IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2023, No. 2, pp. 69–93. DOI:10.46586/tosc.v2023.i2.69-93

EliMAC: Speeding Up LightMAC by around 20%

Christoph Dobraunig[1], Bart Mennink[2] and Samuel Neves[3]

[1] Lamarr Security Research, Graz, Austria
christoph@dobraunig.com
[2] Digital Security Group, Radboud University, Nijmegen, The Netherlands
b.mennink@cs.ru.nl
[3] CISUC, Department of Informatics Engineering, University of Coimbra, Coimbra, Portugal
sneves@dei.uc.pt

Abstract. Universal hash functions play a prominent role in the design of message authentication codes and the like. Whereas it is known how to build highly efficient sequential universal hash functions, parallel non-algebraic universal hash function designs are always built on top of a PRP. In such case, one employs a relatively strong primitive to obtain a function with a relatively weak security model. In this work, we present EliHash, a construction of a parallel universal hash function from non-compressing universal hash functions, and we back it up with supporting security analysis. We use this construction to design EliMAC, a message authentication code similar to LightMAC. We consider a heuristic instantiation of EliMAC with round-reduced AES, and argue that this instantiation of EliMAC is much more efficient than LightMAC, it is around 21% faster, and additionally allows for precomputation of the keys, albeit with a stronger assumption on the AES primitive than in LightMAC. These observations are backed up with an implementation of our scheme.

Keywords: universal hashing · MAC · EliHash · EliMAC · length independence

1 Introduction

Message authentication codes (MACs) are symmetric cryptographic functions that assure authenticity of data. Given the importance of data authenticity in our daily communication, MACs have undergone extensive research in the last decades. As they are employed a lot, they need to be lightning fast, and as data is often of variable length, much research has been performed on how to process variable-length data as fast as possible.

A good example of this is the hash-then-PRF construction. Let n be a natural number, F_K be a pseudorandom function from n to n bits, and H_L be a universal hash function that compresses arbitrarily long data into short fingerprints of n bits. The hash-then-PRF construction authenticates an arbitrarily long message M as

$$T = F_K(H_L(M)) .$$

Here, the function that processes the bulk of the data, H_L, need not be cryptographically strong but simply needs to satisfy lighter probabilistic properties. The cryptographically strong function, F_K, is only applied to a small input block. This way, the efficiency of the function is not significantly affected if longer data is input. A comparable phenomenon appears in nonce-based MAC functions like Wegman-Carter [WC81, Bra82] and Wegman-Carter-Shoup [Sho96, Ber05].

Received: 2023-03-01 Accepted: 2023-05-01 Published: 2023-06-16

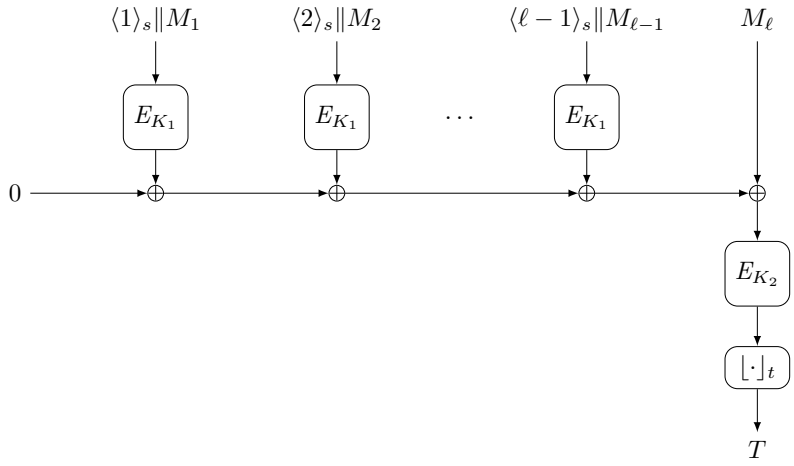

Figure 1: LightMAC message authentication [LPTY16]. Here, $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. Not depicted is the injective padding of an arbitrarily length message M into $\ell \leq 2^s$ $(n-s)$-bit blocks $M_1, \ldots, M_\ell$.

1.1 Parallel Universal Hashing

A popular approach to variable-length universal hashing is chaining or cascading well-analyzed cryptographic operations together with data absorption. A notable example is the universal hashing part of CBC-MAC [BKR94]. Even with round-reduced versions of cryptographic operations, good heuristic designs are known, e.g., Pelican 2.0 [DR05b]. For parallel variable-length universal hashing, the situation is less explored.

The typical approach in constructing a non-algebraic parallel universal hash function is by building it from a pseudorandom permutation (PRP) or a pseudorandom function (PRF). A well-known example of this is PMAC [BR02]. PMAC internally uses an n-bit block cipher E, parses its input to n-bit blocks and masks these with $2^i \cdot E_K(0)$, where i is the block counter. The masked blocks apart from the last one are then fed to the block cipher in parallel, the outcomes of which are added to the last block, and the resulting checksum is transformed with a final call to the block cipher. PMAC is a direct variant of the, older, protected counter sum of Bernstein [Ber99]. This construction takes a function f with input size $n+s$ bits and output size n bits, and parses its inputs to n-bit blocks. The blocks are evaluated as $f(\langle i\rangle_s, M_i)$, the outcomes are added and the final checksum is transformed with a final call to $f(\langle 0\rangle_s, \cdot)$. A final scheme worth mentioning in the context of this paper is LightMAC of Luykx et al. [LPTY16]. LightMAC is a variant of protected counter sum that parses the messages into $(n-s)$-bit blocks and *concatenates* these blocks with an s-bit counter. This way, the authors managed to prove a security bound for LightMAC that is independent of the message length [LPTY16], a property not met by the earlier MAC functions [LPSY16]. LightMAC is depicted in Figure 1. Recently, Chattopadhyay et al. [CJN21] proved security of two single-key variants of LightMAC, namely 1k-LightMAC and LightMAC-ds, and Shen et al. [SWG21] described a double-key and a single-key variant of LightMAC that use domain separation to distinguish between integral and fractional data, namely LedMAC1 and LedMAC2.

Ultimately, despite their efficiency, parallelism, strong security, and conceptual elegance, all these MAC functions use a parallel variable-length universal hash function built on top of a PRP. This seems to be an unnecessarily strong assumption: intuitively, one should be able to get away with fixed-length universal hash functions, or at least with round-reduced versions of these PRPs. We stress that this is a very practical matter. The typical choice for the PRP is the 10-round AES block cipher [DR02] and this choice is for instance also

adopted in the standardization of LightMAC in ISO/IEC 29192-6:2019 [ISO19]. However, it is known that 4 rounds of AES with independent round keys already yield an $1.881 \cdot 2^{-114}$-almost XOR universal hash function [KS07]. In an Utopian setting, this might give a requirement of $4(\ell - 1) + 10$ rounds of AES for the authentication of an ℓ-block message, noting that the finalization still must be performed with a full-round AES. Unfortunately, it is not that easy: simply replacing the block cipher calls before the checksum by universal hashes is not necessarily secure, and for the particular case of 4-round AES, the resulting scheme can be easily broken: if the message is encoded in a certain way, it is possible to take 2^{32} evaluations of 4-round AES that sum to 0 [DR02, §10.2.5].

A truly parallelizable MAC function based on a universal hash function like 4-round AES is MARVIN of Simplício Jr. et al. [SBB+09]. The mode is very elegant: it is a fully parallelizable evaluation of a "square-complete transform", a function that is required to satisfy certain XOR-universality property. If instantiated with 4-round AES, this function seems to solve the problem. However, the formal security analysis of MARVIN, given by Simplício Jr. and Barreto [SB12], misses a crucial aspect in the reasoning: in a nutshell, it relies on XOR-universality to also reason about 4-sums or even larger sums. This makes the proof invalid. We elaborate on this oversight in Appendix A, and even stronger: we give an attack on MARVIN for a specific instantiation of the square-complete transform that works fine for 2-sums but not for larger sums. The fact that MARVIN achieves a length-independent bound despite using a PMAC-style masking (as opposed to a LightMAC style concatenation) should have given away already that the bound is flawed for variable-length data. We refer to Luykx et al. [LPSY16] for a discussion of the influence of the length in the security bound of PMAC.

Overall, we lack a provable parallelizable variable-length universal hash function built from fixed-length universal hash functions. A construction that comes close is APA-then-MTH of Minematsu and Tsunoo [MT06]. This construction takes a universal hash function with the same domain and range, doubles that range with a so-called Add-Permute-Add construction, and then evaluates this function for independent keys in a tree fashion. One can then, indeed, take 4-round AES for this construction, but in order to obtain independent keys, one must generate them using full 10-round AES, for example. This gives significant overhead to the universal hash function, and makes it worse than the universal hash function currently used in LightMAC. In addition, this proposal is ultimately tree-based and not parallelizable. A similar remark applies to MACH of Jakimoski and Subbalakshmi [JS07].

1.2 EliHash: Parallel Universal Hashing from Universal Hash Functions

The first contribution of this work is a fully parallelizable variable-length universal hash function from non-compressing keyed hash functions. The hash function proposal is simple and more efficient than the earlier universal hash functions. It is parametrized by the block size n, and the maximum number of blocks μ, and it is built from two universal hash functions $H : \{0,1\}^{k'} \times [1, \ldots, \mu] \to \{0,1\}^k$ and $I : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$. The main ingredient is a nested universal hash function evaluation $I(H(K,i), M_i)$ that is put in a checksum mode:

$$\mathrm{EliHash}(K, (M_1, \ldots, M_\ell)) = \bigoplus_{i=1}^{\ell} I(H(K,i), M_i)\,,$$

where $\ell \leq \mu$. For data not of size a multiple of n bits, any injective padding on the message completes the picture. The function is described in more detail in Section 3. Note that

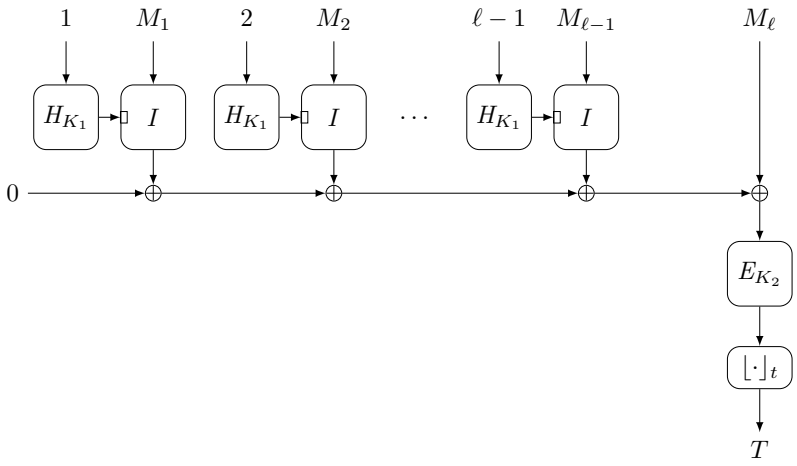

Figure 2: EliMAC message authentication. Here, $H : \{0,1\}^{k'} \times [1,\ldots,\mu] \to \{0,1\}^k$ and $I : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ are two universal hash functions and $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. Not depicted is the injective padding of an arbitrarily length message M into $\ell \leq \mu$ n-bit blocks $M_1,\ldots,M_\ell$.

EliHash can be seen as a generalization of the multilinear hash construction [GMS74],

$$MH((K_1,\ldots,K_\ell),(M_1,\ldots,M_\ell)) = \bigoplus_{i=1}^{\ell} K_i \cdot M_i\,,$$

with an arbitrary universal hash function I instead of finite field multiplication, and with a configurable distribution on the key.

In Section 3, we also derive a bound on the XOR-universality of EliHash, under the assumption that H is μ-independent and I is XOR-universal. The former condition, μ-independence, is the strongest assumption of the two and is comparable to the maximum interpolation probability as employed, e.g., by the Wegman-Carter-Shoup proof of Bernstein [Ber05]. It appears like the strongest condition of the two. However, the condition works in our use case, as H has a very small domain $[1,\ldots,\mu]$. See also Section 1.4.

1.3 EliMAC: Improving LightMAC Using Parallelizable Universal Hashing

We use EliHash to define the EliMAC (Extremely Light MAC) message authentication code in Section 4. EliMAC is like LightMAC, but with the universal hashing part replaced by EliHash. In other words, to authenticate a padded message $(M_1,\ldots,M_\ell)$, with $\ell \leq \mu$, one first evaluates EliHash$(K,\cdot)$ on these message blocks bar the last one that is added in plain. This yields an n-bit value which is subsequently fed to a block cipher E with an independent key. EliMAC is depicted in Figure 2. We prove PRF security and unforgeability of EliMAC in Section 4.1 and Section 4.2, respectively. As the scheme is, basically, a universal hash followed by a (truncated) permutation, the analysis is simple and easy to verify. The PRF security proof is inspired by the analysis of LightMAC. For MAC security, we obtain a tighter analysis. This analysis centers around a generic reduction from truncated to untruncated hash-then-encrypt constructions, that is broadly applicable. Most importantly, it immediately allows for a tighter MAC security bound of LightMAC. The bounds are tight, as discussed in Section 4.3.

1.4 Instantiation

The, disputably, most logical instantiation of EliHash and of EliMAC is to take 4-round AES-128 as universal hash function H and I, and full 10-round AES-128 as finalization function E in the message authentication. (In the remainder, we simply write "AES" and drop the "-128" for brevity.) For 4-round AES (with independent round keys), we know a bound on its XOR-universality and thus on its 2-independence [KS07], however, for larger values of μ, the situation gets increasingly worse as the μ-independence of H is not always easy to estimate. Clearly, if we instantiate H by full 10-round AES, we can rely on the assumption that AES is a secure PRP, which would subsequently imply μ-independence for sufficiently large μ. However, this approach would give an inefficient construction while at the same time general PRP security is for arbitrary inputs and is a too strong condition for our purpose. For EliMAC, a reduced-round variant of AES might do the job: we can use the fact that μ-independence is only required to hold for a restricted number and choice of inputs that the adversary can feed to the function, and by proper encoding of $[1, \ldots, \mu] \mapsto \{0,1\}^n$, it appears [DFJ13] that 7 rounds of AES suffice if $\mu \leq 2^{32}$.

This way, in summary, we obtain a MAC function where H is instantiated with 7-round AES, I with 4-round AES, and E with 10-round AES. The choice for the number of rounds in this instantiation is based on the *best* attacks against round-reduced versions of AES to date. (Details about this instantiation are given in Section 5.) A quick computation shows that, this way, EliMAC makes *slightly* more AES round evaluations as LightMAC instantiated with 10-round AES. However, as a bonus, it can process more message bits. In detail, if we stick to $\mu = 2^{32}$, LightMAC operates with $s = 32$ and thus processes 96 bits of message per 10 rounds of AES (an exception applies to the last block that may be n bits), whereas EliMAC processes 128 bits of message per 11 rounds of AES. A more detailed comparison is given in Table 1. We can observe that EliMAC is, at a similar number of AES rounds, more efficient than LightMAC, with a significant improvement of around 21%. In addition, in EliMAC, the subkeys can be precomputed and stored in memory. This reduces the number of AES rounds per block even to 4 (except for the last block). Of course, this only works if sufficient storage for the subkeys is available.

The efficiency gain in EliMAC also comes at a few differences in security. First off, focusing on the instantiation with AES, LightMAC generically achieves around 64-bit security, whereas EliMAC achieves around 56-bit security. Additionally, the analyses differ in the assumption on the underlying primitive. To be precise, LightMAC is proven secure under the assumption that 10-round AES is PRP-secure against an attacker querying q blocks, i.e., that $\mathbf{Adv}^{\mathrm{prp}}_{\mathrm{AES}_{10}}(q, \tau)$ small, where τ denotes the adversarial time complexity. EliMAC, in addition, requires that also $\mathbf{Adv}^{\mathrm{prp}}_{\mathrm{AES}_7}(\mu, \tau)$ is small, and that 4-round AES is XOR-universal. We believe that these assumptions are reasonable [KS07, DFJ13].

We implemented EliMAC and compared it with LightMAC [LPTY16], PMAC2 [CCJN21], and ZMAC [IMPS17]. A discussion of this comparison is given in Section 6. The implementation results, outlined in Table 2, confirm above-made efficiency claims. In particular, EliMAC achieves higher efficiency than LightMAC and PMAC2 on different microarchitectures, and in particular, EliMAC turns out to stay closest to the theoretical asymptotic performance, mostly due to its implementation simplicity. A similar comparison applies to the recently introduced variants of LightMAC by Chattopadhyay et al. [CJN21] and Shen et al. [SWG21].

2 Preliminaries

For non-empty finite sets $\mathcal{X}, \mathcal{Y}$, we denote by $\mathrm{func}(\mathcal{X}, \mathcal{Y})$ the set of all functions $\mathcal{X} \to \mathcal{Y}$. We let $\mathrm{func}(\mathcal{X}) = \mathrm{func}(\mathcal{X}, \mathcal{X})$ denote the set of all functions on $\mathcal{X}$, and write $\mathrm{perm}(\mathcal{X})$ for the set of all permutations on $\mathcal{X}$. We denote by $X \xleftarrow{\$} \mathcal{X}$ the uniform random sampling of an

Table 1: Comparison of EliMAC with LightMAC, in case the PRPs are instantiated with 10-round AES and the universal hash evaluations with padded 7-round and 4-round AES, where *padded* refers to the fact that the inputs are integer encodings padded with sufficiently many zeros. Comparison is performed for authentication of ℓ message blocks, where $\ell \leq \mu = 2^{32}$. Furthermore, $n = 128$ denotes the state size, and $s = \log_2 \mu = 32$ denotes the counter size.

scheme	bit length of ℓ-block message	# AES rounds for ℓ blocks			reference
		pre	online	total	
LightMAC	$96\ell + 95$	0	10ℓ	10ℓ	[LPTY16]
EliMAC	128ℓ	$7(\ell - 1)$	$4(\ell - 1) + 10$	$11\ell - 1$	Section 5

element X from $\mathcal{X}$. For $m, n \in \mathbb{N}$ with $m \leq n$, we denote by $\mathcal{X}^n$ the set of all $\mathcal{X}$-sequences of length n, by $\mathcal{X}^{[m \ldots n]}$ the set of all $\mathcal{X}$-sequences of length between m and n, by $\mathcal{X}^{<n}$ the set of all $\mathcal{X}$-sequences of length less than n, and by $\mathcal{X}^*$ the set of all $\mathcal{X}$-sequences of arbitrary length. For $m, n \in \mathbb{N}$ with $m \leq n$, we denote by $[n]_m = n(n-1)\cdots(n-m+1)$ the falling factorial.

We let $\mathrm{pad}_n : \{0,1\}^* \to (\{0,1\}^n)^*$ be any injective padding function that transforms arbitrarily long strings to strings of length a positive multiple of n. A minimal working example is the 10-padding, that appends its input with a 1 and a minimal number of 0s to reach a string of length a multiple of n. For $X \in \{1, \ldots, 2^n\}$ we write $\langle X \rangle_n$ as an encoding of X as an n-bit string.

Adversaries are algorithms that have the objective to win a certain security game or produce a certain output. For an adversary A, we denote by A^O that A has query access to a randomized oracle (or list of oracles) O.

2.1 Universal Hash Functions

For three non-empty finite sets $\mathcal{K}, \mathcal{X}, \mathcal{Y}$, consider a family of hash functions $H : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. Let $\mu \in \mathbb{N}$. Following Wegman and Carter [WC81], H is said to be δ-almost μ-wise independent, or more concisely δ-μ-independent, if for any distinct $X_1, \ldots, X_\mu \in \mathcal{X}$ and (not necessarily distinct) $Y_1, \ldots, Y_\mu \in \mathcal{Y}$,

$$\mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}} \left(\forall_{i=1}^{\mu}\, H_K(X_i) = Y_i\right) \leq \delta^\mu \,.$$

We remark that, in general, $\delta \geq |\mathcal{Y}|^{-1}$. In addition, as the probability is taken over the randomness of the key, δ is also lower bounded by a value depending on the key size. Indeed, for an arbitrary function H and fixed $X_1, \ldots, X_\mu \in \mathcal{X}$ and $Y_1, \ldots, Y_\mu \in \mathcal{Y}$, we necessarily have

$$\delta \geq \left(\frac{|\{K \in \mathcal{K} \mid \forall_{i=1}^{\mu}\, H_K(X_i) = Y_i\}|}{|\mathcal{K}|} \right)^{1/\mu} .$$

We can reach this bound in the random setting. For example, if H is the family of permutations on $\mathcal{X} = \mathcal{Y} = \{0,1\}^n$ and $\mathcal{K}$ is the set of permutation indices (i.e., $|\mathcal{K}| = 2^n!$), then H is δ-μ-independent with $\delta = ([2^n]_\mu)^{-1/\mu}$. Subsequently, if H_K is a PRP-secure function with $\mathcal{X} = \mathcal{Y} = \{0,1\}^n$, we can replace it by a random permutation at the cost of its PRP-advantage and subsequently rely on this bounding. We will use this observation in Section 5.

The hash function family H is said to be ε-XOR-universal if for any distinct $X, X' \in \mathcal{X}$ and $Y \in \mathcal{Y}$,

$$\mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}} \left(H_K(X) \oplus H_K(X') = Y\right) \leq \varepsilon \,.$$

2.2 Pseudorandom Permutations and Pseudorandom Functions

For two non-empty finite sets $\mathcal{K}, \mathcal{M}$, a block cipher $E : \mathcal{K} \times \mathcal{M} \to \mathcal{M}$ is a family of permutations on $\mathcal{M}$ indexed by keys from $\mathcal{K}$. We write $E_K(\cdot) = E(K, \cdot)$. Security of a block cipher is defined by its pseudorandom permutation (PRP) security: we consider an adversary A that has access to either E_K for $K \xleftarrow{\$} \mathcal{K}$ or $p \xleftarrow{\$} \mathrm{perm}(\mathcal{M})$, and its goal is to guess with which oracle it is communicating. Formally:

$$\mathbf{Adv}_E^{\mathrm{prp}}(A) = \mathbf{Pr}\left(A^{E_K} = 1\right) - \mathbf{Pr}\left(A^{p} = 1\right) , \tag{1}$$

where the probabilities are drawn over the random selection of $K \xleftarrow{\$} \mathcal{K}$, $p \xleftarrow{\$} \mathrm{perm}(\mathcal{M})$, and the adversarial choices of A (recall that A may be probabilistic). We denote by $\mathbf{Adv}_E^{\mathrm{prp}}(q, \tau)$ the supremal advantage over any adversary with query complexity q and time complexity τ.

For three non-empty sets $\mathcal{K}, \mathcal{M}, \mathcal{T}$, a pseudorandom function $F : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ is a family of functions from $\mathcal{M}$ to $\mathcal{T}$ indexed by keys from $\mathcal{K}$. We write $F_K(\cdot) = F(K, \cdot)$. Security of a pseudorandom function is defined by its pseudorandom function (PRF) security. Now, the adversary has access to either F_K for $K \xleftarrow{\$} \mathcal{K}$ or $f \xleftarrow{\$} \mathrm{func}(\mathcal{M}, \mathcal{T})$:

$$\mathbf{Adv}_F^{\mathrm{prf}}(A) = \mathbf{Pr}\left(A^{F_K} = 1\right) - \mathbf{Pr}\left(A^{f} = 1\right) , \tag{2}$$

where the probabilities are drawn over the random selection of $K \xleftarrow{\$} \mathcal{K}$, $f \xleftarrow{\$} \mathrm{func}(\mathcal{M}, \mathcal{T})$, and the adversarial choices of A. We denote by $\mathbf{Adv}_E^{\mathrm{prf}}(q, \sigma, \tau)$ the supremal advantage over any adversary with query complexity q, total block complexity σ, and time complexity τ. The total block complexity is scheme-dependent; it may be omitted if irrelevant or less trivial.

2.3 Message Authentication Codes

For three non-empty finite sets $\mathcal{K}, \mathcal{M}, \mathcal{T}$, a message authentication code $MAC : \mathcal{K} \times \mathcal{M} \to \mathcal{T}$ is a family of one-way functions from $\mathcal{M}$ to $\mathcal{T}$ indexed by keys from $\mathcal{K}$. One way to measure the security of message authentication codes is by its PRF security. Alternatively, one can consider unforgeability, that quantifies the success probability of an adversary to deliver a correct message-tag pair that was not generated by MAC_K. Formally:

$$\mathbf{Adv}_{MAC}^{\mathrm{mac}}(A) = \mathbf{Pr}\left(A^{MAC_K} \text{ forges}\right) , \tag{3}$$

where "forges" means that A outputs a tuple (M, T) that is not the result of an oracle query but that nevertheless satisfies $MAC_K(M) = T$. As before, the probability is drawn over the random selection of $K \xleftarrow{\$} \mathcal{K}$ and the adversarial choices of A. We denote by $\mathbf{Adv}_{MAC}^{\mathrm{mac}}(q, q_f, \tau)$ the supremal advantage over any adversary with query complexity q, forgery attempt complexity q_f, and time complexity τ.

2.4 Bound Falling Factorial

We will use the following elementary bound related to the falling factorial $[a]_b$. The result is comparable to a derivation of Bernstein [Ber05, Theorem 4.2], be it with a different proof approach.

Lemma 1. *Let $m, n \in \mathbb{N}$ with $m^2 \leq n$. Then,*

$$\frac{n^m}{[n]_m} \leq 2 . \tag{4}$$

Proof. Note that

$$\begin{aligned}\frac{n^m}{[n]_m} &= \prod_{i=0}^{m-1} \frac{n}{n-i} \\ &= \prod_{i=0}^{m-1} \left(1 + \frac{i}{n-i}\right) \\ &\leq \prod_{i=0}^{m-1} e^{\frac{i}{n-i}} \\ &= e^{\sum_{i=0}^{m-1} \frac{i}{n-i}} \\ &\leq e^{\frac{m(m-1)}{2(n-m)}} \\ &\leq e^{1/2}\,,\end{aligned}$$

where the last step holds under the condition that $m^2 \leq n$. The proof is completed by observing that $e^{1/2} \leq 2$. □

3 EliHash

Let $\mu \in \mathbb{N}$, and let $\mathcal{K}', \mathcal{K}, \mathcal{X}, \mathcal{Y}$ be four non-empty finite sets. Consider two – not necessarily distinct – families of hash functions $H : \mathcal{K}' \times [1, \ldots, \mu] \to \mathcal{K}$ and $I : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$. Define EliHash : $\mathcal{K}' \times \mathcal{X}^{[1\cdots\mu]} \to \mathcal{Y}$ as

$$\text{EliHash}(K, (X_1, \ldots, X_\mu)) = \bigoplus_{i=1}^{\mu} I(H(K, i), X_i)\,. \tag{5}$$

Our goal is to prove that EliHash is XOR-universal as long as H and I satisfy certain conditions. This way, it could be used as hash function within a hash-then-PRF construction, similar to LightMAC [LPTY16] or any other universal hash function based MAC function (such as [CS16, MN17, DDNY18, DNT19]). Unfortunately, it is not possible to easily prove XOR-universality of EliHash from XOR-universality of H and I, the main reason being that in EliHash typically more than two evaluations of I are added. Thus, we will prove XOR-universality of EliHash based on slightly stronger properties of H and I. This will be done in Section 3.1.

Remark 1. It is possible to extend the domain of EliHash to $\mathcal{K}' \times \mathcal{X}^{[1\cdots\mu]} \times \mathcal{X}^{[1\cdots\mu]}$ and to let it be a checksum of $I(H(K, X_i), X'_i)$, i.e., with a simple encoding of i replaced with an arbitrary value from $\mathcal{X}$. However, this change would make the proof in Section 3.1 significantly more complex, and the instantiation of the universal hash functions H and I with round-reduced AES-128 in Section 5 meaningless.

3.1 XOR-Universality of EliHash

We prove XOR-universality of EliHash. This is the main result that will be used in the EliMAC construction of Section 4.

Proposition 1. *Let $\mu \in \mathbb{N}$, and let $\mathcal{K}', \mathcal{K}, \mathcal{X}, \mathcal{Y}$ be four non-empty finite sets. Let $H : \mathcal{K}' \times [1, \ldots, \mu] \to \mathcal{K}$ be a δ-μ-independent hash function family and $I : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ an ε-XOR-universal hash function family. Then,* EliHash : $\mathcal{K}' \times \mathcal{X}^{[1\cdots\mu]} \to \mathcal{Y}$ *of (5) is ε'-XOR-universal for*

$$\varepsilon' = (|\mathcal{K}|\delta)^\mu \varepsilon\,.$$

Proof. Consider any $\ell, \ell' \leq \mu$, any distinct $(X_1, \ldots, X_\ell) \in \mathcal{X}^\ell$ and $(X'_1, \ldots, X'_{\ell'}) \in \mathcal{X}^{\ell'}$, and any $Y \in \mathcal{Y}$. Our goal is to bound

$$\mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left(\underbrace{\bigoplus_{i=1}^{\ell} I(H(K,i), X_i) \oplus \bigoplus_{i=1}^{\ell'} I(H(K,i), X'_i) = Y}_{\Theta} \right) . \tag{6}$$

Without loss of generality, $\ell \leq \ell'$, and for all $i \leq \ell$ we have $X_i \neq X'_i$. The reason that the latter argument is fair, is that one can discard the indices for which $X_i = X'_i$ from the checksums and apply below reasoning for two subsets of indices of $\{1, \ldots, \ell\}$ and $\{1, \ldots, \ell'\}$. Then, below reasoning holds with a smaller value of ℓ'.

Define

$$\mathcal{L} := \left\{ (L_1, \ldots, L_{\ell'}) \;\middle|\; \bigoplus_{i=1}^{\ell} I(L_i, X_i) \oplus \bigoplus_{i=1}^{\ell'} I(L_i, X'_i) = Y \right\} .$$

This set describes all possible subkeys $(L_1, \ldots, L_{\ell'})$ for which Θ would hold. Using this, we can condition the event in the probability of (6) depending on whether the key $K \xleftarrow{\$} \mathcal{K}'$ results in ℓ' subkeys that are in $\mathcal{L}$ or not:

$$\begin{aligned}
(6) &= \mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left(\Theta \;\middle|\; (H(K,1), \ldots, H(K,\ell')) \in \mathcal{L} \right) \cdot \mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left((H(K,1), \ldots, H(K,\ell')) \in \mathcal{L} \right) \\
&\quad + \mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left(\Theta \;\middle|\; (H(K,1), \ldots, H(K,\ell')) \notin \mathcal{L} \right) \cdot \mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left((H(K,1), \ldots, H(K,\ell')) \notin \mathcal{L} \right) \\
&= 1 \cdot \mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left((H(K,1), \ldots, H(K,\ell')) \in \mathcal{L} \right) + 0 \\
&= \sum_{(L_1, \ldots, L_{\ell'}) \in \mathcal{L}} \mathbf{Pr}_{K \xleftarrow{\$} \mathcal{K}'} \left(\textstyle\bigwedge_{i=1}^{\ell'} H(K,i) = L_i \right) \\
&\leq |\mathcal{L}| \cdot \delta^{\ell'} .
\end{aligned} \tag{7}$$

What remains is to bound $|\mathcal{L}|$. Clearly, we have $|\mathcal{K}|$ possible choices for each of $L_2, \ldots, L_{\ell'}$, and thus:

$$|\mathcal{L}| \leq |\mathcal{K}|^{\ell'-1} \cdot \max_{\substack{X \neq X' \in \mathcal{X}, \\ Y \in \mathcal{Y}}} |\{L_1 \in \mathcal{K} \mid I(L_1, X) \oplus I(L_1, X') = Y\}| .$$

As I is ε-XOR-universal, the size of the right hand side set is at most $|\mathcal{K}| \cdot \varepsilon$. The proof is completed by maximizing over ℓ' and noticing that $\delta \geq |\mathcal{K}|^{-1}$. □

4 EliMAC

We use the universal hash function design EliHash of (5) to build the EliMAC hash function. EliMAC is parametrized by $\mu, k', k, n, t \in \mathbb{N}$, and we set $\mathcal{K}' = \{0,1\}^{k'}$, $\mathcal{K} = \{0,1\}^k$, and $\mathcal{X} = \mathcal{Y} = \{0,1\}^n$. It thus operates on top of two – not necessarily distinct – universal hash functions $H : \{0,1\}^{k'} \times [1, \ldots, \mu] \to \{0,1\}^k$ and $I : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$. In addition, it employs a block cipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$. EliMAC is specified in Algorithm 1, and it is depicted in Figure 2. Note that it processes a message M of length at most $2^{(\mu+1)n} - 1$ bits, i.e., which might get padded into $\mu + 1$ blocks.

Algorithm 1 EliMAC message authentication

Input: $K_1 \in \{0,1\}^{k'}$, $K_2 \in \{0,1\}^k$, $M \in \{0,1\}^{<(\mu+1)n}$
Output: $T \in \{0,1\}^t$
1: $M_1 \dots M_\ell \leftarrow \mathrm{pad}_n(M)$
2: $S \leftarrow 0^n$
3: **for** $i = 1, \dots, \ell - 1$ **do**
4: $\quad S \leftarrow S \oplus I(H(K_1, i), M_i)$
5: $T \leftarrow \lfloor E_{K_2}(S \oplus M_\ell) \rfloor_t$
6: **return** T

A PRF security bound is given in Section 4.1 and a MAC security bound in Section 4.2. Tightness of these bounds is discussed in Section 4.3.

4.1 PRF Security of EliMAC

We derive a formal PRF security bound for EliMAC. The proof is inspired by the PRF security analysis of LightMAC [LPTY16, Theorem 1], with a small difference that we rely on the PRF security of truncation [Sta78, BN18, Men19] for the last primitive call. We note that the bound includes a term involving μ, seemingly making the bound length-dependent. This, however, is rather caused by the general definition of μ-independence in Section 2.1. As we will see in Section 5, and in particular in Corollary 1, for good universal hash functions H, i.e., where μ is very small relative to 2^n, the value δ is very small (even up to $([2^n]_\mu)^{-1/\mu}$) and the dependence on μ vanishes.

Theorem 1. *Let $\mu, k', k, n, t \in \mathbb{N}$. Let $H : \{0,1\}^{k'} \times [1, \dots, \mu] \to \{0,1\}^k$ be a δ-μ-independent hash function family and $I : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ an ε-XOR-universal hash function family. Then,*

$$\mathbf{Adv}^{\mathrm{prf}}_{\mathrm{EliMAC}}(q, \tau) \leq \binom{q}{2}(2^k\delta)^\mu\varepsilon + \left(\binom{q}{2}/2^{2n-t}\right)^{1/2} + \mathbf{Adv}^{\mathrm{prp}}_E(q, \tau')\,, \tag{8}$$

where $\tau' = \tau + \mathcal{O}(q)$.

Proof. Consider any adversary A making q construction queries and operating in time τ. Let $p \xleftarrow{\$} \mathrm{perm}(\{0,1\}^n)$ be a random permutation. As a first step, we replace E_{K_2} by p. This comes at the cost of

$$\mathbf{Adv}^{\mathrm{prp}}_E(A') \tag{9}$$

for some adversary A' with complexity q and time $\tau' = \tau + \mathcal{O}(q)$.

We next replace $\lfloor p(\cdot) \rfloor_t$ by a random function $f \xleftarrow{\$} \mathrm{func}(\{0,1\}^n, \{0,1\}^t)$. By the PRF security of truncation [Sta78, BN18, Men19], this step comes at the cost of

$$\left(\binom{q}{2}/2^{2n-t}\right)^{1/2}. \tag{10}$$

Denote the resulting scheme by $\mathrm{EliMAC}_{K_1,f}$. For clarity, this function is defined as

$$\mathrm{EliMAC}_{K_1,f}(M) = f\left(\mathrm{EliHash}(K_1, M_1, \dots, M_{\ell-1}) \oplus M_\ell\right),$$

where M is first injectively padded to $M_1 \dots M_\ell$.

The function $\mathrm{EliMAC}_{K_1,f}$ is perfectly indistinguishable from random as long as the adversary never makes two different queries M, M' such that

$$\mathrm{EliHash}(K_1, M_1, \dots, M_{\ell-1}) \oplus M_\ell = \mathrm{EliHash}(K_1, M'_1, \dots, M'_{\ell'-1}) \oplus M'_{\ell'},$$

where $\ell, \ell' \leq \mu$ are the block lengths of padded M, M', respectively, because in this case the inputs to the final f are all distinct. By Proposition 1, that would happen with probability at most

$$\binom{q}{2}(2^k\delta)^\mu\varepsilon\,. \tag{11}$$

The proof is completed by combining the individual terms (9), (10), and (11). □

4.2 MAC Security of EliMAC

We derive a formal MAC security bound for EliMAC. For LightMAC, Luykx et al. [LPTY16, Theorem 2] relied on an unforgeability result of Dodis and Pietrzak [DP07]. However, some subtleties arise: Dodis and Pietrzak considered a hash-then-permute construction whereas LightMAC, and also EliMAC, is a hash-then-permute-then-truncate construction, i.e., a hash-then-permute construction followed by truncation. Therefore, we perform the analysis in greater detail and derive a tighter bound. The core of this analysis is a reduction from truncated to untruncated EliMAC, equation (15), that in fact applies to any hash-then-permute-then-truncate construction, and in particular immediately allows for a slight improvement of the MAC security bound of LightMAC of [LPTY16, Theorem 2].[1]

Theorem 2. *Let $\mu, k', k, n, t \in \mathbb{N}$. Let $H : \{0,1\}^{k'} \times [1,\ldots,\mu] \to \{0,1\}^k$ be a δ-μ-independent hash function family and $I : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ an ε-XOR-universal hash function family. Then,*

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EliMAC}}(q, q_f, \tau) \leq \binom{q}{2}(2^k\delta)^\mu\varepsilon + \frac{q_f 2^n}{2^t} \cdot \max\left\{(2^k\delta)^\mu\varepsilon, (2^n - q)^{-1}\right\} + \mathbf{Adv}^{\mathrm{prp}}_E(q + q_f, \tau')\,. \tag{12}$$

where $\tau' = \tau + \mathcal{O}(q + q_f)$.

Proof. Consider any adversary A making q construction queries and operating in time τ. Let $p \xleftarrow{\$} \mathrm{perm}(\{0,1\}^n)$ be a random permutation. As a first step, we replace E_{K_2} by p. This comes at the cost of

$$\mathbf{Adv}^{\mathrm{prp}}_E(A') \tag{13}$$

for some adversary A' with complexity q and time $\tau' = \tau + \mathcal{O}(q + q_f)$.

Now, unlike in the proof of Theorem 1, we do *not* replace $\lfloor p(\cdot)\rfloor_t$ by a random function $f \xleftarrow{\$} \mathrm{func}(\{0,1\}^n, \{0,1\}^t)$: it turns out to be conceptually simpler to reason about hash-then-permute-then-truncate. Therefore, we denote the resulting scheme by $\mathrm{EliMAC}_{K_1,p,t}$. For clarity, this function is defined as

$$\mathrm{EliMAC}_{K_1,p,t}(M) = \lfloor p\left(\mathrm{EliHash}(K_1, M_1, \ldots, M_{\ell-1}) \oplus M_\ell\right)\rfloor_t\,,$$

where M is first injectively padded to $M_1 \ldots M_\ell$. Our goal is to bound

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EliMAC}_{K_1,p,t}}(q, q_f)\,,$$

where we remark that we consider idealized functions, based on K_1 and p, and hence we have dropped the time complexity τ.

If $t = n$, the function $\mathrm{EliMAC}_{K_1,p,n}$ is a hash-then-permute construction with

$$\mathrm{EliHash}(K_1, M_1, \ldots, M_{\ell-1}) \oplus M_\ell$$

[1] This improvement can be obtained by replacing $(2^k\delta)^\mu\varepsilon$ in the bound and proof of Theorem 2 by LightMAC's $\varepsilon = 1/(2^{n/2} - 1)^2$.

as universal hash. By Proposition 1, this function is $(2^k\delta)^\mu\varepsilon$-universal. Therefore, we immediately obtain from Dodis and Pietrzak [DP07, Proposition 1] that $\mathrm{EliMAC}_{K_1,p,n}$ is unforgeable up to bound

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EliMAC}_{K_1,p,n}}(q,q_f) \leq \binom{q}{2}(2^k\delta)^\mu\varepsilon + q_f \cdot \max\left\{(2^k\delta)^\mu\varepsilon, (2^n-q)^{-1}\right\}. \tag{14}$$

For arbitrary $t \leq n$, we claim that

$$\mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EliMAC}_{K_1,p,t}}(q,q_f) \leq \mathbf{Adv}^{\mathrm{mac}}_{\mathrm{EliMAC}_{K_1,p,n}}(q,2^{n-t}q_f), \tag{15}$$

Obviously, under the hypothesis that this claim holds, the proof is completed, by combining the individual terms (13), (14), and (15).

Reduction (15), finally, has already been demonstrated by Cogliati et al. [CLS17, Lemma 8], but we repeat it in our terminology for completeness. Let A_t be any forger against $\mathrm{EliMAC}_{K_1,p,t}$ making q MAC queries and q_f forgery attempts. We construct a forger A_n against $\mathrm{EliMAC}_{K_1,p,n}$ making q MAC queries and $2^{n-t}q_f$ forgery attempts with at least the same success probabilities. Forger A_n operates as follows:

- If A_t makes a MAC query M, A_n queries its own MAC oracle on input M, truncates the result to t bits, and forwards the resulting value to A_t.
- If A_t makes a forgery attempt (M,T), A_n makes 2^{n-t} forgery attempts $(M,T\|Z_i)$ for all $Z_i \in \{0,1\}^{n-t}$. If any of these 2^{n-t} forgery attempts succeeds, it informs that A_t has mounted a successful forgery. Otherwise, it informs that A_t's forgery attempt failed.

Clearly, A_n perfectly simulates the oracles of A_t, and A_n mounts a successful forgery if A_t mounts a successful forgery. This proves (15), and completes the proof of this theorem. □

4.3 Tightness of the Security Bounds

In both reductions, the hybrid step that replaces E_{K_2} by a random permutation serves to capture any undesired properties of the block cipher E. We will argue tightness of the resulting, idealized, construction.

The PRF security bound of Theorem 1 has two terms, namely $\binom{q}{2}(2^k\delta)^\mu\varepsilon + \left(\binom{q}{2}/2^{2n-t}\right)^{1/2}$.

- Term $\left(\binom{q}{2}/2^{2n-t}\right)^{1/2}$. If the adversary restricts itself to 1-block padded messages, it has direct access to the truncated permutation $\lfloor p(\cdot)\rfloor_t$. The second term of the bound corresponds to the distance of this function from random, and Gilboa and Gueron proved that this term is tight [GG16]. In other words, there exists a distinguisher that can distinguish the truncated permutation from random in around $q = 2^{n-t/2}$ queries.
- Term $\binom{q}{2}(2^k\delta)^\mu\varepsilon$. This term is harder to parse, but we argue based on the separate cryptographic building blocks H and I, and that way indirectly on the separate terms δ and ε.
  - Assume that $H : \{0,1\}^{k'} \times [1,\ldots,\mu] \to \{0,1\}^k$ is a random function. This means that $\delta = 2^{-k}$ and the bound simplifies to $\binom{q}{2}\varepsilon$. If the adversary focuses on 2-block padded messages $M_1\|M_2$, any two queries collide at the input to p with probability at most ε. Under the assumption that the function is reasonably regular (i.e., each image has an approximately equal amount of preimages), and this term ε is met for many query tuples, this allows the adversary to distinguish the scheme from random in around $\varepsilon^{-1/2}$ queries.

– Assume that $I : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is a finite field multiplication with $k = n$. This means that $\varepsilon = 2^{-n}$. If the adversary focuses on 2-block padded messages $M_1 \| M_2$, the bound is of the form $\binom{q}{2}\delta$. Any two queries collide at the input to p if

$$H_{K_1}(1) \otimes (M_1 \oplus M_1') = (M_2 \oplus M_2')\,.$$

This happens with probability at most δ. Under the assumption that the function is reasonably regular, and this term δ is met for many query tuples, this allows the adversary to distinguish the scheme from random in around $\delta^{-1/2}$ queries.

For larger messages, the term in the bound increases. For example, for 3-block padded message $M_1 \| M_2 \| M_3$, the bound is of the form $\binom{q}{2}2^k\delta^2$. Any two queries collide at the input to p if

$$H_{K_1}(1) \otimes (M_1 \oplus M_1') = R \otimes (M_2 \oplus M_2') \oplus (M_3 \oplus M_3')\,, \qquad H_{K_1}(2) = R\,,$$

for some $R \in \{0,1\}^k$. This happens with probability at most $2^k\delta^2$. Under the assumption that the function is reasonably regular, and this term δ^2 is met for many query tuples, this allows the adversary to distinguish the scheme from random in around $\delta^{-1}2^{-k/2}$ queries.

The attacks have an addendum that they hold under the assumption that H and I are reasonably regular. For example, for the presence of $\binom{q}{2}\varepsilon$, an attacker can closely match this bound if the probability for a collision is close to ε for *many* pairs of queries. If H has few outliers and is secure otherwise, the attacker might not meet this term.

For the bound of MAC security of Theorem 2, the analysis is slightly different. The bound of this theorem has two terms, namely $\binom{q}{2}(2^k\delta)^\mu\varepsilon$ and $\frac{q_f 2^n}{2^t} \cdot \max\left\{(2^k\delta)^\mu\varepsilon, (2^n-q)^{-1}\right\}$.

- Term $\binom{q}{2}(2^k\delta)^\mu\varepsilon$. The first term already appeared for PRF-security, and corresponds to input collisions to p. Clearly, if such an input collision between two messages M and M' occurs, the adversary might add a constant to M_ℓ and $M'_{\ell'}$, query one of the adjusted messages and use the outcome to forge for the other message.

- Term $\frac{q_f 2^n}{2^t} \cdot \max\left\{(2^k\delta)^\mu\varepsilon, (2^n-q)^{-1}\right\}$. First consider $t = n$. Any forgery attempt with fresh tag succeeds with probability at most $1/(2^n - q)$, and any forgery attempt with repeated tag succeeds only if the input to p of the forgery collides with the input to p of the earlier query for that tag. This attack matches the term. Now, for $t \leq n$, one may note that any forgery attempt has a success probability amplified by 2^{n-t}, as it only needs to be correct on t bits of the output; the remaining $n - t$ bits can be any value.

5 Instantiation and Application

A logical choice for the block cipher E is AES-128 [DR02] (or, simply, AES, recalling that we would drop "-128" for brevity). It is defined for keys and message blocks of size $n = 128$. For the universal hash function families, one can take round-reduced versions of AES. The 4-round AES is a popular choice for universal hashing. Keliher and Sui [KS07] derived an upper bound of $\varepsilon = 1.881 \cdot 2^{-114}$ on the MEDP, the average differential probability taken over all round keys, of 4-round AES. This bound, however, is not tight; Daemen and Rijmen estimate that it is closer to 2^{-126}, and that the differential probability (DP) for the keyless 4-round AES is at most 2^{-119} [DR10]. Furthermore, the current state of

cryptanalysis of Alred [DR05a] constructions based on 4-round AES has not been able to exploit higher differential probabilities than those that would be expected from the independently keyed version [DKS15]. As such, here we make the added assumption that replacing $\mathrm{AES}_4(K, X)$ by $\mathrm{AES}_4(0, K \oplus X)$ does not result in a significant differential probability gain.

We can thus use this 4-round AES function to instantiate $I : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ as

$$I(K, M) = \mathrm{AES}_4(0, K \oplus M)\,. \tag{16}$$

With respect to H, we require μ-independence, and this property does not directly follow from XOR-universality. Fortunately, we only need μ-independence for a *restricted use of the hash function*, namely where the input is an integer from $[1, \ldots, \mu]$ that is first encoded into an n-bit string. We can still be hopeful about instantiating H using round-reduced a-round AES, for $a \in [0, 10]$. Formally, we instantiate $H : \{0,1\}^n \times [1, \ldots, \mu] \to \{0,1\}^n$ as

$$H(K, i) = \mathrm{AES}_a(K, \langle i \rangle_n)\,, \tag{17}$$

where $\langle i \rangle_n$ is any injective encoding of $i \in [1, \ldots, \mu]$ as an n-bit string. One suitable encoding is the naive one that transforms i into an n-bit string and prepends it with zeros, e.g., $\langle 5 \rangle_n = 0^{n-3}101$. Depending on the value μ, better options for the encoding might exist, as we will see in the next section.

Clearly, μ-independence is implied by PRP-security of H_K. Stated differently, if H_K is PRP-secure, at the cost of its PRP-advantage we can replace it by a random permutation, which is δ-μ-independent for $\delta = ([2^n]_\mu)^{-1/\mu} \geq 2^{-n}$ (see also Section 2.1). Note that, in this case, Theorem 1 indeed leads to a length-independent bound. Formally seen, one replaces H by a random function family R which is indexed by key space $\mathrm{perm}(\{0,1\}^n)$. Full 10-round AES is commonly considered to be PRP-secure. In our setting, however, the adversary is significantly restricted: it has a limited data complexity $\mu \ll q$ and also the input values are fixed to $\langle i \rangle_n$ for $i \in [1, \ldots, \mu]$.

For $\mu = 2$, we can nevertheless rely on the XOR-universality and take $a = 4$. Apart from this case, to judge the number of rounds a needed considering a certain μ, we consider the best attacks on round-reduced AES. To the best of our knowledge, the best attack on 7-round AES requires 2^{97} chosen plaintexts and has a time complexity of 2^{99} [DFJ13]. Hence, we conjecture that for $\mu \leq 2^{32}$, $a = 7$ rounds are sufficient. In the case of 6 rounds of AES, to the best of our knowledge, the *best attack* in terms of data complexity needs 2^8 chosen plaintexts and has a time complexity of 2^{106} [DF13]. Therefore, we conjecture that $\mu \leq 2^4$, $a = 6$ rounds are sufficient. Here, we recall that in our case, we *do not* require the round-reduced versions of AES to be PRP-secure: it is sufficient that they are μ-independent. The recent research in secret-key distinguishers shows that the best distinguishers on 6 rounds of AES require more than 2^{80} chosen plaintexts [BR19]. Hence, distinguishing or even guessing output distributions of round-reduced AES without guessing the key seems to be a more challenging problem than recovering the key given input and output data. We finally remark that, following Jha et al. [JMNS19], it may suffice to only take $a = 2$ rounds of AES, but only in a very restricted setting on the size of the counter i and at the cost of a larger key.

5.1 Concrete Instantiation

Based on above reasoning, we define the following concrete instantiation EliMAC-AES. Let $k = n = 128$ and $t \in \mathbb{N}$ satisfying $t \leq n$. Let K_1, K_2 be two k-bit keys. Let $s = 32$, so $\mu = 2^{32}$. Instantiate I using 4-round AES as prescribed in (16) and H using ($a = 7$)-round AES as prescribed in (17). Define the encoding function $\langle i \rangle_{128}$ by

encoding i as a 32-bit string naively and concatenating this string four times. For example, $\langle 5 \rangle_{128} = 0^{29}101 \| 0^{29}101 \| 0^{29}101 \| 0^{29}101$. Finalize EliMAC with $\mathrm{AES}_{10}(K_2, \cdot)$.

We can prove security of EliMAC-AES under the assumption that AES_{10} is PRP-secure against an adversary making q queries, that AES_7 is PRP-secure against an adversary making $\mu \ll q$ predetermined queries (namely $\langle i \rangle_n$ for $i \in [1, \ldots, \mu]$), and that AES_4 is XOR-universal. The proof is, in fact, a direct corollary of Theorem 1.

Corollary 1. *Let $\mu = 2^{32}$, $k = n = 128$, and $t \in \mathbb{N}$ with $t \leq n$. Assume that $\mathrm{AES}_4 : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is a ε-XOR-universal hash function family. Then,*

$$\mathbf{Adv}^{\mathrm{prf}}_{\text{EliMAC-AES}}(q, \tau) \leq \binom{q}{2} 2\varepsilon + \left(\binom{q}{2} / 2^{2n-t} \right)^{1/2} + \mathbf{Adv}^{\mathrm{prp}}_{\mathrm{AES}_{10}}(q, \tau') + \mathbf{Adv}^{\mathrm{prp}}_{\mathrm{AES}_7}(\mu, \tau'') \,, \tag{18}$$

where $\tau' = \tau + \mathcal{O}(q)$ and $\tau'' = \tau + \mathcal{O}(\mu)$.

Proof. Consider any adversary A making q construction queries and operating in time τ. Let $p' \xleftarrow{\$} \mathrm{perm}(\{0,1\}^n)$ be a random permutation. As a first step, we replace H_{K_1} by p'. This comes at the cost of

$$\mathbf{Adv}^{\mathrm{prp}}_{\mathrm{AES}_7}(A'') \tag{19}$$

for some adversary A'' with complexity μ and time $\tau'' = \tau + \mathcal{O}(\mu)$. Formally seen, this transition moves EliMAC-AES to a scheme where the hash function family H is replaced by a random function family R which is indexed by key space $\mathrm{perm}(\{0,1\}^n)$ and defined as $R(p', i) = p'(\langle i \rangle_n)$. This function family is δ-μ-independent with $\delta = ([2^n]_\mu)^{-1/\mu}$.

The remainder of the proof is identical to the proof of Theorem 1, and we obtain the exact same bound, plus (19) maximized over any adversary with complexity μ and time τ''. The bound still contains a term

$$(2^k \delta)^\mu = \frac{2^{n\mu}}{[2^n]_\mu} \,,$$

where we use that $k = n$. Using Lemma 1, we can bound this term by 2, and this completes the proof. □

Given that AES_4 can be considered ε-XOR-universal with $\varepsilon \approx 1.881 \cdot 2^{-114}$ [KS07], Corollary 1 guarantees security of EliMAC-AES up to $q \lesssim 2^{56}$.

5.2 Comparison with LightMAC

In total, EliMAC-AES performs 11 AES rounds per message block, making it slightly more expensive than LightMAC instantiated with AES in terms of the number of AES rounds. However, EliMAC has significant advantages:

- LightMAC glues together the counter and the message block, meaning that all message blocks in LightMAC are of size 96 bits (except for the last block). In EliMAC, all message blocks are of size 128 bits. This gives – considering the number of AES rounds processed – an asymptotic efficiency gain of 21%.
- In EliMAC, the subkeys $H(K, 1), \ldots, H(K, \mu)$ can be precomputed and stored in memory. This reduces the number of AES rounds per block to 4 (except for the last block). Of course, this only works if sufficient storage for the subkeys is available.

In conclusion, we remark that the security models are different. Security of LightMAC is proven up to the PRP-security of 10-round AES against adversaries that can make up to $q + q_f$ queries. For EliMAC, we additionally require that 7-round AES is PRP-secure against restricted adversaries that can only make μ evaluations for fixed inputs, and we require that 4-round AES is XOR-universal.

Table 2: Measured cycles per byte when authenticating messages of 64, 1536, and 4096 bytes on the Ivy Bridge, Broadwell, Skylake, and Zen 2 microarchitectures with Turbo Boost disabled. "EliMAC p.c." indicates the EliMAC variant with precomputed keys.

		64	1536	4096
Ivy Bridge	LightMAC	3.43	1.13	1.11
	EliMAC	2.18	1.02	0.98
	EliMAC p.c.	2.00	0.46	0.43
	PMAC2	4.50	1.28	1.22
	ZMAC	5.70	1.49	1.26
Broadwell	LightMAC	8.75	0.98	1.08
	EliMAC	1.94	0.76	0.74
	EliMAC p.c.	1.75	0.30	0.27
	PMAC2	3.25	1.13	1.09
	ZMAC	6.97	1.34	1.23
Skylake	LightMAC	2.53	0.86	0.85
	EliMAC	1.56	0.70	0.69
	EliMAC p.c.	1.31	0.27	0.26
	PMAC2	1.71	0.67	0.64
	ZMAC	4.64	0.91	0.84
Zen 2	LightMAC	2.18	0.58	0.58
	EliMAC	1.31	0.45	0.42
	EliMAC p.c.	0.87	0.14	0.13
	PMAC2	1.31	0.58	0.56
	ZMAC	4.34	0.88	0.81

6 Implementation

We implemented and compared EliMAC-AES with some related schemes, namely LightMAC [LPTY16], PMAC2 [CCJN21], and ZMAC [IMPS17] instantiated with `Deoxys-TBC-256` [JNPS21]. We chose these modes because they, like EliMAC, are parallel hash-then-PRF modes that also have length independent bounds.[2] To compare them, we measured the cycle counts of authenticating messages for various message lengths, and computed the median of 2^{16} such runs. Where necessary, Turbo Boost was disabled to prevent dynamic frequency scaling to distort the timings. Table 2 presents the results for several CPUs featuring hardware-accelerated AES instructions: Intel Core i7-3770 (Ivy Bridge), Intel Xeon E5-2686 v4 (Broadwell), Intel Core i7-6770HQ (Skylake), and AMD EPYC 7402P (Zen 2).

Because in AES-based modes the most costly operation is often the evaluation of AES, the performance of such modes is often approximated by how many AES rounds are executed, divided by how many bytes are processed. In parallel modes the latency of the `aesenc` instruction is mostly irrelevant, and the important metric becomes how many new AES instructions can be started per cycle. In most cases, one new AES instruction can be issued at every cycle (cf., Table 3); this results in parallel AES modes such as ECB, CTR, or OCB to have asymptotic performance of 10 cycles (rounds) divided by 16 bytes per block, i.e., 0.625 cycles per byte. In some architectures, like Intel's Ice Lake or AMD's Zen 3, two AES rounds can be started per cycle, and furthermore on two parallel blocks using `ymm` registers, making the asymptotic parallel throughput $10/64 = 0.15625$ [AR19]. Here, however, we will stick to the 1-round per cycle for simplicity, though it is straightforward

[2] In the case of PMAC2, the length-independence only applies if lengths stay below 2^{32}, which is the case considered here. In the case of ZMAC, length-independence only applies to the $\mathbb{ZMAC}$ variant.

to adapt to these other cases.

Implementation of EliMAC-AES is quite straightforward on processors with pipelined AES instructions, such as Intel and AMD chips featuring AES-NI, ARMv8 chips, and others. As such, the expectation was that the implementation matched the asymptotic performance given by $11/16 = 0.6875$. This is indeed the case on Skylake, while on previous Intel microarchitectures there is some noticeable overhead from other instructions competing with the AES instructions for execution units, cf., Table 3 for details.

LightMAC is almost as simple to implement as EliMAC, with one caveat – the 96-bit blocks are not quite natural to the 128-bit `xmm` registers, and thus the message loads become more elaborate, involving either overlapping loads, blends, or multiple small loads to load the message plus counter into a 128-bit register. Which option is better depends on the target architecture. Table 2 shows that this caveat is not without overhead. The asymptotic performance of LightMAC with 96-bit blocks is expected to be $\frac{10}{16} \cdot \frac{128}{96} \approx 0.83$ cycles per byte on the 1 `aesenc` per cycle model, but on the Broadwell and Ivy Bridge microarchitectures we see that the overhead can be quite noticeable.

PMAC2 is not fully specified in [CCJN21], thus where unspecified we chose the option that was most convenient for the implementer. We use the primitive polynomial $x^{128} + x^7 + x^2 + x + 1$ to instantiate $\mathbb{F}_{2^{128}}$. The polynomial evaluation of the processed message blocks at x, i.e., $\bigoplus_{i=0}^{l-1} x^{l-1-i}\pi(m_i \oplus x^i L)$ presents some implementation challenges comparable to the authentication part of AES-GCM [GK10]; we used parallel polynomial evaluation and the carryless multiplication instruction `PCLMULQDQ` where it presented an advantage relatively to using several shift and xor instructions. The asymptotic performance of PMAC2 is expected to be $10/16 = 0.625$. However, since the overhead of non-AES instructions is much larger on PMAC2 than the other modes, only the Skylake microarchitecture achieves this theoretical figure.

ZMAC [IMPS17] presented similar implementation challenges as PMAC2, but heightened. Like PMAC2, ZMAC is also not fully specified; we chose the same primitive polynomial as above for the relevant finite field operations. Furthermore, one can implement ZMAC in several forms:

- Using 1 byte of tweak to signal the tweakable blockcipher instance and finite field multiplications for the message offset;
- Using 5 bytes of tweak to signal both instance and message offset; no finite field operations for masking;
- Using finite field operations but no separate tweakable blockcipher instantiations, as suggested by Naito [Nai18].

We implemented all 3 variants. Ultimately we chose ZMAC1, Naito's variant, as it seemed to perform marginally better.

On paper, ZMAC1 implemented with `Deoxys-TBC-256` has an asymptotic lower bound of $14/(16+16) = 0.4375$ cycles per byte. The ZMAC authors [IMPS17, §6.2] estimated an overhead factor of 1.4 to the asymptotic $14/16 = 0.875$ when processing random tweaks, and divided it by 2 to obtain an estimated $0.875 \cdot 1.4/2 = 0.61$ cycles per byte. However, the overhead incurred here is twofold. Firstly, the Deoxys tweak schedule involves a shuffle and xor at every round, which keeps all the available execution ports busy in most cases (cf. Table 3). Secondly, the ZMAC mode adds more overhead of its own in the form of multiple finite field multiplications both before and after the tweakable blockcipher evaluation. As such, we have not been able to come close to the asymptotic or estimated performance in any of the tested architectures.

While we do not cover the various ARMv8 microarchitectures that contain AES instructions, similar considerations apply there as well. In conclusion, we can argue that in this class of length-independent hash-then-PRF modes of operation, EliMAC stays

Table 3: Instruction port breakdown for several relevant instructions operating on `xmm` registers. Obtained from [AR19]. The notation `p015` means that the instruction can be dispatched via ports 0, 1, or 5 in a given cycle; `p0+p015` means that the instruction involves 2 μops, one of which dispatched to port 0, and the other dispatched to one of ports 0, 1, or 5.

Instruction	Ivy Bridge	Broadwell	Skylake	Zen 2
`aesenc,aesenclast`	`p0+p015`	`p5`	`p0`	`p01`
`paddd`	`p15`	`p15`	`p015`	`p013`
`pxor,pand,por`	`p015`	`p015`	`p015`	`p0123`
`psrld,pslld`	`p0`	`p0`	`p01`	`p2`
`psrldq,pslldq`	`p15`	`p5`	`p5`	`p12`
`pclmulqdq`	complex	`p0`	`p5`	`4*p12`
`blendps,blendd`	`p05`	`p015`	`p015`	`p013`
`pshufb`	`p15`	`p5`	`p5`	`p12`

closest to asymptotic performance due to its implementation simplicity. From Table 3 one can heuristically estimate that, all else being equal, modes which mainly consist of AES rounds plus bitwise logical operations and integer additions are generally expected to have less overhead than modes that are heavier on shifts, permutations, or polynomial multiplications, will generally perform worse.

The implementation of EliMAC using precomputation is vastly faster than any of the other modes, costing only ≈ 4 AES rounds per block after initialization; however this comes with added memory requirements, which might involve more overall cache misses when employed as part of a larger application.

7 Conclusion

It seems that the reduction of the XOR-universality of EliHash to the μ-independence of H and the XOR-universality of I (Proposition 1) leaves little room for improvement. Whereas the XOR-universality assumption is well-established, the μ-independence assumption is a bit stronger. It is satisfied by a random permutation, and thus also by functions that are hard to distinguish from a random permutation (up to their PRP-security). In EliMAC-AES, we even instantiate it using a round-reduced version of AES, noting that the function H is only evaluated a limited number of times on restricted inputs. Nonetheless, the instantiation is not tight, meaning that PRP-security guarantees much more than what we need and a weaker primitive might be employed. It is an interesting question to investigate how many rounds of AES are required to guarantee μ-independence.

For EliHash, we proved XOR-universality. We could have gotten away with blinded keyed hashing (BKH) [GDM19]. However, for the specific setting, proving BKH security of EliHash is equivalent to proving PRF-security of EliMAC, and thus resorting to XOR-universality of EliHash was the most logical option.

Acknowledgments

We would like to thank the reviewers for their valuable feedback and comments. Bart Mennink is supported by the Netherlands Organisation for Scientific Research (NWO) under grant VI.Vidi.203.099.

References

[AR19] Andreas Abel and Jan Reineke. uops.info: Characterizing Latency, Throughput, and Port Usage of Instructions on Intel Microarchitectures. In Iris Bahar, Maurice Herlihy, Emmett Witchel, and Alvin R. Lebeck, editors, *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2019, Providence, RI, USA, April 13-17, 2019*, pages 673–686. ACM, 2019.

[BCC10] Céline Blondeau, Anne Canteaut, and Pascale Charpin. Differential properties of power functions. In *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*, pages 2478–2482. IEEE, 2010.

[Ber99] Daniel J. Bernstein. How to Stretch Random Functions: The Security of Protected Counter Sums. *J. Cryptology*, 12(3):185–192, 1999.

[Ber05] Daniel J. Bernstein. Stronger Security Bounds for Wegman-Carter-Shoup Authenticators. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 164–180. Springer, 2005.

[BKR94] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The Security of Cipher Block Chaining. In Yvo Desmedt, editor, *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings*, volume 839 of *Lecture Notes in Computer Science*, pages 341–358. Springer, 1994.

[BN18] Srimanta Bhattacharya and Mridul Nandi. A note on the chi-square method: A tool for proving cryptographic security. *Cryptography and Communications*, 10(5):935–957, 2018.

[BR02] John Black and Phillip Rogaway. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In Lars R. Knudsen, editor, *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, volume 2332 of *Lecture Notes in Computer Science*, pages 384–397. Springer, 2002.

[BR19] Navid Ghaedi Bardeh and Sondre Rønjom. The Exchange Attack: How to Distinguish Six Rounds of AES with $2^{88.2}$ Chosen Plaintexts. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part III*, volume 11923 of *Lecture Notes in Computer Science*, pages 347–370. Springer, 2019.

[Bra82] Gilles Brassard. On Computationally Secure Authentication Tags Requiring Short Secret Shared Keys. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology: Proceedings of CRYPTO '82, Santa Barbara, California, USA, August 23-25, 1982*, pages 79–86. Plenum Press, New York, 1982.

[CCJN21] Bishwajit Chakraborty, Soumya Chattopadhyay, Ashwin Jha, and Mridul Nandi. On Length Independent Security Bounds for the PMAC Family. *IACR Trans. Symmetric Cryptol.*, 2021(2):423–445, 2021.

[CJN21] Soumya Chattopadhyay, Ashwin Jha, and Mridul Nandi. Fine-Tuning the ISO/IEC Standard LightMAC. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part III*, volume 13092 of *Lecture Notes in Computer Science*, pages 490–519. Springer, 2021.

[CLS17] Benoît Cogliati, Jooyoung Lee, and Yannick Seurin. New Constructions of MACs from (Tweakable) Block Ciphers. *IACR Trans. Symmetric Cryptol.*, 2017(2):27–58, 2017.

[CS16] Benoît Cogliati and Yannick Seurin. EWCDM: An Efficient, Beyond-Birthday Secure, Nonce-Misuse Resistant MAC. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 121–149. Springer, 2016.

[DDNY18] Nilanjan Datta, Avijit Dutta, Mridul Nandi, and Kan Yasuda. Encrypt or Decrypt? To Make a Single-Key Beyond Birthday Secure Nonce-Based MAC. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 631–661. Springer, 2018.

[DF13] Patrick Derbez and Pierre-Alain Fouque. Exhausting Demirci-Selçuk Meet-in-the-Middle Attacks Against Reduced-Round AES. In Shiho Moriai, editor, *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, volume 8424 of *Lecture Notes in Computer Science*, pages 541–560. Springer, 2013.

[DFJ13] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 371–387. Springer, 2013.

[DKS15] Orr Dunkelman, Nathan Keller, and Adi Shamir. Almost universal forgery attacks on AES-based MAC's. *Des. Codes Cryptogr.*, 76(3):431–449, 2015.

[DNT19] Avijit Dutta, Mridul Nandi, and Suprita Talnikar. Beyond Birthday Bound Secure MAC in Faulty Nonce Model. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, volume 11476 of *Lecture Notes in Computer Science*, pages 437–466. Springer, 2019.

[DP07] Yevgeniy Dodis and Krzysztof Pietrzak. Improving the Security of MACs Via Randomized Message Preprocessing. In Alex Biryukov, editor, *Fast Software Encryption, 14th International Workshop, FSE 2007, Luxembourg, Luxembourg,*

March 26-28, 2007, Revised Selected Papers, volume 4593 of *Lecture Notes in Computer Science*, pages 414–433. Springer, 2007.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.

[DR05a] Joan Daemen and Vincent Rijmen. A New MAC Construction ALRED and a Specific Instance ALPHA-MAC. In Henri Gilbert and Helena Handschuh, editors, *Fast Software Encryption: 12th International Workshop, FSE 2005, Paris, France, February 21-23, 2005, Revised Selected Papers*, volume 3557 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2005.

[DR05b] Joan Daemen and Vincent Rijmen. The MAC function Pelican 2.0. Cryptology ePrint Archive, Report 2005/088, 2005. https://eprint.iacr.org/2005/088.

[DR10] Joan Daemen and Vincent Rijmen. Refinements of the ALRED construction and MAC security claims. *IET Information Security*, 4(3):149–157, 2010.

[GDM19] Aldo Gunsing, Joan Daemen, and Bart Mennink. Deck-Based Wide Block Cipher Modes and an Exposition of the Blinded Keyed Hashing Model. *IACR Trans. Symmetric Cryptol.*, 2019(4):1–22, 2019.

[GG16] Shoni Gilboa and Shay Gueron. The Advantage of Truncated Permutations. *CoRR*, abs/1610.02518, 2016.

[GK10] Shay Gueron and Michael E. Kounavis. Efficient implementation of the Galois Counter Mode using a carry-less multiplier and a fast reduction algorithm. *Inf. Process. Lett.*, 110(14-15):549–553, 2010.

[GMS74] Edgar N. Gilbert, F. Jessie MacWilliams, and Neil J. A. Sloane. Codes which detect deception. *Bell System Technical Journal*, 53:405–424, 1974.

[IMPS17] Tetsu Iwata, Kazuhiko Minematsu, Thomas Peyrin, and Yannick Seurin. ZMAC: A Fast Tweakable Block Cipher Mode for Highly Secure Message Authentication. In Katz and Shacham [KS17], pages 34–65.

[ISO19] ISO/IEC 29192-6:2019. Information technology – Lightweight cryptography – Part 6: Message authentication codes (MACs), 2019.

[JMNS19] Ashwin Jha, Cuauhtemoc Mancillas-López, Mridul Nandi, and Sourav Sen Gupta. On Random Read Access in OCB. *IEEE Trans. Inf. Theory*, 65(12):8325–8344, 2019.

[JNPS21] Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. The Deoxys AEAD Family. *J. Cryptol.*, 34(3):31, 2021.

[JS07] Goce Jakimoski and K. P. Subbalakshmi. On Efficient Message Authentication Via Block Cipher Design Techniques. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Malaysia, December 2-6, 2007, Proceedings*, volume 4833 of *Lecture Notes in Computer Science*, pages 232–248. Springer, 2007.

[KS07] Liam Keliher and Jiayuan Sui. Exact maximum expected differential and linear probability for two-round Advanced Encryption Standard. *IET Information Security*, 1(2):53–57, 2007.

[KS17] Jonathan Katz and Hovav Shacham, editors. *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*. Springer, 2017.

[LPSY16] Atul Luykx, Bart Preneel, Alan Szepieniec, and Kan Yasuda. On the Influence of Message Length in PMAC's Security Bounds. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 596–621. Springer, 2016.

[LPTY16] Atul Luykx, Bart Preneel, Elmar Tischhauser, and Kan Yasuda. A MAC Mode for Lightweight Block Ciphers. In Thomas Peyrin, editor, *Fast Software Encryption - 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers*, volume 9783 of *Lecture Notes in Computer Science*, pages 43–59. Springer, 2016.

[Men19] Bart Mennink. Linking Stam's Bounds with Generalized Truncation. In Mitsuru Matsui, editor, *Topics in Cryptology - CT-RSA 2019 - The Cryptographers' Track at the RSA Conference 2019, San Francisco, CA, USA, March 4-8, 2019, Proceedings*, volume 11405 of *Lecture Notes in Computer Science*, pages 313–329. Springer, 2019.

[MN17] Bart Mennink and Samuel Neves. Encrypted Davies-Meyer and Its Dual: Towards Optimal Security Using Mirror Theory. In Katz and Shacham [KS17], pages 556–583.

[MT06] Kazuhiko Minematsu and Yukiyasu Tsunoo. Provably Secure MACs from Differentially-Uniform Permutations and AES-Based Implementations. In Matthew J. B. Robshaw, editor, *Fast Software Encryption, 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers*, volume 4047 of *Lecture Notes in Computer Science*, pages 226–241. Springer, 2006.

[Nai18] Yusuke Naito. On the Efficiency of ZMAC-Type Modes. In Jan Camenisch and Panos Papadimitratos, editors, *Cryptology and Network Security - 17th International Conference, CANS 2018, Naples, Italy, September 30 - October 3, 2018, Proceedings*, volume 11124 of *Lecture Notes in Computer Science*, pages 190–210. Springer, 2018.

[SB12] Marcos A. Simplício Jr. and Paulo S. L. M. Barreto. Revisiting the Security of the ALRED Design and Two of Its Variants: Marvin and LetterSoup. *IEEE Trans. Inf. Theory*, 58(9):6223–6238, 2012.

[SBB+09] Marcos A. Simplício Jr., Pedro d'Aquino F. F. S. Barbuda, Paulo S. L. M. Barreto, Tereza Cristina M. B. Carvalho, and Cintia B. Margi. The MARVIN message authentication code and the LETTERSOUP authenticated encryption scheme. *Security and Communication Networks*, 2(2):165–180, 2009.

[Sho96] Victor Shoup. On Fast and Provably Secure Message Authentication Based on Universal Hashing. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 313–328. Springer, 1996.

[Sta78] A. J. Stam. Distance between sampling with and without replacement. *Statistica Neerlandica*, 32(2):81–91, 1978.

[SWG21] Yaobin Shen, Lei Wang, and Dawu Gu. LedMAC: More Efficient Variants of LightMAC. Cryptology ePrint Archive, Report 2021/1210, 2021. https://eprint.iacr.org/2021/1210.

[WC81] Mark N. Wegman and Larry Carter. New Hash Functions and Their Use in Authentication and Set Equality. *J. Comput. Syst. Sci.*, 22(3):265–279, 1981.

A Note on Security Analysis of MARVIN

A simplified depiction of MARVIN is given in Figure 3. In this figure, R is a subkey derived as $E_K(c) \oplus c$ for some constant c. Data is transformed by first adding $R \cdot (x^w)^i$ for some finite field generator x^w and message index i and then feeding the value through a square complete transform Θ. This, in turn, is a primitive on n bits such that any differential over Θ has at least n/w active S-boxes. The outcomes are then added, the value is blinded with R and the length of the message and tag, and the resulting checksum is fed through E_K.

In the security proof of MARVIN [SB12], the property that Θ is a square complete transform is merely interpreted as the fact that the function $H : M_i \mapsto \Theta(M_i \oplus R \cdot (x^w)^i)$ has differential property at most p:

$$\forall i\,, \forall M_i \neq M_i'\,, \forall C \,:\, \mathbf{Pr}\left(\Theta(M_i \oplus R \cdot (x^w)^i) \oplus \Theta(M_i' \oplus R \cdot (x^w)^i) = C\right) \leq p\,. \tag{20}$$

I.e., the authors basically argue based on the XOR-universality of H. This does not become clear from the definition of a square complete transform, but rather from the proof itself and from the statement of Theorem 1 of [SB12].[3] A specific part of the security analysis of MARVIN, i.e., Section V.D of [SB12], considers zero-sums. First, it considers the case that two evaluations of MARVIN differ in exactly two positions. In this case, one might indeed argue from the XOR-universality of H. However, then, the authors consider the case where two blocks are modified. In other words, one looks at collisions of the form

$$H(M_i) \oplus H(M_i') \oplus H(M_j) \oplus H(M_j') = 0\,,$$

where $i \neq j$ and $(M_i, M_j) \neq (M_i', M_j')$. Inspired by (20), the authors [SB12] maximize over the choices of M_i, M_j and consider the plain differential probability

$$H(M_i') \oplus H(M_j') = H(M_i) \oplus H(M_j)\,. \tag{21}$$

However, this requires one to maximize over the choice of M_i, M_j and to multiply the bound by a factor 2^{2n}.

A.1 Attack with Keyed Θ

We can use this observation in a constructive way to design a function Θ for which H is XOR-universal as requested by Theorem 1 and its proof of [SB12], but for which the security of MARVIN breaks badly. As a first attempt, we select a *keyed* Θ. We admit that this is unfair, as the authors of MARVIN clearly state that Θ is unkeyed.

For our attack, we instantiate Θ as multiplication with secret key L:

$$\Theta(X) = L \cdot X\,.$$

[3] To detail: the theorem statement only states that Θ must be XOR-universal, the the proof considers XOR-universality of H.

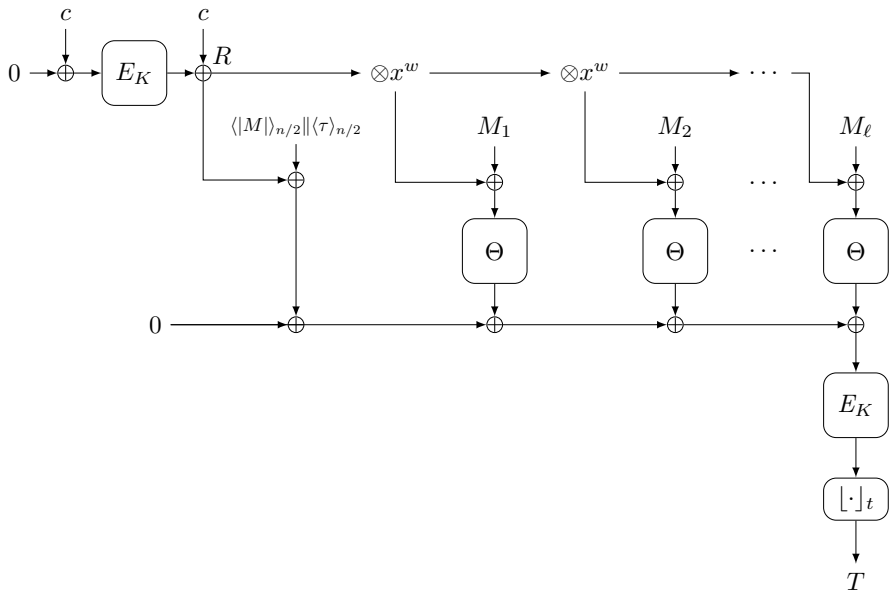

Figure 3: MARVIN message authentication [SBB+09]. Here, $\Theta : \{0,1\}^n \to \{0,1\}^n$ is a square complete transform and $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. Not depicted is the injective padding of an arbitrarily length message M into ℓ n-bit blocks $M_1, \ldots, M_\ell$.

Clearly, above-defined function H fulfills the XOR-universality of (20) for $p = 1/2^n$. However, an adversary can make (21) happen with probability 1 by taking $M_i = M'_j$ and $M_j = M'_i$:

$$\begin{aligned}
&H(M_i) \oplus H(M'_i) \oplus H(M_j) \oplus H(M'_j) \\
&= L \cdot (M_i \oplus R \cdot (x^w)^i) \oplus L \cdot (M'_i \oplus R \cdot (x^w)^i) \oplus L \cdot (M_j \oplus R \cdot (x^w)^j) \oplus L \cdot (M'_j \oplus R \cdot (x^w)^j) \\
&= L \cdot \Big(M_i \oplus R \cdot (x^w)^i \oplus M'_i \oplus R \cdot (x^w)^i \oplus M_j \oplus R \cdot (x^w)^j \oplus M'_j \oplus R \cdot (x^w)^j \Big) \\
&= 0 \,.
\end{aligned}$$

A.2 Attack with Unkeyed Θ

One may argue that the above attack is cheating by using a keyed linear hash for Θ. Clearly, the design states that Θ is unkeyed, and presumably based on some non-linear block cipher building block. It is, nevertheless, possible to create a variant of MARVIN that respects its spirit and nevertheless is hopelessly broken. Let Θ be

$$\Theta(X) = X^7 \,,$$

over $F_{2^{128}}$. This Θ is both a permutation since $\gcd(7, 2^{128} - 1) = 1$, is 6-differentially uniform [BCC10, Theorem 6], and it is non-linear; its algebraic degree is, in fact, 3. Therefore, by the security proof of MARVIN [SB12], this seems to be a perfectly acceptable Θ to employ.

Now let the masking be defined as in PMAC: $\gamma_i \cdot R$, where γ_i is the i-th Gray codeword. Although this is not the MARVIN masking, it again does not seem to contradict any requirement of the proof. As pointed out by Luykx et al. [LPSY16], a sequence of $2^k - 1$ sequential Gray codewords contains a subgroup of size 2^{k-1}. This property, combined with

the low algebraic degree of Θ, can be used to force a short sequence blocks to sum to zero and forge messages.

In particular, the 10-block message pair

$$\begin{aligned} M &= (\alpha, \alpha, \alpha, \alpha, \alpha, \alpha, \alpha, \alpha, \alpha, \alpha) \\ M' &= (\alpha, \alpha, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta)\,, \end{aligned}$$

where α is an arbitrary constant and $\Delta \neq 0$, collides with probability 1. If the non-bijective but 2-differentially uniform $\Theta(X) = X^3$ would be used,

$$\begin{aligned} M &= (\alpha, \alpha, \alpha, \alpha) \\ M' &= (\alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta, \alpha \oplus \Delta) \end{aligned}$$

suffices.