MATTER: A Wide-Block Tweakable Block Cipher

Roberto Avanzi[1], Orr Dunkelman[2] and Kazuhiko Minematsu[3]

[1] Caesarea Rothschild Institute, University of Haifa, Haifa, Israel
roberto.avanzi@gmail.com
[2] Computer Science Department, University of Haifa, Haifa, Israel
orrd@cs.haifa.ac.il
[3] NEC Corporation, Kawasaki, Japan
k-minematsu@nec.com

Abstract. In this note, we introduce the MATTER Tweakable Block Cipher, designed principally for low latency in low-area hardware implementations, but that can also be implemented in an efficient and compact way in software.
MATTER is a 512-bit wide balanced Feistel network with three to six rounds, using the ASCON permutation as the round function. The Feistel network defines a keyed, non-tweakable core, which is made tweakable by using the encryption of the tweak as its key. Key and tweak are 320-bit inputs.
MATTER is particularly suitable for use in an OCB-like mode of operation, with an encrypted checksum for authentication.

Keywords: Tweakable Block Ciphers · Lightweight Cryptography · Wide-Block Ciphers · Memory Encryption

1 Introduction

NIST has recently shown interest in standardizing tweakable, wide-block block ciphers, focusing on "accordion" ciphers based on the AES [DR02]. Software implementations of such a design can be efficient if the *Instruction Set Architecture* (ISA) exposes AES instructions. In hardware, however, the AES can lead to large or slow implementations — this is one of the reasons a large amount of research has been poured into lightweight primitives during the last several years, for instance CLEFIA [SSA+07], ChaCha20 [Ber08], KATAN and KTANTAN [CDK09], KLEIN [GNL11], LED [GPPR11], PRESENT [BKL+07], PRINCE [BCG+12], SIMON and SPECK [BSS+13], MIDORI [BBI+15], QARMA [Ava17], SKINNY and MANTIS [BJK+16], ASCON [DEMS21], BipBip [BDD+23], QARMAv2 [ABD+23], and many other ciphers.

In this note, we propose a solution that prioritizes hardware implementations, based on ASCON, which is set to be standardized by NIST. Since ASCON has been designed to be efficient not only on high-end CPUs but also on resource-constrained devices, both in hardware and in software, our proposal targets the same use cases as well.

The initial motivation for the present work is memory encryption. The security model is straightforward: hardware inside the physical perimeter of a *System-on-a-Chip* (SoC), including the memory controllers, is trusted, while the external memory bus and memory itself are untrusted. We focus mainly on *memory confidentiality* where memory is encrypted in *Cache Line* (CL) sized granules, but also briefly touch integrity and modes of operation.

There are two main approaches to memory encryption: In the first approach, *direct encryption*, clear data is input to a block cipher and the output is written to external memory. A common requirement is that encryption provides *spatial uniqueness*, meaning

the same plaintext at different memory locations produces different ciphertexts. This can be achieved using a *Tweakable Block Cipher* (TBC) [LRW02, LRW11], i.e., a block cipher with three inputs: the secret key, a text, and a *tweak*. The permutation computed by the cipher depends on the tweak as unpredictably as it does on the key. An adversary, however, is able to control the tweak but cannot use this capability to help recover the key. This makes the memory address suitable as a tweak. In fact, one of the earliest uses of TBCs was memory encryption [HT13].

The second approach is suitable if *temporal uniqueness* is required, meaning repeated writes of the same plaintext to the same location produce different ciphertexts. A keystream is generated by a block cipher or a hash function in counter mode, or a stream cipher, and is XOR-ed with the plaintext to produce the ciphertext. The keystream generator uses a secret key, the memory address, and a counter or nonce as inputs. Each CL has its own counter or nonce, which is refreshed before each memory write.

Keystream-based encryption keeps the cipher off the critical path between CPU and external memory, reducing the additional read latency to a single XOR. However, the nonces must be stored in RAM, reducing the amount of the latter available to applications and increasing memory traffic. This can lead to severe performance degradation [AMS+22].

Another problem with the second approach is ciphertext malleability, as flipping a bit in the ciphertext flips the corresponding bit in the plaintext. This enables, for instance, RowHammer attacks [KDK+14], unless integrity tags are generated and verified. These tags also use memory space, adding pressure on the memory subsystem.

Direct encryption, even in the absence of authentication, partially mitigates RowHammer attacks, because any ciphertext change makes its decryption uncorrelated with the original plaintext. This and the absence of ciphertext expansion make direct encryption attractive for practical deployments. This note deals with direct encryption.

ASCON is a 320-bit permutation, and common CL sizes (such as 512 or 1024 bits) cannot be partitioned into 320-bit blocks, preventing its use for direct encryption of CLs. ASCON is also not inherently keyed or tweakable. Our construction addresses these issues.

Outline of the Paper: In Section 2 we define the non-tweakable core of MATTER and in Section 3 we convert it into a TBC. The derivation of the round keys is described in Section 4. In Section 5 we present suitable modes of operation, covering memory encryption and general-purpose use. We sketch provable security in Section 6. Implementation is briefly discussed in Section 7. Finally, in Section 8 we conclude with open questions.

2 Definition of the non-tweakable (core) version of MATTER

The block cipher MATTER is a 512-bit balanced Feistel network [Sor84, LR85, LR88]. It is designed around a non-tweakable, keyed core function, aptly named $\texttt{MATTER}^{\text{core}}$, and a process for deriving its key from a main key and a tweak. $\texttt{MATTER}^{\text{core}}$ comes in versions determined by the number of Feistel rounds, between three and six. Its Feistel function is represented in Figure 1, and consists of the following operations, in the given order:

1. Expansion of the input from 256 to 320 bits by padding it with zero;
2. Addition of a round key k_i (this round key is 320 bits long);
3. An application of $\texttt{ASCON-p}^{r_i}$, where the notation $\texttt{ASCON-p}^{r_i}$ denotes the r_i-round ASCON permutation;
4. Truncation of the result to 256 bits; and
5. Addition of a round key k'_i (this round key needs to be only 256 bits).

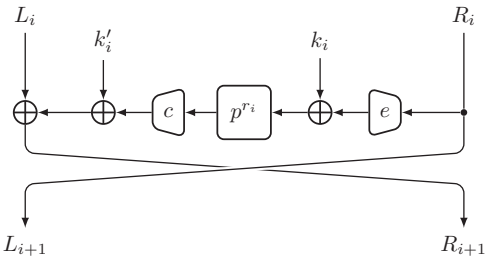

Figure 1: Structure of a MATTER Feistel round.

We restrict $r_i \geq 4$ to guarantee full diffusion in the ASCON function and the number of Feistel rounds is limited to six since we want the total latency of the core function not to exceed twice that of ASCON-p[12] — in other words, 24 ASCON rounds — and a few XORs. The function F_r is computed on one branch and the result added to the other branch, then the two branches are swapped. This operation is performed three to six times. We write $\texttt{MATTER}^{\text{core}}_{r_0,r_1,r_2}$ for the three Feistel rounds version, $\texttt{MATTER}^{\text{core}}_{r_0,r_1,r_2,r_3}$ for the four Feistel rounds version, and so on.

Regardless of the claimed security level associated with a parameter set, keys are always understood as 320-bit strings. This approach mirrors the methodology in BipBip [BDD+23], where a 256-bit main key is used even if the claimed security level is 96 bits, and dispenses with the need for a potentially costly key schedule.

Shorter keys can of course be used and they can be just zero padded, but we strongly recommend the use of a secure *Key Derivation Function* (KDF) to extend short keys to 320 bits. This can be useful in the case the use of separate keys is mandated by architecture, system design, or regulation, as shorter reduce keys storage requirements.

3 Definition of the tweakable version of MATTER

Our approach is inspired by the Masked Even-Mansour *Masked Even-Mansour* (MEM) construction [GJMN16]. In this method, the round keys are not directly derived from the main key, but from a *core key* which is the encryption of the tweak using the main key. The tweak encryption function is a *XOR, Encrypt, and XOR* (XEX) construction [Rog04].

If K and T are the 320-bit key and tweak, then $k = \texttt{ASCON}_{r'}(K \oplus T) \oplus K$, for some positive integer r'. The notation $\texttt{MATTER}^{r'}_{r_0,r_1,r_2,\ldots}$, denotes the tweakable block cipher obtained by adding tweak encryption to $\texttt{MATTER}_{r_0,r_1,r_2,\ldots}$.

4 Derivation of the round keys

Put $k = K$, resp., $k = \texttt{ASCON}_{r'}(K \oplus T) \oplus K$ for the non-tweakable, resp., tweakable version of the cipher. The round keys k_i are given by $k_i = (2^i \cdot k) \oplus \mathfrak{c}_i$, where 2 represents the image of x in $\mathbb{F}_{2^{320}} = \frac{\mathbb{F}_2[x]}{\langle x^{320}+x^4+x^3+x+1\rangle}$. This is an inexpensive operation — a shift and four single-bit XORs — and therefore it does not impact latency upon decryption (cf. Section 7). It is $\mathfrak{c}_0 = 0$ and for $i > 0$ the values $\mathfrak{c}_i$ are 64-bit constants, padded with zeros, taken from the hexadecimal expansion of the fractional part of π:

$$\mathfrak{c}_1 = \texttt{13198A2E03707344}\,,\ \mathfrak{c}_2 = \texttt{A4093822299F31D0}\,,\ \mathfrak{c}_3 = \texttt{082EFA98EC4E6C89}\,,$$
$$\mathfrak{c}_4 = \texttt{452821E638D01377}\,,\ \text{and}\ \mathfrak{c}_5 = \texttt{BE5466CF34E90C6C}\,.$$

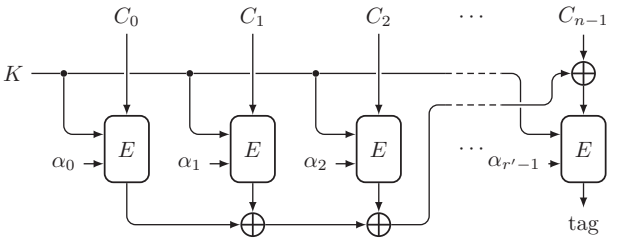

Figure 2: tPMAC, the Tweaked Parallel Message Authentication Code.

The round keys k'_i are defined by $k'_i = \tau_{256}(k_i)$, where τ_s is the function that retains only the s least significant bits of its input. The 64 most significant bits of k_i are not used to determine k'_i. The 256 least significant bits of the difference between k'_i and k_{i+1} is the truncation of $3 \cdot k_i$ (3 represents $x + 1$) and therefore there are no fixed cancellation relations between k'_i and k_{i+1}.

5 Integrity Tags and Modes of Operation

5.1 Memory Protection

MATTER encrypts 512-bit blocks, so, in the context of encryption of CLs, we only consider CLs whose lengths are a multiple of 512 bits.

Let us first discuss encryption only. In the context of direct encryption, avoiding ciphertext expansion is desirable, otherwise a keystream could be used. For CLs longer than 512 bits, once the first block has been encrypted by MATTER and thus the first core key k has been computed, each further block is encrypted by using $\texttt{MATTER}^{\text{core}}$, with each core key derived by multiplying the previous one by, say, 5 (i.e. $x^2 + 1$) in $\mathbb{F}_{2^{320}}$. Other field elements can be used, as long as they are not a power of 2 or 3 times a power of 2, to avoid reusing the round key differences mentioned in Section 4.

Let us now discuss the case where a tag is generated. If the length of a CL is equal to 512 bits, a tweaked encryption of the 64 or 128 least significant bits of the CL is used as a tag, possibly truncated to a shorter length. The tweak must contain the physical memory address of the CL. Following [JLK+23], good candidates for this encryption operation include reduced-round versions of $\texttt{QARMA}_{64}$ or $\texttt{QARMA}_{128}$ [Ava17], or the corresponding versions of QARMAv2 [ABD+23], depending on the required tweak size to hold the physical address and other system information. The TBC-based *Parallel MAC* (PMAC) [Rog04] (or tPMAC) is particularly suitable for memory integrity. It is represented in Figure 2, where the tweaks α_0 contain the memory addresses of the individual 64-bit or 128-bit ciphertext blocks. This way of computing the tag does not only provide authenticity, but also allows error detection and correction of errors up to few bits [JLK+23].

Note that replay attacks are possible unless the tags are themselves protected.

The integrity key must be generated independently of the encryption key. It is a common requirement that memory regions belonging to different processes have different encryption keys. However, a single integrity key for the system is sufficient, as the tag, viewed as a function of the ciphertext, depends on the encryption key.

5.2 General Purpose Usage

For conciseness, we do not present a full description of a general purpose *Authenticated Encryption with Associated Data* (AEAD) mode of encryption based on MATTER, but we

outline the key points. The mode is a variant of Flat-ΘCB [IMO+22], itself derived from ΘCB, and its cost is one encryption per block, plus one encryption to finalize the tag.

When using MATTER with variable length messages, the tweak for the first 512-bit block contains a unique *Initialization Vector* (IV). Once this tweak is encrypted to obtain the first core key, the successive ones can be obtained in one of two ways: either encrypting successive values of the tweak, or by multiplying the first encrypted tweak repeatedly by some element of $\mathbb{F}_{2^{320}}$, say 5, as in the memory encryption case. For the final partial block, ciphertext stealing [Dwo11] (the method can be traced back to [MM82, p. 78]) is used.

An alternative to ciphertext stealing is to treat MATTER as format preserving encryption, i.e., for a b-bit final partial block, to have two branches of $b_l = \lfloor b/2 \rfloor$ and $b_r = \lceil b/2 \rceil$ bits. This can be obtained by simply masking to the intermediate results of the 512-bit wide MATTER function, that is the inputs and the compression function. This requires $b > 1$ (this is usually not problem as the bit-lengths of messages are often multiples of 8). Domain separation is necessary when this happens.

To compute an s-bit tag, an encrypted checksum of the plaintext is used, similarly to ΘCB [Rog04], where the total length of the message is included in the tweak of the final encryption, together with domain separation. Generalizing the approach in [IM19], we generate the tags by taking the 64, 128, or 256 least significant bits of each 512-bit plaintext block, adding these values, and encrypting the result including the length of the message in the tweak (for instance, using QARMAv2 for the smaller tags, but many choices are available including MATTER itself).

In presence of *Associated Data* (AD) A, the contributions to the tag from A are computed on an encryption of the latter. There is no need to use ciphertext stealing or a special format preserving version of MATTER to encrypt the last block, if fractional, of the AD: The last block is simply padded using the string 10^*, if necessary, before encryption, with the length of A included in the tweak with domain separation. Then, the sum of the 64, 128, or 256 least significant bits of each 512-bit encrypted block is saved, to be added to the encrypted checksum of the public part to form the tag.

6 Security

Assume that ASCON with sufficiently many rounds (e.g., 12 or more) is used, to ensure it can be considered a PRP in an Even-Mansour [EM91] construction.

Provable security analysis of MATTER follows known results, assuming the ASCON permutation is ideal. For simplicity we sketch the proof for non-tweakable case, with four Feistel rounds. Let q be the number of encryption or decryption queries of 512-bit messages, and p be the number of primitive queries to ASCON-p.

First, using a hybrid argument on the round function, similar to MEM, we get ASCON using four independent 320-bit random permutations with expansion and truncation (for 256-bit I/O) at a cost of $q \cdot p/2^{320} + q^2/2^{320}$. Applying the PRP-PRF switching lemma (cf. [BR06] and references therein) brings an additional cost of $q^2/2^{320}$. Thus, MATTER with four independent 256-bit random functions with a distinguishing advantage of $q^2/2^{256}$. The total distinguishing advantage is $O(q \cdot p/2^{320} + q^2/2^{256})$.

Improvements are possible. Using the classical result of Naor and Reingold [NR97] instead of the original arguments of Luby-Rackoff, the top and bottom only need to be almost XOR-universal hash functions, relaxing the first hybrid argument, i.e., allowing fewer ASCON rounds for the top and bottom Feistel functions. This permits constructions such as $\texttt{MATTER}_{6,12,6}$ in place of $\texttt{MATTER}_{8,8,8}$, with the same latency but possibly better security. Advanced constructions such as [GW18, CLL19] can reduce the number of independent keys. However, the main bottleneck would be a collision on 256-bit branches, meaning 128-bit security.

Assuming $p \ll 2^{64}\, q$, the first summand in the distinguishing advantage is negligible

Table 1: Comparison of Direct Memory Encryption Primitives in TSMC 5nm Process.

				Area optimized			*Latency optimized*		
				Area		Delay	Area		Delay
Cipher		Width	Rounds	μm^2	GE	ps	μm^2	GE	ps
AES-128		128	10	2304.1	28873	3064	4520.6	56648	1791
AES-128/XEX	(write)	128	$10+10$	4688.3	58750	6156	9122.9	114320	3610
	(read)					3092			1819
AES-192		128	12	2635.4	33025	3686	5023.6	62952	2153
AES-192/XEX	(write)	128	$12+12$	5352.6	67074	7400	10129.0	126928	4334
	(read)					3713			2178
AES-256		128	14	3238.7	40585	4290	6191.5	77587	2513
AES-256/XEX	(write)	128	$14+14$	6559.2	82194	8607	12464.8	156198	5053
	(read)					4316			2538
QARMA-128	($r=11$)	128	24	1635.6	20496	1561	3078.3	38575	1091
QARMAv2-128-128	($r=11$)	128	24	1620.3	20305	1409	2875.8	36037	1068
QARMAv2-128-192	($r=13$)	128	28	1893.5	23727	1645	3333.0	41778	1248
QARMAv2-128-256	($r=15$)	128	32	2166.8	27152	1879	3797.8	47592	1425
ASCON-p^{12}		320	12	2228.3	27923	826	2766.8	34671	507
MATTER$^{8}_{8,8,8}$	(write)	512	$8+24$	6309.9	79069	2299	7745.8	97064	1448
	(read)					1724			1086
MATTER$^{6}_{6,6,6,6}$	(write)	512	$6+24$	6040.4	75567	2085	7376.6	92437	1288
	(read)					1668			1030

compared to the second. Thus, with memory for q plaintext/ciphertext pairs, the advantage is roughly $q^2/2^{256}$, requiring $\propto 2^{256}/q^2$ attempts to succeed. The total time is dominated by the invocations to the encryption and decryption oracles, as well as to the ASCON permutation, so we have approximately $\propto 2^{256}/q$ data and time with q memory. Note that the memory requirements and running time may increase if $p > q$.

Now, distinguishability often leads to key-recovery attacks, with usually comparable complexity. This leads to a memory-time tradeoff $M \cdot T \propto 2^{256}$.

7 Implementations

In Table 1 we pit MATTER$^{8}_{8,8,8}$ and MATTER$^{6}_{6,6,6,6}$ against two other families of choices of block ciphers for memory encryption in a direct mode: the AES and QARMA-128/QARMAv2-128. These two particular instantiations of MATTER have area and latency similar to other options such as, say, MATTER$^{8}_{4,4,4,4,4,4}$. The actual security level of these instantiations of MATTER has not been determined yet through cryptanalysis, but we expect that it will be above 128 bits and approach 256 bits in the memory-time tradeoff sense of Section 6.

For the ciphers we either report implementation results in a low-voltage TSMC 5nm lithography with the tsmc_sch280pp57_cln05fb41001 library, taken from [ABD+23] or extrapolated, using the known area and latency values for XOR gates and intermediate registers in the given process. Area and latency are reported for both area optimized and latency optimized implementations.

Two delay (i.e., additional latency) values are reported for the XEX construction and for MATTER, namely for memory write and read operations. The reason is that when a memory read request is issued, the computation of the encrypted tweak can almost always be completed while the memory controller is waiting for the requested data to reach it. Therefore, in both designs only the encryption latency is on the critical path for memory reads. But, on memory writes also the tweak encryption must always be taken

into account.

Fully unrolled AES-XEX can approach the area of MATTER, but with worse latency. Using a single fully unrolled AES instance twice almost halves the area with a minor latency increase. However, since MATTER is 512 bits wide and AES is 128 bits wide, AES-XEX must be invoked four times, either as monolithic circuit or in a pipelined implementation: both solutions negatively affect latency and area. Replicating the circuit to process four blocks simultaneously makes the AES-based solution significantly more expensive in terms of area, but the latency remains much higher than MATTER.

QARMA and QARMAv2 are significantly lighter on resources, with similar latency to MATTER and lower area. However, as with the AES, invoking the circuit four times or pipelining it increases latency, and replicating it four times makes the area too large, unless the smallest 24-round version is used, in which cases the latter's area is comparable to MATTER.

For software implementations, MATTER inherits the most important advantages of ASCON [DEMS21, Section 7.2], such as suitability to bit-sliced implementations and compact code, making it ideal for restricted environments.

8 (Temporary) Conclusion: Open Questions

While we can expect that $\texttt{MATTER}^{12}_{12,12,12}$ and $\texttt{MATTER}^{12}_{12,12,12,12}$ offer high security levels — say at 256 bits of memory-time product as in Section 6 — they have large area and high latency. To reduce latency, we can lower the number of rounds in ASCON, for example, from 12 to 8 in the three Feistel rounds version and even to 6 in the four Feistel rounds version. For five- or six Feistel round versions, we can go as low as 6 rounds in ASCON, keeping the core latency about twice that of the $\texttt{ASCON-p}^{12}$ permutation.

The question is however whether we can achieve a desired level bit-security for specific instances such as $\texttt{MATTER}^{8}_{8,8,8}$, $\texttt{MATTER}^{6}_{6,12,6}$, $\texttt{MATTER}^{6}_{6,6,6,6}$, or even other variants such as $\texttt{MATTER}^{8}_{4,8,8,4}$ and $\texttt{MATTER}^{8}_{4,4,4,4,4}$ or $\texttt{MATTER}^{8}_{4,4,4,4,4,4}$. In particular, an important goal is to find lower bounds for the classical time complexity when at most 2^{64}, 2^{80}, or 2^{96} data is available, first for the non-tweakable core and then for the entire TBC. Ad-hoc cryptanalysis is required, and it will be the subject of future investigations.

References

[ABD+23] Roberto Avanzi, Subhadeep Banik, Orr Dunkelman, Maria Eichlseder, Shibam Ghosh, Marcel Nageler, and Francesco Regazzoni. The QARMAv2 Family of Tweakable Block Ciphers. *IACR Transactions on Symmetric Cryptology*, (3):25–73, Sep. 2023. doi:10.46586/tosc.v2023.i3.25-73. Cited on pages 1, 4, and 6.

[AMS+22] Roberto Avanzi, Ionut Mihalcea, David Schall, Héctor Montaner, and Andreas Sandberg. Hardware-Supported Cryptographic Protection of Random Access Memory. Cryptology ePrint Archive, Paper 2022/1472, 2022. https://eprint.iacr.org/2022/1472. Available from: https://eprint.iacr.org/2022/1472. Cited on page 2.

[Ava17] Roberto Avanzi. The QARMA Block Cipher Family — Almost MDS Matrices over Rings with Zero Divisors, Nearly Symmetric Even-Mansour Constructions with Non-Involutory Central Rounds, and Search Heuristics for Low-Latency S-Boxes. *IACR Trans. on Symmetric Cryptology*, 2017(1):4–44, 2017. doi:10.13154/tosc.v2017.i1.4-44. Cited on pages 1 and 4.

[BBI+15] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A Block Cipher for Low Energy. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology — ASIACRYPT 2015 — 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November*

29–December 3, 2015, Proceedings, Part II, volume 9453 of *LNCS*, pages 411–436. Springer, 2015. doi:10.1007/978-3-662-48800-3_17. Cited on page 1.

[BCG+12] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander, Ventzislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçin. PRINCE — A Low-Latency Block Cipher for Pervasive Computing Applications — Extended Abstract. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology — ASIACRYPT 2012 — 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 208–225. Springer, 2012. doi:10.1007/978-3-642-34961-4_14. Cited on page 1.

[BDD+23] Yanis Belkheyar, Joan Daemen, Christoph Dobraunig, Santosh Ghosh, and Shahram Rasoolzadeh. BipBip: A Low-Latency Tweakable Block Cipher with Small Dimensions. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(1):326–368, 2023. Available from: https://doi.org/10.46586/tches.v2023.i1.326-368, doi:10.46586/TCHES.V2023.I1.326-368. Cited on pages 1 and 3.

[Ber08] Daniel J. Bernstein. The ChaCha family of stream ciphers. See https://cr.yp.to/chacha.html, January 2008. Cited on page 1.

[BJK+16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. 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 II*, volume 9815 of *LNCS*, pages 123–153. Springer, 2016. doi:10.1007/978-3-662-53008-5_5. Cited on page 1.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems — CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007. Available from: https://doi.org/10.1007/978-3-540-74735-2, doi:10.1007/978-3-540-74735-2_31. Cited on page 1.

[BR06] Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006. doi:10.1007/11761679_25. Cited on page 5.

[BSS+13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK Families of Lightweight Block Ciphers. Cryptology ePrint Archive, Report 2013/404, 2013. Available from: http://eprint.iacr.org/2013/404. Cited on page 1.

[CDK09] Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. KATAN and KTANTAN — A Family of Small and Efficient Hardware-Oriented Block Ciphers. In Christophe Clavier and Kris Gaj, editors, *CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, 2009. Cited on page 1.

[CLL19] Wonseok Choi, ByeongHak Lee, and Jooyoung Lee. Indifferentiability of Truncated Random Permutations. 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 I*, volume 11921 of *Lecture Notes in Computer Science*, pages 175–195. Springer, 2019. doi:10.1007/978-3-030-34578-5_7. Cited on page 5.

[DEMS21] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1.2: Lightweight Authenticated Encryption and Hashing. *J. Cryptol.*, 34(3):33, 2021. doi:10.1007/s00145-021-09398-9. Cited on pages 1 and 7.

[DR02] Joan Daemen and Vincent Rijmen. AES and the Wide Trail Design Strategy. 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 108–109. Springer, 2002. doi:10.1007/3-540-46035-7_7. Cited on page 1.

[Dwo11] Morris Dworkin. Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode. Addendum to NIST Special Publication 800-38A [NIS01]. Technical report, National Institute of Standards and Technology, Gaithersburg, MD, United States, October 2011. Cited on page 5.

[EM91] Shimon Even and Yishay Mansour. A Construction of a Cipher from a Single Pseudorandom Permutation. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *Advances in Cryptology — ASIACRYPT '91, International Conference on the Theory and Applications of Cryptology, Fujiyoshida, Japan, November 11-14, 1991, Proceedings*, volume 739 of *LNCS*, pages 210–224. Springer, 1991. doi:10.1007/3-540-57332-1_17. Cited on page 5.

[GJMN16] Robert Granger, Philipp Jovanovic, Bart Mennink, and Samuel Neves. Improved Masking for Tweakable Blockciphers with Applications to Authenticated Encryption. 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 *LNCS*, pages 263–293. Springer, 2016. doi:10.1007/978-3-662-49890-3_11. Cited on page 3.

[GNL11] Zheng Gong, Svetla Nikova, and Yee Wei Law. KLEIN: A New Family of Lightweight Block Ciphers. In Ari Juels and Christof Paar, editors, *RFID. Security and Privacy — 7th International Workshop, RFIDSec 2011, Amherst, USA, June 26-28, 2011, Revised Selected Papers*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011. doi:10.1007/978-3-642-25286-0_1. Cited on page 1.

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems — CHES 2011 — 13th International Workshop, Nara, Japan, September 28 — October 1, 2011. Proceedings*, volume 6917 of *Lecture Notes in Computer Science*, pages 326–341. Springer, 2011. doi:10.1007/978-3-642-23951-9_22. Cited on page 1.

[GW18] Chun Guo and Lei Wang. Revisiting Key-Alternating Feistel Ciphers for Shorter Keys and Multi-user Security. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*, volume 11272 of *Lecture Notes in Computer Science*, pages 213–243. Springer, 2018. doi:10.1007/978-3-030-03326-2_8. Cited on page 5.

[HT13] Michael Henson and Stephen Taylor. Memory Encryption: A Survey of Existing Techniques. *ACM Comput. Surv.*, 46(4):53:1–53:26, 2013. doi:10.1145/2566673. Cited on page 2.

[IM19] Akiko Inoue and Kazuhiko Minematsu. Parallelizable Authenticated Encryption with Small State Size. In Kenneth G. Paterson and Douglas Stebila, editors, *Selected Areas in Cryptography — SAC 2019 — 26th International Conference, Waterloo, ON, Canada, August 12-16, 2019, Revised Selected Papers*, volume 11959 of *Lecture Notes in Computer Science*, pages 618–644. Springer, 2019. doi:10.1007/978-3-030-38471-5_25. Cited on page 5.

[IMO+22] Akiko Inoue, Kazuhiko Minematsu, Maya Oda, Rei Ueno, and Naofumi Homma. ELM: A Low-Latency and Scalable Memory Encryption Scheme. *IEEE Trans. Inf.*

Forensics Secur., 17:2628–2643, 2022. doi:10.1109/TIFS.2022.3188146. Cited on page 4.

[JLK+23] Jonas Juffinger, Lukas Lamster, Andreas Kogler, Moritz Lipp, Maria Eichlseder, and Daniel Gruss. CSI:Rowhammer — Cryptographic Security and Integrity against Rowhammer. In *Proceedings of IEEE S&P '23, San Francisco, California, USA, May 22–26, 2023*, 2023. Cited on page 4.

[KDK+14] Yoongu Kim, Ross Daly, Jeremie S. Kim, Chris Fallin, Ji-Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ISCA*, pages 361–372. IEEE Computer Society, 2014. Cited on page 2.

[LR85] Michael Luby and Charles Rackoff. How to Construct Pseudo-Random Permutations from Pseudo-Random Functions (Abstract). In Hugh C. Williams, editor, *Advances in Cryptology — CRYPTO '85, Santa Barbara, California, USA, August 18-22, 1985, Proceedings*, volume 218 of *Lecture Notes in Computer Science*, page 447. Springer, 1985. doi:10.1007/3-540-39799-X_34. Cited on page 2.

[LR88] Michael Luby and Charles Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. *SIAM J. Comput.*, 17(2):373–386, 1988. doi:10.1137/0217022. Cited on page 2.

[LRW02] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable Block Ciphers. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, volume 2442 of *LNCS*, pages 31–46. Springer, 2002. doi:10.1007/3-540-45708-9_3. Cited on page 1.

[LRW11] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable Block Ciphers. *Journal of Cryptology*, 24(3):588–613, 2011. doi:10.1007/s00145-010-9073-y. Cited on page 1.

[MM82] Carl H. Meyer and Stephen M. Matyas. *Cryptography: a new dimension in computer data security.* John Wiley & Sons, 1982. Cited on page 5.

[NIS01] NIST. Recommendation for Block Cipher Modes of Operation: Methods and Techniques. Technical report, National Institute of Standards and Technology, Gaithersburg, MD, United States, January 2001. Cited on page 9.

[NR97] Moni Naor and Omer Reingold. On the Construction of Pseudo-Random Permutations: Luby-Rackoff Revisited (Extended Abstract). In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 189–199. ACM, 1997. doi:10.1145/258533.258581. Cited on page 5.

[Rog04] Phillip Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In *Advances in Cryptology — ASIACRYPT 2004, 10th International Conference on the Theory and Application of Cryptology and Information Security, Jeju Island, Korea, December 5-9, 2004, Proceedings*, pages 16–31, 2004. doi:10.1007/978-3-540-30539-2_2. Cited on pages 3, 4, and 5.

[Sor84] Arthur Sorkin. Lucifer, a Cryptographic Algorithm. *Cryptologia*, 8(1):22–42, 1984. doi:10.1080/0161-118491858746. Cited on page 2.

[SSA+07] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-Bit blockcipher CLEFIA. In Alex Biryukov, editor, *FSE 2007*, volume 4593 of *LNCS*, pages 181–195. Springer, 2007. doi:10.1007/978-3-540-74619-5. Cited on page 1.