1 implementation is easier than writing an optimized PMAC implementation because one is freed from worrying about Gray codes, precomputing $L(i)$-values, or finding the most efficient way to bring in the right $L(i)$ value. Finally, PMAC1 enjoys a simpler proof.

12 Comments

To make an authenticated encryption scheme that handles associated-data, combine OCB1 and PMAC1 [14]. Encryption of message M under key K, nonce N, and header H is defined as OCB1.$\mathsf{Encrypt}^N_K(M) \oplus \mathrm{PMAC1}_K(H)$ where the $\oplus$ xors into the end. Omit the $\oplus\, \mathrm{PMAC1}_K(H)$ if $H = \varepsilon$. We call this scheme AEM.

Under the approach suggested by this paper, to get good efficiency for a design that uses a tweakable-blockcipher, the designer must accept certain design rules. In particular, the tweak space needs to look like $\{0,1\}^n \times \mathbb{BIG} \times \mathbb{SMALL}$ for appropriate sets $\mathbb{BIG}$ and $\mathbb{SMALL}$, and one needs to arrange that most tweaks be obtained by incrementing the prior one. It is a thesis implicit in this work that these restrictions are not overly severe.

Besides simplifying the design and proof for OCB and PMAC, we have improved their efficiency. The improvement are not large (the modes were already highly efficient), but performance improvements, of any size, was not a benefit formerly envisaged as flowing from the tweakable-blockcipher abstraction.

Somewhat strangely, our constructions depend on the relative *easiness* of computing discrete logarithms. I know of no other example where one needs to compute discrete logs in order to design or verify a mode of operation.

I end this paper by acknowledging that everyone writes *block cipher*, not *blockcipher*. Still, the time has come to spell this word solid. I invite you to join me.

Acknowledgments

Thanks to David Wagner for pointing out an oversight in an early draft. Useful comments were also received from John Black and the anonymous referees.

This research was supported by NSF 0208842 and by a gift from Cisco System. Thanks to the NSF (particularly Carl Landwehr) and to Cisco (particularly David McGrew) for their kind support of my research.

References

[1] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A concrete security treatment of symmetric encryption: Analysis of the DES modes of operation. *Symposium on Foundations of Computer Science, FOCS '97*, IEEE Computer Society, pp. 394–403, 1997.

[2] M. Bellare, J. Kilian, and P. Rogaway. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, vol. 61, no. 3, Dec 2000. Earlier version in *CRYPTO '94*.

[3] M. Bellare, P. Rogaway, and D. Wagner. The EAX Mode of operation. *Fast Software Encryption, FSE 2004.* Lecture Notes in Computer Science, vol. 3017, Springer-Verlag, pp. 389–407, 2004.

[4] J. Black and P. Rogaway. A block-cipher mode of operation for parallelizable message authentication. *Advances in Cryptology — Eurocrypt '02.* Lecture Notes in Computer Science, vol. 2332, Springer-Verlag, pp. 384–397, 2002.

[5] J. Black and P. Rogaway. CBC MACs for arbitrary-length messages: the three-key constructions. To appear in *J. of Cryptology*. Earlier version in *Advances in Cryptology — CRYPTO '00*.

[6] V. Gligor and P. Donescu. Fast encryption and authentication: XCBC encryption and XECB authentication modes. *Fast Software Encryption, FSE 2001.* Lecture Notes in Computer Science, vol. 2355, Springer-Verlag, pp. 92–108, 2001.

[7] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, vol. 28, April 1984, pp. 270–299.

[8] S. Halevi and P. Rogaway. A parallelizable enciphering mode. *Topics in Cryptology — CT-RSA 2004.* Lecture Notes in Computer Science, vol. 2964, Springer-Verlag, pp. 292–304, 2004.

[9] J. Kilian and P. Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *J. of Cryptology*, vol. 14, no. 1, pp. 17–35, 2001.

[10] C. Jutla. Encryption modes with almost free message integrity. *Advances in Cryptology — EUROCRYPT 2001.* Lecture Notes in Computer Science, vol. 2045, Springer-Verlag, pp. 529–544, 2001.

[11] M. Liskov, R. Rivest, and D. Wagner. Tweakable block ciphers. *Advances in Cryptology — CRYPTO '02.* Lecture Notes in Computer Science, vol. 2442, Springer-Verlag, pp. 31–46, 2002.

[12] S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Transactions on Information Theory*, vol 24, pp. 106–110, 1978.

[13] J. Pollard. Monte Carlo methods for index computation (mod p). *Mathematics of Computation*, vol. 32, pp. 918–924, 1978.

[14] P. Rogaway. Authenticated-encryption with associated-data. *ACM Conference on Computer and Communications Security 2002, CCS 2002.* ACM Press, pp. 98–107, 2002.

[15] P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. *Advances in Cryptology — Asiacrypt '04.* Lecture Notes in Computer Science, Springer-Verlag, 2004. Proceedings version of this paper.

[16] P. Rogaway, M. Bellare, and J. Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Transactions on Information and System Security*, vol. 6, no. 3, pp. 365–403, 2003. Earlier version, with T. Krovetz, in *CCS 2001*.

[17] R. Schroeppel. The hasty pudding cipher. AES candidate submitted to NIST, 1998.

[18] D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). Network Working Group RFC 3610. The Internet Society, September 2003.

A Tweakable Blockciphers Implicit in Prior Work

When tweaks increase in sequence, the most efficient constructions formerly known for a tweakable blockcipher are those implicit in earlier modes [4, 6, 10, 16], recast in view of Liskov, Rivest, and Wagner [11]. In particular:

- Jutla [10] might be seen as suggesting a construction (among others) of $\widetilde{E}$: $(\mathcal{K}\times\mathcal{K}')\times(\{0,1\}^n\times \mathbb{Z}_p^+)\times\{0,1\}^n \to \{0,1\}^n$ by way of $\widetilde{E}^{N,i}_{KK'}(X) = E_K(X\oplus\Delta)\oplus\Delta$ where $\Delta = i\ell \bmod p$ and $\ell = E_{K'}(N)$ and p is the largest prime less than 2^n.
- Gligor and Donescu [6] might be seen as suggesting constructions like $\widetilde{E}$: $(\mathcal{K}\times\{0,1\}^n)\times [1\,..\,2^n-1]\to\{0,1\}^n$ by $\widetilde{E}^{i}_{K,r}(X) = E_K(X+\delta)$ where $\delta = ir$ and addition is done modulo 2^n.
- Rogaway, Bellare, and Black [16] might be seen as implicitly suggesting making a tweakable blockcipher $\widetilde{E}$: $\mathcal{K}\times(\{0,1\}^n\times[0\,..\,2^{n-2}])\times\{0,1\}^n\to\{0,1\}^n$ from an ordinary blockcipher E: $\mathcal{K}\times\{0,1\}^n\to\{0,1\}^n$ by way of $\widetilde{E}^{N,i}_{K}(X) = E_K(X\oplus\Delta)\oplus\Delta$ where $\Delta = \gamma_i L\oplus R$ and $L = E_K(0^n)$ and $R = E_K(N\oplus L)$ and γ_i is the i-th Gray-code coefficient.
- Black and Rogaway [4] might be seen as making $\widetilde{E}$: $\mathcal{K}\times[0\,..\,2^{n-2}]\times\{0,1\}^n\to\{0,1\}^n$ out of E: $\mathcal{K}\times\{0,1\}^n\to\{0,1\}^n$ by $\widetilde{E}^{i}_{K}(X) = E_K(X\oplus\Delta)$ where $\Delta = \gamma_i L$ and $L = E_K(0^n)$ and γ_i is as before.
- The last two definitions ignore the "special" treatment afforded to blocks modified by xoring in $2^{-1}L$. The implicit intent [4, 16] was to use this mechanism to enlarge the tweak space by one bit, effectively taking the cross product with $\{0,1\}$.

B Proof of Theorem 7 — Security of XE

Let A be an adversary trying to distinguish $\widetilde{E}$ from the family all tweakable permutations with the same tweak space. Say that A runs in time t and makes exactly q queries. Without loss of generality assume that A is deterministic. We give a hybrid argument. In particular, we introduce the following hybrids:

(1) $p_1 = \Pr[K \xleftarrow{\$} \mathcal{K}:\ A^{\widetilde{E}_K(\cdot,\cdot)} \Rightarrow 1]$

(2) $p_2 = \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(n):\ A^{\widetilde{\pi}(\cdot,\cdot)} \Rightarrow 1]$

(3) $p_3 = \Pr[\rho \xleftarrow{\$} \mathrm{Rand}(n):\ A^{\widetilde{\rho}(\cdot,\cdot)} \Rightarrow 1]$

(4) $p_4 = \Pr[\rho \xleftarrow{\$} \mathrm{Rand}(\mathcal{T},n):\ A^{\rho(\cdot,\cdot)} \Rightarrow 1]$

(5) $p_5 = \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(\mathcal{T},n):\ A^{\pi(\cdot,\cdot)} \Rightarrow 1]$

Let us explain the notation. In the experiment associated to (1) a query $(N, i_1,\ldots,i_k)$, M is answered with $E_K(M\oplus\Delta)$ where $\Delta = \alpha_1^{i_1}\cdots\alpha_k^{i_k}\,\mathsf{N}$ and $\mathsf{N} = E_K(N)$. Here K is a random key for the tweakable blockcipher. In the experiment associated to (2) a query $(N, i_1,\ldots,i_k)$, M is answered with $\pi(M\oplus\Delta)$ where $\Delta = \alpha_1^{i_1}\cdots\alpha_k^{i_k}\,\mathsf{N}$ and $\mathsf{N} = \pi(N)$. Here π is a random permutation on n bits. In the experiment associated to (3) a query $(N, i_1,\ldots,i_k)$, M is answered with $\rho(M\oplus\Delta)$ where $\Delta = \alpha_1^{i_1}\cdots\alpha_k^{i_k}\mathsf{N}$ and $\mathsf{N} = \rho(N)$. Here ρ is a random function from n bits to n bits. In the experiment associated to (4) a query $(N, i_1,\ldots,i_k)$, M is answered with $\rho((N, i_1,\ldots,i_k), M)$. Here ρ is a random function that takes a tweak $(N, i_1,\ldots,i_k)$ and an n-bit string M and returns n bits. In the experiment associated to (5) a query $(N, i_1,\ldots,i_k)$, M is answered with $\pi((N, i_1,\ldots,i_k), M)$. Here $\pi(T,\cdot)$ is a random function permutation on n bits for each tweak $T = (N, i_1,\ldots,i_k)$.

The value we must bound is $p_1 - p_5 = (p_1 - p2) + (p_2 - p_3) + (p_3 - p_4) + (p_4 - p_5)$. Three of these four addends are easy to bound:

Initialization:

000 $\quad bad \leftarrow \mathsf{false};\quad$ **for** all $X \in \{0,1\}^n$ **do** $\rho(X) \leftarrow \mathsf{undefined}$

Respond to the jth query, $(N^j, i_1^j \cdots i_k^j)$, M^j, as follows:

100 $\quad$ **if** $N^j = N^i$ for some $i < j$ **then** $\mathsf{N}^j \leftarrow \mathsf{N}^i$

101 $\quad$ **else** $\mathsf{N}^j \xleftarrow{\$} \{0,1\}^n$

102 $\qquad$ **if** $N^j \in \mathrm{Domain}(\rho)$ **then** $bad \leftarrow \mathsf{true}$, $\mathsf{N}^j \leftarrow \rho(N^j)$

103 $\quad \rho(N^j) \leftarrow \mathsf{N}^j$

104 $\quad Y^j \xleftarrow{\$} \{0,1\}^n$

105 $\quad X^j \leftarrow M^j \oplus \alpha_1^{i_1^j} \cdots \alpha_k^{i_k^j}\, \mathsf{N}^j$

106 $\quad$ **if** $X^j \in \mathrm{Domain}(\rho)$ **then** $bad \leftarrow \mathsf{true}$, $Y^j \leftarrow \rho(X^j)$

107 $\quad \rho(X^j) \leftarrow Y^j$

108 $\quad$ **return** Y^j

Figure 6: Game P3 (as written) and Game P4 (with the shaded statements omitted). The former accurately simulates the environment defining p_3, while the latter accurately simulates the environment defining p_4.

- $p_1 - p_2 \leq \mathbf{Adv}_E^{\mathrm{prp}}(t', 2q)$ where $t' = t + ckn(q+1)$ for some constant c depending on details of the model of computation. This is because one strategy for distinguishing an oracle $f = E_K(\cdot)$ (for a random key K) from an oracle $f = \pi(\cdot)$ (where π is a random permutation on n bits) is to run A, answering each query according to the construction $\widetilde{f}$, namely, answering $((N, i_1, \cdots, i_k), M)$ by $f(M + \Delta)$ where $\Delta = \alpha_1^{i_1} \cdots \alpha_k^{i_k} f(N)$. The time to do this is $cq + c'$ beyond the time that the adversary A runs, while the number of queries asked to the f-oracle is $2q$.
- $p_2 - p_3 \leq 2q^2/2^n$. This is the standard replacement of a random permutation by a random function (see, for example, [2]). The number of queries is at most $2q$, so the discrepancy between what the permutation gives and what the function gives is at most $0.5 \cdot 2q(2q-1)/2^n \leq 2q^2/2^n$.
- $p_5 - p_4 \leq 0.5q^2/2^n$. This is again the standard switching of a random function and a random permutation (the tweak that indexes the function makes no difference).

We are left with the task of bounding $p_4 - p_3$. This is done with a game-playing argument, as used in works like [9].

Refer to Figure 6, which defines Games P3 and P4. The former is an accurate simulation of the experiment defining p_3 (the CPA powering-up construction applied to a random function ρ). Thus $p_3 = \Pr[A^{\mathrm{GameP3}} \Rightarrow 1]$. The latter is an accurate simulation of the experiment defining p_4 (every value returned to the adversary is a random n-bit string). (Recall that, by convention, an adversary may not repeat a query.) Thus $p_4 = \Pr[A^{\mathrm{GameP4}} \Rightarrow 1]$. Since Games P3 and P4 are syntactically identical until the flag *bad* gets set to true, the usual game-playing argument ensures that $p_3 - p_4 \leq \Pr[A \text{ sets } bad \text{ in Game P4}]$. Our task is to bound that probability.

The flag *bad* gets set to true in Game P4 if there is ever a collision among: the distinct N^j-values together with all of the X^j-values. The N^j-values cannot collide among themselves because of the **if** statement at line 100. But N^i-values can collide with X^j-values (for any $i, j \in [1\,..\,q]$) and X^i-values can collide with X^j-values (for distinct $i, j \in [1\,..\,q]$).

We observe that the Y^i-values returned to the adversary are independent of the X^j-values placed into $\mathrm{Domain}(\rho)$. (Observe that Y^j is selected at random on line 104 and then returned to the adversary at line 108. It never has an impact in determining $\mathrm{Domain}(\rho)$ and its being placed in the range at line 108 is irrelevant, as range-values are never consulted in Game P4.) Since the values that the adversary is learning have nothing to do with the adversary's task at hand (to set *bad*) we may assume that the adversary is non-adaptive. Thus it asks some fixed sequence $((N^1, i_1^1 \cdots i_k^1), M^1), \ldots, ((N^q, i_1^q \cdots i_k^q), M^q)$ hoping that a resulting N^i and X^j collide, or that a resulting X^i and X^j that collide.

Fix $i, j \in [1 \,..\, q]$. We claim that the chance that $N^i = X^j$ is exactly 2^{-n}. For $\Pr[N^i = X^j] = \Pr[N^i = M^j \oplus \alpha_1^{i_1^j} \cdots \alpha_k^{i_k^j} \, \mathsf{N}^j]$. The N^j value was selected at random and its multiplier is nonzero, so this probability is just 2^{-n}.

Fix distinct $i, j \in [1 \,..\, q]$. We claim that the chance that $X^i = X^j$ is exactly 2^{-n}. We are considering $\Pr[X^i = X^j] = \Pr[M^i \oplus \alpha_1^{i_1^i} \cdots \alpha_k^{i_k^i} \, \mathsf{N}^i = M^j \oplus \alpha_1^{i_1^j} \cdots \alpha_k^{i_k^j} \, \mathsf{N}^j]$. Now if $N^i \neq N^j$ then N^i and N^j are independent random n-bit strings, the multiplier in front of them is nonzero, and so this probability is certainly 2^{-n}. If $N^i = N^j$ then $\mathsf{N}^i = \mathsf{N}^j$ is a random n-bit string and we are looking at $\Pr[M^i \oplus M^j = (\alpha_1^{i_1^i} \cdots \alpha_k^{i_k^i} - \alpha_1^{i_1^j} \cdots \alpha_k^{i_k^j}) \, \mathsf{N}^i]$. Since $N^i = N^j$ and queries are distinct, if $(i_1^i, \ldots, i_k^i) = (i_1^j, \ldots, i_k^j)$ then $M^i \neq M^j$ and the probability above is 0. If, instead, $(i_1^i, \ldots, i_k^i) \neq (i_1^j, \ldots, i_k^j)$ then $\alpha_1^{i_1^i} \cdots \alpha_k^{i_k^i} \neq \alpha_1^{i_1^j} \cdots \alpha_k^{i_k^j}$ because of the unique-representation condition and so the multiplier in front of N^i is nonzero and the the probability in question is 2^{-n}.

We have shown that for every pair of values added into the domain of ρ the probability that they collide is at most 2^{-n}. Since there are at most $\binom{}{2q2 \leq 2q^2}$ such values the sum bound tells us that $p_3 - p_4 \leq 2q^2/2^n$. Adding this to $(p_1 - p_2) + (p_2 - p_3) + (p_5 - p_4) \leq 2.5q^2/2^n + \mathbf{Adv}_E^{\mathrm{prp}}(t', 2q)$ gives the desired result.

C Proof of Theorem 11 — Security of XEX*

Let A be an adversary that attacks $\widetilde{E}$. Say that A runs in time at most t and makes exactly q queries (if it makes fewer oracle queries on a given run then have it ask additional, valid queries until it has made q calls). Without loss of generality assume that A is deterministic.

We start off with a hybrid argument. In particular, we introduce the following hybrids:

(1) $p_1 = \Pr[K \xleftarrow{\$} \mathcal{K} : \; A^{\widetilde{E}_K(\cdot,\cdot)\, \widetilde{D}_K(\cdot,\cdot)} \Rightarrow 1]$

(2) $p_2 = \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(n) : \; A^{\widetilde{\pi}(\cdot,\cdot)\, \widetilde{\pi}^{-1}(\cdot,\cdot)} \Rightarrow 1]$

(3) $p_3 = \Pr[A^{\$(\cdot,\cdot)\, \$(\cdot,\cdot)} \Rightarrow 1]$

(4) $p_4 = \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(\mathcal{T}, n) : \; A^{\pi(\cdot,\cdot)\, \pi^{-1}(\cdot,\cdot)} \Rightarrow 1]$

Let us explain the notation above. In the experiment associated to (1) a query $(b, N, i_1, \ldots, i_k)$, M to the adversary's left oracle is answered with $E_K(M \oplus \Delta)$ if $b = 0$ and with $E_K(M \oplus \Delta) \oplus \Delta$ if $b = 1$, where $\Delta = \alpha_1^{i_1} \cdots \alpha_k^{i_k} \, \mathsf{N}$ and $\mathsf{N} = E_K(N)$. A query to the adversary's right oracle is answered with $D_K(M \oplus \Delta)$ if $b = 0$ and $D_K(M \oplus \Delta) \oplus \Delta$ if $b = 1$. Here K is a random key for the blockcipher E. In the experiment associated to (2) a query $(b, N, i_1, \ldots, i_k)$, M to the adversary's left oracle is answered with $\pi(M \oplus \Delta)$ if $b = 0$ and $\pi(M \oplus \Delta) \oplus \Delta$ if $b = 1$, where $\Delta = \alpha_1^{i_1} \cdots \alpha_k^{i_k} \, \mathsf{N}$ and $\mathsf{N} = \pi(N)$. A query to the adversary's right oracle is answered with $D_K(M \oplus \Delta)$ if $b = 0$ and $D_K(M \oplus \Delta) \oplus \Delta$ if $b = 1$. Here π is a random permutation on n bits. In the experiment associated to (3) a query $(b, N, i_1, \ldots, i_k)$, M to either oracle is answered with n random bits. In the experiment associated to (4) a left query of $(b, N, i_1, \ldots, i_k)$, M is answered by $\pi_{N, i_1, \ldots, i_k}(M)$ and a right query of $(b, N, i_1, \ldots, i_k)$, M is answered with $\pi^{-1}_{N, i_1, \ldots, i_k}(M)$ where each π_T is a random permutation on n bits.

The value we must bound is $p_1 - p_4 = (p_1 - p_2) + (p_2 - p_3) + (p_3 - p_4)$. Two of these three addends are easy to bound:

- $p_1 - p_2 \le \mathbf{Adv}_E^{\pm\mathrm{prp}}(t', 2q)$ where $t' = t + ckn(q+1)$ for some constant c depending only on details of the model of computation. This is because one strategy for distinguishing oracles $(F(\cdot), G(\cdot)) = (E_K(\cdot), D_K(\cdot))$ from $(F, G) = (\pi(\cdot), \pi^{-1}(\cdot))$ is embodied by the following adversary $B^{F(\cdot)\,G(\cdot)}$: run $A^{f\,g}$, answering each query $f((b, N, \alpha_1, \ldots, \alpha_k), M)$ by $F(M \oplus \Delta)$ if $b = 0$ and $F(M \oplus \Delta) \oplus \Delta$ if $b = 1$, and answering each query $g((b, N, \alpha_1, \ldots, \alpha_k), M)$ by $G(M \oplus \Delta)$ if $b = 0$ and $G(M \oplus \Delta) \oplus \Delta$ if $b = 1$, where $\Delta = \alpha_1^{i_1} \cdots \alpha_k^{i_k} \mathsf{N}$ and $\mathsf{N} = F(N)$. When A halts with output β have B halt with output β. The time for this adversary is $ckn(q+1)$ beyond the time that the adversary A runs, for some constant c; the total number of queries asked by B is $2q$; and $\mathbf{Adv}_E^{\pm\mathrm{prp}}(B) = p_1 - p_2$.
- $p_3 - p_4 \le 2q^2/2^n$. This is just the standard replacement of a random permutation and its inverse by a pair of random functions. The number of queries is at most $2q$, so the discrepancy between what the permutation gives and what the function gives is at most $0.5 \cdot 2q(2q-1)/2^n \le 2q^2/2^n$. Recall that we have forbidden the types of queries that trivially allow the distinguishing of these two kinds of oracles, such as repeating an oracle query.

We are left with the task of bounding $p_2 - p_3$. This is done with a game-playing argument, as used in works like [9]. We begin with a game, call it Game 2, that accurately simulates the pair of oracles $(\widetilde{\pi}, \widetilde{\pi}^{-1})$ that we have defined. See Figure 7. Also defined in Figure 7 is Game 3, which is obtained by omitting the eight shaded statements.

An inspection of Game 2 will make clear that it provides to the adversary an identical view as that which is obtained by giving the adversary a random $(\widetilde{\pi}(\cdot,\cdot), \widetilde{\pi}^{-1}(\cdot,\cdot))$ oracle. As such,

$$p_2 \;=\; \Pr[\pi \xleftarrow{\$} \mathrm{Perm}(n) :\; A^{\widetilde{\pi}(\cdot,\cdot)\,\widetilde{\pi}^{-1}(\cdot,\cdot)} \Rightarrow 1] \;=\; \Pr[A^{\text{Game 2}} \Rightarrow 1]\;. \tag{1}$$

On the other hand, it is easy to see that Game 3 simply returns n random bits in response to each query, and so

$$p_3 \;=\; \Pr[A^{\$(\cdot,\cdot)\,\$(\cdot,\cdot)} \Rightarrow 1] \;=\; \Pr[A^{\text{Game 3}} \Rightarrow 1]\;. \tag{2}$$

The quantity that we must bound is $p_2 - p_3$, and so

$$p_2 - p_3 \;=\; \Pr[A^{\text{Game 2}} \Rightarrow 1] - \Pr[A^{\text{Game 3}} \Rightarrow 1]\;. \tag{3}$$

Since Games 2 and 3 have been setup to be syntactically identical apart from that which happens following the setting of the flag *bad* to true, the usual game-playing method informs us that

$$p_2 - p_3 \;\le\; \Pr[A^{\text{Game 3}} \text{ sets } \mathit{bad}\,]\;. \tag{4}$$

To understand Game 3 we will make some modifications to it. Begin by *eliminating* lines 24, 30, 44, and 50. For clarity, we have rewritten Game 3A as Figure 8 This results in a different game, call it Game 3A. In eliminating these lines we may decrease the probability that *bad* gets set to true, but it is easy to see by how much we have lessened this probability: by at most $(1 + 2 + \cdots + 2q - 1)/2^n \le 2q^2/2^n$. We record this for future reference:

$$\Pr[A^{\text{Game 3}} \text{ sets } \mathit{bad}\,] \;\le\; \Pr[A^{\text{Game 3A}} \text{ sets } \mathit{bad}\,] + \frac{2q^2}{2^n} \tag{5}$$

We now make some changes to Game 3A, changes that have no adversarially-visible effect. See Figure 9. (1) We start of by indexing the calls, numbering them from 1 to q. This lets us to drop the

Initialization:
    Let $\pi[\cdot]$ be everywhere $\mathsf{undefined}$
    Let $\mathit{haveSeen}[\cdot]$ be everywhere false
    $\mathit{bad} \leftarrow \mathsf{true}$

On a left query of (T, M):
    Parse T into $(b, N, i_1, \ldots, i_k)$
    if $\mathit{haveSeen}[N]$ **then** $\mathsf{N} \leftarrow \pi[N]$ **else**
        $\mathit{haveSeen}[N] \leftarrow \mathsf{true}$
        $\mathsf{N} \xleftarrow{\$} \{0,1\}^n$
        if $\mathsf{N} \in \mathrm{Range}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $\mathsf{N} \xleftarrow{\$} \overline{\mathrm{Range}}(\pi)$
        if $\pi[N] \neq \mathsf{undefined}$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $\mathsf{N} \leftarrow \pi[N]$
        $\pi[N] \leftarrow \mathsf{N}$
    $\Delta \leftarrow \alpha_1^{i_1} \cdots \alpha_k^{i_k}\, \mathsf{N}$
    $X \leftarrow M \oplus \Delta$
    $Y \xleftarrow{\$} \{0,1\}^n$
    if $Y \in \mathrm{Range}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $Y \xleftarrow{\$} \overline{\mathrm{Range}}(\pi)$
    if $X \in \mathrm{Domain}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $Y \leftarrow \pi[X]$
    $\pi[X] \leftarrow Y$
    if $b = 0$ **then** $C \leftarrow Y$
    if $b = 1$ **then** $C \leftarrow Y \oplus \Delta$
    return C

On a right query of (T, C):
    Parse T into $(N, i_1, \ldots, i_k)$
    if $\mathit{haveSeen}[N]$ **then** $\mathsf{N} \leftarrow \pi[N]$ **else**
        $\mathit{haveSeen}[N] \leftarrow \mathsf{true}$
        $\mathsf{N} \xleftarrow{\$} \{0,1\}^n$
        if $\mathsf{N} \in \mathrm{Range}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $\mathsf{N} \xleftarrow{\$} \overline{\mathrm{Range}}(\pi)$
        if $\pi[N] \neq \mathsf{undefined}$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $\mathsf{N} \leftarrow \pi[N]$
        $\pi[N] \leftarrow \mathsf{N}$
    $\Delta \leftarrow \alpha_1^{i_1} \cdots \alpha_k^{i_k}\, \mathsf{N}$
    $Y \leftarrow C \oplus \Delta$
    $X \xleftarrow{\$} \{0,1\}^n$
    if $X \in \mathrm{Domain}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $X \xleftarrow{\$} \overline{\mathrm{Domain}}(\pi)$
    if $Y \in \mathrm{Range}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$, $X \leftarrow \pi^{-1}[Y]$
    $\pi[X] \leftarrow Y$
    $M \leftarrow X \oplus \Delta$
    return M

Figure 7: Definition of Game 2, as written, and Game 3, which is obtained by omitting the eight shaded statements.

haveSeen predicate, replacing it by something equivalent. (2) We move the choice of return values up to the initialization step. (3) On a left oracle-query we define the internal value Y from the return value $Z = C$, as as opposed to defining the return value from Y. (4) On a right oracle-query we define the internal value X from the return value $Z = M$, as opposed to defining the return

Initialization:
Let $\pi[\cdot]$ be everywhere $\mathsf{undefined}$
Let $\mathit{haveSeen}[\cdot]$ be everywhere false
$\mathit{bad} \leftarrow \mathsf{true}$

On a left query of (T, M):
Parse T into $(b, N, i_1, \ldots, i_k)$
if $\mathit{haveSeen}[N]$ **then** $\mathsf{N} \leftarrow \pi[N]$ **else**
    $\mathit{haveSeen}[N] \leftarrow \mathsf{true}$
    $\mathsf{N} \xleftarrow{\$} \{0,1\}^n$
    if $\pi[N] \neq \mathsf{undefined}$ **then** $\mathit{bad} \leftarrow \mathsf{true}$
    $\pi[N] \leftarrow \mathsf{N}$
$\Delta \leftarrow \alpha_1^{i_1} \cdots \alpha_k^{i_k}\, \mathsf{N}$
$X \leftarrow M \oplus \Delta$
$Y \xleftarrow{\$} \{0,1\}^n$
if $X \in \mathrm{Domain}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$
$\pi[X] \leftarrow Y$
if $b = 0$ **then** $C \leftarrow Y$
if $b = 1$ **then** $C \leftarrow Y \oplus \Delta$
return C

On a right query of (T, C):
Parse T into $(N, i_1, \ldots, i_k)$
if $\mathit{haveSeen}[N]$ **then** $\mathsf{N} \leftarrow \pi[N]$ **else**
    $\mathit{haveSeen}[N] \leftarrow \mathsf{true}$
    $\mathsf{N} \xleftarrow{\$} \{0,1\}^n$
    if $\pi[N] \neq \mathsf{undefined}$ **then** $\mathit{bad} \leftarrow \mathsf{true}$
    $\pi[N] \leftarrow \mathsf{N}$
$\Delta \leftarrow \alpha_1^{i_1} \cdots \alpha_k^{i_k}\, \mathsf{N}$
$Y \leftarrow C \oplus \Delta$
$X \xleftarrow{\$} \{0,1\}^n$
if $Y \in \mathrm{Range}(\pi)$ **then** $\mathit{bad} \leftarrow \mathsf{true}$
$\pi[X] \leftarrow Y$
$M \leftarrow X \oplus \Delta$
return M

Figure 8: Game 3A, obtained by dropping four statements from Game 3. The probability that bad is set to true may be lessened compared to Game 3, but not by more than by $2q^2/2^n$.

value from X. (5) We delay the computation of *bad* until a finalization step that runs after the adversary has asked its queries. The choices above do not impact the probability that *bad* gets set to true and, in particular,

$$\Pr[A^{\text{Game 3A}} \text{ sets } \mathit{bad}\,] \quad = \quad \Pr[A^{\text{Game 3B}} \text{ sets } \mathit{bad}\,] \tag{6}$$

and our task has been reduced to bounding the probability that *bad* gets set to true at in Game 3B.

At this point we make the observation that, in Game 3B, the association of domain points to range points that is maintained by π is of no significance beyond the "bookkeeping" that π does for recording which values are in the domain of π and which values are in the range of π and what random value is associated to each $\pi[N^r]$. We could just as well have collected up all the domain points which get added to π in a multiset $\mathcal{X}$, and looked for collision, and gathered up all the range

Initialization:

Let $\pi[\cdot]$ be everywhere $\mathsf{undefined}$
$Z^1, \ldots, Z^q \xleftarrow{\$} \{0,1\}^n$
$bad \leftarrow \mathsf{true}$

If the r-th query is a left query (T^r, M^r):

return Z^r

If the r-th query is a right query (T^r, C^r):

return Z^r

Finalization:

for $r \leftarrow 1$ **to** q **do**

    if the r-th query was a left query (T^r, M^r) **then**
        Parse T^r into $(b^r, N^r, i_1^r, \ldots, i_k^r)$
        if $N^r = N^p$ for some $p < r$ **then** $\mathsf{N}^r \leftarrow \pi[N^r]$ **else**
            $\mathsf{N}^r \xleftarrow{\$} \{0,1\}^n$
            if $\pi[N^r] \neq \mathsf{undefined}$ **then** $bad \leftarrow \mathsf{true}$
            $\pi[N^r] \leftarrow \mathsf{N}^r$
        $\Delta^r \leftarrow \alpha_1^{i_1^r} \cdots \alpha_k^{i_k^r}\, \mathsf{N}^r$
        $X^r \leftarrow M^r \oplus \Delta^r$
        if $X^r \in \mathrm{Domain}(\pi)$ **then** $bad \leftarrow \mathsf{true}$
        if $b^r = 0$ **then** $Y^r \leftarrow Z^r$
        if $b^r = 1$ **then** $Y^r \leftarrow Z^r \oplus \Delta^r$
        $\pi[X^r] \leftarrow Y^r$

    if the r-th query was a right query (T^r, C^r) **then**
        Parse T^r into $(b^r, N^r, i_1^r, \ldots, i_k^r)$
        if $N^r = N^p$ for some $p < r$ **then** $\mathsf{N}^r \leftarrow \pi[N^r]$ **else**
            $\mathsf{N}^r \xleftarrow{\$} \{0,1\}^n$
            if $\pi[N^r] \neq \mathsf{undefined}$ **then** $bad \leftarrow \mathsf{true}$
            $\pi[N^r] \leftarrow \mathsf{N}^r$
        $\Delta^r \leftarrow \alpha_1^{i_1^r} \cdots \alpha_k^{i_k^r}\, \mathsf{N}^r$
        $Y^r \leftarrow C^r \oplus \Delta^r$
        if $Y^r \in \mathrm{Range}(\pi)$ **then** $bad \leftarrow \mathsf{true}$
        $X^r \leftarrow Z^r \oplus \Delta^r$
        $\pi[X^r] \leftarrow Y^r$

Figure 9: Definition of Game 3B, which sets flag *bad* with the same probability as in Game 3A.

points which get added to π in a multiset $\mathcal{Y}$, and looked for collisions. As such, we can eliminate π, accomplish the bookkeeping differently, and check for what would have been collision in the domain or range of π at the very end. All of this is done in Game 3C, show in in Figure 10. Though it looks quite different from Game 3B it sets *bad* under exactly the same circumstances, so

$$\Pr[A^{\mathrm{GameB}} \text{ sets } bad\,] \quad = \quad \Pr[A^{\mathrm{GameC}} \text{ sets } bad\,] \tag{7}$$

and our task is to evaluate the probability that *bad* gets set in Game 3C.

In Game 3C the adversary asks a sequence of questions and gets back a sequence of random answers. It can only help the adversary if we were to give it all of the answers in advance. The adversary is then at liberty to choose its questions in a way that depends on knowledge of future

Initialization:

$Z^1, \ldots, Z^q \xleftarrow{\$} \{0,1\}^n$

If the r-th query is a left query (T^r, M^r):

  return Z^r

If the r-th query is a right query (T^r, C^r):

  return Z^r

Finalization:

  for $r \in [1\,..\,q]$ **do** parse T^r into $(b^r, N^r, i_1^r, \ldots, i_k^r)$
  Let $N_1, \ldots, N_p$ be the distinct strings from $N^1, \ldots, N^q$
  for $i \in [1\,..\,p]$ **do** $\mathsf{N}_{N_i} \xleftarrow{\$} \{0,1\}^n$

  for $r \in [1\,..\,q]$ **do**

    $\Delta^r \leftarrow \alpha_1^{i_1^r} \cdots \alpha_k^{i_k^r}\, \mathsf{N}_{N^r}$

    if the r-th query was a left query (T^r, M^r) **then**
      $X^r \leftarrow M^r \oplus \Delta^r$
      if $b^r = 0$ **then** $Y^r \leftarrow Z^r$
      if $b^r = 1$ **then** $Y^r \leftarrow Z^r \oplus \Delta^r$

    if the r-th query was a right query (T^r, C^r) **then**
      $X^r \leftarrow Z^r \oplus \Delta^r$
      $Y^r \leftarrow C^r \oplus \Delta^r$

  $\mathcal{X} \leftarrow [N_1, \ldots, N_p,\ X^1, \ldots, X^q]$
  $\mathcal{Y} \leftarrow [\mathsf{N}_{N_1}, \ldots, \mathsf{N}_{N_p},\ Y^1, \ldots, Y^q]$

  $bad \leftarrow$ (there is a repetition in $\mathcal{X}$) **or** (there is a repetition in $\mathcal{Y}$)

Figure 10: Definition of Game 3C, which is adversarially indistinguishable from Game 3B.

answers. Since our adversary is deterministic the probability that it sets *bad* to true is over the random values $Z^1, \ldots, Z^q$ returned to the adversary and the random values $\mathsf{N}_1, \ldots, \mathsf{N}_p$ selected as the game runs. We now make the stronger claim that for *any* sequence of responses $Z^1, \ldots, Z^q$ to the adversary's queries, the probability that *bad* gets set to true is small (the probability now being over only the $\mathsf{N}_1, \ldots, \mathsf{N}_p$ values). Once we have fixed $Z^1, \ldots, Z^q$ the adversary's own queries are all fixed, as is whether each query is a left oracle query or a right oracle query. What we aim to show, then, is that for any sequence of valid queries and responses $(b^1, N^1, i_1^1, \ldots, i_k^1, M^1, C^1, Z^1, lr^1), \ldots, (b^q, N^q, i_1^q, \ldots, i_k^q, M^q, C^q, Z^q, lr^q)$ the probability that *bad* will get set to true is small. Here lr^r is an indication if the r-th query was a left oracle query (to encipher, coded as 0) or a right oracle query (to decipher, coded as 1). Thus we fix constants $\mathsf{b}^1, \mathsf{N}^1, \mathsf{i}_1^1, \ldots, \mathsf{i}_k^1, \mathsf{M}^1, \mathsf{C}^1, \mathsf{Z}^1, \mathsf{lr}^1, \ldots, \mathsf{b}^q, \mathsf{N}^q, \mathsf{i}_1^q, \ldots, \mathsf{i}_k^q, \mathsf{M}^q, \mathsf{C}^q, \mathsf{Z}^q, \mathsf{lr}^q)$ and look at the probability that *bad* gets set to true for this vector of constants. The vector of constants must be *valid*, in the sense that the assumptions that we have made about adversarial behaviors are reflected in the constants: no repeated queries; no deciphering when the resulting plaintext is known because of a prior enciphering; and no enciphering when the resulting ciphertext is known because of a prior deciphering. In addition, we insist that the adversary select constants $\mathsf{Z}^1, \ldots, \mathsf{Z}^q$ that are all distinct and that each Z^s differs from M^r and C^r for all $r < s$. This assumption entails a possible decrease in the probability that *bad* gets set to true by at most $1.5\, q^2/2^n$. Call a collection of constants as above *valid*. Now fix a vector of valid

Let $\mathsf{N}_1, \ldots, \mathsf{N}_p$ be the distinct strings from $\mathsf{N}^1, \ldots, \mathsf{N}^q$
for $i \in [1\,..\,p]$ **do** $\mathsf{N}_{\mathsf{N}_i} \xleftarrow{\$} \{0,1\}^n$

for $r \in [1\,..\,q]$ **do**

  if $\mathsf{lr}^r = 0$ **then**
    $X^r \leftarrow \mathsf{M}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}$
    $Y^r \leftarrow \mathsf{Z}^r \oplus \mathsf{b}^r \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}$

  if $\mathsf{lr}^r = 1$ **then**
    $X^r \leftarrow \mathsf{Z}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}$
    $Y^r \leftarrow \mathsf{C}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}$

$\mathcal{X} \leftarrow [\mathsf{N}_1, \ldots, \mathsf{N}_p,\ X^1, \ldots, X^q]$
$\mathcal{Y} \leftarrow [\mathsf{N}_{\mathsf{N}_1}, \ldots, \mathsf{N}_{\mathsf{N}_p},\ Y^1, \ldots, Y^q]$

bad $\leftarrow$ (there is a repetition in $\mathcal{X}$) **or** (there is a repetition in $\mathcal{Y}$)

Figure 11: Definition of Game 3D. All interaction and adaptivity has been eliminated, effectively replaced by universal quantification over the constants (in sans serif font, apart from $p, q, \alpha_1, \ldots, \alpha_k$).

constants that maximizes the probability that *bad* will get set to true in Game 3C and regard those constants as fixed. The resulting game, which we call Game 3D, is shown in Figure 11. We have then that

$$\Pr[A^{\text{Game 3C}} \Rightarrow 1] \quad \leq \quad \Pr[\text{Game 3D sets } bad\,] + \frac{1.5q^2}{2^n} \tag{8}$$

and our job is now to bound $\Pr[\text{Game 3D sets } bad\,]$.

We proceed to analyze the probability that *bad* gets set in Game 3D Let us first evaluate the probability of a collision within $\mathcal{X}$,

$$\mathcal{X} = [\mathsf{N}_1, \ldots, \mathsf{N}_p,\ X^1, \ldots, X^q]\ .$$

There can be no collision among the N_r-values because they are, by definition, all distinct. What about collision between an N_s and an X^r? If $\mathsf{lr}^r = 0$ then we are considering

$$\Pr\left[\mathsf{N}_s = \mathsf{M}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}\right]$$

which clearly occurs with probability 2^{-n}. If $\mathsf{lr}^r = 1$ then we are considering

$$\Pr\left[\mathsf{N}_s = \mathsf{Z}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}\right]$$

which likewise occurs with probability 2^{-n}. What about collision between an X^r and an X^s, where $r < s$? Then depending on lr^r and lr^s we are considering one of:

$$\Pr\left[\mathsf{M}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{M}^s \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$
$$\Pr\left[\mathsf{M}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{Z}^s \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$
$$\Pr\left[\mathsf{Z}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{M}^s \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$
$$\Pr\left[\mathsf{Z}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{Z}^s \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$

Clearly these are 2^{-n} if $\mathsf{N}^r \neq \mathsf{N}^s$. If $\mathsf{N}^r = \mathsf{N}^s$, however, we have to look at:

$$\begin{aligned}
&\Pr\left[\mathsf{M}^r \oplus \mathsf{M}^s = \left(\alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r} \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\right) \mathsf{N}_{\mathsf{N}^r}\right] \\
&\Pr\left[\mathsf{M}^r \oplus \mathsf{Z}^s = \left(\alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r} \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\right) \mathsf{N}_{\mathsf{N}^r}\right] \\
&\Pr\left[\mathsf{Z}^r \oplus \mathsf{M}^s = \left(\alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r} \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\right) \mathsf{N}_{\mathsf{N}^r}\right] \\
&\Pr\left[\mathsf{Z}^r \oplus \mathsf{Z}^s = \left(\alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r} \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\right) \mathsf{N}_{\mathsf{N}^r}\right]
\end{aligned}$$

For all of these, if $(\mathsf{i}_1^r, \ldots, \mathsf{i}_k^r) \neq (\mathsf{i}_1^s, \ldots, \mathsf{i}_k^s)$ then, by the uniqueness of representation for $\alpha_1, \ldots, \alpha_k$ over $\mathbb{I}_1 \times \cdots \times \mathbb{I}_k$, the value multiplying $\mathsf{N}_{\mathsf{N}^r}$ is nonzero and the indicated probability is 2^{-n}. We have left to consider the case of $(\mathsf{N}^r, \mathsf{i}_1^r, \ldots, \mathsf{i}_k^r) = (\mathsf{N}^s, \mathsf{i}_1^s, \ldots, \mathsf{i}_k^r)$, whence we are looking at

$$\begin{aligned}
&\Pr[\mathsf{M}^r = \mathsf{M}^s] \\
&\Pr[\mathsf{M}^r = \mathsf{Z}^s] \\
&\Pr[\mathsf{Z}^r = \mathsf{M}^s] \\
&\Pr[\mathsf{Z}^r = \mathsf{Z}^s]
\end{aligned}$$

Enciphering queries may not be repeated, so the first probability is zero. We have insisted in our choice of valid constants that $M^r \neq Z^s$, so the second probability is zero. An adversary that makes a deciphering query with a given tweak is not allowed to subsequently make an enciphering query of the resulting answer with the same tweak, so the third probability is zero. And deciphering queries may not be repeated, so the forth probability is zero. We may conclude that the probability that there is a collision in $\mathcal{X}$ is at most $2q^2/2^n$.

We next bound the probability of a collision in the multiset

$$\mathcal{Y} = [\mathsf{N}_{\mathsf{N}_1}, \ldots, \mathsf{N}_{\mathsf{N}_p},\ Y^1, \ldots, Y^q]\ .$$

For $r < s$ the probability of a collision between an $\mathsf{N}_{\mathsf{N}_r}$ and an $\mathsf{N}_{\mathsf{N}_s}$ is 2^{-n} since these are random and independent n-bit strings. What about a collision between an $\mathsf{N}_{\mathsf{N}_s}$ and a Y^r? If $\mathsf{Ir}^r = 0$ we are considering

$$\Pr\left[\mathsf{N}_{\mathsf{N}_s} = \mathsf{Z}^r \oplus \mathsf{b}^r \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}\right]\ .$$

If $\mathsf{N}_s \neq \mathsf{N}^r$ then this probability is 2^{-n}. If $\mathsf{N}_s = \mathsf{N}^r$ and $\mathsf{b}^r = 0$ then the probability is 2^{-n}. If $\mathsf{N}_s = \mathsf{N}^r$ and $\mathsf{b}^r = 1$ then the probability we are considering is

$$\Pr\left[\mathsf{Z}^r = \left(1 \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\right) \mathsf{N}_{\mathsf{N}^r}\right]\ .$$

By the unique-representation property and the prohibition of the vector of indices $(\mathsf{i}_1^r, \ldots, \mathsf{i}_k^r) = (0, \ldots, 0)$ the parenthesized quantity is not zero and the indicated probability is 2^{-n}. Continuing, a collision between an $\mathsf{N}_{\mathsf{N}_s}$ and a Y^r can also occur when $\mathsf{Ir}^r = 1$, in which case we are looking at

$$\Pr\left[\mathsf{N}_{\mathsf{N}_s} = \mathsf{C}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r}\right]\ .$$

If $\mathsf{N}_s \neq \mathsf{N}^r$ then the above is 2^{-n}. Otherwise we are looking at

$$\Pr\left[\mathsf{C}^r = \left(1 \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\right) \mathsf{N}_{\mathsf{N}^r}\right]$$

and the parenthesized quantity is nonzero for the same reason as before, making the probability 2^{-n}. Finally, we must look at the probability that $Y^r = Y_s$. Then depending on the values of lr^r and lr^s we are considering one of:

$$\Pr\left[\mathsf{Z}^r \oplus \mathsf{b}^r \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{Z}^s \oplus \mathsf{b}^s \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$
$$\Pr\left[\mathsf{Z}^r \oplus \mathsf{b}^r \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{C}^s \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$
$$\Pr\left[\mathsf{C}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{Z}^s \oplus \mathsf{b}^s \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$
$$\Pr\left[\mathsf{C}^r \oplus \alpha_1^{\mathsf{i}_1^r} \cdots \alpha_k^{\mathsf{i}_k^r}\, \mathsf{N}_{\mathsf{N}^r} = \mathsf{C}^s \oplus \alpha_1^{\mathsf{i}_1^s} \cdots \alpha_k^{\mathsf{i}_k^s}\, \mathsf{N}_{\mathsf{N}^s}\right]$$

The first of these is 0 if $\mathsf{b}^r = \mathsf{b}^s = 0$ because we have arranged that Z^i-values are all distinct. It is 2^{-n} if one of b^r and b^s is zero and the other is 1. Otherwise, $\mathsf{b}^r = \mathsf{b}^s = 1$. The probability above is clearly 2^{-n} if $\mathsf{N}^r \neq \mathbb{N}^s$, while if is the same value when $\mathsf{N}^r = \mathsf{N}^s$ by the unique-representation property. The remaining three probabilities are all at most 2^{-n} by reasoning exactly analogous to that which has been given. We conclude that the probability that there is a collision in $\mathcal{Y}$ is at most $2q^2/2^n$. Summing up, we have that

$$\Pr[\text{Game 3D sets } bad] \quad \leq \quad \frac{4q^2}{2^n} \tag{9}$$

and, combining everything, the advantage of our original adversary A is at most $\mathbf{Adv}_E^{\widetilde{\mathrm{prp}}}(t' + 2q) + (2 + 2 + 1.5 + 4)q^2/2^n$, establishing our result.

D Proof of Theorem 12 — Security of OCB1

Consulting Figure 2 may be helpful. In particular, look at the figure on the top and understand that each π_i^N and $\bar{\pi}_i^N$ is a random permutation on n bits. All of these permutations are independent. In effect, the key is the infinite collection of random permutations π_i^N, $\bar{\pi}_i^N$ for $i \in \mathbb{N}$ and $N \in \{0,1\}^n$.

The privacy statement is immediate. During the adversary's attack it asks a sequence of queries $(N^1, M^1), \ldots, (N^q, M^q)$ where the N^i-values are distinct. Since the N^i-values are distinct each π_i^N and $\bar{\pi}_i^N$ that gets used gets used exactly once. We are thus applying a number of independent random permutations each to a *single* point. The image of a single point under a random permutation is uniform, so the output is perfectly uniform. That is all that is needed to ensure that an adversary has no advantage to distinguish the output from random bits.

The authenticity statement is more involved, but still straightforward. Before we launch into it, consider the following simple game. Suppose that you know that an n-bit string X is *not* some particular value X_0. All of the $2^n - 1$ other values are equally likely. Then your chance of correctly predicting the τ-bit prefix of X is at most $2^{n-\tau}/(2^n - 1)$. That's because the best strategy is to guess any τ-bit string other than the τ-bit prefix of X_0. The probability of being right under this strategy is $2^{n-\tau}/(2^n - 1)$. We will use this fact in the sequel.

Now suppose that the adversary asks a sequence of queries $(N^1, M^1), \ldots, (N^q, M^q)$ and then makes its forgery attempt $(N, \mathcal{C})$. Let $M^i = M_1^i \cdots M_{m_i}^i$ be the queries and let $C^i = C_1^1 \cdots C_{m_i}^i$ be the responses. Let the tags produced as the adversary asks its queries be $Pad^1, \ldots, Pad^q$ and let the checksums produced be $\Sigma^1, \ldots, \Sigma^q$. Let $\mathcal{C} = C \parallel T$ where T is the first τ bits of *Tag* and $C = C_1 \ldots C_c$. Let the pad and checksum for the forgery attempt be *Pad* and Σ. We consider a number of cases.

(1) Suppose $N \notin \{N^1, \ldots, N^q\}$—the adversary tries to forge using a new nonce. Then the adversary needs to find the correct value of T but has seen no image of the random permutation $\bar{\pi}_c^N$. The chance that the adversary can guess the correct value for a random τ-bit string, given no information about it, is $2^{-\tau}$.

(2) Suppose $N = N^i$ but $c \neq m_i$. As above, the adversary needs to find the correct value of T but has seen no image of the random permutation $\bar{\pi}_c^N$. Thus the chance that the adversary can guess the correct value is $2^{-\tau}$.

(3) Suppose $N = N^i$ and $c = m_i$ but $|C| \neq |M^i|$. First, we may ignore the queries other than the i^{th} since their answers are unrelated to the adversary's task of producing a valid ciphertext $(N, \mathcal{C})$ with $N = N^i$. This time the adversary has seen one point of relevance to determining *Pad*, namely, the adversary may have seen some or all of Pad^i. No other values returned to the adversary are correlated to π_c^N. Among the possible values of *Pad*, all but Pad^i are equally likely, so there are $2^n - 1$ equally plausible values for *Pad*. Thus there are $2^n - 1$ equally plausible values for Σ even if we make public to the adversary all permutations other than π_c^N. Even then there are $2^n - 1$ equally likely inputs to $\bar{\pi}_c^N$ so there are at least $2^n - 1$ equally plausible outputs as *Tag*. By the analysis at the beginning of the authenticity analysis, the adversary's chance of correctly guessing T is thus at most $2^{n-\tau}/(2^n - 1)$.

(4) Suppose $N = N^i$ and $|C| = |M^i|$ and $C_j \neq C_j^i$ for some $j \in [1..c-1]$. We may again ignore the queries other than the i^{th} since their answers are unrelated to the adversary's task of producing a valid ciphertext $(N, \mathcal{C})$ with $N = N^i$. The adversary has seen the preimage under π_j^N of the point C_j^i, namely M_j^i, but the preimage under π_j^N of every point other than C_j^i is equally possible. In particular, the preimage of C_j under π_j^N is equally likely to be any of $2^n - 1$ different strings string. We imagine providing to the adversary all permutations other than the permutation π_j^N. Even then, the value of Σ can be any of $2^n - 1$ strings, each with equal probability. By the analysis at the beginning of the authenticity analysis, the adversary's chance of correctly guessing T is thus at most $2^{n-\tau}/(2^n - 1)$.

(5) Finally, suppose $N = N^i$ and $|C| = |M^i|$ and $C_j = C_j^i$ for all $j \in [1..c-1]$ but $C_c \neq C_c^i$. In this case the value Σ is known and differs from Σ^i by the non-zero value $C_c \oplus C_c^i$. The value *Tag* can be any of $2^n - 1$ values, each with equal likelihood. By the analysis at the beginning of the authenticity analysis, the adversary's chance of correctly guessing T is at most $2^{n-\tau}/(2^n - 1)$.

This completes the proof.

E Proof of Theorem 15 — Security of PMAC1

The proof is relatively standard and so we are brief in its exposition. We replace the random tweakable permutation $\pi \in \mathrm{Perm}(\mathcal{T}, n)$ by a random tweakable function $\rho \in \mathrm{Rand}(\mathcal{T} \times \{0,1\}^n, n)$ noting that, by the obvious extension to the standard PRP/PRF switching lemma,

$$\begin{aligned} \mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Perm}(\mathcal{T},n),\tau]}(\sigma) &\leq \mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Rand}(\mathcal{T}\times\{0,1\}^n,n),\tau]}(\sigma) + 0.5\,\sigma^2/2^n \\ &\leq \mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Rand}(\mathcal{T}\times\{0,1\}^n,n),n]}(\sigma) + 0.5\,\sigma^2/2^n \end{aligned}$$

Next we bound $\mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Rand}(\mathcal{T}\times\{0,1\}^n,n),n]}(\sigma)$. Let A be an adversary that tries to distinguish $f \in \mathrm{PMAC1}[\mathrm{Rand}(\mathcal{T} \times \{0,1\}^n, n), n]$ from $\rho \in \mathrm{Rand}(\{0,1\}^*, n)$ and assume that A's queries total at most σ blocks (the empty block counting as one block). Recognizing that we are in the three-key version of the Carter-Wegman paradigm, a conventional game-playing argument, given in [5], can be used to justify that

$$\mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Rand}(\mathcal{T}\times\{0,1\}^n,n),n]}(A) \quad \le \quad \max_{\substack{q,M^1,\ldots,M^q \\ |M^1|_n+\cdots+|M^q|_n\le\sigma}} \left\{ \sum_{1\le i<j\le q} \Pr[M^i \text{ and } M^j \text{ collide}] \right\}$$

where M and M' are said to collide if they are distinct and either:

- *Collision between two full-final-block messages.* $M = M_1 \cdots M_m$ and $M' = M'_1 \cdots M'_{m'}$ are nonempty and have length is divisible by n and it happens that $X = X'$ where $X = Y_1 \oplus \cdots \oplus Y_{m-1} \oplus M_m$ is the input to final random function $\bar{\rho}_m$ when M is processed and $X' = Y'_1 \oplus \cdots \oplus Y'_{m'-1} \oplus M_{m'}$ is the input to final random function $\bar{\rho}_{m'}$ when M' is processed; or
- *Collision between two partial-final-block messages.* $M = M_1 \cdots M_m$ and $M' = M'_1 \cdots M'_{m'}$ are either empty or have length not divisible by n and it happens that $X = X'$ where $X = Y_1 \oplus \cdots \oplus Y_{m-1} \oplus M_m 10^*$ is the input to final random function $\bar{\bar{\rho}}_m$ when M is processed and $X' = Y'_1 \oplus \cdots \oplus Y'_{m'-1} \oplus M_{m'} 10^*$ is the input to final random function $\bar{\bar{\rho}}_{m'}$ when M' is processed.

The probability above is over the choice of $\rho \in \mathrm{Rand}(\mathcal{T} \times \{0,1\}^n, n)$. A case analysis is then used to show that if $|M|_n = m$ and $|M'|_n = m'$ and $M \ne M'$ then the probability that they collide is at most $1/2^n$. Suppose first that M and M' are distinct, nonempty, and have lengths divisible by n. If $m \ne m'$ then the probability that M and M' collide is clearly $1/2^n$. If $m = m'$ and for some $i < m$ we have that $M_i \ne M'_i$ then the probability that M and M' collide is $1/2^n$. If $m = m'$ and $M_i = M'_i$ for all $i < m$ then necessarily $M_m \ne M'_m$ and the probability that M and M' collide is 0. If one supposes instead that M and M' are distinct and either empty or have lengths divisible by n then the argument is analogous. We conclude that in all cases the probability that M and M' collide is at most $1/2^n$ and so

$$\begin{aligned}
\mathbf{Adv}^{\widetilde{\mathrm{prf}}}_{\mathrm{PMAC1}[\mathrm{Rand}(\mathcal{T}\times\{0,1\}^n,n),n]}(A) \quad &\le \quad \max_{\substack{q,M^1,\ldots,M^q \\ |M^1|_n+\cdots+|M^q|_n\le\sigma}} \left\{ \sum_{1\le i<j\le q} \frac{1}{2^n} \right\} \\
&\le \quad 0.5\, q^2/2^n \\
&\le \quad 0.5\, \sigma^2/2^n
\end{aligned}$$

Summing with the prior bound of $0.5\, \sigma^2/2^n$ completes the proof.