The Design Space of Lightweight Cryptography

Nicky Mouha[1,2]

[1] Dept. Electrical Engineering-ESAT/COSIC, KU Leuven, Leuven and iMinds, Ghent, Belgium.
[2] Project-team SECRET, Inria, France.
Firstname.Lastname@esat.kuleuven.be

Abstract. For constrained devices, standard cryptographic algorithms can be too big, too slow or too energy-consuming. The area of lightweight cryptography studies new algorithms to overcome these problems. In this paper, we will focus on symmetric-key encryption, authentication and hashing. Instead of providing a full overview of this area of research, we will highlight three interesting topics. Firstly, we will explore the generic security of lightweight constructions. In particular, we will discuss considerations for key, block and tag sizes, and explore the topic of instantiating a pseudorandom permutation (PRP) with a non-ideal block cipher construction. This is inspired by the increasing prevalence of lightweight designs that are not secure against related-key attacks, such as PRINCE, PRIDE or Chaskey. Secondly, we explore the efficiency of cryptographic primitives. In particular, we investigate the impact on efficiency when the input size of a primitive doubles. Lastly, we provide some considerations for cryptographic design. We observe that applications do not always use cryptographic algorithms as they were intended, which negatively impacts the security and/or efficiency of the resulting implementations.

Keywords: Symmetric-key, encryption, authentication, hash function, lightweight, constrained devices, scaling law.

1 Introduction

Lightweight cryptography is commonly defined as cryptography for resource-constrained devices, for which RFID tags and wireless sensor networks are typically mentioned as examples [23]. The goal of lightweight cryptography is to enable a diverse range of modern applications, such as smart meters, vehicle security systems, wireless patient monitoring systems, Intelligent Transport Systems (ITS) and the Internet of Things (IoT) [21, 76, 79].

As such, lightweight cryptography targets a very wide variety of devices. They can be implemented on a broad range of hardware and software, and their communication can be either wired or wireless. Wireless devices can be powered either by electromagnetic induction or by a battery, which can be disposable or rechargeable.

Therefore for some applications, either energy or power consumption is critical, whereas for other applications a low latency is much more important. It can

be that hardware area or software code size is a limiting factor, or that only a small amount of RAM is available. More often than not, a combination of the aforementioned criteria needs to be satisfied.

What sets lightweight cryptography apart from "conventional cryptography?" Although the question seems simple, this appears to be a quite controversial subject. The reason is that the distinction between "lightweight" and "conventional" blurs when we ask ourselves: "If a lightweight algorithm outperforms conventional algorithms, should it then not be considered to be a replacement for conventional algorithms?"

The goal of this paper is to shed some light on this discussion, yet we do not claim to provide a definitive answer. We hope to give a better insight into lightweight cryptography, and explain how to achieve design goals that cannot be met by conventional cryptographic algorithms. To do this, we will combine knowledge of provable security, cryptanalysis and implementation, with a particular focus on recent academic advancements.

It is important to note that lightweight cryptography should not be equated with weak cryptography. Lightweight designs have been proposed that are significantly weaker than conventional algorithms, for example by using short key or block sizes. We strongly discourage the use of such algorithms, and will counter common arguments that are used to defend them.

The focus of this paper will be on symmetric-key encryption, authentication and hashing, nevertheless some of its insights may also apply to public-key cryptography.

2 How to Measure Security

A key insight of modern cryptography is to split the security analysis of any protocol or construction into two parts. The first part is the analysis of the protocol or construction with ideal primitives. This allows us to evaluate the security against *generic attacks*, that is, attacks that are inherent to the construction and not due to any flaws in the underlying primitives. The second part is the analysis of the algorithms by which these idealized primitives are instantiated.

2.1 The Security against Generic Attacks

A commonly used model to analyze the security against generic attacks, is the information-theoretic framework. It considers the ideal primitives not as deterministic algorithms, but as statistical objects. In the same way that the outcome of a dice roll is not known before the dice is rolled, the output of an ideal primitive is not known until it is *queried*. It therefore becomes possible to consider *computationally unbounded adversaries*, that are only restricted by the amount of information that they can collect (expressed in the number of queries to the ideal primitives), and not by any computational restrictions on how this information is processed.

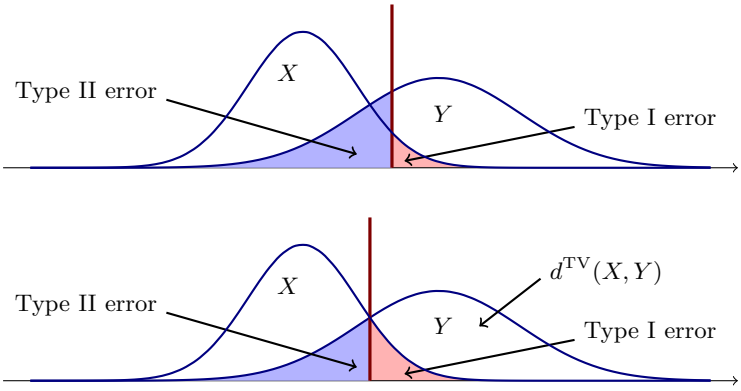

Fig. 1. (Top) A hypothesis test to distinguish between random variables X (null hypothesis) and Y (alternative hypothesis), showing the probability to make Type I and Type II errors; (Bottom) The hypothesis test for X and Y where the probability of making an error (Type I or Type II) is minimized. The two white regions are equal in area, each of which corresponds to the total variation distance $d^{\mathrm{TV}}(X,Y)$.

The information-theoretic framework not only greatly simplifies the security analysis; it also means that the analysis still remains valid, regardless of future algorithmic improvements. For example, the collision resistance of a hash function is then measured by the number of messages required for a collision to *exist* with a sufficiently high probability, and not by the currently best-known algorithm to efficiently *find* a collision. In fact, for this particular example, it has been shown that it is not necessary to store all messages and hash values in a table. Collisions for meaningful messages can be found with negligible memory requirements [66], even with efficient parallel implementations [62].

The *total variation distance* is often used in cryptography to analyze the security against generic attacks. It corresponds to the maximum success probability[3] to distinguish between two distributions: the "real world" and the "ideal world." The total variation distance can be defined as follows.

Definition 1 (Total Variation Distance). *Let X and Y be two random variables on a finite*[4] *set Ω. The* total variation distance *between X and Y is defined as*

$$d^{\mathrm{TV}}(X,Y) \triangleq \max_{A \subset \Omega} |\Pr[X \in A] - \Pr[Y \in A]| \ .$$

[3] As there are several ways to calculate this success probability, it is often preferable to use more precise terminology instead. Following Bellare and Rogaway [13, Chapter 3], we therefore clarify that this term should be understood "in a specific, technical way based on the definition."

[4] The finiteness assumptions are only made for simplicity. It is straightforward to generalize the notions, as explained for example in [74].

The total variation distance satisfies all the requirements of a distance function (non-negativity, identity of indiscernibles, symmetry, triangle inequality). For an accessible introduction to the total variation distance and its properties, we refer to [54, Chapter 4]. It is interesting for readers with an engineering background to note that the total variation distance is related to hypothesis testing. If we consider the hypothesis test with the lowest probability of making an error (Type I or Type II), then the total variation distance is equal to the probability that this hypothesis test makes the right decision [77]. The concept of total variation distance and its relation to hypothesis testing is illustrated in Fig. 1.

In cryptographic literature, we often find another, equivalent way to calculate the total variation distance between two distributions. It is known as the *maximum adversarial advantage*, and can be defined as follows.

Definition 2 (Maximum Adversarial Advantage). *Let X and Y be two random variables on a finite set Ω. The class of algorithms (probabilistic or deterministic – it does not matter) with input $x \in \Omega$ and output 0 or 1 is denoted by $\mathbf{A}$, and let $\mathcal{A} \in \mathbf{A}$. Then the* maximum adversarial advantage *to distinguish between X and Y is defined as*

$$d^{\mathrm{Adv}}(X, Y) \triangleq \max_{\mathcal{A} \in \mathbf{A}} |\Pr[\mathcal{A}(X) = 1] - \Pr[\mathcal{A}(Y) = 1]| \ .$$

The total variation distance and the maximum adversarial advantage are equal, see e.g. [60] for a proof.

A number of proof techniques exist to calculate the total variation distance, including game-playing [14], Patarin's H-coefficient technique [63] (see Chen and Steinberger [34] for detailed discussion of this technique) and coupling (a well-known technique from the theory of Markov chains [54, Chapter 5]).

For symmetric-key encryption and authentication, it makes sense to consider two types of queries:

- D-queries refer to queries that are evaluated under a secret key, which is unknown to the adversary. This can be, for example, encryption with a block cipher under a secret key.
- T-queries do not involve a secret key. An example is the same block cipher, where the key is not secret but provided as an extra input for the adversary.

Let D be the number of D-queries, and T the number of T-queries. Often D is referred to as the *data complexity* and T is referred to as the *time complexity* of the attack [37]. Note that the actual time complexity may be higher. As we explained earlier, we consider information-theoretic adversaries and therefore the cost of processing the queries is not taken into account. This also explains why we do not consider the *memory complexity* of attacks. Clearly, this analysis is very optimistic from the adversary's point of view, which is why it is useful to construct security bounds.

The encryption algorithm is assumed to be known to the adversary (due to Kerckhoffs's principle), which is why we give the adversary access to T-queries. Security results for block-cipher-based modes of operation (e.g. [12]) typically

do not involve such T-queries, as they are usually expressed in terms of pseudorandom permutation (PRP) security. The T-queries appear when this PRP is instantiated with a block cipher construction, such as an ideal block cipher, an Even-Mansour [42, 43] block cipher, a key-alternating cipher [28],...

Note that we obviously do not intend to critique proofs based on PRP security. It is precisely thanks to these proofs that we can instantiate PRPs in different ways, and have the security results carry over. For a rigorous explanation of why the original PRP-based proofs still hold when additional access to T-queries is introduced, we refer to [59].

2.2 Considerations for Key, Block and Tag Sizes

It is often reasonable to assume that the number of D-queries is limited, as they rely on the secret key and must therefore be generated by the device under attack. The number of T-queries is typically much higher: as the algorithm is public, these can be performed without access to the device, for example using custom-built hardware or cloud computing.

Although it is reasonable to assume that the maximum number of D-queries that an attacker can make is smaller than the maximum number of T-queries, we must be very careful when putting specific limits on D and T. Unfortunately all too frequently, especially in the area of lightweight cryptography, it is argued that the attacker is very limited in D and T, because "the key is changed every half hour," or "the data is not worth a million dollars." The argument is that the attacker will not be able to break the encryption as only a limited amount of time and data is available, especially if the monetary value of the secret is too low to justify the attacker's investment. We will now explain the flaws in this argument.

Multi-key Attacks. It is often implicitly assumed that the attacker's goal is to break the encryption under *one* particular key. Under that assumption, it makes sense to change the key frequently to limit the amount of data available to the attacker. However, the confidentiality of encryption breaks down if an attack is successful for *any* of the keys that are being used. This setting should be of particular concern. As shown by Biham [17, 18], faster generic key-recovery attacks exist on any block cipher when multiple keys are used. Mouha and Luykx explained how modern Internet protocols are particularly vulnerable to this attack [58], for example when JavaScript malware is generating a large number of connections.

For MAC functions, multi-key attacks seem to be of a lesser concern. When any of the old keys is compromised, the attacker can generate forgeries under this key. However, unlike in the case of encryption, compromise of an old authentication key does not result in the compromise of other secrets. This is because in a scenario where messages are only authenticated but not encrypted, messages are sent in the clear and therefore known to the attacker already. The leak of an old key should not impact the authenticity of messages under the current key. A

possible concern, however, is that an attacker may target several devices at the same time. This possibility should be carefully considered.

The Power of Precomputation. The use of short keys should be discouraged for any application. Exhaustive key search may become feasible when short keys are used, although this may require a large amount of time and memory. However, this large investment only needs to be made once as part of a precomputation phase, after which any number of keys can be attacked. This is the basis of the time-memory tradeoff by Hellman [50]. During a precomputation phase that is equal to exhaustive search, a table is constructed. By means of this table, individual keys can then be recovered quickly.

This was done, for example, by Nohl et al. [61] for the A5/1 encryption algorithm used by GSM. A5/1 has an effective key size of 61 bits. A large precomputation equivalent to dozens of GPU years was used to construct a table of about 1.6 terabytes, using a variant of Hellman's technique. Using this table, any A5/1 key can be broken in about five seconds using commodity hardware.

As this example clearly shows, it makes little sense to argue that short keys are acceptable because they are changed frequently, or because the investment to break one particular key would be too high for the attacker. This is because the large precomputation needs to be done only once for every algorithm, after which individual keys can be broken at a low cost and in a very short amount of time.

Hellman's time-memory trade-off requires exhaustive key search in a precomputation phase. However, the time-memory-data tradeoffs for stream ciphers of Babbage-Golić [7,48] and Biryukov-Shamir [22] can have a time complexity that is far below exhaustive search. As Biryukov et al. [20] showed, the time-memory-data tradeoffs also apply to block ciphers when we consider the aforementioned multi-key scenario. For example, if a block cipher with a 112-bit key size is used, one key out of 2^{32} can be found in 2^{32} time, after a one-time precomputation of 2^{80} to generate a table of size 2^{64} [20].

In fact, Nohl et al.'s attack on A5/1 is actually also a time-memory-data tradeoff, although with a very limited amount of data. The attack uses one 114-bit GSM burst which therefore contains 51 ciphertexts of 64 bits [31].

Just Keep Guessing. For MAC functions and authenticated encryption algorithms, a forgery attack is possible even when T and D are zero. Forgeries can be attempted by simply guessing the tag value, which is especially a concern when short tags are used. The NIST recommendations for CMAC [38] and GCM/GMAC [39] state that in order to avoid guessing attacks, the system should "limit the number of unsuccessful verification attempts for each key." However, rekeying obviously does not give any protection against guessing attacks, as the success probability of a guessing attack depends on the tag length and not on the key. The attacker can thus continue the guessing attack, regardless of how many times the key is changed.

Short Block Sizes? We previously discussed the dangers of short key lengths. But short block lengths can also become a concern, especially when the message is longer than the block size. For randomized encryption, such as CBC mode with a random IV, the attacker will have an increased probability to guess the IV value. For on-line deterministic encryption, it becomes easier to recover the plaintext block by block by means of a chosen-plaintext attack [51].

These problems do not present themselves for plaintexts with a length of at most one block. However, the key will then need to be changed for every block to avoid the problems that are inherent to ECB mode. In the context of format-preserving encryption, a tweak is often used here instead of a new key for efficiency reasons [40].

2.3 Instantiating Ideal Primitives with Algorithms

Care must be taken when an ideal primitive is instantiated with a particular algorithm. The "trick" here is that when an ideal primitive is assumed to be "random until queried," this informally means that the algorithm by which it is instantiated should then be "random until evaluated." It does not seem to be possible to formulate the instantiation with an algorithm in a rigorous way. In fact, a variety of constructions [10,32,47,55] show theoretical problems with this type of instantiation, although these constructions differ significantly from the algorithms that are used in practice.

Another problem with instantiation is that speed-ups of exhaustive search exist for any algorithm, for example the "distributive technique" and the "early abort technique" that are well-known in cryptographic folklore [19]. Even faster techniques to speed-up exhaustive search may exist for certain algorithms, for example using the recently introduced framework of biclique attacks [24]. It therefore makes sense to be conservative when estimating the generic time complexity that is required for exhaustive search.

One way to take these observations into account, is to introduce a safety margin by assuming that exhaustive search can always be sped up by a small factor. This is similar to the motivation behind the "green category" of the ECRYPT II SHA-3 Zoo [41]. Allowing such "bruteforce-like attacks" by design seems to be an interesting way to satisfy the requirements of lightweight cryptography.

It should be pointed out that bruteforce-like attacks still correspond to exhaustive search, although on a smaller part of the algorithm. An example of this type of attack are the biclique attacks on the full AES [24].

2.4 Insights for Lightweight Cryptography

In the previous sections, we provided a thorough explanation of how to measure security against generic attacks. More specifically, we explained how for a particular symmetric-key cryptographic algorithm (for example with key size k, block size n and tag size t), we can evaluate the success probability of any generic attack with a given amount of time and data complexity.

It should be taken into account that the attacker may perform a large precomputation, and that multiple devices may be under attack. In many protocols, it is possible that keys are changed frequently. For encryption algorithms, this leads to an inevitable erosion of security and should therefore be considered.

Our explanation of generic attacks was not only meant as a warning against bad parameter choices, but also to provide insight into how we can realize the lightweight cryptography goal of "getting more for less." Below we provide some examples of how that can be done.

- If the implementation can guarantee that keys are chosen independently and uniformly at random, resistance against related-key attacks is not necessary. This insight is used in several recent lightweight designs, such as PRINCE [29], PRIDE [1] and Chaskey [59]. Note that designs without security against related-key attacks are not new: DES [56] and Triple-DES [8] are trivially vulnerable to related-key attacks due to the complementation property.
- Many block-cipher-based modes of operation are only secure up to about $2^{n/2}$ blocks [12]. The underlying block ciphers could then be designed to be secure only up to $2^{n/2}$ plaintext-ciphertext pairs, as higher security would be overkill for the mode of operation in which it is used. This design philosophy is used, for example, in the CAESAR candidates Prøst [53], Minalpher [72], and the MAC function Chaskey.
- The design of a secure permutation algorithm is typically a very difficult task. However, when this permutation is used inside a keyed primitive, a wide variety of attacks become inapplicable. This insight was used in the Chaskey MAC function: the security is reduced to that of the permutation-based block cipher, and not to the permutation itself. This allows a much lighter permutation to be used. Of course, the security of the block cipher algorithm must still be investigated by cryptanalysis.

3 The Efficiency of Underlying Primitives

3.1 Measuring Efficiency

To evaluate the efficiency of cryptographic protocols, modes of operation, or primitives, an efficiency metric must be introduced. Typical examples are the number of modular exponentiations, the number of block cipher calls per plaintext block, or the number of permutation calls per message block.

We should be very careful when introducing such metrics, as we risk measuring the "theoretical efficiency" instead of the "actual efficiency" of a protocol or algorithm.

Observe that certain designs are only efficient for long messages, whereas messages in lightweight applications are typically very short. Some designs rely on nonces (numbers used only once) for their security, which are difficult to ensure in low-end systems, as they require either non-volatile memory to store a counter or a hardware source of randomness. We also note that some designers

focus only on the number of calls to an underlying primitive to measure efficiency, whereas the potential for parallelism is much more important for low-latency applications.

Our intention is not to criticize any particular design. In fact, we are confident that most designs will find their way to applications for which they are particularly well-suited. Instead, what we want to argue is that the communication between theoreticians, cryptanalysts and implementers has failed when a design does not meet the requirements of the application for which it was intended. One solution here can be to introduce more refined efficiency metrics, which will hopefully lead to new designs that can match the stringent requirements imposed by lightweight cryptography.

The goal of this paper is not to give a definitive answer on how to measure efficiency. However, we intend to provide a step in the right direction, that will hopefully lead to a better understanding. More specifically, the following section will introduce a scaling law for symmetric-key cryptography.

3.2 Scaling Law for Symmetric-Key Cryptography

Motivation. In the context of hash function design, Rogaway and Steinberger proved that at least three n-bit compression functions are needed to construct a $2n$-to-n-bit compression function with an optimal collision resistance of $2^{n/2}$ [68]. They presented a construction that achieves this bound in [67]. In later work, a construction was given by Mennink and Preneel that uses only XOR operations [57].

We note that the same $2n$-to-n-bit compression function can also be achieved with one $2n$-bit permutation, where half of the output is truncated. In fact, this construction even reaches a preimage resistance of 2^n, compared to only $2^{2n/3}$ for designs based on three n-bit permutations. This raises the question: is it more efficient to use three n-bit permutations, or one $2n$-bit permutation?

Scaling Law. To answer this question, we need to introduce a model of computation. In what follows, we will use a biclique-style approach [24] where the computational cost is determined by the number of basic operations: S-box evaluations, Skein MIX functions,... We will assume that a 64-bit addition is twice as costly as a 32-bit addition, which is roughly true on hardware, as well as on software if two 32-bit additions can be performed by one SIMD instruction.

This model obviously has its limitations, and is clearly not "science" but "engineering." It should nevertheless be a useful starting point to compare primitives of different sizes.

We formulate the scaling law for symmetric-key primitives (block ciphers, compression functions, cryptographic permutations,...) as follows:

> *"When the input size of a symmetric-key primitive doubles, the number of operations (roughly) doubles as well".*

This scaling law is seemingly contradicted by information theory. We would expect a superexponential increase of the number of operations if the input size doubles, as there are $(2^n)^{2^n} = 2^{n2^n}$ functions from n to n bits. This reasoning does not hold for practical designs, however. An obvious counterexample is that the running time of the 128-bit block cipher AES [35] is not superexponentially higher than that of the 64-bit block cipher DES [56].

We will therefore not use information theory to support the scaling law, but we will use cryptographic designs as examples. These are in turn influenced by the best known attacks, as well as by the designers' intuitions for an acceptable security margin against known attacks. We consider two cases: algorithms with a fixed word size, and algorithms where the word size is variable. It seems difficult to define unambiguously what the "word size" of a given algorithm is, however hopefully the examples below will clarify this concept.

Fixed Word Size. The PHOTON family of lightweight hash functions has a word size of 4 bits, corresponding to the size of the S-box. This holds for the 100-bit, 144-bit, 196-bit and 256-bit PHOTON permutations, we will not consider the 288-bit permutation as it uses an 8-bit S-box. As the number of rounds for PHOTON is always 12, this means that the number of S-box evaluations doubles if the permutation size doubles.

In the case of Rijndael [35] with a 256-bit key, the number of rounds is always 14, regardless of the block size (128, 192 or 256 bits). Therefore the number of S-box evaluations doubles if the block size doubles.

The SHA-3 finalist Skein [45] processes words of 64 bits. When the key and block size of the underlying Threefish block cipher is 256 or 512 bits, the number of rounds is 72. Therefore the number of Skein MIX operations doubles when the word size doubles. When the key and block size are 1024 bits, Threefish uses 80 rounds. Note that Skein seems to have a very large security margin, as the best known non-biclique attack is on only 36 rounds [80].

Variable Word Size. The SHA-3 finalist BLAKE [6] processes a 960-to-256-bit compression function in 14 rounds using 32-bit words. The 1920-to-512-bit compression function has 16 rounds and uses 64-bit words.

The hash function SHA-256 [69] uses a 768-to-256-bit compression function with 64 rounds using 32-bit words, whereas SHA-512 uses a 1536-to-512 bit compression function with 80 rounds using 64-bit words.

Keccak [16], the winner of the SHA-3 competition, uses a 800-bit permutation with 22 rounds when the word size is 32 bits, and a 1600-bit permutation with 24 rounds when the word size is 64 bits. It should be noted that a zero-sum distinguisher exists for the full-round 1600-bit permutation [30, 36].

From these variable word-size examples, it seems that the number of operations increases slightly more than twice when the word size doubles. This is likely to the fact that the diffusion between bits becomes weaker when every word doubles in size. Nevertheless, the number of operations is far lower than three times as much when the word size doubles.

Counterexamples? It is interesting to see if there are cryptographic designs that do not follow the scaling law.

One algorithm of interest is the SHA-3 finalist Grøstl [46], which uses 10 rounds for the 512-bit permutation, and 14 rounds for the 1024-bit permutation. When the number of S-box evaluations is used as a metric for Grøstl's efficiency, the 1024-bit permutation has almost three times as many operations as the 512-bit version. Nevertheless, the best attack on these two Grøstl-permutations are on 9 and 10 rounds respectively [52], indicating that the 1024-bit permutation may be overdesigned.

A clear counterexample to the scaling law is the SPONGENT family of lightweight hash functions [25, 26], which are based on the PRESENT block cipher [27]. When an n-bit permutation is used, SPONGENT evaluates $n/4$ four-bit S-boxes per round. The actual number of rounds depends on the permutation, but is at least $n/2$. This brings the total number of S-boxes to at least $n^2/8$, which means that number of operations quadruples when the permutation size doubles.

For SPONGENT, the number of S-boxes grows very quickly when the permutation size increases. For the 768-bit permutation, the number of S-boxes is 73,920, which corresponds to 770 S-box evaluations for every byte of input. This has a significant impact on the performance: the 272-bit SPONGENT has a five times lower throughput than 256-bit PHOTON, yet the hardware area of both implementations are comparable [26].

The most notable difference between PHOTON and SPONGENT is that PHOTON uses a recursive MDS matrix, whereas SPONGENT uses a PRESENT-style bit permutation. Recursive MDS matrices were pioneered by PHOTON, and later adopted by other designs such as LED [49] and PRIMATEs [2]. They show that it is possible to significantly reduce the hardware area while still maintaining high throughput, which is one of the key problems in lightweight cryptography. This explains the high interest in recursive MDS matrices in recent academic literature [4, 5, 15, 70, 71, 73, 78].

In this section, we introduced the scaling law for symmetric-key cryptography to measure efficiency in a more accurate way. Instead of counting the number of calls to cryptographic primitives, we suggest to weigh in the cost of every call, which becomes relevant when primitives of different sizes are used. More specifically, the scaling law states that amount of "computations" will double when the size of the primitive is doubled. Although this model clearly has its limitations, it helps to give the designers of protocols and modes of operation a more accurate way to evaluate the efficiency of their designs.

4 Picking the Right Tool for the Job

The previous sections focused on how to express security against generic attacks, and gave some first insights in how efficiency can be measured. Although we regularly gave hints on how to make everything as lightweight as possible, we did not consider the requirements of any particular application. We now discuss

some constraints in lightweight applications, and explain how they can be taken into account.

Perhaps most importantly, we should point out that lightweight cryptography is intended for applications with very stringent requirements that conventional algorithms fail to satisfy. As such, we focus specifically at applications where cryptography is the bottleneck. Conventional algorithms were of course never designed to be inefficient, but typically aim to have good performance on a very wide range of platforms. For lightweight cryptography, it seems that the goal is then to focus on algorithms that are tailored to more specific use cases, at the cost of having a more narrow range of applicability.

The goal should be to do an iterative design. Ideally, the lightweight requirements should be formulated such that they can support a wide variety of use cases. We do not always know where the algorithm may eventually end up, and do not want to restrict ourselves too much to any particular platform or application. An iterative design process allows the requirements to gradually become more strict until all design constraints are met.

At the protocol level, we should make the following considerations.

- Are all requirements on the primitives necessary? In a survey by Feldhofer and Rechberger on nine hash-function-based RFID protocols, it was found that only one protocol required collision resistance [44]. Similarly, HMAC [11] does not require a collision resistant hash function [9]. When a secret key is available, it therefore seems preferable to use a block-cipher-based instead of a hash-function-based construction, as argued in [59].
- What is the cost of computation compared to the cost of communication? A survey by Struik indicates that in highly constrained environments, the cost of communication is usually much higher [75]. Ciphertext expansion should then be avoided at all cost. Nevertheless, in some situations we need to add padding to hide the length of the plaintext to the attacker.
- How expensive is it to store a counter, or to generate a random number? This can be very difficult on constrained devices, and still the risk may exist that nonces are reused. It may be worth considering algorithms that do not use a nonce, or that are resistant against nonce reuse. This may lead us in the direction of (authenticated) encryption algorithms that pass over the data twice. Especially for short messages, the implementation benefits may outweigh the overhead of the algorithm.
- In the same spirit, we should be careful with unverified plaintext in authenticated encryption algorithms. Implementations should either store unverified plaintext in secure memory, or use algorithms that remain secure when unverified plaintext is released to the attacker. For a detailed discussion of this scenario, we refer to [3].

We can then look at the platform under consideration, to see which algorithms are best suited to instantiate the protocol. This will depend on whether the main target is performance in hardware or software, and to what extent parallelism can be used. What are the restrictions on hardware area or code size,

and how much RAM is available? For a given platform, how do we interpret the application's restrictions on latency, throughput, power and/or energy?

Last but not least, it is typically not sufficient to have a fast implementation, but this implementation should also be secure against side-channel and fault attacks. Although these problems were in the past often considered as an afterthought, we increasingly see that efficient side-channel and fault attack countermeasures are taken into account during the design phase.

5 Conclusion

We explored the domain of lightweight symmetric-key cryptography, in order to gain insight into some of its challenges.

Firstly, we investigated the security of lightweight algorithms against generic attacks. We warned against designs with short block and key sizes, a problem that exacerbates for applications that frequently rekey. Lightweight designs that are not secure against related-key attacks are becoming increasingly common. We approach the security of these constructions from theoretical point of view, where we discuss various ways to instantiate a PRP, for example with an ideal block cipher or with an Even-Mansour cipher. In line with Kerckhoffs's principle, we also allow the adversary to query this cipher under keys of its choice. This leads to an information-theoretic analysis where we distinguish between data and time queries, depending on whether or not the secret key is involved.

Secondly, we introduced a scaling law for symmetric-key cryptography to express the observation that the number of operations roughly doubles when the input size of a symmetric-key primitive doubles. This law seems to hold for most cryptographic designs, except for SPONGENT (a hash function family based on PRESENT), where the relation is not linear but quadratic. We mentioned recursive MDS matrices as an interesting way to improve implementation efficiency in hardware, while still maintaining a high throughput.

Lastly, we discussed how design decisions in lightweight cryptography should be driven by the application. It seems that the key to lightweight cryptography is not only to apply new academic insights to cryptographic designs, but also to match protocols with the cryptographic primitives that are best suited for them.

Acknowledgments. This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007), by Research Fund KU Leuven, OT/13/071, and by the French Agence Nationale de la Recherche through the BLOC project under Contract ANR-11-INS-011. Nicky Mouha is supported by a Postdoctoral Fellowship from the Flemish Research Foundation (FWO-Vlaanderen), and by a JuMo grant from KU Leuven (JuMo/14/48CF).

References

1. Albrecht, M.R., Driessen, B., Kavun, E.B., Leander, G., Paar, C., Yalçin, T.: Block Ciphers - Focus on the Linear Layer (feat. PRIDE). In: Garay, J.A., Gennaro,

R. (eds.) Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I. LNCS, vol. 8616, pp. 57–76. Springer (2014)
2. Andreeva, E., Bilgin, B., Bogdanov, A., Luykx, A., Mendel, F., Mennink, B., Mouha, N., Wang, Q., Yasuda, K.: PRIMATEs v1.02. Submission to the CAESAR Competition (Round 1) (2014)
3. Andreeva, E., Bogdanov, A., Luykx, A., Mennink, B., Mouha, N., Yasuda, K.: How to Securely Release Unverified Plaintext in Authenticated Encryption. In: Sarkar, P., Iwata, T. (eds.) Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014. Proceedings, Part I. LNCS, vol. 8873, pp. 105–125. Springer (2014)
4. Augot, D., Finiasz, M.: Exhaustive Search for Small Dimension Recursive MDS Diffusion Layers for Block Ciphers and Hash Functions. In: Proceedings of the 2013 IEEE International Symposium on Information Theory, Istanbul, Turkey, July 7-12, 2013. pp. 1551–1555. IEEE (2013)
5. Augot, D., Finiasz, M.: Direct Construction of Recursive MDS Diffusion Layers Using Shortened BCH Codes. In: Cid, C., Rechberger, C. (eds.) Fast Software Encryption - 21st International Workshop, FSE 2014, London, UK, March 3-5, 2014. Revised Selected Papers. Lecture Notes in Computer Science, vol. 8540, pp. 3–17. Springer (2014)
6. Aumasson, J.P., Henzen, L., Meier, W., Phan, R.C.W.: SHA-3 proposal BLAKE. Submission to the NIST SHA-3 Competition (Round 3) (2010), `http://131002.net/blake/blake.pdf`
7. Babbage, S.H.: Improved "Exhaustive Search" Attacks on Stream Ciphers. In: ECOS 95 (European Convention on Security and Detection), Conference publication No. 408. pp. 161–166 (May 1995)
8. Barker, W.C., Barker, E.: SP 800-67 Revision 1: Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher (January 2012), `http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf`
9. Bellare, M.: New Proofs for NMAC and HMAC: Security Without Collision-Resistance. In: Dwork, C. (ed.) Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings. LNCS, vol. 4117, pp. 602–619. Springer (2006)
10. Bellare, M., Boldyreva, A., Palacio, A.: An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem. In: Cachin and Camenisch [32], pp. 171–188
11. Bellare, M., Canetti, R., Krawczyk, H.: Keying Hash Functions for Message Authentication. In: Koblitz, N. (ed.) Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings. LNCS, vol. 1109, pp. 1–15. Springer (1996)
12. Bellare, M., Desai, A., Jokipii, E., Rogaway, P.: A Concrete Security Treatment of Symmetric Encryption. In: 38th Annual Symposium on Foundations of Computer Science, FOCS '97, Miami Beach, Florida, USA, October 19-22, 1997. pp. 394–403. IEEE Computer Society (1997)
13. Bellare, M., Rogaway, P.: Introduction to Modern Cryptography. In: UCSD CSE 207 Course Notes. p. 283 (2005)
14. Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) 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. LNCS, vol. 4004, pp. 409–426. Springer (2006)
15. Berger, T.P.: Construction of Recursive MDS Diffusion Layers from Gabidulin Codes. In: Paul, G., Vaudenay, S. (eds.) Progress in Cryptology - INDOCRYPT 2013 - 14th International Conference on Cryptology in India, Mumbai, India, December 7-10, 2013. Proceedings. LNCS, vol. 8250, pp. 274–285. Springer (2013)
16. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: The Keccak SHA-3 submission. Submission to the NIST SHA-3 Competition (Round 3) (2011), http://keccak.noekeon.org/Keccak-submission-3.pdf
17. Biham, E.: How to Forge DES-Encrypted Messages in 2^{28} Steps. Technical Report CS0884, Technion Computer Science Departement, Israel (1996)
18. Biham, E.: How to decrypt or even substitute DES-encrypted messages in 2^{28} steps. Inf. Process. Lett. 84(3), 117–124 (2002)
19. Biham, E., Dunkelman, O., Keller, N., Shamir, A.: New Data-Efficient Attacks on Reduced-Round IDEA. Cryptology ePrint Archive, Report 2011/417 (2011), http://eprint.iacr.org/
20. Biryukov, A., Mukhopadhyay, S., Sarkar, P.: Improved Time-Memory Trade-Offs with Multiple Data. In: Preneel, B., Tavares, S.E. (eds.) Selected Areas in Cryptography, 12th International Workshop, SAC 2005, Kingston, ON, Canada, August 11-12, 2005, Revised Selected Papers. LNCS, vol. 3897, pp. 110–127. Springer (2005)
21. Biryukov, A., Perrin, L.: Lightweight cryptography lounge. http://cryptolux.org/index.php/Lightweight_Cryptography (2015)
22. Biryukov, A., Shamir, A.: Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers. In: Okamoto, T. (ed.) Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings. LNCS, vol. 1976, pp. 1–13. Springer (2000)
23. Bjørstad, T.E., Bogdanov, A., Gilbert, H., Ideguchi, K., Indesteege, S., Küçük, Ö., Leander, G., Mouha, N., Nakahara Jr., J., Poschmann, A., Rechberger, C., Rijmen, V., Sekar, G., Shibutani, K., Schläffer, M., Standaert, F.X., Tischhauser, E., Velichkov, V., Visconti, I.: D.SYM.5: WG2 Lightweight Cryptographic Algorithms. ECRYPT II Symlab Report (July 2010), Available at: http://www.ecrypt.eu.org/documents/D.SYM.5.pdf
24. Bogdanov, A., Khovratovich, D., Rechberger, C.: Biclique Cryptanalysis of the Full AES. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT. LNCS, vol. 7073, pp. 344–371. Springer (2011)
25. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: SPONGENT: A Lightweight Hash Function. In: Preneel and Takagi [65], pp. 312–325
26. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: SPONGENT: The Design Space of Lightweight Cryptographic Hashing. IEEE Trans. Computers 62(10), 2041–2053 (2013)
27. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings. LNCS, vol. 4727, pp. 450–466. Springer (2007)
28. Bogdanov, A., Knudsen, L.R., Leander, G., Standaert, F., Steinberger, J.P., Tischhauser, E.: Key-Alternating Ciphers in a Provable Setting: Encryption Using a

Small Number of Public Permutations - (Extended Abstract). In: Pointcheval and Johansson [64], pp. 45–62
29. Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E.B., Knezevic, M., Knudsen, L.R., Leander, G., Nikov, V., Paar, C., Rechberger, C., Rombouts, P., Thomsen, S.S., Yalçin, T.: PRINCE - A Low-Latency Block Cipher for Pervasive Computing Applications - Extended Abstract. In: Wang, X., Sako, K. (eds.) 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. LNCS, vol. 7658, pp. 208–225. Springer (2012)
30. Boura, C., Canteaut, A., Cannière, C.D.: Higher-Order Differential Properties of KECCAK and *Luffa*. In: Joux, A. (ed.) FSE. LNCS, vol. 6733, pp. 252–269. Springer (2011)
31. van den Broek, F., Poll, E.: A Comparison of Time-Memory Trade-Off Attacks on Stream Ciphers. In: Youssef, A., Nitaj, A., Hassanien, A.E. (eds.) Progress in Cryptology - AFRICACRYPT 2013, 6th International Conference on Cryptology in Africa, Cairo, Egypt, June 22-24, 2013. Proceedings. LNCS, vol. 7918, pp. 406–423. Springer (2013)
32. Cachin, C., Camenisch, J. (eds.): Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings, LNCS, vol. 3027. Springer (2004)
33. Canteaut, A. (ed.): Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers, LNCS, vol. 7549. Springer (2012)
34. Chen, S., Steinberger, J.P.: Tight Security Bounds for Key-Alternating Ciphers. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology - EUROCRYPT 2014 - 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings. LNCS, vol. 8441, pp. 327–350. Springer (2014)
35. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer (2002)
36. Duan, M., Lai, X.: Improved zero-sum distinguisher for full round KECCAK-f permutation. Cryptology ePrint Archive, Report 2011/023 (2011), `http://eprint.iacr.org/`
37. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In: Pointcheval and Johansson [64], pp. 336–354
38. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. NIST special publication 800-38b, National Institute of Standards and Technology (NIST) (May 2005), `http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf`
39. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC (November 2007), `http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf`
40. Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Methods for Format-Preserving Encryption (July 2013), `http://csrc.nist.gov/publications/drafts/800-38g/sp800_38g_draft.pdf`
41. ECRYPT II: Cryptanalysis Categories - The ECRYPT Hash Function Website. `http://ehash.iaik.tugraz.at/wiki/Cryptanalysis_Categories`
42. Even, S., Mansour, Y.: A Construction of a Cipher From a Single Pseudorandom Permutation. In: Imai, H., Rivest, R.L., Matsumoto, T. (eds.) ASIACRYPT. LNCS, vol. 739, pp. 210–224. Springer (1991)

43. Even, S., Mansour, Y.: A Construction of a Cipher from a Single Pseudorandom Permutation. J. Cryptology 10(3), 151–162 (1997)
44. Feldhofer, M., Rechberger, C.: A Case Against Currently Used Hash Functions in RFID Protocols. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM Workshops (1). LNCS, vol. 4277, pp. 372–381. Springer (2006)
45. Ferguson, N., Lucks, S., Schneier, B., Whiting, D., Bellare, M., Kohno, T., Callas, J., Walker, J.: The Skein Hash Function Family. Submission to the NIST SHA-3 Competition (Round 3) (2010), http://www.skein-hash.info/sites/default/files/skein1.3.pdf
46. Gauravaram, P., Knudsen, L.R., Matusiewicz, K., Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: Grøstl – a SHA-3 candidate. Submission to the NIST SHA-3 Competition (Round 3) (2011), http://www.groestl.info/Groestl.pdf
47. Goldwasser, S., Kalai, Y.T.: On the (In)security of the Fiat-Shamir Paradigm. In: 44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings. pp. 102–113. IEEE Computer Society (2003)
48. Golić, J.D.: Cryptanalysis of Alleged A5 Stream Cipher. In: Fumy, W. (ed.) Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding. LNCS, vol. 1233, pp. 239–255. Springer (1997)
49. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.J.B.: The LED Block Cipher. In: Preneel and Takagi [65], pp. 326–341
50. Hellman, M.E.: A Cryptanalytic Time-Memory Trade-off. Information Theory, IEEE Transactions on 26(4), 401–406 (1980)
51. Hoang, V.T., Reyhanitabar, R., Rogaway, P., Vizár, D.: Online Authenticated-Encryption and its Nonce-Reuse Misuse-Resistance. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, Springer (2015 (To appear))
52. Jean, J., Naya-Plasencia, M., Peyrin, T.: Improved Rebound Attack on the Finalist Grøstl. In: Canteaut [33], pp. 110–126
53. Kavun, E.B., Lauridsen, M.M., Leander, G., Rechberger, C., Schwabe, P., Yalçın, T.: Prøst v1.1. Submission to the CAESAR Competition (Round 1) (2014)
54. Levin, D.A., Peres, Y., Wilmer, E.L.: Markov Chains and Mixing times. American Mathematical Society (2009), Available at: http://pages.uoregon.edu/dlevin/MARKOV/markovmixing.pdf
55. Maurer, U.M., Renner, R., Holenstein, C.: Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In: Naor, M. (ed.) Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004, Cambridge, MA, USA, February 19-21, 2004, Proceedings. LNCS, vol. 2951, pp. 21–39. Springer (2004)
56. Mehuron, W.: Data Encryption Standard (DES) (October 1999), http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf
57. Mennink, B., Preneel, B.: Hash Functions Based on Three Permutations: A Generic Security Analysis. In: Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. LNCS, vol. 7417, pp. 330–347. Springer (2012)
58. Mouha, N., Luykx, A.: Multi-Key Security: The Even-Mansour Construction Revisited. Cryptology ePrint Archive, Report 2015/101 (2015), http://eprint.iacr.org/
59. Mouha, N., Mennink, B., Herrewege, A.V., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers.

In: Joux, A., Youssef, A.M. (eds.) Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers. LNCS, vol. 8781, pp. 306–323. Springer (2014)
60. Nandi, M.: A Simple and Unified Method of Proving Indistinguishability. In: Barua, R., Lange, T. (eds.) Progress in Cryptology - INDOCRYPT 2006, 7th International Conference on Cryptology in India, Kolkata, India, December 11-13, 2006, Proceedings. LNCS, vol. 4329, pp. 317–334. Springer (2006)
61. Nohl, K.: Attacking phone privacy. Presented at Black Hat USA 2010, Las Vegas (July 2010)
62. van Oorschot, P.C., Wiener, M.J.: Parallel Collision Search with Cryptanalytic Applications. J. Cryptology 12(1), 1–28 (1999)
63. Patarin, J.: The "Coefficients H" Technique. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) Selected Areas in Cryptography, 15th International Workshop, SAC 2008, Sackville, New Brunswick, Canada, August 14-15, Revised Selected Papers. LNCS, vol. 5381, pp. 328–345. Springer (2008)
64. Pointcheval, D., Johansson, T. (eds.): Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques,Nohl Cambridge, UK, April 15-19, 2012. Proceedings, LNCS, vol. 7237. Springer (2012)
65. Preneel, B., Takagi, T. (eds.): Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings, LNCS, vol. 6917. Springer (2011)
66. Quisquater, J.J., Delescaille, J.P.: How Easy is Collision Search? Application to DES (Extended Summary). In: EUROCRYPT. pp. 429–434 (1989)
67. Rogaway, P., Steinberger, J.P.: Constructing Cryptographic Hash Functions from Fixed-Key Blockciphers. In: Wagner, D. (ed.) Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings. LNCS, vol. 5157, pp. 433–450. Springer (2008)
68. Rogaway, P., Steinberger, J.P.: Security/Efficiency Tradeoffs for Permutation-Based Hashing. In: Smart, N.P. (ed.) Advances in Cryptology - EUROCRYPT 2008, 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings. LNCS, vol. 4965, pp. 220–236. Springer (2008)
69. Romine, C.H.: Secure Hash Standard (SHS) (March 2012), `http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf`
70. Sajadieh, M., Dakhilalian, M., Mala, H., Sepehrdad, P.: Recursive Diffusion Layers for Block Ciphers and Hash Functions. In: Canteaut [33], pp. 385–401
71. Sajadieh, M., Dakhilalian, M., Mala, H., Sepehrdad, P.: Efficient Recursive Diffusion Layers for Block Ciphers and Hash Functions. J. Cryptology 28(2), 240–256 (2015)
72. Sasaki, Y., Todo, Y., Aoki, K., Naito, Y., Sugawara, T., Murakami, Y., Matsui, M., Hirose, S.: Minalpher v1. Submission to the CAESAR Competition (Round 1) (2014)
73. Sim, S.M., Khoo, K., Oggier, F., Peyrin, T.: Lightweight MDS Involution Matrices. In: Leander, G., Demirci, H. (eds.) FSE 2015. LNCS, Springer (2015 (To appear))
74. Steinberger, J.P.: Improved Security Bounds for Key-Alternating Ciphers via Hellinger Distance. Cryptology ePrint Archive, Report 2012/481 (2012), `http://eprint.iacr.org/`
75. Struik, R.: Aead ciphers for highly constrained networks. DIAC 2013 presentation (August 2013)

76. Verbauwhede, I.: How much crypto in one microJoule? Workshop on Real-World Cryptography (January 2013)
77. Wierzbicki, M.: What is the relationship of $\mathcal{L}_1$ (total variation) distance to hypothesis testing? http://math.stackexchange.com/questions/72721/what-is-the-relationship-of-mathcall-1-total-variation-distance-to-hypoth (October 2015)
78. Wu, S., Wang, M., Wu, W.: Recursive Diffusion Layers for (Lightweight) Block Ciphers and Hash Functions. In: Knudsen, L.R., Wu, H. (eds.) Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers. LNCS, vol. 7707, pp. 355–371. Springer (2012)
79. Yoshida, H., Watanabe, D., Mouha, N.: On the status of techniques and standardization regarding lightweight cryptography -ISO/IEC JTC1/SC27/WG2 status report-. Information and Communication Systems Security (ICSS), IEICE Technical Report (November 2014)
80. Yu, H., Chen, J., Wang, X.: Partial-Collision Attack on the Round-Reduced Compression Function of Skein-256. In: Moriai, S. (ed.) Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers. LNCS, vol. 8424, pp. 263–283. Springer (2013)