Multimixer-128: Universal Keyed Hashing Based on Integer Multiplication

Koustabh Ghosh[1], Parisa Amiri Eliasi[1] and Joan Daemen[1]

Digital Security Group, Radboud University, Nijmegen, the Netherlands
{koustabh.ghosh,parisa.amirieliasi,joan.daemen}@ru.nl

Abstract. In this paper we introduce a new keyed hash function based on 32-bit integer multiplication that we call Multimixer-128. In our approach, we follow the key-then-hash parallel paradigm. So, we first add a variable length input message to a secret key and split the result into blocks. A fixed length public function based on integer multiplication is then applied on each block and their results are added to form the digest. We prove an upper bound of 2^{-127} for the universality of Multimixer-128 by means of the differential probability and image probability of the underlying public function.
There are vector instructions for fast 32-bit integer multiplication on many CPUs and in such platforms, Multimixer-128 is very efficient. We compare our implementation of Multimixer-128 with NH hash function family that offers similar levels of security and with two fastest NIST LWC candidates. To the best of our knowledge, NH hash function is the fastest keyed hash function on software and Multimixer-128 outperforms NH while providing same levels of security.

Keywords: Keyed Hashing · Parallel Construction · Multimixer-128

1 Introduction

Keyed hashing is a class of cryptographic primitives that compresses variable-length messages into fixed-length digests, using a secret key. The security of keyed hashing can be measured in terms of its ε-universality and ε-Δuniversality [Sti95]. The ε-universality of a keyed hash function $F_{\mathbf{K}}$ upper bounds the probability, taken over the key space and for an optimal attacker, to generate collisions at the output of $F_{\mathbf{K}}$: Finding $\mathbf{M}$ and $\mathbf{M}^*$ such that $F_{\mathbf{K}}(\mathbf{M}) = F_{\mathbf{K}}(\mathbf{M}^*)$. ε-Δuniversality is a natural generalization of the ε-universality as it upper bounds the success probability, taken over the key space and for an optimal attacker, to find a particular output difference Δ at the output of $F_{\mathbf{K}}$: Finding $\mathbf{M}$ and $\mathbf{M}^*$ such $F_{\mathbf{K}}(\mathbf{M}) - F_{\mathbf{K}}(\mathbf{M}^*) = \Delta$.

Keyed hash functions exhibiting good uniformity can be used to construct secure Message Authentication Code (MAC) functions [WC81] and doubly extendable cryptographic keyed (Deck) functions [BDH$^+$17]. Deck functions generalize both MAC functions and stream ciphers by supporting both variable-length inputs and outputs and can be used to build authenticated encryption schemes and wide block ciphers.

There are various approaches to building keyed hash functions: They can be built as modes of strong cryptographic primitive, like constructions based on cryptographic hash functions such as HMAC [BCK96] and NMAC [BCK96], or block ciphers such as CBC-MAC [BKR94], CMAC [BR00, IK03, 80005] and PMAC [BR02]. A more efficient way to build keyed hash functions is by using iterated finite field multiplication using Horner's rule. Examples include GHASH [MV04], the function used to compute a MAC in GCM mode and Poly-1305 [Ber05]. GHASH and Poly-1305 operate over fields $\mathbb{F}_{2^{128}}$ and

$\mathbb{F}_{2^{130}-5}$ respectively and consequently, an important cost in these functions is the modular reduction that is part of the finite field multiplication.

In order to design efficient keyed hash functions in software, the authors of [HK97] take an interesting approach. They take as starting point the MMH [CW79] and NMH family of hash functions that are based on multiplication over a finite field with prime elements. The authors modify these algorithms such that instead of field multiplication, they can make use of integer multiplication without any reductions and only need to do the modular reduction once at the very end of the algorithm. They choose the prime to be $2^{32} + 15$ and are able to implement these functions using division-less modular reductions making them more efficient. Black et al. further simplified these designs to build a remarkable family of keyed hash functions called NH hash function family [BHK+99] used in UMAC [BHK+99], Adiantum [CB18] and HS1-SIV [Kro15]. This function makes use of integer multiplication of 32-bit integers and thus all modular reductions are simple truncation to 32 or 64 bits. The NH hash function takes as input a variable-length message and a key with length the maximum message length supported. Both the message and key are first split into 32-bit blocks and then added block by block modulo 2^{32}. These resultant integers are multiplied in pairs, and these products are summed modulo 2^{64} to produce a 64-bit digest. This function is reported as 2^{-32}-Δuniversal, restricting to collisions between messages of equal length.

To decrease the universality to 2^{-128}, they make use of the Toeplitz-NH denoted as NH^{T}. NH^{T} applies the NH function 4 times to the input with 4 different keys and concatenates the outputs, resulting in a 256-bit digest. In this paper we investigate whether we can build a keyed hash function that is more efficient than NH^{T} for the same level of security.

1.1 Our Contribution

We propose a new keyed hash function as an alternative to NH^{T} that we call *Multimixer-128*. In order to build this secure and efficient keyed hash function, we adopt the framework of [FRD23] that was applied in [GFAD23] using finite field multiplication.

To study the security of Multimixer-128, we thoroughly analyze the differential properties of integer multiplication, where the inputs are w-bit integers. We prove that Multimixer-128 is an ε-Δuniversal hash function with $\varepsilon = 2^{-127}$ and show that our function is faster than the NH^{T} hash function on a typical target CPU.

The ε achieved by Multimixer-128 is twice that reported for the NH^{T} universal hash function. Still Multimixer-128 achieves $\varepsilon = 2^{-128}$ when restricted to collisions between messages of equal length. For messages of unequal length the universality of NH^{T} turns out to be only $\varepsilon = 2^{-124}$ (see Sect. 6). The designers of NH^{T} avoid collisions between messages of different length by defining a mode on top of NH^{T} that applies NH^{T} on fixed-length inputs and NH^{T} used with this mode attains the universality of 2^{-128} over messages of all string length.

Multimixer-128 can be used as the compression phase of a MAC function. It has a digest of 512-bits, which requires further compression when a 128-bit tag is required. NH^{T} with its digest size of 256-bits also has this requirement. In various current NH^{T} applications, e.g., in Adiantum [CB18] and HS1-SIV [Kro15], that is indeed the case. Furthermore, Multimixer-128 can be used as the compression phase in the Farfalle construction [BDH+17] to build DECK functions. The 512-bit digest does not cause any problems in that case as a wide cryptographic permutation can be used for the expansion phase, e.g., the 1600-bit permutation KECCAK-f [BDH+08] or the 512-bit permutation in ChaCha [Berrs].

1.2 Outline of the Paper

The paper is organized as follows. In Sections 2 and 3 we remind the readers of keyed-hash-functions, their universalities and how a public function can be parallelized to form such a keyed-hash-function [GFAD23]. We then introduce the notations that will be used throughout the paper and then in Section 4, take a look at the differential properties of the w-bit multiplication. In Section 5 we look at our first proposed public function $\mathcal{F}$-128, whose parallelization is called Multimixer-128. We look at the security of this construction and concretely prove that it is 2^{-127}-Δuniversal. In Section 6 we briefly talk about the NH hash function family and compare its security to Multimixer-128. Finally in Section 8 we report on our implementation of Multimixer-128 and compare its efficiency with NH hash function.

2 Preliminaries and Notations

A public function is denoted as $f \colon G \to G'$, where G and G' are abelian groups $\langle G, + \rangle$ and $\langle G', + \rangle$. The elements of G are called *blocks*. The set containing ℓ-block string is denoted as G^ℓ, i.e., $G^\ell = \{(x_0, x_1, \ldots, x_{\ell-1}) \mid x_i \in G \text{ for each } i = 0, 1, \ldots, \ell - 1\}$. The set of strings of length 1 upto κ is denoted as $\mathrm{BS}(G, \kappa) = \cup_{\ell=1}^{\kappa} G^\ell$. We denote strings in bold uppercase letters, like $\mathbf{M}$, its blocks by M_i, where indexing starts from 0 and the length of that string by $|\mathbf{M}|$. Given any set S, the cardinality of that set is denoted by $\#S$.

2.1 ε and ε-Δuniversality

Let $F_\mathbf{K}$ denote a keyed hash function where the key $\mathbf{K}$ is sampled uniformly at random from the key space. The security of $F_\mathbf{K}$ is measured by the probability of generating a collision at the output of $F_\mathbf{K}$ or more strongly, by the probability of two distinct inputs strings exhibiting a specific output difference. These probabilities are upper bounded respectively by the ε-universality and the ε-Δuniversality of the keyed hash function.

Definition 1 (ε-universality [Sti95])**.** A keyed hash function F is said to be ε-universal if for any distinct strings $\mathbf{M}, \mathbf{M}^*$

$$\Pr[F_\mathbf{K}(\mathbf{M}) = F_\mathbf{K}(\mathbf{M}^*)] \leq \varepsilon \, .$$

Definition 2 (ε-Δuniversality [Sti95])**.** A keyed hash function F is said to be ε-Δuniversal if for any distinct strings $\mathbf{M}, \mathbf{M}^*$ and for all $\Delta \in G$

$$\Pr[F_\mathbf{K}(\mathbf{M}) - F_\mathbf{K}(\mathbf{M}^*) = \Delta] \leq \varepsilon \, .$$

2.2 Key-then-hash Functions

Key-then-hash functions are a special type of keyed hash functions. They take as input elements of $\mathrm{BS}(G, \kappa)$ and return an element of G'. The keys are elements of G^κ. When processing an input, the key is first added to the input and then an unkeyed function is applied to the result. A key-then-hash function is defined as: $F \colon \mathrm{BS}(G, \kappa) \to G'$ with $F_\mathbf{K}(\mathbf{M}) := F(\mathbf{K} + \mathbf{M})$. The addition of two strings $\mathbf{M} = (M_0, M_1, \ldots, M_{|\mathbf{M}|-1})$ and $\mathbf{M}^* = (M_0^*, M_1^*, \ldots, M_{|\mathbf{M}^*|-1}^*)$ with $|\mathbf{M}| \leq |\mathbf{M}^*|$ is defined as $\mathbf{M}' := \mathbf{M} + \mathbf{M}^* = (M_0 + M_0^*, M_1 + M_1^*, \ldots, M_{|\mathbf{M}|-1} + M_{|\mathbf{M}|-1}^*)$ with $|\mathbf{M}'| = |\mathbf{M}|$. In Section 3 we demonstrate how to build such functions using a public function as the underlying primitive [GFAD23].

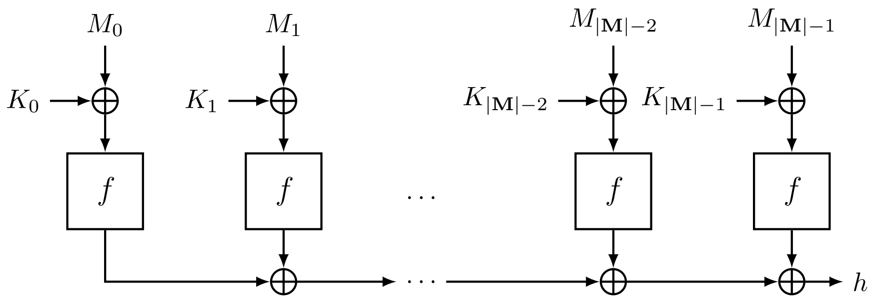

Figure 1: The parallelization Parallel $[f]$ adapted from [FRD23].

3 Parallel Universal Hashing

The parallelization of a public function to build key-then-hash function is described in Algorithm 1 and depicted in Figure 1 [GFAD23]. The construction takes as parameters a public function $f\colon G \to G'$ and a maximum string length κ. The inputs to the construction are a key $\mathbf{K} \in G^{\kappa}$ and a string $\mathbf{M} \in \mathrm{BS}(G, \kappa)$. The construction returns a digest $h \in G'$.

Algorithm 1: The parallelization Parallel $[f]$ [GFAD23]

```
Parameters: A public function $f\colon G \to G'$ and a maximum string length $\kappa$
Inputs    : A key $\mathbf{K} \in G^{\kappa}$ and a message $\mathbf{M} \in \mathrm{BS}(G, \kappa)$
Output    : A digest $h \in G'$
$\mathbf{X} \leftarrow \mathbf{M} + \mathbf{K}$
$h \leftarrow 0$
for $i \leftarrow 0$ to $|\mathbf{M}| - 1$ do
    $h \leftarrow h + f(X_i)$
end
return $h$
```

Given any public function f, its parallelization is the key-then-hash function denoted as Parallel $[f]$. The key space of Parallel $[f]$ is G^{κ} and as such we assume the existence of long keys with independent key blocks. This is a commonly made assumption and other keyed hash functions including NH^{T} also require long keys. In fact such a long key can be precomputed from a short secret key by any Pseudorandom generator (PRG) and this computation has to be made only when a change of key is required.

The universality of Parallel $[f]$ is upper-bounded by the propagation probabilities of the underlying fixed length function f, which are defined as follows.

A differential defined over f is the tuple (A, Δ), where $A \in G/\{0\}$ is called the input difference and $\Delta \in G'$ is called the output difference. We now remind the reader of differential probability of a differential over fixed-length public functions.

Definition 3 (Differential probability)**.** Let $f\colon G \to G'$ be a public function. The differential probability of a differential (A, Δ) of f, denoted as $\mathsf{DP}_f(A, \Delta)$, is:

$$\mathsf{DP}_f(A, \Delta) = \frac{\#\{X \in G \mid f(X + A) - f(X) = \Delta\}}{\#G} .$$

We say that input difference A propagates to output difference Δ with probability $\mathsf{DP}_f(A, \Delta)$.

Definition 4 (Solution set)**.** Given any public function f, the solution set of a differential (A,Δ) to f denoted as $\mathrm{S}_f(A,\Delta)$ is

$$\mathrm{S}_f(A,\Delta) = \{X \in G \mid f(X+A) - f(X) = \Delta\} \ .$$

Definition 5 (Differential weight)**.** Let $f\colon G \to G'$ be a public function. The differential weight of a differential (A,Δ) of f denoted as $\mathrm{w}_f(A,\Delta)$ is:

$$\mathrm{w}_f(A,\Delta) = -\log_2(\mathsf{DP}_f(A,\Delta)) = \log_2(\#G) - \log_2(\#\mathrm{S}_f(A,\Delta)) \ .$$

The relative frequency of outputs of a non-bijective public function f is not necessarily constant. As such, the *image probability* of an output of a public function is defined as follows.

Definition 6 (Image probability [GFAD23])**.** Let $f\colon G \to G'$ be a public function. The image probability of an output $Z \in G'$ of f, denoted as $\mathrm{IP}_f(Z)$, is the number of inputs that f maps to Z divided by the total number of possible inputs, namely,

$$\mathrm{IP}_f(Z) = \frac{\#\{X \in G \mid f(X) = Z\}}{\#G} \ .$$

The maximum possible value of DP_f and IP_f over all differentials and outputs of the underlying fixed length public function f respectively are denoted as:

$$\mathrm{MDP}_f = \max_{(A,\Delta)} \mathsf{DP}_f(A,\Delta) \quad \text{and} \quad \mathrm{MIP}_f = \max_{Z} \mathrm{IP}_f(Z) \ .$$

Theorem 1 (Theorem 1 [GFAD23])**.** *The parallelization of a public function f, Parallel$[f]$, is* $\max\{\mathrm{MDP}_f, \mathrm{MIP}_f\}$*-*$\Delta$*universal.*

Thus the problem of bounding universality of Parallel $[f]$ is reduced to bounding MDP_f and MIP_f of the underlying fixed length function f. The tightness of the ε-Δuniversality bound in Theorem 1 depends solely on the tightness of the bounds for MDP_f and MIP_f of the underlying public function f.

3.1 Notations

In this paper, $\mathbb{Z}/2^w\mathbb{Z}$ denotes the ring of integer residues modulo 2^w with the addition and multiplication. $(\mathbb{Z}/2^w\mathbb{Z})^n$ denotes the cartesian product of $\mathbb{Z}/2^w\mathbb{Z}$ n-times. For the public functions proposed in this paper, $G = (\mathbb{Z}/2^w\mathbb{Z})^8$ and $G' = \left(\mathbb{Z}/2^{2w}\mathbb{Z}\right)^8$ with $w = 32$. So the input and the output to our public function are both 8-tuples, where each element of the input tuple and output tuple are elements of $\mathbb{Z}/2^w\mathbb{Z}$ and $\mathbb{Z}/2^{2w}\mathbb{Z}$ and we call them as input word and output word respectively.

We represent $X \in G = (\mathbb{Z}/2^w\mathbb{Z})^8$ as $X = (x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3)^\intercal$. For simplicity, we slightly abuse the notations to denote $X = (\mathbf{x}, \mathbf{y})$, where $\mathbf{x} = (x_0, x_1, x_2, x_3)^\intercal \in (\mathbb{Z}/2^w\mathbb{Z})^4$ and $\mathbf{y} = (y_0, y_1, y_2, y_3)^\intercal \in (\mathbb{Z}/2^w\mathbb{Z})^4$. Similarly input differences and key blocks are denoted as $A = (\mathbf{a}, \mathbf{b})$ and $K = (\mathbf{h}, \mathbf{k})$ respectively. An output $Z \in G' = \left(\mathbb{Z}/2^{2w}\mathbb{Z}\right)^8$ is denoted as $Z = (z_0, z_1, \ldots, z_7)^\intercal$ and similarly an output difference Δ is given by $\Delta = (\delta_0, \delta_1, \ldots, \delta_7)^\intercal$. This means that throughout this paper for $i \in \{0,1,2,3\}$, each of $x_i, y_i, a_i, b_i, h_i, k_i \in \mathbb{Z}/2^w\mathbb{Z}$ and for $i \in \{0,1,\ldots,7\}$, each of $z_i, \delta_i \in \mathbb{Z}/2^{2w}\mathbb{Z}$.

For $n \geq 1$, the number of non-zero components of a vector $\mathbf{x} \in (\mathbb{Z}/2^w\mathbb{Z})^n$ is its hamming weight denoted as $\mathrm{w}(\mathbf{x})$. We further denote $(0,0,\ldots,0) \in (\mathbb{Z}/2^w\mathbb{Z})^n$ as 0^n. $\mathbb{Z}_{\geq 0}$ is used to denote the set of positive integers including 0.

For two elements $x, y \in \mathbb{Z}/2^w\mathbb{Z}$, $x \boxplus y$, $x \boxminus y$ and $x \cdot y$ denote respectively $(x+y) \bmod 2^w$, $(x-y) \bmod 2^w$ and $(x \cdot y) \bmod 2^w$. For any element $x \in \mathbb{Z}/2^w\mathbb{Z}$, $\overline{x}$ denotes the additive inverse of x, i.e., $\overline{x} = 2^w \boxminus x$.

The integer multiplication of two elements of $\mathbb{Z}/2^w\mathbb{Z}$ is called the *w-bit multiplication* and is denoted as $M[w]$. This operation is clearly not closed in $\mathbb{Z}/2^w\mathbb{Z}$ and is defined as

$$M[w]\colon\ (\mathbb{Z}/2^w\mathbb{Z})^2 \to \mathbb{Z}/2^{2w}\mathbb{Z}\colon\ M[w](x,y) = x \times y\ . \tag{1}$$

Example 1. Let $w = 4$. Then $M[w](5,6) = 5 \times 6 = 30$.

To differentiate between w-bit multiplication and the ring multiplication in $\mathbb{Z}/2^w\mathbb{Z}$, $\times$ will be used to denote the w-bit multiplication and $\cdot$ will be used to denote the ring multiplication throughout this paper.

We now take a brief look at the differential properties of the w-bit multiplication.

4 Differential Properties of w-bit Multiplication

$M[w]$ is a binary operation in $\mathbb{Z}/2^w\mathbb{Z}$. To that end an input difference to $M[w]$ has the form $A = (a,b)$, where $a, b \in \mathbb{Z}/2^w\mathbb{Z}$. The co-domain of $M[w]$ is $\mathbb{Z}/2^{2w}\mathbb{Z}$ and thus output difference $\delta \in \mathbb{Z}/2^{2w}\mathbb{Z}$. Naturally the solution set and DP of a differential $((a,b),\delta)$ to the w-bit multiplication are denoted as $\mathrm{S}_{M[w]}((a,b),\delta)$ and $\mathrm{DP}_{M[w]}((a,b),\delta)$ respectively. To simplify notations, we denote $\mathrm{N}((a,b),\delta) = \#\mathrm{S}_{M[w]}((a,b),\delta)$. For the sake of further notational simplicity we will use $\mathrm{DP}((a,b),\delta)$ and $\mathrm{S}((a,b),\delta)$ without any subscript in this section. Now, $\mathrm{S}((a,b),\delta)$ is given by:

$$\mathrm{S}((a,b),\delta) = \{(h,k) \in (\mathbb{Z}/2^w\mathbb{Z})^2 \mid ((a \boxplus h) \times (b \boxplus k) - h \times k) \bmod 2^{2w} = \delta\}\ . \tag{2}$$

Clearly $\mathrm{DP}((a,b),\delta) = \frac{\mathrm{N}((a,b),\delta)}{2^{2w}}$.

Corollary 1. *For any differential $((a,b),\delta)$ to $M[w]$, $\mathrm{DP}((a,b),\delta)$ is symmetric in the components of its input difference. So, $\mathrm{DP}((a,b),\delta) = \mathrm{DP}((b,a),\delta)$.*

Proof. The proof follows from (2) and the commutativity of $M[w]$. □

Obtaining the cardinality of $\mathrm{S}((a,b),\delta)$ for an input difference (a,b) with $a = 0$ or $b = 0$ is an interesting case and requires special attention.

Definition 7 (Unilateral and bilateral differentials)**.** For a pair of inputs from $(\mathbb{Z}/2^w\mathbb{Z})^2$, let their input difference be $(a,b) \neq (0,0)$. When (a,b) is such that $a = 0$ or $b = 0$, we call (a,b) an unilateral difference. Otherwise we call (a,b) a bilateral difference and any differential to $M[w]$ with a unilateral input difference is called a unilateral differential, while a differential to $M[w]$ with a bilateral difference is called a bilateral differential.

Due to Corollary 1 it suffices to only look at unilateral differentials of the form $((a,0),\delta)$.

Lemma 1. *For a unilateral differential $((a,0),\delta)$ to $M[w]$ with $\delta \neq 0$, we have*

$$\begin{aligned}
&\textit{For } \delta < 2^w a: && \mathrm{DP}((a,0),\delta) = \begin{cases} \frac{\overline{a}}{2^{2w}} & , \textit{ if } a \mid \delta \\ 0 & , \textit{ otherwise} \end{cases} \\
&\textit{For } \delta > 2^w a: && \mathrm{DP}((a,0),\delta) = \begin{cases} \frac{a}{2^{2w}} & , \textit{ if } \overline{a} \mid 2^{2w} - \delta \\ 0 & , \textit{ otherwise} \end{cases} \\
&\textit{For } \delta = 2^w a: && \mathrm{DP}((a,0),\delta) = 0\ .
\end{aligned}$$

Proof. For an input difference $(a,0)$, (2) converts into

$$((a \boxplus h) \times k - h \times k) \bmod 2^{2w} = \delta\ .$$

After modular reduction, there are two cases namely

$$h < \overline{a}: \quad a \times k = \delta \,, \tag{3}$$

$$h \geq \overline{a}: \quad -\overline{a} \times k + 2^{2w} = \delta \,. \tag{4}$$

The solutions to (3) and (4) are positive integers smaller than 2^w. When $h < \overline{a}$, $a \times k = \delta$ has at most one solution and that solution exists iff $a \mid \delta$ such that $\delta / a < 2^w$, i.e., $\delta < 2^w a$. Similarly for $h \geq \overline{a}$, $-\overline{a} \times k + 2^{2w} = \delta$ has at most one solution and that solution exists when $\overline{a} \mid 2^{2w} - \delta$ such that $(2^{2w} - \delta)/\overline{a} < 2^w$, i.e., $\delta > 2^w a$. Since $\delta < 2^w a$ and $\delta > 2^w a$ cannot occur simultaneously, we arrive at the lemma.

□

Lemma 2. *For a unilateral differential* $((a, 0), 0)$ *to* $M[w]$, $\mathrm{DP}((a, 0), 0) = \frac{1}{2^w}$.

Proof. For the unilateral differential $((a, 0), 0)$ to $M[w]$, (2) transforms into

$$(a \boxplus h) \times k = h \times k \,.$$

This equation is satisfied iff $k = 0$. Hence $\mathrm{S}((a, b), 0) = \{(h, 0) \mid h \in \mathbb{Z}/2^w\mathbb{Z}\}$, i.e., $\mathrm{N}((a, 0), 0) = 2^w$. Thus $\mathrm{DP}((a, 0), 0) = \frac{1}{2^w}$. □

We now focus on bilateral differentials. Given any δ, obtaining $\mathrm{S}((a, b), \delta)$ from (2) involve modular reduction depending on whether $h + a < 2^w$ and whether $k + b < 2^w$. We deal with these reductions by partitioning the domain in four parts that we denote as *quadrants I,II,III* and *IV*. We describe them along with the simplified form of (2) in Table 1.

Table 1: The Quadrants corresponding to bilateral differential $((a, b), \delta)$.

Quadrant	Domain of quadrants	Reduced form of (2) modulo 2^w
I	$h \in [0, \overline{a}), k \in [0, \overline{b})$	$b \times h + a \times k + a \times b = \delta$
II	$h \in [0, \overline{a}), k \in [\overline{b}, 2^w)$	$\left(-\overline{b} \times h + a \times k - a \times \overline{b}\right) \bmod 2^{2w} = \delta$
III	$h \in [\overline{a}, 2^w), k \in [0, \overline{b})$	$(b \times h - \overline{a} \times k - \overline{a} \times b) \bmod 2^{2w} = \delta$
IV	$h \in [\overline{a}, 2^w), k \in [\overline{b}, 2^w)$	$-\overline{b} \times h - \overline{a} \times k + \overline{a} \times \overline{b} + 2^{2w} = \delta$

For a given bilateral differential $((a, b), \delta)$ and $i \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}, \mathrm{IV}\}$, we use $\mathrm{S}^i((a, b), \delta)$ to denote $\mathrm{S}((a, b), \delta)$ restricted to quadrant i, i.e., $\mathrm{S}^i((a, b), \delta) = \mathrm{S}((a, b), \delta) \cap \text{Quadrant } i$.

We now depict the $\mathrm{S}((a, b), \delta)$ for a concrete case when $w = 4$, $a = 4$, $b = 8$ and $\delta = 208$ in Figure 2. Naturally $\overline{a} = 2^4 - 4 = 12$ and $\overline{b} = 2^4 - 8 = 8$. The horizontal axis represents h and the vertical axis represents k: The whole domain of $(\mathbb{Z}/2^w\mathbb{Z})^2$ is the grid of points with integer coordinates (h, k) . The quadrants are naturally rectangles as depicted in Figure 2. Now, each blue point in the figure is an element of $\mathrm{S}((4, 8), 208)$ for the 4-bit multiplication. Thus $\#\mathrm{S}((4, 8), 208) = 6$. We further see that $\mathrm{S}^{\mathrm{I}}((4, 8), 208) = \mathrm{S}^{\mathrm{IV}}((4, 8), 208) = \phi$ and for $i = \mathrm{II}, \mathrm{III}$, each element of $\mathrm{S}^i((4, 8), 208)$ lies on line segments reflecting the linearity of the equations within each quadrant.

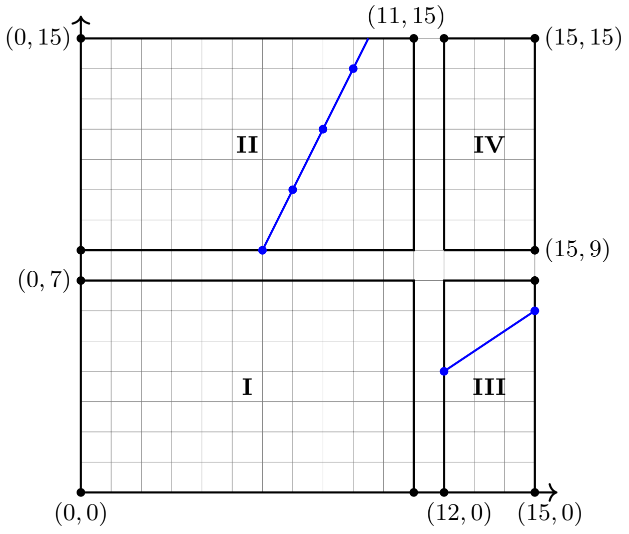

Figure 2: Solution set corresponding to the differential $((4,8),208)$ when $w = 4$.

Lemma 3. *Let $((a,b),\delta)$ be a bilateral differential of $M[w]$ .Then for $i \in \{I,II,III,IV\}$, $\mathrm{S}^i((a,b),\delta)$ denote straight line segments in $(\mathbb{Z}/2^w\mathbb{Z})^2$, whose slopes and maximum cardinalities are given by*

	Slope	Max $\#\mathrm{S}^i((a,b),\delta)$
$\mathrm{S}^{\mathrm{I}}((a,b),\delta)$	$-b/a$	$\left\lceil \gcd(a,b) \min\left(\frac{\overline{a}}{a},\frac{\overline{b}}{b}\right)\right\rceil$
$\mathrm{S}^{\mathrm{II}}((a,b),\delta)$	$\overline{b}/a$	$\left\lceil \gcd(a,\overline{b}) \min\left(\frac{\overline{a}}{a},\frac{b}{\overline{b}}\right)\right\rceil$
$\mathrm{S}^{\mathrm{III}}((a,b),\delta)$	$b/\overline{a}$	$\left\lceil \gcd(\overline{a},b) \min\left(\frac{a}{\overline{a}},\frac{\overline{b}}{b}\right)\right\rceil$
$\mathrm{S}^{\mathrm{IV}}((a,b),\delta)$	$-\overline{b}/\overline{a}$	$\left\lceil \gcd(\overline{a},\overline{b}) \min\left(\frac{a}{\overline{a}},\frac{b}{\overline{b}}\right)\right\rceil$

Proof. We prove this for $i =$I. For $(h,k) \in \mathrm{S}^{\mathrm{I}}((a,b),\delta)$, we see from Table 1 that $b \times h + a \times k + a \times b = \delta$, which denotes a straight line with slope $-b/a$ in $(\mathbb{Z}/2^w\mathbb{Z})^2$.

Every point on this line can be expressed as $(h+x, k-bx/a)$ for some x. This point has integer coordinates iff $a \mid bx$, or equivalently, if $a/\gcd(a,b) \mid x$. This means that these x coordinates of these points are at distances $d_a = a/gcd(a,b)$ from each other. Quadrant I has width $(\overline{a}-1)$ and can fit at most $\lceil \overline{a}/d_a \rceil$ points. The y coordinates of these points are at distances $d_b = b/gcd(a,b)$ from each other and hence quadrant I with its height of $(\overline{b}-1)$ can fit at most $\lceil \overline{b}/d_b \rceil$ points. Both restrictions apply and hence the number of points on a line is at most $\left\lceil \gcd(a,b) \min\left(\frac{\overline{a}}{a},\frac{\overline{b}}{b}\right)\right\rceil$.

The proofs are similar when $i =$II, III or IV. □

Lemma 4. *The solution set of a bilateral differential $((a,b),\delta)$ is fully in quadrants I and IV or in quadrants II and III.*

Proof. We will first show that the solution set must be empty in one of $\mathrm{S}^{\mathrm{I}}((a,b),\delta)$ and $\mathrm{S}^{\mathrm{II}}((a,b),\delta)$. Indeed if that were not the case, from Table 1 it follows that both the following equations must have a solution.

$$b \times h + a \times k + a \times b = \delta \qquad ,0 \leq h < \overline{a}, \quad 0 \leq k < \overline{b}\ , \tag{5}$$

$$\left(-\overline{b} \times h + a \times k - a \times \overline{b}\right) \bmod 2^{2w} = \delta \qquad ,0 \leq h < \overline{a}, \quad \overline{b} \leq k < 2^{w}\ . \tag{6}$$

Now, (6) after reduction modulo 2^{2w} can have one of the following forms

$$-\overline{b} \times h + a \times k - a \times \overline{b} = \delta\ , \tag{6.1}$$

$$-\overline{b} \times h + a \times k - a \times \overline{b} = \delta - 2^{2w}\ . \tag{6.2}$$

From (5) we have,

$$\begin{aligned} 0 \leq k < \overline{b} \implies \delta - a \times \overline{b} < \delta - a \times k \leq \delta &\implies \delta - a \times \overline{b} < b \times h + a \times b \leq \delta \\ &\implies \delta - 2^{w} \times a < b \times h \leq \delta - a \times b\ . \end{aligned} \tag{7}$$

Similarly from (5) we also have:

$$\delta - 2^{w} \times b < a \times k \leq \delta - a \times b\ . \tag{8}$$

From (6.1) we see that

$$\overline{b} \leq k < 2^{w} \implies -\delta \leq \overline{b} \times h < a \times b - \delta\ . \tag{9}$$

Since both $b \times h \geq 0$ and $\overline{b} \times h \geq 0$, $\delta \geq a \times b$ implies (9) cannot hold for any h and $\delta < ab$ implies (7) cannot hold for any h, Thus for all values of δ, (7) and (9) cannot hold simultaneously for any h.

Now from (6.2) we have

$$0 \leq h < \overline{a} \implies \delta - 2^{2w} + a \times \overline{b} \leq a \times k < \delta - 2^{w} \times b\ . \tag{10}$$

But this implies that (8) and (10) cannot both hold simultaneously.

Hence (5) and (6) cannot have a common solution. Thus both $\mathrm{S}^{\mathrm{I}}((a,b),\delta)$ and $\mathrm{S}^{\mathrm{II}}((a,b),\delta)$ cannot be non-empty. It can similarly be shown that both $\mathrm{S}^{\mathrm{I}}((a,b),\delta)$ and $\mathrm{S}^{\mathrm{III}}((a,b),\delta)$ or $\mathrm{S}^{\mathrm{II}}((a,b),\delta)$ and $\mathrm{S}^{\mathrm{IV}}((a,b),\delta)$ or $\mathrm{S}^{\mathrm{III}}((a,b),\delta)$ and $\mathrm{S}^{\mathrm{IV}}((a,b),\delta)$ cannot be non-empty. □

Lemma 5. *Let $((a,b),\delta)$ be a bilateral differential to $M[w]$. Then*

$$\mathrm{N}((a,b),\delta) \leq \max\left(\left\lceil \gcd(a,b) \min\left(\tfrac{\overline{a}}{a}, \tfrac{\overline{b}}{b}\right)\right\rceil + \left\lceil \gcd(\overline{a},\overline{b}) \min\left(\tfrac{a}{\overline{a}}, \tfrac{b}{\overline{b}}\right)\right\rceil, \left\lceil \gcd(a,\overline{b}) \min\left(\tfrac{\overline{a}}{a}, \tfrac{b}{\overline{b}}\right)\right\rceil + \left\lceil \gcd(\overline{a},b) \min\left(\tfrac{a}{\overline{a}}, \tfrac{\overline{b}}{b}\right)\right\rceil\right).$$

Proof. We first note that,

$$\mathrm{S}((a,b),\delta) = \mathrm{S}^{\mathrm{I}}((a,b),\delta) \cup \mathrm{S}^{\mathrm{II}}((a,b),\delta) \cup \mathrm{S}^{\mathrm{III}}((a,b),\delta) \cup \mathrm{S}^{\mathrm{IV}}((a,b),\delta)\ .$$

By Lemma 4 it follows that for every differential $((a,b),\delta)$, one of $\mathrm{S}^{\mathrm{I}}((a,b),\delta) \cup \mathrm{S}^{\mathrm{IV}}((a,b),\delta)$ and $\mathrm{S}^{\mathrm{II}}((a,b),\delta) \cup \mathrm{S}^{\mathrm{III}}((a,b),\delta)$ must be empty. Thus we must have

$$\mathrm{N}((a,b),\delta) \leq \max\left(\#\mathrm{S}^{\mathrm{I}}((a,b),\delta) + \#\mathrm{S}^{\mathrm{IV}}((a,b),\delta), \#\mathrm{S}^{\mathrm{II}}((a,b),\delta) + \#\mathrm{S}^{\mathrm{III}}((a,b),\delta)\right).$$

The rest of the proof follows immediately from Lemma 3. □

For any input difference (a,b) to $M[w]$, Lemma 5 gives us an upper-bound for $\max_{\delta} \mathrm{DP}((a,b),\delta)$. This upper-bound is not tight for all input differences, but is still a reasonably good upper-bound. In fact in practice we only observed the difference between the upper bound obtained in Lemma 5 and $\max_{\delta} \mathrm{DP}((a,b),\delta)$ to be negligible with the difference being $\frac{2}{2^{2w}}$ at most.

Lemma 6. *For any bilateral differential* $((a,b),0)$ *to* $M[w]$*, we have*

$$\mathrm{N}((a,b),0) = \left\lceil \gcd\left(a,\overline{b}\right)\min\left(\frac{\overline{a}}{a},\frac{b}{\overline{b}}\right)\right\rceil + \left\lceil \gcd\left(\overline{a},b\right)\min\left(\frac{a}{\overline{a}},\frac{\overline{b}}{b}\right)\right\rceil .$$

Proof. We first note that it can be verified from Table 1 that $(0,\overline{b}) \in \mathrm{S}^{\mathrm{II}}((a,b),0)$, i.e., $\mathrm{S}^{\mathrm{II}}((a,b),0) \neq \varnothing$. Consequently from Lemma 4, $\mathrm{S}^{\mathrm{I}}((a,b),0) \cup \mathrm{S}^{\mathrm{IV}}((a,b),0) = \varnothing$. Thus,

$$\mathrm{S}((a,b),0) = \mathrm{S}^{\mathrm{II}}((a,b),\delta) \cup \mathrm{S}^{\mathrm{III}}((a,b),\delta) .$$

We now claim that $\#\mathrm{S}^{\mathrm{II}}((a,b),0) = \left\lceil \gcd(a,\overline{b})\min\left(\frac{\overline{a}}{a},\frac{b}{\overline{b}}\right)\right\rceil$. Indeed by Lemma 3, $\mathrm{S}^{\mathrm{II}}((a,b),0)$ denotes a line segment with slope $\overline{b}/a$. $(0,\overline{b})$ is one of the end points of the line segment since $(0,\overline{b})$ is one of the vertices of Quadrant II. Hence for any point $(x,y) \in \mathrm{S}^{\mathrm{II}}((a,b),0)$, $0 \le x < \overline{a}$, $\overline{b} \le y < 2^w$ and x is of the form $\frac{ai}{\gcd(a,\overline{b})}$, y is of the form $\overline{b} + \frac{\overline{b}j}{\gcd(a,\overline{b})}$ for some $i,j \in \mathbb{Z}_{\ge 0}$. Thus $(x,y) \in \mathrm{S}^{\mathrm{II}}((a,b),0)$ for all $i,j \in \mathbb{Z}_{\ge 0}$ whenever

$$0 \le \frac{ai}{\gcd\left(a,\overline{b}\right)} < \overline{a} \implies i < \left\lceil \gcd\left(a,\overline{b}\right)\frac{\overline{a}}{a}\right\rceil ,$$
$$\overline{b} \le \overline{b} + \frac{\overline{b}j}{\gcd\left(a,\overline{b}\right)} < 2^w \implies j < \left\lceil \gcd\left(a,\overline{b}\right)\frac{b}{\overline{b}}\right\rceil .$$

Thus we can conclude that that $\#\mathrm{S}^{\mathrm{II}}((a,b),0) = \left\lceil \gcd(a,\overline{b})\min\left(\frac{\overline{a}}{a},\frac{b}{\overline{b}}\right)\right\rceil$.

It can be similarly shown that $\#\mathrm{S}^{\mathrm{III}}((a,b),0) = \left\lceil \gcd\left(\overline{a},b\right)\min\left(\frac{a}{\overline{a}},\frac{\overline{b}}{b}\right)\right\rceil$. $\mathrm{S}^{\mathrm{II}}((a,b),0)$ and $\mathrm{S}^{\mathrm{II}}((a,b),0)$ are mutually disjoint and thus we arrive at our desired result. □

Corollary 2. *Let* $((a,b),0)$ *be a bilateral differential to* $M[w]$ *such that* $b = \overline{a}$*. Then* $\mathrm{N}((a,b),0) = 2^w$*.*

Proof. Substituting $b = \overline{a}$ in Lemma (6), we see that $\#\mathrm{S}(A,0) = 2^w$. □

We call differences of the form $(a,\overline{a})$ *counter-diagonal differences.* $\mathrm{N}((a,b),0) = 2^w$ only for these bilateral differences and all the unilateral differences. We also call differences of the form (a,a) the *diagonal differences.* Lemma 5 provides the upper-bound for DP of a differential with diagonal difference as $\max_\delta \mathrm{DP}((a,a),\delta) \le 2^{-w}$.

For an input difference (a,b), we are primarily interested in $\max_\delta \mathrm{DP}((a,b),\delta)$. Lemma 5 provides a good upper-bound for this value. Another differential of interest is $((a,b),0)$ since this differential corresponds to collision at the output of the multiplication.

Figure 3 shows the histogram of differential weight vs the number of input differences that attain that weight for some output difference for 16-bit multiplication , $M[16]$. Here a blue point at a coordinate (x,y) means that there are y input differences with $\mathrm{DP}((a,b),0) = x.2^{-32}$. Similarly a red point at a coordinate (x,y) means that there are y input differences with $\max_\delta \mathrm{DP}((a,b),\delta) \le x.2^{-32}$. So the red dots in the figure correspond to the bound of Lemma 5

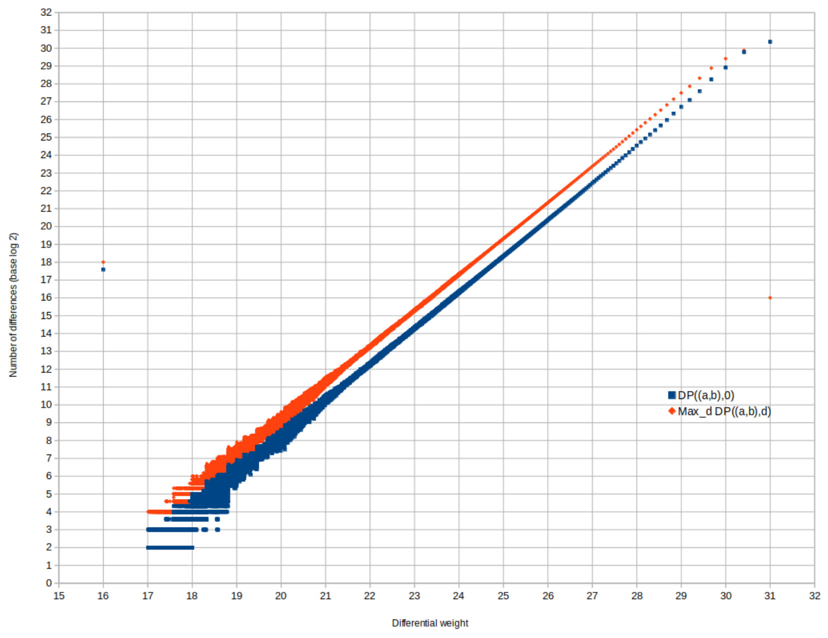

Figure 3: Upper-bound of $\max_\delta \mathrm{DP}((a,b),\delta)$ and $\mathrm{DP}((a,b),0)$-vs Number of differences for $w = 16$.

Figure 3 shows that there are exactly $3 \cdot (2^w - 1)$ differentials with zero output difference that have weight w. These are the unilateral differences and the counter-diagonal differences, i.e., input differences with shape $(a, 0)$, $(0, a)$ or $(a, \overline{a})$. Moreover, there are exactly $4 \cdot (2^w - 1)$ input differences for which the bound of Lemma 5 gives weight w. These are the $3 \cdot (2^w - 1)$ ones with output difference 0 and the diagonal differences with shape (a, a). Disregarding $(2^{w-1}, 2^{w-1})$, for the latter the bound of Lemma 5 is not tight: the output difference with highest DP is attained for $a = \overline{1}$ and $a = 1$ and for these differences, the maximum DP is $\frac{2^w - 1}{2^{2w}}$. From this histrogram, it is clear that while the maximum possible value of $\max_\delta \mathrm{DP}((a,b),\delta) = 2^{-w}$, for most of the differentials this value is actually much smaller. In fact, for about half of the differentials, $\max_\delta \mathrm{DP}((a,b),\delta) \leq \frac{3}{2^{2w}}$. These properties make integer multiplication an excellent choice to be used as a source of non-linearity in symmetric cryptographic functions.

5 Multimixer-128

We now introduce the key-then-hash function that we call Multimixer-128. It is the parallelization of the public function, that we denote as $\mathcal{F}$-128, i.e., Parallel $[\mathcal{F}\text{-}128]$ = Multimixer-128. While Multimixer-128 operates on 32-bit input words, we will prove the results in this section for a generic w unless otherwise specified.

The specification of $\mathcal{F}$-128 is provided in Algorithm 2, where all the additions in the indexes of $\mathbf{x}$ and $\mathbf{y}$ are done modulo 4. This function closely follows the multiply-transform-multiply construction [GFAD23] with the differences being:

- The domain and co-domain of $\mathcal{F}$-128 are $(\mathbb{Z}/2^{32}\mathbb{Z})^8$ and $(\mathbb{Z}/2^{64}\mathbb{Z})^8$ respectively, neither of which is a field.
- All the multiplications in $\mathcal{F}$-128 are integer multiplications.

In order to provide a design rationale for $\mathcal{F}$-128, we first define coordinate-wise product of vectors and circulant matrices.

Algorithm 2: The public function of Multimixer, $\mathcal{F}$-128

```
Inputs   : $X = (x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3)^\intercal \in \left(\mathbb{Z}/2^{32}\mathbb{Z}\right)^8$
Output   : $Z = (z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7)^\intercal \in \left(\mathbb{Z}/2^{64}\mathbb{Z}\right)^8$
for $i \leftarrow 0$ to 3 do
    $u_i \leftarrow x_i \boxplus x_{i+1} \boxplus x_{i+2}$
    $v_i \leftarrow y_i \boxplus y_{i+1} \boxplus y_{i+2}$
end
for $i \leftarrow 0$ to 3 do
    $z_i \leftarrow x_i \times y_i$
    $z_{i+4} \leftarrow u_i \times v_i$
end
$Z \leftarrow (z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7)$
return $Z$
```

Definition 8 (Coordinate-wise product)**.** Given $\mathbf{x} = (x_0, x_1, x_2, x_3)^\intercal, \mathbf{y} = (y_0, y_1, y_2, y_3)^\intercal \in (\mathbb{Z}/2^w\mathbb{Z})^4$, their coordinate wise product denoted as $\mathbf{x} \otimes \mathbf{y}$ is given by

$$\mathbf{x} \otimes \mathbf{y} = (x_0 \times y_0, x_1 \times y_1, x_2 \times y_2, x_3 \times y_3)^\intercal .$$

We let $\operatorname{circ}(\alpha_0, \alpha_1, \ldots, \alpha_{n-1})$ denote the $n \times n$ circulant matrix with $(\alpha_0, \alpha_1, \ldots, \alpha_{n-1})$ as its entries in the first row, where $\alpha_i \in \mathbb{Z}/2^w\mathbb{Z}$ for each $i \in \{0, 1, \ldots, n-1\}$. For example,

$$\operatorname{circ}(1,1,1,0) = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} .$$

We now look at an alternative definition of $\mathcal{F}$-128

Definition 9 (Alternative definition of $\mathcal{F}$-128)**.** $\mathcal{F}$-128 can alternatively be defined as:

$$\mathcal{F}\text{-}128\colon\ (\mathbb{Z}/2^w\mathbb{Z})^8 \to \left(\mathbb{Z}/2^{2w}\mathbb{Z}\right)^8 : \mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \otimes \mathbf{y}, \mathrm{N}_\alpha \cdot \mathbf{x} \otimes \mathrm{N}_\beta \cdot \mathbf{y}) \ , \tag{11}$$

where N_α and N_β are 4×4 circulant matrices: $\mathrm{N}_\alpha = \operatorname{circ}(1,1,1,0)$ and $\mathrm{N}_\beta = \operatorname{circ}(0,1,1,1)$.

By Algorithm 2 and following the notational convention of Section 3.1, we see that $\mathrm{N}_\alpha \cdot \mathbf{x} = \mathbf{u}$ and $\mathrm{N}_\beta \cdot \mathbf{y} = \mathbf{v}$. Thus $\mathcal{F}$-128$(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \otimes \mathbf{y}, \mathbf{u} \otimes \mathbf{v})$.

The branch number of a matrix plays an important role in our security analysis.

Definition 10 (Branch number [DR20])**.** Given a $n \times n$ matrix N defined over $\mathbb{Z}/2^w\mathbb{Z}$, its branch number is defined as $\min\limits_{\mathbf{x} \in (\mathbb{Z}/2^w\mathbb{Z})^n / \{0^n\}} (\mathrm{w}(\mathbf{x}) + \mathrm{w}(\mathrm{N} \cdot \mathbf{x}))$.

To look at the propagation of vectors under the circulant matrices inside $\mathcal{F}$-128, we define the cyclic shift operation over vectors of $(\mathbb{Z}/2^w\mathbb{Z})^4$. For $\mathbf{x} \in (\mathbb{Z}/2^w\mathbb{Z})^4$, we denote by $\tau(\mathbf{x}) \in (\mathbb{Z}/2^w\mathbb{Z})^4$ the vector obtained by shifting the entries of $\mathbf{x}$ by one position to the left cyclically. So for $\mathbf{x} = (x_0, x_1, x_2, x_3)^\intercal$, $\tau(\mathbf{x}) = (x_1, x_2, x_3, x_0)^\intercal$. Naturally for $r < 4$, τ^r corresponds to shifting the entries of a vector belonging to $(\mathbb{Z}/2^w\mathbb{Z})^4$ by r places to the left cyclically.

Definition 11 (Activity pattern)**.** Let $\mathbf{x} = (x_0, x_1, \ldots, x_{n-1})^\intercal \in (\mathbb{Z}/2^w\mathbb{Z})^n$. The activity pattern of $\mathbf{x}$ is given by the binary vector $(g_0, g_1, \ldots, g_{n-1})^\intercal \in \mathbb{F}_2^n$, where $\forall i \in \{0, 1, \ldots, n-1\}$

$$g_i = \begin{cases} 1 & , \text{ if } x_i \neq 0 \\ 0 & , \text{ if } x_i = 0 \ . \end{cases}$$

For $i \in \{0, 1, \ldots, n-1\}$, g_i as called the activity of the x_i.

Example 2. Let $w = 4$. Then $(2, 6, 0, 1)^\intercal \in (\mathbb{Z}/2^w\mathbb{Z})^4$ has the activity pattern $(1, 1, 0, 1)^\intercal$.

We will argue based on the activity patterns of vectors belonging to $(\mathbb{Z}/2^w\mathbb{Z})^8$ and look at the vector's propagation under linear maps. In general, activities at the output of $\mathcal{F}$-128 may depend on the concrete inputs. We denote such activities by $\otimes$.

Example 3. Let $\mathrm{N}_\alpha = \mathrm{circ}(1, 1, 1, 0)$. Then for a vector $\mathbf{x} \in (\mathbb{Z}/2^w\mathbb{Z})^4$ with activity pattern $(1, 1, 0, 0)^\intercal$, $\mathrm{N}_\alpha \cdot \mathbf{x}$ has activity pattern $(\otimes, 1, 1, \otimes)^\intercal$.

The circulant nature of N_α and N_β translates directly to activity patterns: cyclically shifting an input activity pattern results in the same cyclic shift of the output activity pattern. A 4×4 circulant matrix commutes with the mapping τ^r for any $r < 4$.

Lemma 7. *Let* N *be a* 4×4 *circulant matrix and for* $\mathbf{x} \in (\mathbb{Z}/2^w\mathbb{Z})^4$*, let the activity pattern of* $\mathrm{N} \cdot \mathbf{x}$ *be* $\mathbf{g}$*. Then for any* $r < 4$*, the activity pattern of* $\mathrm{N} \cdot (\tau^r(\mathbf{x}))$ *is* $\tau^r(\mathbf{g})$*.*

Proof. Due to N being circulant, we have $\mathrm{N} \cdot (\tau^r(\mathbf{x})) = \tau^r(\mathrm{N} \cdot \mathbf{x})$. Thus the activity pattern of $\mathrm{N} \cdot (\tau^r(\mathbf{x}))$ is $\tau^r(\mathbf{g})$. □

For a matrix N, We similarly denote by $\tau^r(\mathrm{N})$ the matrix obtained by cyclically shifting the columns of N by r places to the left.

When computing the DP of a differential (A, Δ) over $\mathcal{F}$-128, the number of multiplications that has a non-zero input difference plays a crucial role in determining $\mathrm{DP}_{\mathcal{F}\text{-}128}(A, \Delta)$.

Definition 12 (Active multiplication [GFAD23])**.** Let f be a public function that involves w-bit multiplications. A w-bit multiplication in f is said to be active corresponding to an input difference if that multiplication has a non-zero input difference in at least one of its multiplicands as a result of propagation of that difference inside f.

So, our objective is to maximize the minimum number of active multiplication over all possible input differences to $\mathcal{F}$-128. This means that we require the matrices N_α and N_β to have good diffusion properties. But, for efficiency we are limited to small matrix entries so that their computation can be done efficiently with vector addition and subtraction only. We realized this with circulant matrices $\mathrm{circ}(1, 1, 1, 0)$ and $\mathrm{circ}(0, 1, 1, 1)$.

Thanks to the symmetry in the matrices and the fact the $\mathrm{N}_\beta = \tau^3(\mathrm{N}_\alpha)$, the number of cases to consider in security analysis is relatively small.

We now look at the maximum image probability and differential probability of $\mathcal{F}$-128, which in turn determines the ε-Δuniversality of Multimixer-128.

5.1 Maximum Image Probability of $\mathcal{F}$-128

In this section we show that $\mathrm{MIP}_{\mathcal{F}\text{-}128} \leq \frac{1}{2^{127}}$ and it is obtained for 0^8.

Lemma 8. *For the public function* $\mathcal{F}$-128, $\mathrm{IP}_{\mathcal{F}\text{-}128}(0^8) = \frac{2^{4w+1}-1}{2^{8w}} \leq 2^{-127}$.

Proof. We show that $\mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) = 0^8$ if and only if $\mathbf{x} = 0^4$ or $\mathbf{y} = 0^4$.

We first show that if $\mathbf{x} = 0^4$ or $\mathbf{y} = 0^4$, $\mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) = 0^8$. Indeed if $\mathbf{x} = 0^4$, we must have $\mathbf{u} = 0^4$ and if $\mathbf{y} = 0^4$, we must have $\mathbf{v} = 0^4$. Thus if $\mathbf{x} = 0^4$ or $\mathbf{y} = 0^4$, $\mathbf{x} \otimes \mathbf{y} = 0^4$ and $\mathbf{u} \otimes \mathbf{v} = 0^4$; Consequently $\mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) = 0^8$.

We now show that when $\mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) = 0^8$, $\mathbf{x} = 0^4$ or $\mathbf{y} = 0^4$, i.e., when both $\mathbf{x} \neq 0^4$ and $\mathbf{y} \neq 0^4$, $\mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) \neq 0^8$. We argue on the basis of activity patterns of $\mathbf{x}$ and show that whenever $\mathbf{x} \neq 0^4$, we must have $\mathbf{y} = 0^4$. Due to Lemma 7, we only need look at inputs $\mathbf{x}$ that have different relative position of 1s in their activity pattern. So, looking at vectors with activity pattern $(1, 0, 0, 0)$ covers all cases with $\mathrm{w}(\mathbf{x}) = 1$ and so on.

Now for each such possible value of $\mathbf{x}$ based on its activity pattern, we look at the activity pattern of $\mathbf{u}$ via its propagation under N_α. Furthermore, for $\mathcal{F}\text{-}128(\mathbf{x}, \mathbf{y}) = 0^8$, we must have $\mathbf{x} \otimes \mathbf{y} = 0^4$ and $\mathbf{u} \otimes \mathbf{v} = 0^4$. This implies that for $i \in \{0,1,2,3\}$, if the i-th bit in the activity pattern of $\mathbf{x}$ (and $\mathbf{u}$) is 1, then the i-th bit in the activity pattern of $\mathbf{y}$ (and $\mathbf{v}$) must be 0. Based on these arguments, for each possible $\mathbf{x}$, we tabulate the activity patterns of $\mathbf{u}$ and the possible activity pattern in $\mathbf{y}$ and $\mathbf{v}$ that are compatible with a 0^8 output in Table 2.

Table 2: Activity patterns of $\mathbf{x}$, $\mathbf{y}$, $\mathbf{u}$ and $\mathbf{v}$ for $\mathcal{F}$-128$(\mathbf{x},\mathbf{y}) = 0^8$.

Cases	Activity pattern of			
	$\mathbf{x}$	$\mathbf{u}$	$\mathbf{y}$	$\mathbf{v}$
1	$(1,0,0,0)$	$(1,0,1,1)$	$(0,\otimes,\otimes,\otimes)$	$(0,\otimes,0,0)$
2	$(1,1,0,0)$	$(\otimes,1,1,\otimes)$	$(0,0,\otimes,\otimes)$	$(\otimes,0,0,\otimes)$
3	$(1,0,1,0)$	$(\otimes,1,\otimes,1)$	$(0,\otimes,0,\otimes)$	$(\otimes,0,\otimes,0)$
4	$(1,1,1,0)$	$(\otimes,\otimes,\otimes,\otimes)$	$(0,0,0,\otimes)$	$(\otimes,\otimes,\otimes,\otimes)$
5	$(1,1,1,1)$	$(\otimes,\otimes,\otimes,\otimes)$	$(0,0,0,0)$	$(\otimes,\otimes,\otimes,\otimes)$

We now look at the cases, in particular at the difference propagation from $\mathbf{y}$ to $\mathbf{v}$, separately.

In case 1, we have $y_0 = 0$ and also $v_0 = v_2 = v_3 = 0$. Thus from $\mathrm{N}_\beta \cdot \mathbf{y} = \mathbf{v}$ we have

$$\begin{aligned} y_1 \boxplus y_2 \boxplus y_3 &= 0 \ , \\ y_1 \boxplus y_3 &= 0 \ , \\ y_1 \boxplus y_2 &= 0 \ . \end{aligned}$$

However the above set of linear equation has a unique solution given by $y_1 = y_2 = y_3 = 0$ and this in turn implies $\mathbf{y} = 0^4$.

In case 2 we have $y_0 = y_1 = 0$ and $v_1 = v_2 = 0$. Thus, from $\mathrm{N}_\beta \cdot \mathbf{y} = \mathbf{v}$ we have $y_2 \boxplus y_3 = 0$ and $y_3 = 0$, i.e., $y_2 = y_3 = 0$. Thus again we have $\mathbf{y} = 0^4$.

In case 3 we have $y_0 = y_2 = 0$ and $v_1 = v_3 = 0$. Thus, from $\mathrm{N}_\beta \cdot \mathbf{y} = \mathbf{v}$ we directly have $y_1 = y_3 = 0$, i.e., $\mathbf{y} = 0^4$.

In case 4, $y_0 = y_1 = y_2 = 0$. Now, if $y_3 \neq 0$, then from $\mathrm{N}_\beta \cdot \mathbf{y} = \mathbf{v}$ we must have $v_0 = v_1 = v_2 = y_3 \neq 0$. Since $\mathbf{u} \otimes \mathbf{v} = 0^4$, this in turn implies $u_0 = u_1 = u_2 = 0$.Now from $\mathrm{N}_\alpha \cdot \mathbf{x} = \mathbf{u}$ and the fact that $x_3 = 0$, we have

$$\begin{aligned} x_0 \boxplus x_1 \boxplus x_2 &= 0 \ , \\ x_1 \boxplus x_2 &= 0 \ , \\ x_0 \boxplus x_2 &= 0 \ . \end{aligned}$$

But these system of solutions have a unique solution given by $x_0 = x_1 = x_2 = 0$, contradicting the fact that $\mathbf{x}$ has activity pattern $(1,1,1,0)$ and thus y_3 must be 0. Hence once again we have $\mathbf{y} = 0^4$.

In case 5, it follows directly from Table 2 that $\mathbf{y} = 0^4$.

Thus $\mathcal{F}$-128$(\mathbf{x},\mathbf{y}) = 0^8$ iff $\mathbf{x} = 0^4$ or $\mathbf{y} = 0^4$. Thus we conclude $\mathrm{IP}_{\mathcal{F}\text{-}128}(0^8) = \frac{2^{4w+1}-1}{2^{8w}} \leq \frac{1}{2^{4w-1}} = 2^{-127}$. $\square$

We now upper bound the value of $\mathrm{IP}_{\mathcal{F}\text{-}128}(Z)$ for $Z \neq 0^8$. To upper-bound this value we first briefly remind the readers of highly composite numbers.

Definition 13 (Highly composite number)**.** A number $n \in \mathbb{N}$ is called highly composite if it has more divisors than any number smaller than n.

We computed the list of all highly composite numbers smaller than 2^{65} based on the code by Federico Glaudo[1] . We found out that $36,802,111,876,251,321,600$ is the largest

[1]https://gist.github.com/dario2994/fb4713f252ca86c1254d

highly composite number smaller than 2^{65} and has 207360 divisors. For our choice of $w = 32$, we can thus claim that any number smaller than 2^{64} has at most $207360 < 2^{18}$ divisors.

Lemma 9. *Let $Z \neq 0^8$. Then for the public function $\mathcal{F}$-128, $\mathrm{IP}_{\mathcal{F}\text{-}128}(Z) \leq \frac{2^{18}(2^{w+1}-1)^3}{2^{8w}} < 2^{-149}$.*

Proof. When $Z \neq 0^8$, $z_i \neq 0$ for some $i \in \{0, 1, \ldots, 8\}$. Now, we have:

$$x_i \times y_i = z_i \text{ for } i \in \{0, 1, 2, 3\} \ , \tag{12}$$

$$u_i \times v_i = z_{i+4} \text{ for } i \in \{0, 1, 2, 3\} \ . \tag{13}$$

When $z_i \neq 0$ for some $i \in \{0, 1, 2, 3\}$, we assume without loss of generality that $z_0 \neq 0$. Now, clearly the number of solutions to $x_0 \times y_0 = z_0$ is upper bounded by the number of divisors of z_0, i.e., by 2^{18}. Now each of $x_i \times y_i = z_i$ for $i \in \{1, 2, 3\}$ can have at most $2^{w+1} - 1$ solutions. Consequently there can be at most $2^{18}(2^{w+1} - 1)^3$ solutions to (12). Hence for such a Z, $\mathrm{IP}_{\mathcal{F}\text{-}128}(Z) < \frac{2^{18}(2^{w+1}-1)^3}{2^{8w}} < 2^{-149}$.

When $z_i \neq 0$ for some $i \in \{4, 5, 6, 7\}$, we can similarly prove that (13) can have at most $2^{18}(2^{w+1} - 1)^3$ solutions in u_i, v_i for $i \in \{0, 1, 2, 3\}$ and since N_α and N_β are both invertible matrices, each value of u_i, v_i corresponds to a unique value of x_i, y_i and hence for such a Z as well $\mathrm{IP}_{\mathcal{F}\text{-}128}(Z) < 2^{-149}$. □

Corollary 3. *For the public function $\mathcal{F}$-128, we have $\mathrm{MIP}_{\mathcal{F}\text{-}128} \leq 2^{-127}$.*

Proof. The proof follows directly from Lemmas 8 and 9. □

5.2 Maximum Differential Probability of $\mathcal{F}$-128

Given any differential $((\mathbf{a}, \mathbf{b}), \Delta)$ to $\mathcal{F}$-128, where $\mathbf{a} = (a_0, a_1, a_2, a_3)^\intercal$, $\mathbf{b} = (b_0, b_1, b_2, b_3)^\intercal \in (\mathbb{Z}/2^w\mathbb{Z})^4$ and $\Delta = (\delta_0, \delta_1, \ldots, \delta_7)^\intercal \in \left(\mathbb{Z}/2^{2w}\mathbb{Z}\right)^8$, its DP depends on the number of active multiplications in $\mathcal{F}$-128 as a result of propagation of the input difference $A = (\mathbf{a}, \mathbf{b})$. The number of such active multiplications are in turn lower bound by the branch numbers of N_α and N_β. We wanted to make sure that the minimum number of active multiplication in $\mathcal{F}$-128 is at least 4. This meant we were looking for a matrix with branch number 4. We also wanted the entries in our matrix to be small positive integers so that the matrix multiplication could be computed using additions only. N_α satisfies these conditions and $\mathrm{N}_\beta = \tau^3(\mathrm{N}_\alpha)$. As such N_β has similar diffusion properties and in particular has branch number 4. Thus any differential over $\mathcal{F}$-128 must contribute to at least 4 active multiplications.

In the context of our public functions we call an input difference $(\mathbf{a}, \mathbf{b}) \in (\mathbb{Z}/2^w\mathbb{Z})^8$ to be unilateral iff $\mathbf{a} = 0^4$ or $\mathbf{b} = 0^4$. Due to the circulant nature of N_α and N_β and since $\mathrm{N}_\beta = \tau^3(\mathrm{N}_\alpha)$, clearly $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a}, 0^4), \Delta) = \mathrm{DP}_{\mathcal{F}\text{-}128}((0^4, \mathbf{a}), \Delta)$ and thus we only look at unilateral differentials of type $((\mathbf{a}, 0^4), \Delta)$.

Now, the matrices are chosen such that only unilateral differentials lead to four active multiplications in $\mathcal{F}$-128. For $a \neq 0$, we classify all such input differences $(\mathbf{a}, 0^4)$ and compute the corresponding value of $\mathbf{c}$ in Table 3. Due to Lemma 7, we only look at values of $\mathbf{a}$ with different relative position of non-zero components.

In Table 3 $\mathbf{a} = (2^{w-1}, 2^{w-1}, 2^{w-1}, 0)^\intercal$, denotes a single difference, while all other values of $\mathbf{a}$ define a class of differences. This single entry appears since we are operating in $\mathbb{Z}/2^w\mathbb{Z}$, which is not a field and in $\mathbb{Z}/2^w\mathbb{Z}$, $2 \cdot 2^{w-1} = 0$.

We now look at the DP of a unilateral differential to $\mathcal{F}$-128. Throughout this section, we will use the following notation: $\mathbf{c} = \mathrm{N}_\alpha \cdot \mathbf{a}$, $\mathbf{d} = \mathrm{N}_\beta \cdot \mathbf{b}$, $\mathbf{p} = \mathrm{N}_\alpha \cdot \mathbf{h}$ and $\mathbf{q} = \mathrm{N}_\beta \cdot \mathbf{k}$.

Table 3: Unilateral differences that lead to 4 active multiplications in $\mathcal{F}$-128.

$\mathbf{a}^\intercal$	$\mathbf{c}^\intercal$
$(a,0,0,0)$	$(a,a,a,0)$
$(a,\overline{a},0,0)$	$(0,\overline{a},a,0)$
$(a,0,\overline{a},0)$	$(0,\overline{a},0,a)$
$(2^{w-1},2^{w-1},2^{w-1},0)$	$(2^w,0,0,0)$

Lemma 10. *Let $((\mathbf{a},0^4),\Delta)$ be a unilateral differential to $\mathcal{F}$-128. Then we have*

$$\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},0^4),\Delta) \leq \frac{1}{2^{4w}} \ ,$$

where the equality occurs for $\Delta = 0^8$.

Proof. An input difference $(\mathbf{a},0^4)$ propagates to the output difference Δ under the key $(\mathbf{h},\mathbf{k})$ iff $(\mathcal{F}\text{-}128(\mathbf{h}\boxplus\mathbf{a},\mathbf{k}) - \mathcal{F}\text{-}128(\mathbf{h},\mathbf{k})) \bmod 2^{2w} = \Delta$, i.e.,

$$(((\mathbf{h}\boxplus\mathbf{a})\otimes\mathbf{k}, \mathrm{N}_\alpha\cdot(\mathbf{h}\boxplus\mathbf{a})\otimes\mathrm{N}_\beta\cdot\mathbf{k}) - (\mathbf{h}\otimes\mathbf{k},\mathrm{N}_\alpha\cdot\mathbf{h}\otimes\mathrm{N}_\beta\cdot\mathbf{k})) \bmod 2^{2w} = \Delta \ ,$$

where the output difference modulo 2^{2w} is computed component wise. This simplifies to, for $i \in \{0,1,2,3\}$,

$$\begin{aligned}(((h_i\boxplus a_i)\times k_i) - (h_i\times k_i)) \bmod 2^{2w} &= \delta_i \ ,\\ (((p_i\boxplus c_i)\times q_i) - (p_i\times q_i)) \bmod 2^{2w} &= \delta_{i+4} \ .\end{aligned} \tag{14}$$

For each of the 8 equations in (14), if the corresponding a_i or c_i becomes 0, that equation leads to either an identity or a contradiction depending on whether the corresponding δ_i or δ_{i+4} is 0 or not. Now, if any of the 8 equations lead to a contradiction, then $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},0^4),\Delta) = 0$.

Let us now assume that none of the 8 equations lead to a contradiction. Since N_α has branch number 4, at least 4 components of $(\mathbf{a},\mathbf{c})$ must be non-zero. Thus, (14) is a system containing at least 4 equations. This system of equations, upon reduction modulo 2^w, turn into a system of at least 4 linear equations involving only k_i and q_i by Lemma 1. Furthermore, since $\mathbf{q} = \mathrm{N}_\beta\cdot\mathbf{k}$, we have in fact a system of at least 4 linear equations in the four variables k_0, k_1, k_2 and k_3, where the variables all lie in the interval $[0, 2^w)$. N_β is an invertible matrix and as such this system of equations can have at most one solution for each of k_i. This in turn implies that the total number of solutions to (14) can be at most 2^{4w}. Thus, $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},0^4),\Delta) \leq \frac{2^{4w}}{2^{8w}} = \frac{1}{2^{4w}}$ and the upper bound is achieved when the free variables h_0, h_1, h_2, h_3 take every possible value.

When $\Delta = 0^8$, (14) becomes:

$$\begin{aligned}((h_i\boxplus a_i)\times k_i) &= (h_i\times k_i) \ ,\\ ((p_i\boxplus c_i)\times q_i) &= (p_i\times q_i) \ .\end{aligned} \tag{15}$$

Again, by similar arguments we see that (15) leads to a system of at least 4 equations in 4 variables from k_i, q_i and any such system leads to a unique solution given by: $k_0 = k_1 = k_2 = k_3 = 0$. They hold for any choice of $\mathbf{h}$. Thus, we can say $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},0^4),0^8) = \frac{2^{4w}}{2^{8w}} = \frac{1}{2^{4w}}$. □

So, for any differential $((\mathbf{a},0^4),\Delta)$ that leads to 4 active multiplications in $\mathcal{F}$-128, $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},0^4),\Delta) \leq \frac{1}{2^{4w}}$ and in particular $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},0^4),0^8) = \frac{1}{2^{4w}}$.

We now look at bilateral input differences that lead to 5 active multiplications in $\mathcal{F}$-128. We computed all such differences and again, keeping rotational symmetries aside due to Lemma 7, for $a, b \neq 0$ we tabulate these differences along with the value of the corresponding $\mathbf{c}$ and $\mathbf{d}$ in Table 4.

Table 4: Types of bilateral differences that lead to 5 active multiplications.

Type	$\mathbf{a}^\intercal$	$\mathbf{b}^\intercal$	$\mathbf{c}^\intercal$	$\mathbf{d}^\intercal$
1	$(a,0,0,0)$	$(b,0,0,0)$	$(a,0,a,a)$	$(0,b,b,b)$
2	$(a,0,0,0)$	$(0,b,0,0)$	$(a,0,a,a)$	$(b,0,b,b)$
3	$(a,0,0,0)$	$(b,0,0,\overline{b})$	$(a,0,a,a)$	$(\overline{b},0,0,b)$
4	$(a,0,0,0)$	$(b,0,\overline{b},0)$	$(a,0,a,a)$	$(\overline{b},0,b,0)$
5	$(a,0,0,\overline{a})$	$(b,0,0,\overline{b})$	$(a,\overline{a},0,0)$	$(\overline{b},0,0,b)$
6	$(a,0,0,\overline{a})$	$(b,\overline{b},0,0)$	$(a,\overline{a},0,0)$	$(\overline{b},b,0,0)$
7	$(a,0,0,\overline{a})$	$(\overline{b},0,b,b)$	$(a,\overline{a},0,0)$	$(2b,b,0,0)$
8	$(a,0,\overline{a},0)$	$(b,0,b,\overline{b})$	$(0,\overline{a},0,a)$	$(0,b,0,2b)$
9	$(a,\overline{a},0,a)$	$(b,b,0,\overline{b})$	$(0,0,2a,a)$	$(0,0,b,2b)$
10	$(a,a,a,\overline{2a})$	$(\overline{2b},b,b,b)$	$(3a,0,0,0)$	$(3b,0,0,0)$

Lemma 11. *Let* $((\mathbf{a},\mathbf{b}),\Delta)$ *be a bilateral differential of type 1 to* $\mathcal{F}$-128. *So* $\mathbf{a}=(a,0,0,0)^\intercal$ *and* $\mathbf{b}=(b,0,0,0)^\intercal$ *for some* $a,b\neq 0$. *Then,*

$$\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},\mathbf{b}),\Delta \leq \frac{\prod_{i\in\{0,6,7\}}\mathrm{N}((a,b),\delta_i)}{2^{8w}}\ .$$

In particular, for such $(\mathbf{a},\mathbf{b})$, $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},\mathbf{b}),\Delta)\leq 2^{-5w}$.

Proof. An input difference $(\mathbf{a},\mathbf{b})$ of type 1 propagates to the output difference Δ under the key $(\mathbf{h},\mathbf{k})$ iff:

$$((h_0\boxplus a)\times(k_0\boxplus b)-h_0\times k_0) \bmod 2^{2w}=\delta_0\ , \tag{16.1}$$

$$((p_0\boxplus a)\times q_0-p_0\times q_0) \bmod 2^{2w}=\delta_4\ , \tag{16.2}$$

$$(p_1\times(q_1\boxplus b)-p_1\times q_1) \bmod 2^{2w}=\delta_5\ , \tag{16.3}$$

$$((p_2\boxplus a)\times(q_2\boxplus b)-p_2\times q_2) \bmod 2^{2w}=\delta_6\ , \tag{16.4}$$

$$((p_3\boxplus a)\times(q_3\boxplus b)-p_3\times q_3) \bmod 2^{2w}=\delta_7\ . \tag{16.5}$$

Due to Lemma 1, (16.2) and (16.3) are satisfied for at most one value of q_0 and p_1 respectively. Let us assume that those equations are consistent and as such let those values be: $q_0=\beta$ and $p_1=\alpha$.

Now for $i\in\{0,6,7\}$, (16.1), (16.4) and (16.5) has $\mathrm{N}((a,b),\delta_i)$ solutions in variables (h_0,k_0), (p_2,q_2) and (p_3,q_3) respectively. Let (α_0,β_0), (α_6,β_6) and (α_7,β_7) be one such solution to equations (16.1), (16.4) and (16.5)respectively. Then we have

$$\begin{bmatrix}1&0&0&0\\0&1&1&1\\1&0&1&1\\1&1&0&1\end{bmatrix}\cdot\begin{bmatrix}h_0\\h_1\\h_2\\h_3\end{bmatrix}=\begin{bmatrix}\alpha_0\\\alpha\\\alpha_6\\\alpha_7\end{bmatrix}\ \text{and}\ \begin{bmatrix}1&0&0&0\\0&1&1&1\\1&1&0&1\\1&1&1&0\end{bmatrix}\cdot\begin{bmatrix}k_0\\k_1\\k_2\\k_3\end{bmatrix}=\begin{bmatrix}\beta_0\\\beta\\\beta_6\\\beta_7\end{bmatrix}\ . \tag{17}$$

Both the matrices in (17) are invertible and as such for each solution to (16.1), (16.4) and (16.5), we obtain a unique value of $(\mathbf{h},\mathbf{k})$. Thus, we can conclude

$$\#\mathrm{S}_{\mathcal{F}\text{-}128}((\mathbf{a},\mathbf{b}),\Delta)=\begin{cases}\prod\limits_{i\in\{0,6,7\}}\mathrm{N}((a,b),\delta_i) & \text{, when (16.2), (16.3) are consistent}\\ 0 & \text{, otherwise .}\end{cases}$$

Now, for each $i\in\{0,6,7\}$, $\mathrm{N}((a,b),\delta_i)\leq 2^w$. Thus we conclude that $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a},\mathbf{b}),\Delta\leq \frac{\prod_{i\in\{0,6,7\}}\mathrm{N}((a,b),\delta_i)}{2^{8w}}\leq 2^{-5w}$. □

For input difference $(\mathbf{a}, \mathbf{b})$ of types 2,3,4,5 and 6 as well, $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a}, \mathbf{b}), \Delta) \leq 2^{-5w}$. We can see from Table 4 that for these types, just like differences of type 1, 2 of the active multiplications have unilateral difference and the rest 3 have bilateral difference. Thus we can upper-bound their DP in a similar way as described in the proof to Lemma 11.

Lemma 12. *Let $((\mathbf{a}, \mathbf{b}), \Delta)$ be a bilateral differential of type 7 to $\mathcal{F}$-128. So $\mathbf{a} = (a, 0, 0, \overline{a})^\intercal$ and $\mathbf{b} = (\overline{b}, 0, b, b)^\intercal$ for some $a, b \neq 0$. Then, $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a}, \mathbf{b}), \Delta) \leq 2^{-5w}$.*

Proof. An input difference $(\mathbf{a}, \mathbf{b})$ of type 7 propagates to the output difference Δ under the key $(\mathbf{h}, \mathbf{k})$ iff:

$$((h_0 \boxplus a) \times (k_0 \boxplus \overline{b}) - h_0 \times k_0) \bmod 2^{2w} = \delta_0 \ , \tag{18.1}$$
$$(h_2 \times (k_2 \boxplus b) - h_2 \times k_2) \bmod 2^{2w} = \delta_2 \ , \tag{18.2}$$
$$((h_3 \boxplus \overline{a}) \times (k_3 \boxplus b) - h_3 \times k_3) \bmod 2^{2w} = \delta_3 \ , \tag{18.3}$$
$$((p_0 \boxplus a) \times (q_0 \boxplus 2b) - p_0 \times q_0) \bmod 2^{2w} = \delta_4 \ , \tag{18.4}$$
$$((p_1 \boxplus \overline{a}) \times (q_1 \boxplus b) - p_1 \times q_1) \bmod 2^{2w} = \delta_5 \ . \tag{18.5}$$

Due to Lemma 1, (18.2) is satisfied for at most one value of h_2. Let us assume that the solution exists and let the solution be $h_2 = \alpha$.

Now, equations (18.1), (18.3) and (18.5) have $\mathrm{N}((a, \overline{b}), \delta_0)$, $\mathrm{N}((\overline{a}, b), \delta_3)$ and $\mathrm{N}((\overline{a}, b), \delta_5)$ solutions in variables (h_0, k_0), (h_3, k_3) and (p_1, q_1) respectively. Let (α_0, β_0), (α_3, β_3) and (α_5, β_5) be one such solution to equations (18.1), (18.3) and (18.5)respectively. Then corresponding to these solutions we have,

$$p_1 = h_1 \boxplus h_2 \boxplus h_3 = h_1 \boxplus \alpha \boxplus \alpha_3 = \alpha_5 \implies h_1 = \alpha_5 \boxminus \alpha \boxminus \alpha_3 \ ,$$
$$q_1 = k_0 \boxplus k_2 \boxplus k_3 = \beta_0 \boxplus k_2 \boxplus \beta_3 = \beta_5 \implies k_2 = \beta_5 \boxminus \beta_0 \boxminus \beta_3 \ .$$

Now the only unknown variable remaining is k_1, which can be obtained from (18.4). Indeed letting $p_0 = \alpha_0 \boxplus \alpha_5 \boxminus \alpha_3 = \alpha_4$ and $q_0 = k_1 \boxplus \beta_5 \boxminus \beta_0 = k_1'$, (18.4) now becomes

$$((\alpha_4 \boxplus a) \times (k_1' \boxplus 2b) - \alpha_4 \times k_1') \bmod 2^{2w} = \delta_4 \ . \tag{19}$$

Arguing similar to Lemma 1, we can see that (19) has at most one solution in k_1' and consequently for k_1.

Thus, we see that corresponding to every solution of (18.1), (18.3) and (18.5), we obtain a unique solution in $(\mathbf{h}, \mathbf{k})$. Thus we can conclude that

$$\#\mathrm{S}_{\mathcal{F}\text{-}128}((\mathbf{a}, \mathbf{b}), \Delta) \leq \mathrm{N}((a, \overline{b}), \delta_0) \times \mathrm{N}((\overline{a}, b), \delta_3) \times \mathrm{N}((\overline{a}, b), \delta_5) \leq 2^{3w} \ .$$

Hence, $\mathrm{DP}_{\mathcal{F}\text{-}128}((\mathbf{a}, \mathbf{b}), \Delta) \leq 2^{-5w}$. □

We can see that the proofs of Lemmas 11 and 12 are very similar. In fact, the same line of reasoning as used in these proofs can be applied for differences of types 8,9, 10 and also for bilateral differences that lead to more than 5 active multiplication. The basic strategy is to first obtain the solution sets for three $M[w]$s in the corresponding variables (h_i, k_i) or (p_i, q_i). This already finds 6 of the 8 unknown variables. Then corresponding to each such solution, we can use remaining 2 equations to obtain the last 2 unknown variables. These last 2 equations can now have at most 1 solution since these equations turn into modular linear equation in a single variable and can have at most one solution. This upper-bounds the number of solutions to 2^{3w}.

Corollary 4. *For the public function $\mathcal{F}$-128, $\mathrm{MDP}_{\mathcal{F}\text{-}128} = \frac{1}{2^{4w}}$.*

Proof. The proof follows directly from Lemmas 10, 11, 12 and the discussions that follow. □

5.3 ε-Δuniversality of Multimixer-128

Theorem 2. *Multimixer-128 is* 2^{-127}-Δ*universal.*

Proof. The proof follows directly from Corollaries 3 and 4. □

So, we see that $\mathrm{MIP}_{\mathcal{F}\text{-}128} \approx 1/2^{127}$. While it would have been ideal to have $\mathrm{MIP}_{\mathcal{F}\text{-}128} \approx 1/2^{128}$, we would like to point out that for our public function only $\mathrm{IP}_{\mathcal{F}\text{-}128}(0^8)$ attains that value and for all other outputs, the IP is significantly smaller than $1/2^{128}$. In fact $\mathrm{IP}_{\mathcal{F}\text{-}128}(0^8)$ could also be decreased significantly if we used affine maps instead of the linear maps corresponding to the matrices N_α and N_β. However using affine maps will make our function slightly less efficient. Furthermore for unilateral differentials with 4 active multiplications, $\mathrm{DP}_{\mathcal{F}\text{-}128} = 2^{-128}$ and using affine map instead of a linear one, does not decrease the $\mathrm{DP}_{\mathcal{F}\text{-}128}$ of such differentials. Thus difference of 1-bit in epsilon does not outweigh the decrease in efficiency.

6 The NH Hash Function

The NH Hash function was proposed by Black et al. at CRYPTO 1999 [BHK+99]. For an even $\kappa \geq 2$ and a number $w \geq 1$, the family of functions $\mathrm{NH}[\kappa, w]$ is defined as follows. Its domain is $\mathrm{BS}(G, \kappa)$ with $G = \{0,1\}^{2w}$ and range is $\{0,1\}^{2w}$. Let $\mathbf{M} = (M_0, M_1, \ldots, M_{l-1})$ and $\mathbf{K} = (K_0, K_1, ..., K_{\kappa-1})$, where l is an even integer. For each $i \in \{0, 1, \ldots, l-1\}$ both $M_i, K_i \in \{0,1\}^w$ (where e.g. $w = 16$ or 32) and m_i, k_i denote respectively the numbers that M_i and K_i represent as unsigned integers. Now, $\mathrm{NH}[\kappa, w]$ is defined as

$$\mathrm{NH}_{\mathbf{K}}(\mathbf{M}) = \sum_{i=0}^{(l-2)/2} [(m_{2i} \boxplus k_{2i}) \times (m_{2i+1} \boxplus k_{2i+1})] \bmod 2^{2w} \ .$$

$\mathrm{NH}[\kappa, w]$ is 2^{-w}-universal for equal length messages. To reduce this universality from 2^{-w} to 2^{-tw}, the authors apply $\mathrm{NH}[\kappa, w]$ function on the same message $\mathbf{M}$ with independent keys t times, and concatenate the results. But this process requires t times as much key material and to that end they further apply the Toeplitz technique [Kra94, MNT93] to reduce the amount of extra key material. By this approach, they only require $2w(t-1)$-bits of extra key material. They denote the Toeplitz-NH as $\mathrm{NH}^{\mathrm{T}}[\kappa, w, t]$.

We compare Multimixer-128 with $\mathrm{NH}^{\mathrm{T}}[\kappa, 32, 4]$. The $\mathrm{NH}^{\mathrm{T}}[\kappa, 32, 4]$ hash function does achieve the universality of 2^{-128} for equal length messages by Theorems 1,2[BHK+99]. In our security analysis, based on [FRD23] and [GFAD23], we consider messages of variable length and MIP of a public function captures this notion. Thus, an analysis of $\mathrm{NH}^{\mathrm{T}}[\kappa, 32, 4]$ taking into account messages of variable length, will lead to investigating the MIP of that function. Now, $\mathrm{NH}^{\mathrm{T}}[\kappa, 32, 4]$ consists of parallel application of 4 $\mathrm{NH}[\kappa, 32]$ with independent keys on the same message $\mathbf{M}$. For $\mathrm{NH}[\kappa, 32]$, clearly $\mathrm{MIP}_{\mathrm{NH}[\kappa,32]} = \mathrm{IP}_{\mathrm{NH}[\kappa,32]}(0) = 2^{-31}$ and thus $\mathrm{MIP}_{\mathrm{NH}^{\mathrm{T}}[\kappa,32,4]} = 2^{-31\times 4} = 2^{-124}$.

The authors of NH^{T} deal with messages of variable length by defining a mode on top of NH^{T} that applies NH^{T} on fixed-length inputs. But, this results in longer digests for long messages. While Multimixer-128 could also be applied on top of this mode, we do not consider this since our analysis incorporates collisions between messages of different length and as discussed in Section 5.3, offers similar levels of security as NH^{T}.

7 Partial key knowledge

Some cryptographic algorithms have classes of weak keys and classes of weak keys of MAC functions were identified in [HP08], where they were characterized as follows [HP08, Section 3.1]: "*In symmetric cryptology, a class of keys is called a weak key class if for the members of that class the algorithm behaves in an unexpected way and if it is easy to detect whether a particular unknown key belongs to this class. For a MAC algorithm, the unexpected behavior can be that the forgery probability for this key is substantially larger than average.*"

In this paper we investigate the differential uniformities of functions, that are defined as worst case differential probabilities where the probability is taken over the full key space assuming uniformly distributed keys. Clearly, taking this probability over non-uniform key distributions can lead to significantly higher uniformities and thus forgery probabilities. Any such non-uniform distribution could be labeled a class of weak keys (or rather a generalization thereof). As a matter of fact, we think it is more informative to speak of the implications of partial key knowledge on the effect of the forgery probability as done in [HP08].

In general the knowledge of x key bits in Multimixer-128 may, in the worst case, increase the forgery probability per message pair from 2^{-127} to 2^{x-127}. We can illustrate that with an extreme example. Assume the adversary knows the first half of the first block of the key: (h_0, h_1, h_2, h_3). Then two single-block messages $(-h_0, -h_1, -h_2, -h_3, m_0, m_1, m_2, m_3)$ and $(-h_0, -h_1, -h_2, -h_3, m'_0, m'_1, m'_2, m'_3)$ will collide for any values of (m_0, m_1, m_2, m_3) and (m'_0, m'_1, m'_2, m'_3), so forgery can be realized with one generation query and one verification query.

Multimixer-128 is not alone with this property. For example, in NH^{T} knowledge of 128 bits of the key also allows generating a forgery in a single verification query [HP08, Section 3.2]. So, while in classical block ciphers (or stream ciphers) with an n-bit key and claimed pseudorandom permutation (PRP) security strength (or PRG security strength in case of a stream cipher) of n bits, it is plausible that knowledge of x key bits may only reduce the security strength to $n - x$, in Multimixer-128 and other keyed hash based MAC functions only partial knowledge of the long key reduces the security strength to 0. It is therefore essential for security that the long key K has been generated in a way that makes its distribution practically indistinguishable from uniform.

8 Implementation and Benchmarking Results

We wrote optimized code for Multimixer-128 on 32-bit ARMv7 Cortex-A processors. ARMv7 architecture was the first to introduce the Advanced Single Instruction and Multiple Data Stream (ASIMD) extension to the ARMv7-A and ARMv7-R profiles. The implementation of the ASIMD extension used in ARM processors is called NEON. To obtain a highly-optimized keyed hash function, we not only considered the ASIMD extension in our design phase but also utilized it in our implementation.

We chose ARMv7 architecture primarily because it is used inside many embedded devices, and no Cryptography Extensions are available on this architecture. The Cryptography Extensions add new instructions for the ASIMD to accelerate the execution of AES, SHA1, and SHA2-256 algorithms. Although newer A-profile architecture targets allow for these instructions, the availability of such extensions varies among devices and depends on the CPU manufacturer's decision.

We benchmark our code on a popular ARMv7-powered core, i.e., CortexA7. However, our code runs on other ARMv7-A cores supporting NEON and can be adapted easily for the ARMv8 Cortex-A family of processors.

We compare our code for Multimixer-128 with available open-source NH^{T} implementation for the ARMv7 Cortex-A processors. For a fair comparison, all the performance-critical parts in these implementations are written in assembly language, leveraging NEON instructions. Our code for Multimixer-128 outperforms NH^{T} making it the fastest keyed hash with that level of universality on ARMv7-powered devices. Software implementation can be found at https://github.com/Parisaa/Multimixer.

8.1 Multimixer-128 Implementation

NEON instructions make parallel operation on packed units of data, determined by the so-called arrangement specifier, viable. We leveraged these instructions to parallelize our process on data in the Multimixer-128 implementation. The arrangement specifier used in all instructions of our code is 32 bits, as our input words in the design are defined with the same length. The inputs to $\mathcal{F}$-128 are 8 key and 8 message words that can be stored in 4 vector registers. These message words are be loaded from memory and then mapped as $\{x_0, x_1, y_1, y_2\}$ and $\{x_2, x_3, y_3, y_0\}$. This re-mapping makes the implementation of the circulant matrices more efficient and does not impact the security analysis.

The circulant matrix symmetry allows an efficient implementation using only vector additions and a word shuffle called *vector reverse in double-words*. All in all, we implemented both matrix multiplications in $\mathcal{F}$-128 using only two `vrev64` instructions and four `vadd` instructions. Two `vadd` instructions add the key to the message in the beginning, and 4 `vmlal` instructions operate the integer multiplication and add the result to the register that is used to keep the output. We end up with a total of 11 instructions per call to $\mathcal{F}$-128, while NH$^{\mathrm{T}}$ uses 16 instructions to process same amount of data. By keeping most operations in place we also minimize the number of `MOV` operations. We compare the size features and number of arithmetic operations in Multimixer-128 with that of NH$^{\mathrm{T}}$ in Table 5.

Table 5: Feature comparison between NH$^{\mathrm{T}}$ and Multimixer-128.

	length in bits			# ops. per 32-bit word		
Algorithm	digest	block	key	$\times$	+ mod 2^{32}	+ mod 2^{64}
NH$^{\mathrm{T}}$	256	64	64(# blocks) +192	2	4	2
Multimixer-128	512	256	256(# blocks)	1	2.5	1

8.2 Performance

We benchmarked our Multimixer-128 code and that of NH$^{\mathrm{T}}$ on an ARM Cortex-A7 processor in the Broadcom BCM2836 chipset used in the Raspberry Pi 2 model B single board computer and report in the results in Table 6. We also compared the performance of Multimixer-128 and the two fastest NIST LWC candidates [MBA$^+$23], Xoodyak [AK20] and ASCON [DEMS21]. These two cryptographic schemes claim 128-bits of security and they can be used for MAC computation by fixing the plaintext to empty and taking associated data as input. For NH$^{\mathrm{T}}$ and Xoodyak we used the fastest constant-time implementations we could find[2,3]. However, for the ASCON we used the reported performance per authenticated/encrypted byte on an Armv7a architecture [4].

Xoodyak's performance is around 24.5 cycles per authenticated byte for processing 1400 bytes, and the ASCON team reports performance of 30.7 cycles per authenticated/encrypted byte on an Armv7a architecture, while for Multimixer-128, it is less than 2 cycles per byte. Thus we conclude that Multimixer-128 outperforms NH$^{\mathrm{T}}$, Xoodyak, and ASCON on Armv7a architecture platforms.

[2]https://github.com/google/adiantum/tree/master/benchmark/src/arm
[3]https://github.com/XKCP/XKCP/tree/master/lib/low/Xoodoo/ARMv7A-NEON
[4]https://ascon.iaik.tugraz.at/implementations.html

Table 6: Performance on ARM Cortex-A7 in cycles per byte.

Algorithm	Input length in bytes		
	512	4096	32768
Multimixer-128	1.830	1.233	1.396
NH^T	2.033	1.500	1.558

8.2.1 Acknowledgements.

The authors would like to thank Jonathan Fuchs for interesting discussions on weak keys in the context of keyed hash functions. Koustabh Ghosh is supported by the Netherlands Organisation for Scientific Research (NWO) under TOP grant TOP1.18.002 SCALAR, Parisa Amiri Eliasi is supported by the Cryptography Research Center of the Technology Innovation Institute (TII), Abu Dhabi (UAE), under the TII-Radboud project with title Evaluation and Implementation of Lightweight Cryptographic Primitives and Protocols and Joan Daemen is supported by the European Research Council under the ERC advanced grant agreement under grant ERC-2017-ADG Nr. 788980 ESCADA.

References

[80005] NIST. NIST Special Publication 800-38B. Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication, May 2005. https://csrc.nist.gov/pubs/sp/800/38/b/upd1/final.

[AK20] G Van Assche and R Van Keer. Xoodyak, a lightweight cryptographic scheme. 2020.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying Hash Functions for Message Authentication. In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, volume 1109 of *LNCS*, pages 1–15. Springer, 1996.

[BDH+08] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. KECCAK Specifications, NIST SHA-3 Submission, October 2008. http://keccak.noekeon.org/.

[BDH+17] Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Farfalle: parallel permutation-based cryptography. *IACR Trans. Symmetric Cryptol.*, 2017(4):1–38, 2017.

[Ber05] Daniel J. Bernstein. The Poly1305-AES Message-Authentication Code. In *Fast Software Encryption: 12th International Workshop, FSE 2005, Paris, France, February 21-23, 2005, Revised Selected Papers*, volume 3557 of *LNCS*, pages 32–49. Springer, 2005.

[Berrs] Daniel J. Bernstein. Chacha, a variant of salsa20, Workshop Record of SASC 2008: The State of the Art of Stream Ciphers. https://cr.yp.to/papers.html#chacha.

[BHK+99] John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, and Phillip Rogaway. UMAC: Fast and Secure Message Authentication. In *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*, pages 216–233, 1999.

[BKR94] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The Security of Cipher Block Chaining. In *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.

[BR00] John Black and Phillip Rogaway. CBC MACs for Arbitrary-Length Messages: The Three-Key Constructions. In *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *LNCS*, pages 197–215. Springer, 2000.

[BR02] John Black and Phillip Rogaway. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In *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 *LNCS*, pages 384–397. Springer, 2002.

[CB18] Paul Crowley and Eric Biggers. Adiantum: length-preserving encryption for entry-level processors. *IACR Trans. Symmetric Cryptol.*, 2018(4):39–61, 2018.

[CW79] Larry Carter and Mark N. Wegman. Universal classes of hash functions. *J. Comput. Syst. Sci.*, 18(2):143–154, 1979.

[DEMS21] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight authenticated encryption and hashing. *J. Cryptol.*, 34(3):33, 2021.

[DR20] Joan Daemen and Vincent Rijmen. *The Design of Rijndael - The Advanced Encryption Standard (AES), Second Edition.* Information Security and Cryptography. Springer, 2020.

[FRD23] Jonathan Fuchs, Yann Rotella, and Joan Daemen. On the security of keyed hashing based on public permutations. In *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 607–627. Springer, 2023.

[GFAD23] Koustabh Ghosh, Jonathan Fuchs, Parisa Amiri-Eliasi, and Joan Daemen. Universal hashing based on field multiplication and (near-)mds matrices. In *Progress in Cryptology - AFRICACRYPT 2023 - 14th International Conference on Cryptology in Africa, Sousse, Tunisia, July 19-21, 2023, Proceedings*, volume 14064 of *Lecture Notes in Computer Science*, pages 129–150. Springer, 2023.

[HK97] Shai Halevi and Hugo Krawczyk. MMH: software message authentication in the gbit/second rates. In Eli Biham, editor, *Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, Israel, January 20-22, 1997, Proceedings*, volume 1267 of *Lecture Notes in Computer Science*, pages 172–189. Springer, 1997.

[HP08] Helena Handschuh and Bart Preneel. Key-recovery attacks on universal hash function based MAC algorithms. In *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *Lecture Notes in Computer Science*, pages 144–161. Springer, 2008.

[IK03] Tetsu Iwata and Kaoru Kurosawa. OMAC: one-key CBC MAC. In *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153. Springer, 2003.

[Kra94] Hugo Krawczyk. Lfsr-based hashing and authentication. 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 129–139. Springer, 1994.

[Kro15] T. Krovetz. HS1-SIV (v2), 2015. Submission to CAESAR competition.

[MBA+23] Kamyar Mohajerani, Luke Beckwith, Abubakr Abdulgadir, Eduardo Ferrufino, Jens-Peter Kaps, and Kris Gaj. Sca evaluation and benchmarking of finalists in the nist lightweight cryptography standardization process. *Cryptology ePrint Archive*, 2023.

[MNT93] Yishay Mansour, Noam Nisan, and Prasoon Tiwari. The computational complexity of universal hashing. *Theor. Comput. Sci.*, 107(1):121–133, 1993.

[MV04] David A. McGrew and John Viega. The Security and Performance of the Galois/Counter Mode (GCM) of Operation. In *Progress in Cryptology - INDOCRYPT 2004, 5th International Conference on Cryptology in India, Chennai, India, December 20-22, 2004, Proceedings*, volume 3348 of *LNCS*, pages 343–355. Springer, 2004.

[Sti95] Douglas R. Stinson. On the Connections Between Universal Hashing, Combinatorial Designs and Error-Correcting Codes. *Electron. Colloquium Comput. Complex.*, TR95-052, 1995.

[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.