State of the Art in Lightweight Symmetric Cryptography

Alex Biryukov[1] and Léo Perrin[2]

[1] SnT, CSC, University of Luxembourg, alex.biryukov@uni.lu
[2] SnT, University of Luxembourg, perrin.leo@gmail.com

Abstract. Lightweight cryptography has been one of the "hot topics" in symmetric cryptography in the recent years. A huge number of lightweight algorithms have been published, standardized and/or used in commercial products.
In this paper, we discuss the different implementation constraints that a "lightweight" algorithm is usually designed to satisfy. We also present an extensive survey of all lightweight symmetric primitives we are aware of. It covers designs from the academic community, from government agencies and proprietary algorithms which were reverse-engineered or leaked. Relevant national (NIST...) and international (ISO/IEC...) standards are listed. We then discuss some trends we identified in the design of lightweight algorithms, namely the designers' preference for ARX-based and bitsliced-S-Box-based designs and simple key schedules.
Finally, we argue that lightweight cryptography is too large a field and that it should be split into two related but distinct areas: *ultra-lightweight* and *IoT* cryptography. The former deals only with the smallest of devices for which a lower security level may be justified by the very harsh design constraints. The latter corresponds to low-power embedded processors for which the AES and modern hash function are costly but which have to provide a high level security due to their greater connectivity.

Keywords: Lightweight cryptography · Ultra-Lightweight · IoT · Internet of Things · SoK · Survey · Standards · Industry

1 Introduction

The Internet of Things (IoT) is one of the foremost buzzwords in computer science and information technology at the time of writing. It is a very broad term describing the fact that, in the near future, the Internet will be used more and more to connect devices to one another other rather than to connect people together.

Some of these devices use powerful processors and can be expected to use the same cryptographic algorithms as standard desktop PCs. However, many of them use extremely low power micro-controllers which can only afford to devote a small fraction of their computing power to security. Similarly, regular algorithms may incur too high a latency or too high a power consumption for such platforms.

A common example of such use is that of sensor networks. Such networks are intended to connect vast amounts of very simple sensors to a central hub. These sensors would run on batteries and/or generate their own energy using for example solar panels. Cryptographic algorithms must be used in the communication channels between the sensors and their hub in order to provide security, authenticity and integrity of the messages. However, because of the very low energy available, and because security is an overhead on top of the actual functionality of the device, the cryptographic algorithms have to be as "small" as possible. Similar reasoning goes for the size of ROM and RAM consumption of the cryptographic algorithm.

Similarly, RFID (Radio-Frequency IDentification) chips are used to identify devices, animals — and even people. In order to prevent an eavesdropper from learning the identification associated to a chip, this information has to be encrypted. Moreover, such RFID tags can be used in challenge response protocols. Because of the very small number of logical gates that can be used in such devices and because of the very little energy available, specially designed algorithms are necessary.

A vast number of symmetric cryptographic algorithms have been proposed to fill these use cases (see Table 1). They are usually referred to as "lightweight". Their designs vary greatly, the only unifying thread between them is the low computing power of the devices intended to run them.

Still, we call "lightweight" a wide range of algorithms with different properties and corresponding to very different use cases. As a consequence, it is difficult to give a clear definition of what this term entails. For example, does it make sense for a "lightweight" algorithm to be secure against attacks requiring a large amount of data? If the device using it is a simple RFID tag, probably not. Conversely, if it is an internet-enabled device downloading e.g. software updates at regular intervals, then such protection would be desirable.

Our Contributions. In this paper, we systematize the knowledge in the area of lightweight cryptography with the aim of better understanding what "lightweightness" is. Our work is based on an extensive literature survey of more than 100 algorithms; the number of algorithms considered is summarized in Table 1. Beyond merely listing these algorithms, we identify the main trends in their design and discuss their standardization. Finally, we argue that the design requirements corresponding to "lightweight cryptography" are too broad. As a consequence, we suggest splitting this area into two sub-fields and provide a general scope statement for algorithms fitting in each category.

Table 1: The number of lightweight symmetric algorithms surveyed in this paper.

	Stream	Block	Hash	Auth.	Total
Academia	13	49	9	11	82
Proprietary	17	5	0	1	23
Government	1	5	0	0	6
Total	31	59	9	12	**111**

Outline. Section 2 lists the design constraints that make an algorithm lightweight. Both the hardware and software cases are described. Then, we list all symmetric cryptographic lightweight algorithms we are aware of in Section 3. We consider algorithms published at cryptography/security conferences as well as those designed by government agencies, those specified in standards and those designed by the industry which were reverse-engineered from actual products. In particular, the following tables list lightweight algorithms along with a reference for their specification and some of their properties such as key and block sizes:

- Table 4 lists ciphers from the industry. None of them provides more than 64 bits of security either by design (short key), because of weaknesses in their design, or both;
- algorithms published at cryptography/security conferences or submitted to cryptography competitions are listed in Table 5 (block ciphers), Table 6 (hash functions), Table 7 (stream ciphers), and Table 8 (authenticated ciphers); and

- Table 9 lists public ciphers designed by government agencies.

We also survey the standardization of lightweight crypto by both national and international bodies in Section 4. We also consider the primitives used by some software libraries intended for embedded devices.

In Section 5, we investigate some trends in the field of lightweight cryptography such as the increasing popularity of lighter key schedules. Finally, we argue in Section 6 for a division in the field of lightweight cryptography between *ultra-lightweight* and *IoT* crypto targeting respectively the most constrained devices (e.g RFID tags) and networked embedded devices.

2 Design Constraints

The metrics usually optimised are the memory consumption, the implementation size and the speed or throughput of the primitive. However, the specifics of the comparison depend on whether hardware or software implementations are considered.

2.1 Hardware Case

If the primitive is implemented in hardware, the memory consumption and the implementation size are lumped together into its gate area which is measured in Gate Equivalents (GE). It quantifies how physically large a circuit implementing the primitive is. The throughput is measured in bytes per second and corresponds to the amount of plaintext processed per time unit.

The exact measures depend on the exact type of circuit considered, e.g. the frequency at which it is clocked or the area of each gate. Furthermore, the tools used to simulate those circuits do not give the same results and are usually both proprietary and expensive. Therefore, a fair comparison of the different algorithms proposed regarding their hardware implementation is very difficult. In fact, when comparing their new algorithm with existing ones, designers are usually forced to design their own implementations of preexisting ones too.

Memory is usually the most expensive part of the implementation of a lightweight primitive. In most cases, implementations work by storing the full internal state and key state and then perform one round in c clock cycle, e.g. one round per clock cycle.[1] As a consequence, it is preferable to operate on small blocks using a small key.

However, some space can be saved by hard-coding or "burning" the keys into the device. That is, instead of using read/write memory to store the key, use read-only structures. In order for this method to be viable, the key schedule must build the round keys using only simple operations taking as input the bits of the master key. In particular, no key state can be operated upon. This strategy has been used by several algorithms, both block and stream ciphers, as shown in Section 5.2.2.

More recently, other criteria have emerged for the design of lightweight algorithms. For example, energy and power efficiency of a hardware implementation are at the core of the design of the Midori block cipher published about a year ago [BBI+15]. Another criteria is latency, that is, the time taken to perform a given operation. There are contexts in which the low-latency is crucial, for example for memory encryption. This particular requirement demands specific design choices as illustrated by the lightweight block ciphers PRINCE [BCG+12] followed by Mantis [BJK+16] and Qarma [Ava17].

[1] It is a bit more complicated in the case of a serial implementation. They only update a small part of the state at each round — typically the size of an S-Box — but, due to the simplicity of the logic involved, they allow a far higher clock frequency.

2.2 Software Case

Primitives can be also implemented in software, typically for use on micro-controllers. In this case, the relevant metrics are the RAM consumption, the code size and the throughput of the primitive measured in bytes per CPU cycle. The FELICS framework allows a relevant comparison of these quantities across algorithms and across different implementations of a given algorithm. This framework was presented in [DBG+15]. The name FELICS stands for "Fair Evaluation of LIghtweight Cryptographic Systems".

FELICS takes as input the implementation of a block or stream cipher and outputs the corresponding code size, RAM consumption and time taken to perform a given task. These quantities are obtained for three different micro-controllers: an 8-bit AVR, a 16-bit MSP and a 32-bit ARM. The tasks investigated correspond to different scenarios such as the encryption of a 128-bit block in counter-mode. The information extracted is then summarized into a single quantity called Figure of Merit (FoM), the lower the better. It can be used to rank block ciphers, as shown in Table 2 where the block and key sizes are in bits, the code size and maximum RAM consumption are in bytes and the time is in number of CPU cycles.

Table 2: The current best FELICS results for scenario 2: counter mode encryption of 128 bits.

General info			AVR (8-bit)			MSP (16-bit)			ARM (32-bit)			FoM
Name	block	key	Code	RAM	Time	Code	RAM	Time	Code	RAM	Time	
Chaskey	128	128	770	84	1597	490	86	1351	178	80	614	4.7
SPECK	64	96	448	53	2829	328	48	1959	256	56	1003	4.8
SPECK	64	128	452	53	2917	332	48	2013	276	60	972	4.9
Chaskey-LTS	128	128	770	84	2413	492	86	2064	178	80	790	5.4
SIMON	64	96	600	57	4269	460	56	2905	416	64	1335	6.6
SIMON	64	128	608	57	4445	468	56	3015	388	64	1453	6.8
LEA	128	128	906	80	4023	722	78	2814	520	112	1171	7.6
RECTANGLE	64	128	602	56	4381	480	54	2651	452	76	2432	8.1
RECTANGLE	64	80	606	56	4433	480	54	2651	452	76	2432	8.1
SPARX	64	128	662	51	4397	580	52	2261	654	72	2338	8.3
SPARX	128	128	1184	74	5478	1036	72	3057	1468	104	2935	12.4
RC5-20	64	128	1068	63	8812	532	60	15925	372	64	1919	13.5
AES	128	128	1246	81	3408	1170	80	4497	1348	124	4044	14.1
HIGHT	64	128	636	56	6231	636	52	7117	670	100	5532	14.8
Fantomas	128	128	1712	76	9689	1920	78	3602	2184	184	4550	18.8
Robin	128	128	2530	108	7813	1942	80	4913	2188	184	6250	22.0

The three quantities measured are not independent. For example, loading information from the RAM into CPU registers is a costly operation and so is its inverse. Therefore, limiting the number of such operations leads to a decrease in both RAM consumption and time complexity.

2.3 Side-Channel Attack Resilience

Side-channel attacks (SCAs) use some special knowledge about the implementation of a cipher to break its security. For example, observing the power consumption of an encryption can leak information about the Hamming weight of the output of an S-Box.

Such attacks demand that the cryptanalyst has physical access to the device attacked. However, this requirement is particularly easy to fulfill in the context of the IoT: a desktop computer can be expected to be reasonably hard to physically interact with because it is in a locked room but a sensor measuring traffic in an open street may not enjoy such

protection.

As a consequence, lightweight algorithms are often built in such a way as to decrease the vulnerability of their implementation to such attacks. This can be done through the use of inherently less leaky operations or by simplifying the use of a masked implementation. These topics are further discussed in Section 5.1.

2.4 Common Trade-Offs

To accommodate these constraints, most lightweight algorithms are designed to use smaller internal states and smaller key sizes. Indeed, while a 128-bit block and at least 128-bit key was demanded from the AES candidates, most lightweight block ciphers use only 64-bit blocks. This smaller size leads to a smaller memory footprint in both software and hardware. It also means that the algorithm is better suited for processing smaller messages.

Nevertheless, the small block size can be a problem as the security of some modes of operation such as CBC erodes very quickly when the number of n-bit blocks encrypted approaches $2^{n/2}$, as exploited for example in [BL16]. As a consequence, dedicated modes of operation such as [LPTY16] have been proposed to mitigate these issues. Furthermore, key sizes are often as small as 80 bits which offers little security margin against brute-force search. Such an attack is likely to be infeasible nowadays for all but the most powerful state sponsored adversaries, but how long will this last? As pointed out in [BMS06], time-memory-data tradeoff can become an issue if the key size is too small, especially in the multi-key setting.

In the case of lightweight block ciphers, it is also common for the components used to be involutions so as to decrease the cost of the implementation of decryption. This can be done by using non-linear involutions as S-Boxes or by using a Feistel structure. On the other hand, this issue can be mitigated through the use of modes of operations that do not require block cipher decryption. For example, if a block cipher is used in counter mode, the area/ROM which would be needed to store the description of the inverse block cipher can be saved.

Finally, due to the importance of their performance, lightweight algorithms often have thinner security margins. For example, KETJE [BDP+16] is a sponge-based authenticated cipher where the sponge transformation uses only one round. That is, it does not even provide full diffusion. This obvious issue is offset by the use of a non-repeating nonce.

2.5 Are Dedicated Algorithms Needed?

As lightweightness is mostly a property of the implementation of an algorithm, we can wonder if dedicated algorithms are actually needed. Would it not be sufficient to use lightweight *implementations* of regular algorithms?

It is often possible. For example, many implementers have worked on optimising the implementation of the AES with some success in both hardware [BJM+14, BBR16, UMHA16] and software [SS16]. Even if an efficient implementation of the AES is impossible using the instructions available on a micro-controller, those devices are sometimes shipped with a hardware acceleration module for this task, effectively adding a new set of instructions dedicated entirely to a quick evaluation of this block cipher.

In this context, lightweight symmetric algorithms may seem unnecessary. However, block cipher hardware acceleration has its limitations. As summarized for example in Table 1 of [OC16] (which is reproduced in Table 3), the hardware-accelerated encryptions used by many devices are vulnerable to various forms of side-channel attacks. These attacks do not only target these devices "in a vacuum". For example, Philips light bulbs using the Zigbee protocol to communicate have been recently shown to be insecure [ROSW16]. One of the key components of this attack is a subversion of the update mechanism of the

light bulb. Updates are normally authenticated with an AES-based MAC using a secret key which is constant across all devices. Ronen *et al.* recovered this key via an SCA and were therefore able to push malicious updates to these devices.

Table 3: Several micro-processors whose hardware accelerated cryptography is vulnerable to SCA (reproduced from [OC16]).

Product	Cipher
DESFire, MF3ICD40	3-DES
DS2432, DS28E01	SHA-1
Microchip HCSXXX	Keeloq
ProASIC3	AES
Spartan-6	AES
Stratix II	AES
Stratix III	AES
Virtex-II	3-DES
Virtext-4, Virtex-5	AES
XMEGA	AES
Yubikey	AES

Still, there are cases where side-channel attacks are not really relevant. For example, a yubikey[2] is supposed to be always carried by its owner, so that studying its power consumption is not practical for the adversary. However, if attackers can easily access devices with the secret key they are after, e.g. in the case of a wireless sensor network, such a weakness is not acceptable. This problem could be mitigated by using protected implementations such as masked ones.

On micro-controllers without hardware support for cryptographic functions, their assembly implementation must be as small and as fast as possible. In these cases, the AES is decently fast, especially on 8-bit micro-controllers where it is in fact one of the fastest. However, its implementation requires storing at least the full look-up-table of its 8-bit S-Box, meaning that its code size cannot be as small as that of dedicated algorithms.

All in all, while the AES is a reasonably lightweight block cipher, its large S-Box, large block size and inherent vulnerability to SCA caused by its look-up-based S-Box make it a suboptimal choice in many cases.

Another case where dedicated lightweight algorithms are needed is for hashing. Indeed, standard hash functions need large amounts of memory to store both their internal states — 1600 bits in the case of SHA-3 — and the block they are operating on — 512 bits in the case of the SHA-2 family. These memory requirements significantly hinder performance on lightweight platforms and justify the need for dedicated lightweight hash functions.

3 A Survey of Symmetric Lightweight Algorithms

Several very distinct actors are involved in the field of lightweight cryptography. In fact, they are the same that discuss "regular" cryptography: academia, industry, standardizing bodies and government agencies — including spying agencies.

The industry is supposed to be the implementer of those algorithms, designing or choosing the best one for their purpose. Unfortunately, until the 2000's and the spread of the AES, many of the algorithms had been designed in-house with little regard to what is considered best practice. The corresponding algorithms are described in Section 3.1.

[2] A yubikey is a commercial USB drive designed to store cryptographic keys securely and use them in authentication protocols.

Academics have published dozens of symmetric cryptographic algorithms claiming to be lightweight. Those are listed in Section 3.2. Said publications always contain a description of the cryptanalysis attempts by the authors of the algorithm.

This creates a significant contrast with the algorithms proposed by government agencies: even their public algorithms have been designed in a secret way. Furthermore, these agencies are more often than not also in charge of spying, (see the American NSA and the Russian FSB), meaning that they have contradictory incentives. On the one hand, it is their task to ensure the security of their own citizens which may imply designing strong encryption. On the other hand, as evidenced by the first crypto wars and the current ongoing debate surrounding the alleged fear of some law enforcement agencies of "going dark", they may also seek to purposefully weaken these standards. What might have been discounted as mere conspiracy theory a few years ago is now an established fact: the Snowden documents show that the NSA has a budget dedicated to the subversion of cryptographic standards and has pushed for the standardization by NIST of the easily trapdoored Dual EC pseudo-random number generator [BLN15]. In a similar vein, as shown in [BPU16] the latest Russian standards designed by FSB share an S-Box with a hidden structure, which still leaves cryptographers puzzled over its purpose. This situation also resembles controversy over S-boxes of DES, whose design criteria were not published at the time of standardization of DES. For any user familiar with the trapdoored horse story [HomBC], such developments are highly unsettling. We list lightweight algorithms designed by government agencies in Section 3.3.

3.1 Algorithms from the Industry

Many lightweight algorithms used by industrial products are surprisingly weak. Many of those algorithms were designed in the 80's or early 90's, a time during which cipher design had to accommodate for the stringent American export laws which forbid selling devices with strong cryptography. Still, like modern lightweight algorithms, those were intended to run on devices with little computing power devoted to encryption.

They were also intended to be kept secret, their secrecy hopefully enhancing their security. However, They were eventually released through leaks or reverse-engineering.[3] In a clear vindication of Kerchkoffs' law [Ker83], they were broken as soon as they were made public. Attacks with a time complexity under 2^{40} evaluations of the primitive exist for all of these algorithms except for CSA ciphers. Many of them are the target of even more powerful cryptanalyses.

These primitives are described in Appendix A and listed in Table 4. Block ciphers are marked with "†", MACs with "‡" and unmarked primitives are stream ciphers. The internal state (IS), key and initialization vector (IV) sizes are expressed in bits. For block ciphers, the internal state size corresponds to the block size. The time complexity of the best attack targeting the full round primitive is given in the column "Att. time".

3.2 A Semi-Exhaustive List of Public Algorithms

Throughout the last 25 years and especially since 2011, a lot of algorihtms intended to be lightweight have been published in cryptography- and security-related conference proceedings and journals. Those are listed in this section.

The algorithms in these lists have either been advertised as lightweight in their specification, have a very small implementation or have been standardized as such. Figure 1 provides an overview of all lightweight symmetric algorithms published by academics, sorted by publication date and by type.

[3] Except for E0 and for PC-1.

Table 4: A summary of all lightweight primitives from the industry we are aware of.

Name	Things	Reference	Key	IS	IV	Att. time
A5/1	Cell phones	[And94]	64	64	22	2^{24}
A5/2		[BBK08]	64	81	22	2^{16}
CMEA †		[WSK97]	64	16–48	–	2^{32}
ORYX		[WSD+99]	96	96	–	2^{16}
A5-GMR-1	Satellite phones	[DHW+12]	64	82	19	$2^{38.1}$
A5-GMR-2		[DHW+12]	64	68	22	2^{28}
DSC	Cordless phones	[LST+09]	64	80	35	2^{34}
SecureMem.	Atmel chips	[GvRVWS10]	64	109	128	$2^{29.8}$
CryptoMem.			64	117	128	2^{50}
Hitag2	Car key/ immobilizer	[VGB12]	48	48	64	2^{35}
Megamos		[VGE13]	96	57	56	2^{48}
Keeloq †		[BSK96]	64	32	–	$2^{44.5}$
DST40 †		[BGS+05]	40	40	–	2^{40}
iClass	Smart cards	[GdKGV14]	64	40	–	2^{40}
Crypto-1		[NESP08]	48	48	96	2^{32}
CSS	DVD players	[BD04]	40	42	–	2^{40}
Cryptomeria †		[BKLM09]	56	64	–	2^{48}
CSA-BC †	Digital televisions	[WW05]	64	64	–	2^{64}
CSA-SC			64	103	64	$2^{45.7}$
PC-1	Amazon Kindle	[BLR13]	128	152	–	2^{31}
SecurID ‡	Secure token	[BLP04]	64	64	–	2^{44}
E0	Anything	[FL01]	128	128	–	2^{38}
RC4		[Nob94]	128	2064	–	2^{32}

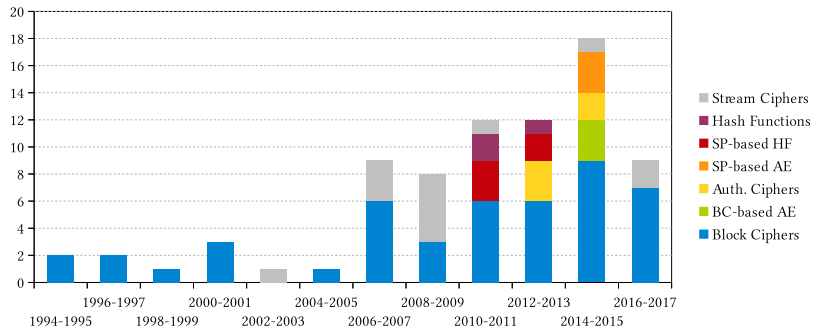

Figure 1: Lightweight algorithms published by academics. Block cipher-based authenticated ciphers are listed as "BC-based AE"; sponge-based algorithms are shown as such.

3.2.1 Block ciphers

Block ciphers are the most common choice for academic designers trying to build a lightweight symmetric algorithm. All those designed and published by academics we are aware of are listed in Table 5 where they are sorted by date of publication. We use "†" to mark block ciphers published as part of a higher level construction, e.g. an authenticated cipher submitted to CAESAR[4] or a MAC; and "‡" to indicate tweakable block ciphers.

[4]The project CAESAR (Competition for Authenticated Encryption: Security, Applicability, and Robustness) aims at identifying the best authenticated ciphers. All submissions are listed on the following webpage: http://competitions.cr.yp.to/caesar-submissions.html.

Table 5: A summary of all lightweight block ciphers published at cryptography and security conference we are aware of, sorted by publication date.

Description		Parameters		
Name	Ref.	Key	Block	Rounds
3-Way	[DGV94]	96	96	11
RC5	[Riv95]	0–2040	32/64/128	12
Misty1	[Mat97]	128	64	8
Xtea	[NW97]	128	64	64
Aes	[DR98]	128/192/256	128	10/12/14
Bksq	[DR00]	96	96/144/192	10/14/18
Khazad	[BR00]	128	64	8
Noekeon	[DPVAR00]	128	128	16
Iceberg	[SPR+04]	128	64	16
mCrypton	[LK06]	64/96/128	64	12
Hight	[HSH+06]	128	64	32
Sea	[SPGQ06]	96	96	93
Clefia	[SSA+07]	128/192/256	128	18/22/26
Deslx	[LPPS07]	184	64	16
Present	[BKL+07]	80/128	64	31
Mibs	[ISSK09]	64/80	64	32
Katan/Ktantan	[CDK09]	80	32/48/64	254
Gost revisited	[PLW10]	256	64	32
PrintCipher	[KLPR10]	48/96	80/160	48/96
Epcbc	[YKPH11]	96	48/96	32
Klein	[GNL11]	64/80/96	64	12/16/20
LBlock	[WZ11]	80	64	32
Led	[GPPR11]	64/128	64	32/48
Piccolo	[SIH+11]	80/128	64	25/31
Picaro	[PRC12]	128	128	12
Prince	[BCG+12]	128	64	12
ITUbee	[KDH13]	80	80	20
Twine	[SMMK13]	80/128	64	36
Zorro	[GGNS13]	128	128	24
Chaskey †	[MMH+14]	128	128	8/12/16
Pride	[ADK+14]	128	64	20
Joltik † ‡	[JNP14]	64/80/96/128	64	24/32
Lea	[HLK+14]	128/192/256	128	24/28/32
iScream † ‡	[GLS+14]	128	128	12/14
LBlock-s †	[ZWW+14]	80	64	32
Scream † ‡	[GLS+14]	128	128	10/12
Lilliput	[BFMT15]	80	64	30
Rectangle	[ZBL+15]	80/128	64	25
Fantomas	[GLSV15]	128	128	12
Robin	[GLSV15]	128	128	16
Midori	[BBI+15]	128	64/128	16/20
Simeck	[YZS+15]	64/96/128	32/48/64	32/36/44
RoadRunneR	[BŞ16]	80/128	64	10/12
Fly	[KG16]	128	64	20
Mantis ‡	[BJK+16]	128	64	14
Skinny ‡	[BJK+16]	64–384	64/128	32–56
Sparx	[DPU+16]	128/256	64/128	24–40
Mysterion	[JSV17]	128/256	128/256	12
Qarma ‡	[Ava17]	128/256	64/128	16/24

3.2.2 Hash functions

It is more difficult to implement a lightweight hash function than a lightweight block cipher. Indeed, they usually require a much larger internal state which is reasonable on a desktop computer but would have a prohibitive cost on a lightweight device. For example, SHA-3 uses a 1600-bit internal state which dwarfs the 64-bit block of most lightweight block ciphers.

And yet, since a collision in the internal state leads to a collision in the final digest, it has to have at least a size corresponding to the desired security level.

As an answer to this problem, several designers chose the use of a sponge construction with a very small rate. Indeed, the internal state of a sponge is divided into two distinct parts:

- the r-bit rate decides how fast the plaintext is processed and how fast the final digest is produced, and
- the c-bit capacity determines the security level as for example a birthday collision search succeeds with a time complexity of roughly $2^{c/2}$ independently of the digest size.

For example, using a capacity of 128 bits along with a rate of 8 bits, as done in some versions of the hash functions listed below, minimizes the memory footprint at the cost of a slower data processing.

All lightweight hash functions we are aware of are in Table 6. The internal state (IS) size usually refers to the size of the chaining value. However, for example in BLAKE2, the update function operates on the chaining value and on an IV of the same size, so that its internal state is twice as big. Sponge-based hash functions are marked with a "‡" symbol.

Table 6: A summary of all lightweight hash functions from academia. The digest, internal state (IS) and block sizes are expressed in bits.

Name	Reference	Digest	Block	IS
Armadillo	[BDN+10]	80/128/160/256	48/64/80/128	256/384/576/768
BLAKE2s/b	[ANWOW13]	8–256/8–512	512/1024	512/1024
GLUON ‡	[BDM+12]	128/160/224	8/16/32	136/176/256
Lesamnta-LW	[HIK+11]	256	128	256
PHOTON ‡	[GPP11]	80/128/160/224/256	16/32/36	100/144/196/256/288
QUARK ‡	[AHMN10]	136/176/256	8/16/32	136/176/256
SipHash ‡	[AB12]	64	64	256
SPN-Hash ‡	[CYK+12]	128/256	256/512	128/256
Spongent ‡	[BKL+11]	80/128/160/224/256	8/16	88/136/176/240/272

3.2.3 Stream ciphers

The eSTREAM competition was held in 2008 to choose two portfolios of stream ciphers. The first type of algorithms fits the so-called *software profile*, meaning that they were aimed at software efficiency. The second category was the *hardware profile*. Further, some of them use an internal state so small that they can be considered to be lightweight stream ciphers.

However, lightweight stream ciphers were proposed outside the framework of this competition. For example, SNOW 3G corresponds to a simple modification of the academic-designed SNOW 2.0 tailored for specific industrial needs: it is used in the 3GPP communication standard. All such lightweight stream ciphers we are aware of are listed in Table 7.

Table 7: A summary of all lightweight stream ciphers from academia we are aware of. The key, internal state (IS) and initialization vector (IV) sizes are expressed in bits.

Name	Reference	Key	IV	IS
A2U2	[DRL11]	61	64	95
Chacha20	[Ber08a]	256	64	256
Enocoro-80	[WIK+08]	80	64	176
F-FCSR-H/16	[ABL+09]	80/128	80/128	160/256
Grain	[HJM07]	80/128	64/96	160/256
Lizard	[HKM17]	120 †	64	121
Mickey v2	[BD08]	80/128	0–80/0–128	200/320
Plantlet	[MAM17]	80	90	110 ‡
Salsa20	[Ber08b]	256	64	256
Snow 2.0	[EJ03]	128/256	128	576
Snow 3G	[ETS06]	128	128	576
Sprout	[AM15]	80	70	89 ‡
Trivium	[Can06]	80	80	288

†While Lizard uses a 120-bit key, its designers only claim 80-bit security.
‡The key is stored separately from the internal state in non-volatile memory.

3.2.4 Dedicated Authenticated Encryption Schemes

Following the call for submissions of the Caesar competition, several lightweight authenticated encryption schemes were proposed. Some of them rely on dedicated block ciphers used with specific modes, in which case their underlying block cipher is in Table 5. However, other algorithms based on sponge transformation or stream cipher-like construction were also proposed. These are listed in Table 8. As with hash functions, sponge-based algorithms are marked with the symbol "‡".

Table 8: A summary of all lightweight authenticated ciphers from academia. The key, initialization vector (IV) and internal state (IS) are expressed in bits.

Name	Reference	Key	IV	IS
Acorn	[Wu16]	128	128	293
Ale	[BMR+14]	128	128	128
Asc-1	[JK12]	256	56	384
Ascon ‡	[DEMS16]	96/128	96/128	320
Fides	[BBK+13]	80/96	160/192	80/96
Hummingbird-2	[ESSS12]	128	64	128
Ketje ‡	[BDP+16]	$\leq 182/382$	$182-k/382-k$	200/400
Lac	[ZWW+14]	80	64	144
Norx32 ‡	[AJN16]	128	128	512
Helix	[FWS+03]	256	128	160
Sablier	[ZSX+14]	80	80	208

3.3 Algorithms from Government Agencies

Governmental agencies have published their own lightweight ciphers. The publication is often done via national standards. These ciphers are usually targeting local usage but, due to the interconnection of the markets and the corresponding standardizing efforts, these can end up being used outside of their expected zone of influence.

For example, Simon and Speck are two block ciphers designed by the American National Security Agency (nsa) which were disclosed on `eprint.iacr.org` [BSS+13].

However, they might in the end be used outside of the USA as the NSA is lobbying ISO/IEC to include them as part of the standard for "lightweight cryptography", as mentioned in Section 4.1. The designers of these algorithms were invited to present these algorithms to the Design Automation Conference (DAC) of 2015 but their paper [BTCS+15] provides little insight into their process. In particular, it discloses no information regarding their security analysis.

The Skipjack block cipher is in a similar position. The rationale behind its design is not known. The only public information comes from the report written by external cryptographers after two days spent at the NSA headquarters [BDK+93] and from several attempts at reverse-engineering its structure [KRW99, KW01] and its S-Box [BP15].

While some of the design criteria used to build the S-Boxes of the DES [U.S99] have eventually been published [Cop94], the exact generation process remains a mystery.

Finally, the same can be said of the Russian lightweight block cipher Magma. It is specified as a part of the latest Russian standard for block ciphers, GOST R 34.12–2015 [Fed15], which both describes this algorithm and the heavier Kuznyechik. Again, the rationale behind these designs is not known. Their specification merely discloses the algorithms, not their design process and especially not the best cryptanalysis of their authors.

The stream cipher ZUC was designed by the Data Assurance and Communication Security Research Center (DACAS) of the Chinese Academy of Science. It was published directly as part of the 3GPP standard [ETS11]. This addition was caused by the demand from the Chinese government to use Chinese algorithms when operating in China. ZUC should in theory not be used while in other countries. Unlike in the cases of the Russian and American algorithms, some information is provided regarding its design. In particular, several modifications were made necessary by external cryptanalysis which are described in [ETS11]. Nevertheless, the cryptanalysis (hopefully) performed by its original designers is, to the best of our knowledge, still secret.

The public lightweight ciphers designed by government agencies are listed in Table 9. The stream cipher is marked with the "†" symbol.

Table 9: A summary of all ciphers from government agencies. The key, initialization vector (IV) and internal state (IS) are expressed in bits. For block ciphers, the internal state corresponds to the block.

Name	Reference	Key	IS	Rounds	IV
DES	[U.S99]	56	64	16	–
Magma	[Fed15]	256	64	32	–
SIMON	[BSS+13]	64–256	32–128	32–72	–
Skipjack	[U.S98]	80	64	32	–
SPECK	[BSS+13]	64–256	32–128	22–34	–
ZUC †	[ETS11]	128	560	–	128

4 Lightweight Cryptography in the Wild

Several standards and libraries are aimed at use cases overlapping with those of lightweight cryptography. These are listed in this section and summarized in Table 10. Section 4.1 deals with ISO/IEC standards, Section 4.2 with regional cryptographic ones, Section 4.3 with general purpose communication protocols run by low power devices and, finally, Section 4.4 describes several libraries specifically aimed at the IoT.

Table 10: Standards and libraries involving lightweight algorithms.

Type	Name	Lightweight algorithms standardized
ISO/IEC	29167	AES-128, PRESENT-80, Grain-128A
	29192-2	PRESENT, CLEFIA
	29192-3	Enocoro, Trivium
	29192-5	PHOTON, Lesamnta-LW, Spongent
	18033-3	AES, MISTY1, HIGHT
	18033-4	SNOW 2.0
Regional	FIPS 185 (USA)	Skipjack (now deprecated [BR15])
	FIPS 197 (USA)	AES
	NESSIE (EU)	AES, MISTY1
	eSTREAM portfolio (EU)	Grain, Trivium, Salsa20, MICKEY
	GOST R 34.12–2015 (Russia)	Magma
Protocols	GSM	A5/1, A5/2, A5/3 (KASUMI)
	3G	SNOW 3G, ZUC, AES, KASUMI
	Bluetooth	E0, AES
	Bluetooth smart	AES
	WEP	RC4
	WPA	RC4
	WPA2	AES
	Lora Alliance	AES
	IEEE 802.15.4 (Zigbee)	AES
Embedded Lib.	Tinysec	Skipjack (CBC), (RC5)
	Minisec	Skipjack (OCB)
	mbedTLS (ciphers)	AES, RC4, XTEA, Blowfish, 3-DES, Camellia
	mbedTLS (hash functions)	MD5, SHA-1, SHA-256, SHA-512

4.1 Iso/iec cryptographic standards.

The International Organization for Standards (ISO) and the International Electrotechnical Commission (IEC) are tasked with issuing and maintaining standards regarding information and communication technology.

Three of their standards are particularly relevant for lightweight cryptography. The first is *ISO/IEC 29167: Information technology – Automatic identification and data capture techniques*, in particular parts 10, 11 and 13. Those deal with the symmetric ciphers that should be used for securing "air interface communications", that is, RFID tags. These parts describe respectively AES-128, PRESENT-80 and Grain-128A. Other parts deal with public key cryptography.

Another set of relevant ISO/IEC standards are those with number 29192 which deal specifically with "lightweight cryptography". The following algorithms are part of this series of standards: the block ciphers PRESENT and CLEFIA, the stream ciphers Trivium and Enocoro, and the hash functions PHOTON, Spongent and Lesamnta-LW. The criteria for algorithms to be considered for inclusion in this standard are listed in the following quote from Annex A of said standard.

> a) The security of the cryptographic mechanism. 80-bit security is considered to be the minimum security strength for lightweight cryptography. It is however recommended that at least 112-bit security be applied for systems that will require security for longer periods (refer to SD12 for security strength references, as the period of protection provided is determined by the security strength as well as the computing power of the adversary who wishes to break the algorithm.).

b) The hardware implementation properties (for hardware targeted mechanisms). The chip area occupied by the cryptographic mechanism (reduced compared to existing ISO standards) and the energy consumption. (clear advantage over existing ISO standards, e.g. ISO/IEC 18033, ISO/IEC 9798, ISO/IEC 11770).

c) The software implementation properties (for software targeted mechanisms). In particular, the code size and the required RAM size. (Less resource requirements compared to existing standards on the same platform are considered as potentially lightweight for software environments).

d) The nature of any licensing issues affecting the cryptographic mechanism.

e) The maturity of the cryptographic mechanism.

f) The generality of the lightweight properties claimed for the cryptographic mechanism (i.e. the more independent the claimed lightweight property is from implementation in a specific technology, the better, as it will be useable by a wider audience).

At the time of writing, the block ciphers Simon and Speck designed by the nsa were being considered for inclusion in this standard.

Finally, standard 18033 describes "Encryption algorithms". Some of those can be considered lightweight, such as the block ciphers Aes, Misty1 and Hight (in 18033-3) and the stream cipher Snow 2.0 (in 18033-4).

4.2 Regional Cryptographic Standards

Several regional standards deal with cryptography in general and some of the algorithm specified in them can be considered to be lightweight. In the usa, cryptographic standards are handled by the National Institute for Standards and Technology (nist) which famously standardized the Aes after an open competition. This institution is currently working towards a standard for lightweight cryptography, as explained in a detailed report [MBTM16]. Their intention is to agree upon several *profiles* corresponding to different algorithms, use cases and constraints. Then, possibly different algorithms will be standardized for use in each of these profiles. The legacy cipher Skipjack was a nist standard but it is now deprecated.

In Europe, the Nessie project selected several block ciphers including the Aes and Misty1. Its failure to find good stream ciphers lead to the estream competition. At its end, a portfolio of stream ciphers was published. It is divided into two profiles, one software oriented and one hardware oriented. Several of those stream ciphers can be considered to be lightweight: Trivium, Grain, Mickey and Salsa20.

Finally, the latest Russian standard for block ciphers contains the 64-bit block cipher Magma.

4.3 Communication protocols

Several communication protocols specify a form of encryption which, given the nature of the devices running them, have to be lightweight. For example, cell phones are not nearly as powerful as computers, although Moore's law and modern smartphones complicate this picture.

The GSM and 3G networks deal with cell phone communication. They specify that communications should be encrypted using A5/1, A5/2, A5/3 (Kasumi in counter mode), Snow 3G, Zuc or Kasumi, the latter being a variant of Misty1.

Bluetooth connects devices over short distances. The original specification required the stream cipher E0 but it was later replaced by the Aes. A more recent variant called

"Bluetooth smart" aims at lower energy consumption. It also relies on the AES for its security.

Modern WiFi connections are secured using WPA or WPA2. The former uses RC4 while the latter moved on to the AES. The previous standard was WEP, which used RC4, but practical attacks exist against it.

Several protocols have recently been proposed to connect wireless IoT devices to one another. The one put forward by the Lora Alliance uses the AES. The same is true for IEEE 802.15.4, which is used e.g. in Zigbee.

4.4 IoT Oriented Libraries.

Let us look at two libraries intended for embedded devices. The first we consider is `tinysec`[5] which is used in the security-related stack of the TinyOS operating system. It uses Skipjack in CBC mode, although RC5 was also considered and found to be quite efficient [KSW04]. Its successor is `minisec`[6] and it also uses Skipjack but in OCB mode. The library `mbedTLS` which also targets embedded devices but is not tied to TinyOS offers several algorithms, namely the ciphers AES, RC4, XTEA, Blowfish, 3-DES and Camellia as well as the hash functions MD5, SHA-1, SHA-256 and SHA-512.

5 Trends in Lightweight Design

Lightweightness can be seen as a set of specific design constraints. These are tackled differently by different algorithms but some trends emerge when we look at the evolution of lightweight block ciphers. These are particularly visible on two fronts: the choice of the non-linear operations and the key schedule which are described respectively in Section 5.1 and in 5.2. In Section 5.3, the fact that fewer bad ciphers seem to be in use now than 15 years ago is discussed.

5.1 Non-Linear Operations

Non-linearity is a necessary property of any cryptographic primitive. It can be provided by S-Boxes or through the use of non-linear arithmetic operations. S-Box-based algorithms can further be divided into two categories. The first implements them using Look-Up Tables (LUT) and the second relies on bit-sliced implementations. As for arithmetic operations, only modular additions are considered here, that is, primitives following the ARX paradigm. Although other operations are sometimes used, such as modular multiplication in the block cipher IDEA [LM91] and the PC1 stream cipher, these are extremely uncommon.

5.1.1 Lut-based

LUT-*based* algorithms use S-Boxes which are intended to be implemented using look-up tables in software. Such functions are useful as they can offer (near) optimal cryptographic properties using only one operation. However, their implementation requires storing all possible outputs which, for an 8-bit S-Box such as the one used by the AES, has a significant cost. Furthermore, the table look-up is the operation leaking the most information, as shown in [BDG16].

S-Boxes intended to be implemented using LUTs in software usually correspond to a simple electronic circuit which can be efficiently implemented in hardware, such as the 4-bit S-Boxes used by Piccolo, PRESENT and SKINNY.

[5] It is described in the wiki of the TinyOS operating system: http://tinyos.stanford.edu/tinyos-wiki/index.php/TinySec.

[6] See http://tinyos.stanford.edu/tinyos-wiki/index.php/MiniSec.

5.1.2 Bit-slice-based

Bit-slice-based algorithms also use S-Boxes but, in this case, the S-Box is supposed to be implemented in a bit-sliced fashion: no table look-ups are required to evaluate the S-Box layer. Instead, some bitwise operations such as AND and XOR are performed on words of w bits, thus evaluating the S-Box in parallel w times.

S-Boxes implemented in this fashion are typically designed for this purpose. Thus, they require only a limited number of logical operations: 4-bit ones usually need only 4 ANDs during their evaluation which makes their software implementation particularly easy to mask. At the same time, they allow simple security argument based for example on the wide trail strategy. A simple bit-sliced implementation is also related — but is not equivalent to — a small area for a hardware implementation, meaning that such algorithms can be expected to perform well in hardware as well.

Because of these properties, bit-sliced S-Boxes are a popular choice for the design of lightweight algorithms, especially during the last 4 years. For example, all the algorithms in the following list use such components.

- 3-Way
- Ascon
- Fantomas
- Fly
- iScream
- Ketje
- Mysterion
- Noekeon
- Pride
- Rectangle
- RoadRunneR
- Scream

5.1.3 Arx-based

Arx-*based* algorithms rely on modular addition to provide non-linearity while word-wise rotations and XOR provide diffusion, hence the name: Addition, Rotation, XOR.

The bits of highest weight in the output of a modular addition are highly non-linear functions due to the propagation of the carry. However, the lower weight bits retain a simple dependency. Furthermore, some differentials and linear approximations have probability 1, meaning that the structure of the linear part must be studied carefully. In fact, to the best of our knowledge, Sparx [DPU+16] is the only ARX-based primitive designed to be provably secure against differential and linear attacks.

Modular addition is fairly expensive to implement in hardware, especially if the size of the words is larger as this significantly increases the length of the critical path. On the other hand, it is extremely cheap in software. Not only does it consist in one operation, it also uses fewer or no additional registers as it can often be performed in place using the "+=" operator.

As a consequence, ARX-based ciphers are among the best performers for micro-controllers identified using Felics. Some ARX-based block and stream ciphers are listed below.

- Chacha20
- Chaskey
- Hight
- Lea
- RC5
- Salsa20
- Sparx
- Speck
- Xtea

5.2 Key Schedule

The key schedule is the area where lightweight algorithms differ the most from their non-lightweight counterparts. Indeed, for algorithms intended to run on standard computers, it is fine to have a complex key schedule as it would typically be run only once, the corresponding subkeys being subsequently stored. For lightweight algorithms, the incurred cost in terms of RAM or gate area is unacceptable. Furthermore, it is common for lightweight algorithms to dismiss resilience against related key attacks, a design decision which authorizes the use of much simpler key schedules.

Different attitudes regarding related-key attacks are discussed in Section 5.2.1. Popular strategies for building simple key schedules are described in Sections 5.2.2 and 5.2.3.

Some recent proposals provide a tweak in addition to the secret key [LRW02]. It is a public parameter which enables the use of more sophisticated modes of operation. In fact, most of these algorithms are parts of authenticated ciphers submitted to the CAESAR competition. However, an overwhelming majority of lightweight block ciphers do not provide this functionality.

5.2.1 On Related-Key Attacks

Some cipher designers claim resilience against related-key attacks while some other algorithms are trivially vulnerable against such attacks. For example, the block cipher PRINCE has a very simple related-key distinguisher because of its α-reflection. On the other hand, other algorithms explicitly give resilience against related-key as a design criteria.

Preventing related-key attacks is a more conservative choice. Whatever the setting, from a security standpoint, being protected against such adversaries can only be an advantage. And yet this resilience has a cost since it implies the use of more rounds and/or more complex key schedules which lead to a performance degradation particularly unwelcome in the lightweight setting. Furthermore, for devices using a unique factory-defined key throughout their lifetimes, the probability of finding two devices with the appropriate key relation is small enough that it is of no practical concern. Similarly, if the protocols using the ciphers are properly implemented, related-key attacks should not be possible.

The following algorithms were explicitly *not* designed to prevent related-key attacks. Still, the approach used for Noekeon and FLY is a bit more subtle. Indeed, for cases where related-key attacks might be of concern, the authors provide a modified key schedule. While normally the master key is simply XORed in the state, as for the ciphers in Section 5.2.2, the related-key protected version imposes that the master key first goes through several rounds of the round function so as to break any pattern relating the keys.

- Fantomas
- FLY
- Mysterion
- Noekeon
- PRIDE
- PRINCE
- Zorro
- ...

Some designers prefer to make the most conservative choice by providing related-key security. Some of the corresponding algorithms are listed below. These usually employ a more complex key-schedule but, since they remain lightweight ciphers, those can be evaluated "on the fly" cheaply. It means that the subkeys are obtained by extracting bits from a key state which is updated in every round, just like the internal state of the block cipher. However, this update function is kept simple to limit the performance overhead.

- EPCBC
- LBlock
- SEA
- SIMON
- SKINNY
- SPARX
- TWINE
- ...

5.2.2 Even-Mansour and "Selecting" Key Schedules

It is popular for lightweight algorithms to use a key schedule which merely selects different bits of the master key in each round for use as subkey material along with some round constants. If the master key of a block cipher is simply XORed to the internal state during each round along with a round constant, the key schedule can be seen as a variant of the Even-Mansour construction [EM97]. Of course, it is possible to use said construction directly, as is the case for the Chaskey cipher. It is also possible to use different chunks of the master key during encryption. For example, Skipjack uses a 32-bit subset of its 80-bit master key in each round. The subkeys therefore repeat themselves every 5 rounds. We

call such a key schedule a *selecting* key schedule since it merely selects some bits of the master key for use as subkeys.

The main advantage of such methods is that they require very little logic to compute the round keys. Furthermore, they have no need for a key state getting updated at each round which would be particularly expensive in hardware. This observation is what lead the designers of the stream cipher Sprout, followed later by those of Lizard and Plantlet, to fix the content of one of their registers to be the master key without modifying it. In fact, the paper introducing Plantlet [MAM17] provides a detailed analysis of the way a key stored in non-volatile memory can be accessed and its impact on both performance and algorithm design. For example, it is better to access master key bits sequentially, like in Skipjack and in LED, than to use master key bits that are far apart to build a given round key.

Below, we list all ciphers using the master key in such a way that no key state needs to be maintained. It encompasses the (iterated) Even-Mansour construction, the "selecting" key schedules and the stream ciphers that do not modify their key register. Stream ciphers are indicated by the "†" symbol.

- 3-Way
- Chaskey
- DES
- Fantomas
- FLY
- GOST revisited
- HIGHT
- iScream
- ITUbee
- KTANTAN
- LED
- Lizard †
- Magma
- Midori
- Mysterion
- Noekeon
- Piccolo
- Plantlet
- PRIDE
- PRINCE
- RoadRunneR
- Robin
- Scream
- Skipjack
- Sprout †
- XTEA
- Zorro

The impact of such a key schedule in terms of gate area in hardware is extensively discussed in the recent paper [MAM17] which introduced Plantlet.

5.2.3 Round Function Based

A simple strategy to have a substantial key schedule while minimising its cost is to reuse significant parts of the round function to update the key state. The whole round function can be used, as in SPECK, or only parts of it, as in SPARX. Several block ciphers using this principle are listed below. For FLY and Noekeon, only the key schedule protecting against related-key attacks is concerned.

- EPCBC
- FLY
- Noekeon
- SEA
- SIMECK
- SPARX
- SPECK

5.3 No More Non-Standard Ciphers?

So far, we have discussed trends regarding algorithm design. But there is another trend at a higher level: questionable ciphers such as those listed in Section 3.1 are being phased out. Nowadays, the prevalence of the AES means that using algorithms such as A5/1 would be unacceptable. Not only new standards are concerned: previously existing standards such as Bluetooth have been amended to move away from their previous *ad hoc* solutions (here, the E0 stream cipher) and towards more common choices (for Bluetooth, the AES). The reason behind this change is probably two-fold.

First, the lessons from the attacks targeting proprietary algorithms have likely been learnt. Thus, once these standards had to be replaced by more modern ones, the cryptography used was updated at the same time.

Second, the AES likely played a significant role. The fact that it performs decently — even if not optimally — on a wide variety of platforms, means that it a priori constitutes

a satisfactory choice in most situations. As explained in Section 2.5, the situation is unfortunately a bit more complicated but using this algorithm is nevertheless a significant step up from what was done before. As a consequence of this versatility, the AES has been formally standardized for use in most areas, as explained in Section 4. Still, the time it took for the algorithms from Section 3.1 to be phased out shows the importance of getting an algorithm choice right from the start.

6 Two Faces for Lightweight Crypto

Providing a formal definition of "lightweightness" is a difficult task because different algorithms corresponding to different sets of requirements claim to fall under its umbrella. In this section, we argue that the area of lightweight should be split into two distinct fields. The first is *ultra-lightweight cryptography*, described in Section 6.1. The second is *IoT cryptography* and is discussed in Section 6.2.

As evidenced by all the primitives listed in Section 3, *a lot* have been proposed: the list in Section 3.2.1 contains only block ciphers published by academics and it contains 47 entries! More importantly, even within this a priori narrower subset, algorithms differ greatly. Let us look at two extreme cases:

- KTANTAN encrypts blocks consisting of at most 64 bits using an 80-bit key and 254 very simple rounds, while
- LEA uses 32-bit modular additions, XORs and rotations to encrypt 128-bit blocks with keys of length 128, 192 or 256.

Both of these algorithms would be considered lightweight, and rightfully so: the circuit needed to evaluate the non-linear function of KTANTAN consists only in a few gates while LEA is one of the top performers in the FELICS triathlon. And yet, a category so wide that two algorithms so different both fit comfortably in it must be of little use in terms of classification.

The distinction between these block ciphers goes beyond their intended target — although KTANTAN is indeed hardware-oriented and LEA software oriented. In fact, their differences highlight another gap: what security level is desirable in the context of lightweight cryptography? There are two broad schools of thought on this matter.

- Lightweight algorithms are intended to run on cheap devices securing cheap objects, for example RFID tags used to track an inventory of T-shirts. What would be the point in paying for a high level of security in this context? Indeed, the consequences of an adversary successfully attacking these tags would be local at worst.
- On the other hand, IoT devices are, by definition, connected to the Internet. It implies that they can be used e.g. for Denial-of-Service attacks, as has already happened. In this context, can we afford *not to* have a maximum level of security?

Below, we argue that both points of view are correct. Rather, the mistake lies in lumping both use cases together.

6.1 Ultra-Lightweight Crypto

As its name indicates, this type of algorithm deals with the most constrained use cases. On top of the power of the devices, what defines this field is also their connectivity.

For example, RFID tags used for challenge-response based access control may not need full-on 256-bit security against adversaries having access to the full code-book. As the throughput of these devices is very limited, it makes sense to discard attacks requiring

too much data. Similarly, should the secret key used by this card be recovered, the consequences would be restricted geographically to whatever place this card was used in. It would be the same as a physical key being lost and revoking the access of this card would be the same as changing door locks. In this context, "weak" cryptography with only 80-bit keys may be understandable. However, a great emphasis on side-channel protection is likely to be necessary in many cases.

Definition 1 (Ultra-Lightweight Crypto)**.** An *ultra-lightweight* cryptographic algorithm is one running on very cheap devices which are not connected to the internet, which are easily replaced if necessary and have a limited shelf-life.

Possible use cases for such algorithms include RFID tags, RAIN [Rob16] tags, smart cards, remote car keys, memory encryption... Besides, many algorithms already fit this bill such as Grain, KTANTAN, PHOTON, PRESENT, PRINCE, SKINNY and Trivium to name a few.

Because the implementation constraints are particularly stringent in this context, some specific trade-offs can be relevant. For example, PRINCE is unlikely to make it to the top of the FELICS triathlon because its design was aimed at low-latency in hardware. Lower block sizes are also acceptable.

Here are some properties an ideal ultra-lightweight algorithm and its implementation may have.

- *Type:* block/stream cipher for versatility and small memory footprint.
- *Block size:* 64 bits (or more if possible).[7]
- *Key size:* at least 80 bits, more if possible.
- *Relevant attacks:* since the devices running ultra-lightweight algorithms have very little computing power, they cannot be expected to produce large amounts of data. Thus, it makes sense to only consider attacks with rather low data complexity.[8] By only considering attackers with access to, say, less than 2^{50} plaintext/ciphertext pairs, it is possible to reduce the total number of rounds of the primitive while retaining k bits of security. Given the low power of the devices running ultra-lightweight algorithms, saving several rounds may be a welcome performance gain.
- SCA *resilience:* countermeasures must be easy to implement by design.
- *Use of non-volatile memory:* using non-volatile memory to store the key is cheaper, even though it may require extra care when designing the algorithm.
- *Functionality:* only one type of operation per device — for example, the block cipher used on a given smart card will only be used in a challenge-response protocol, meaning that this device has no need for a hash function or a MAC. The versatility of the primitive is therefore not that important in this context.
- *Implementation flexibility:* it must be possible to optimise any of the relevant efficiency metrics. In other words, if a low latency is needed, the algorithm should allow it at the cost of a possible increase in area or decrease in throughput. Similarly, a low area should be possible, etc. To quote [Rob16] (emphasis his): "*flexibility* gives the opportunity to find the right trade-off".

The ecosystem of ultra-lightweight algorithm can be expected to be diverse. While a unique algorithm capable of fitting in every niche of the design space would be welcome, it is likely that different algorithms are used in different cases.

[7] In [Rob16], Robshaw explains that even RAIN RFID tags can afford blocks of 64 bits: "there is no demand for very short block lengths (*e.g.* 48 bits)".

[8] This limitation has already been suggested by the designers of PRINCE when they issued the "PRINCE challenge" https://www.emsec.rub.de/research/research_startseite/prince-challenge/.

6.2 IoT Crypto

The other subfield of lightweight cryptography is *IoT cryptography*. It is oriented toward the IoT in its most literal sense, that is, it deals with objects connected to the internet. While remaining computationally weak compared to a desktop computer or a higher end smartphone, these devices perform multiple tasks. Accordingly, the primitive they use must be versatile: unlike ultra-lightweight devices which only need one cryptographic operation, IoT ones need to both encrypt and authenticate their communications with their user, authenticate the updates from their manufacturers, etc.

Another key difference is the importance of their security. The consequences are further reaching in the case of IoT ones due to their network connection. For example, the attacks against the"smart" light bulbs presented in [ROSW16] can spread from one light bulb to the next. Furthermore, it could be used effectively to jam the WiFi signal in a vast geographical area because of the overlap between the frequencies involved. Similarly, an insecure IoT-enabled device can be used in a DoS attack to take down websites. As we can see, the security level needed in those cases is much higher: 80-bit keys are not acceptable in this context, at least 128 bits are necessary.

Definition 2 (IoT Crypto)**.** An *IoT* cryptographic algorithm is one running on a low-power device connected to a global network such as the Internet.

Unlike in the ultra-lightweight case, there should ideally be only one algorithm or one suite of algorithms for all IoT devices: as they are all networked, they must all use the same primitives. Since some of these devices will run in conditions in which an attacker may physically access them, such as outdoor security cameras, it is crucial that SCA counter-measures be easy to implement.

Because IoT devices perform multiple tasks, the cryptographic operations will be performed by their multi-purpose micro-controllers rather than an electronic circuit. Thus, software efficiency is much more important in this case. Several lightweight algorithms have been explicitly designed for software rather than hardware implementation: Chaskey, FLY, LEA, PRIDE, SPARX...

In light of all this, let us list some of the properties such an IoT algorithm should have.

- *Type:* block cipher or sponge. In the IoT case, the device must be able to perform many different operations so a versatile primitive is needed. Furthermore, as the intended target is a micro-controller, the idea of "burning" the key into the circuit does not make sense. Instead, the key will have to occupy some registers which may as well be used to build the larger internal state of a sponge.

- *Block size:* 96 bits[9] is the minimum, higher sizes must be preferred. Ideally, the internal state and the key — if any — should all fit into the registers of a typical micro-controller.

- *Key size:* at least 128 bits. In recent years, many block cipher designers have provided several versions of their algorithms with different key sizes, typically 80 and 128 bits. While this might make an ultra-lightweight implementation of the smallest one more efficient, it makes little sense in the IoT context. Indeed, in most cases, two versions of a given algorithm have very similar performance. In the FELICS ranking,[10] we can see that SIMON-96/64 and SIMON-128/128 follow one another and have very similar FOM in all scenarios. The same holds for SPECK-96/64 and SPECK-128/128,

[9] A block size of 96 bits is such that the data complexity of an attack based on the birthday paradox is 2^{16} times higher than for a 64-bit block. Attacking 64-bit block cipher is practical, as illustrated by [BL16], but multiplying the data complexity of such attacks by 2^{16} is sufficient to have some security margin.

[10] A summary of which is available at https://www.cryptolux.org/index.php/FELICS_Block_Ciphers_Brief_Results.

RECTANGLE-80 and RECTANGLE-128, etc. In the IoT case, a slew of key sizes as provided e.g. by SPECK is thus not necessary. Furthermore, since the algorithms will be used by more sophisticated protocols, providing small key sizes may lead to downgrade attacks.

- *Relevant attacks:* the security model must be more conservative than in the ultra-lightweight case. For example, limiting the amount of data available to the adversary would be too restrictive.
- SCA *resilience:* countermeasures must be easy to implement by design.
- *Use of non-volatile memory:* the use of non-volatile memory is less relevant in this context as the main target is software implementation.
- *Functionalities:* encryption, authentication, hashing... Such devices communicate non-trivial data with different actors, which means that they require several cryptographic functionalities.
- *Flexibility:* the algorithm must be decently efficient on a wide range of micro-controllers. Hardware efficiency is less important but it may help with hardware acceleration.

7 Conclusion

Lightweight cryptography has received significant attention in the last two decades and even more so in the last 5 years. The need for such algorithms is well established, as evidenced by the NIST effort to standardize such algorithms and the short-comings of the AES in this context.

Table 11: A summary of the differences between ultra-lightweight and IoT cryptography.

	Ultra-Lightweight	IoT
Block size	64 bits	$\geq$ 128 bits
Security level	$\geq$ 80 bits	$\geq$ 128 bits
Relevant attacks	low data/time complexity	Same as "regular" crypto
Intended platform	dedicated circuit (ASIC, RFID...)	micro-controllers, low-end CPUs
SCA resilience	important	important
Functionality	one per device, e.g. authentication	encryption, authentication, hashing...
Connection	temporary, only to a given hub	permanent, to a global network

However, the spectrum of use cases encompassed by "lightweightness" has become *too* wide. Thus, we propose to split the field of lightweight cryptography into two areas corresponding to two different types of requirements: ultra-lightweight and IoT cryptography, whose particularities are summarized in Table 11. There are of course common criteria between those cases such as the emphasis that needs to be put on resilience against side channel attacks. Still, we think that the difference between the two is relevant. In particular, we think this separation is necessary because of the different levels of security they demand: connecting a family of devices to a global network and protecting them with an 80-bit key is not a desirable situation, and yet it is what may happen if an ultra-lightweight algorithm is used where an IoT one is needed.

8 Acknowledgement

The authors thank Daniel Dinu for fruitful discussions about implementation and side-channel protection issues. The work of Léo Perrin was supported by the CORE project

ACRYPT (ID C12-15-4009992) funded by the Fonds National de la Recherche, Luxembourg.

References

[AB12] Jean-Philippe Aumasson and Daniel J. Bernstein. SipHash: A fast short-input PRF. In Steven D. Galbraith and Mridul Nandi, editors, *Progress in Cryptology - INDOCRYPT 2012: 13th International Conference in Cryptology in India*, volume 7668 of *Lecture Notes in Computer Science*, pages 489–508. Springer, Heidelberg, December 2012.

[ABL+09] François Arnault, Thierry P. Berger, Cédric Lauradoux, Marine Minier, and Benjamin Pousse. A new approach for FCSRs. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *SAC 2009: 16th Annual International Workshop on Selected Areas in Cryptography*, volume 5867 of *Lecture Notes in Computer Science*, pages 433–448. Springer, Heidelberg, August 2009.

[ABP+13] Nadhem J. AlFardan, Daniel J. Bernstein, Kenneth G. Paterson, Bertram Poettering, and Jacob CN Schuldt. On the security of RC4 in TLS. In *Proceedings of the 22nd USENIX Security Symposium*, volume 2013, Washington DC, USA, 2013. USENIX.

[ADK+14] Martin R. Albrecht, Benedikt Driessen, Elif Bilge Kavun, Gregor Leander, Christof Paar, and Tolga Yalçin. Block ciphers - focus on the linear layer (feat. PRIDE). In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 57–76. Springer, Heidelberg, August 2014.

[AHMN10] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and María Naya-Plasencia. Quark: A lightweight hash. In Mangard and Standaert [MS10], pages 1–15.

[AJN16] Jean-Philippe Aumasson, Philipp Jovanovic, and Samuel Neves. NORX v3.0. Candidate for the CAESAR Competition. See also https://norx.io, 2016.

[AM15] Frederik Armknecht and Vasily Mikhalev. On lightweight stream ciphers with shorter internal states. In Gregor Leander, editor, *Fast Software Encryption – FSE 2015*, volume 9054 of *Lecture Notes in Computer Science*, pages 451–470. Springer, Heidelberg, March 2015.

[And94] Ross Anderson. A5 (Was: HACKING DIGITAL PHONES). uk.telecom (Usenet), https://groups.google.com/forum/?msg/uk.telecom/TkdCaytoeU4/Mroy719hdroJ#!msg/uk.telecom/TkdCaytoeU4/Mroy719hdroJ, June 1994.

[ANWOW13] Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein. BLAKE2: simpler, smaller, fast as MD5. Available online: https://blake2.net/blake2.pdf, 2013.

[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 Transactions on Symmetric Cryptology*, 2017(1):4–44, 2017.

[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, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pages 411–436. Springer, Heidelberg, November / December 2015.

[BBK03] Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 600–616. Springer, Heidelberg, August 2003.

[BBK08] Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. *Journal of Cryptology*, 21(3):392–429, July 2008.

[BBK+13] Begül Bilgin, Andrey Bogdanov, Miroslav Knežević, Florian Mendel, and Qingju Wang. Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware. In Bertoni and Coron [BC13], pages 142–158.

[BBR16] Subhadeep Banik, Andrey Bogdanov, and Francesco Regazzoni. Atomic-AES: A compact implementation of the AES encryption/decryption core. In Orr Dunkelman and Somitra Kumar Sanadhya, editors, *Progress in Cryptology – INDOCRYPT 2016*, volume 10095 of *Lecture Notes in Computer Science*, pages 173–190, Cham, 2016. Springer International Publishing.

[BC13] Guido Bertoni and Jean-Sébastien Coron, editors. *Cryptographic Hardware and Embedded Systems – CHES 2013*, volume 8086 of *Lecture Notes in Computer Science*. Springer, Heidelberg, August 2013.

[BCG+12] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knežević, 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*, volume 7658 of *Lecture Notes in Computer Science*, pages 208–225. Springer, Heidelberg, December 2012.

[BD04] M. Becker and A. Desoky. A study of the DVD content scrambling system (CSS) algorithm. In *Proceedings of the Fourth IEEE International Symposium on Signal Processing and Information Technology, 2004.*, pages 353–356, Dec 2004.

[BD08] Steve Babbage and Matthew Dodd. The MICKEY stream ciphers. In Matthew Robshaw and Olivier Billet, editors, *New Stream Cipher Designs*, volume 4986 of *Lecture Notes in Computer Science*, pages 191–209. Springer Berlin Heidelberg, 2008.

[BDG16] Alex Biryukov, Daniel Dinu, and Johann Großschädl. Correlation power analysis of lightweight block ciphers: From theory to practice. In *International Conference on Applied Cryptography and Network Security – ACNS 2016*, volume 9696 of *Lecture Notes in Computer Science*, pages 537–557. Springer, 2016.

[BDK+93] Ernest F. Brickell, Dorothy E. Denning, Stephen T. Kent, David P. Maher, and Walter Tuchman. SKIPJACK review: Interim report. This note is available at http://faculty.nps.edu/dedennin/publications/SkipjackReview.txt., 1993.

[BDM+12] Thierry P. Berger, Joffrey D'Hayer, Kevin Marquet, Marine Minier, and Gaël Thomas. The GLUON family: A lightweight hash function family based on FCSRs. In Mitrokotsa and Vaudenay [MV12], pages 306–323.

[BDN+10] Stéphane Badel, Nilay Dagtekin, Jorge Nakahara, Khaled Ouafi, Nicolas Reffé, Pouyan Sepehrdad, Petr Susil, and Serge Vaudenay. ARMADILLO: A multi-purpose cryptographic primitive dedicated to hardware. In Mangard and Standaert [MS10], pages 398–412.

[BDP+16] Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Caesar submission: Ketje v2. Candidate for the CAESAR Competition. See also http://ketje.noekeon.org/, 2016.

[Ber08a] Daniel J. Bernstein. Chacha, a variant of Salsa20. SASC 2008 – the State of the Art in Stream Ciphers. See also https://cr.yp.to/chacha.html, 2008.

[Ber08b] Daniel J. Bernstein. The salsa20 family of stream ciphers. In Matthew Robshaw and Olivier Billet, editors, *New Stream Cipher Designs: The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 84–97, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[BFMT15] Thierry Pierre Berger, Julien Francq, Marine Minier, and Gaël Thomas. Extended generalized feistel networks using matrix representation to propose a new lightweight block cipher: Lilliput. *IEEE Transactions on Computers*, PP(99), August 2015.

[BGS+05] Stephen C. Bono, Matthew Green, Adam Stubblefield, Ari Juels, Aviel D. Rubin, and Michael Szydlo. Security analysis of a cryptographically-enabled RFID device. In *Proceedings of the 14th Conference on USENIX Security Symposium - Volume 14*, SSYM'05, pages 1–1, Berkeley, CA, USA, 2005. USENIX Association.

[Bir07] Alex Biryukov, editor. *Fast Software Encryption – FSE 2007*, volume 4593 of *Lecture Notes in Computer Science*. Springer, Heidelberg, March 2007.

[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, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, Heidelberg, August 2016.

[BJM+14] Lejla Batina, Domagoj Jakobovic, Nele Mentens, Stjepan Picek, Antonio De La Piedra, and Dominik Sisejkovic. S-box pipelining using genetic algorithms for high-throughput AES implementations: How fast can we go? In Willi Meier and Debdeep Mukhopadhyay, editors, *Progress in Cryptology - INDOCRYPT 2014: 15th International Conference in Cryptology in India*, volume 8885 of *Lecture Notes in Computer Science*, pages 322–337. Springer, Heidelberg, December 2014.

[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*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, Heidelberg, September 2007.

[BKL+11] Andrey Bogdanov, Miroslav Knežević, Gregor Leander, Deniz Toz, Kerem Varici, and Ingrid Verbauwhede. Spongent: A lightweight hash function. In Preneel and Takagi [PT11], pages 312–325.

[BKLM09] Julia Borghoff, Lars R. Knudsen, Gregor Leander, and Krystian Matusiewicz. Cryptanalysis of C2. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 250–266. Springer, Heidelberg, August 2009.

[BKZ11] Alex Biryukov, Ilya Kizhvatov, and Bin Zhang. Cryptanalysis of the Atmel cipher in SecureMemory, CryptoMemory and CryptoRF. In Lopez and Tsudik [LT11], pages 91–109.

[BL16] Karthikeyan Bhargavan and Gaëtan Leurent. On the practical (in-)security of 64-bit block ciphers: Collision attacks on HTTP over TLS and OpenVPN. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 456–467, New York, NY, USA, 2016. ACM.

[BLN15] Daniel J. Bernstein, Tanja Lange, and Ruben Niederhagen. Dual EC: A standardized back door. Cryptology ePrint Archive, Report 2015/767, 2015. http://eprint.iacr.org/2015/767.

[BLP04] Alex Biryukov, Joseph Lano, and Bart Preneel. Cryptanalysis of the alleged SecurID hash function. In Mitsuru Matsui and Robert J. Zuccherato, editors, *SAC 2003: 10th Annual International Workshop on Selected Areas in Cryptography*, volume 3006 of *Lecture Notes in Computer Science*, pages 130–144. Springer, Heidelberg, August 2004.

[BLR13] Alex Biryukov, Gaëtan Leurent, and Arnab Roy. Cryptanalysis of the "kindle" cipher. In Knudsen and Wu [KW13], pages 86–103.

[BMR+14] Andrey Bogdanov, Florian Mendel, Francesco Regazzoni, Vincent Rijmen, and Elmar Tischhauser. ALE: AES-based lightweight authenticated encryption. In Moriai [Mor14], pages 447–466.

[BMS06] Alex Biryukov, Sourav Mukhopadhyay, and Palash Sarkar. Improved time-memory trade-offs with multiple data. In Bart Preneel and Stafford Tavares, editors, *SAC 2005: 12th Annual International Workshop on Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 110–127. Springer, Heidelberg, August 2006.

[BP15] Alex Biryukov and Léo Perrin. On reverse-engineering S-boxes with hidden design criteria or structure. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *Advances in Cryptology – CRYPTO 2015, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 116–140. Springer, Heidelberg, August 2015.

[BPU16] Alex Biryukov, Léo Perrin, and Aleksei Udovenko. Reverse-engineering the S-box of streebog, kuznyechik and STRIBOBr1. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 372–402. Springer, Heidelberg, May 2016.

[BR00] Paulo Barreto and Vincent Rijmen. The Khazad legacy-level Block Cipher. First Open NESSIE Workshop, see also https://www.cosic.esat.kuleuven.be/nessie/workshop/submissions/khazad.zip, 2000.

[BR15] Elaine Barker and Allen Roginsky. NIST special publication 800-131a revision 1. *NIST Special Publication*, 800(131A):1–29, 2015.

[BŞ16] Adnan Baysal and Sühap Şahin. RoadRunneR: A small and fast bitslice block cipher for low cost 8-bit processors. In Tim Güneysu, Gregor Leander, and Amir Moradi, editors, *Lightweight Cryptography for Security and Privacy — LightSec 2015*, volume 9542 of *Lecture Notes in Computer Science*, pages 58–76, Berlin, Heidelberg, 2016. Springer International Publishing.

[BSK96] F.J. Bruwer, W. Smit, and G.J. Kuhn. Microchips and remote control devices comprising same, May 1996. US Patent 5,517,187.

[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. http://eprint.iacr.org/2013/404.

[BSW01] Alex Biryukov, Adi Shamir, and David Wagner. Real time cryptanalysis of A5/1 on a PC. In Bruce Schneier, editor, *Fast Software Encryption – FSE 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 1–18. Springer, Heidelberg, April 2001.

[BTCS+15] Ray Beaulieu, Stefan Treatman-Clark, Douglas Shors, Bryan Weeks, Jason Smith, and Louis Wingers. The SIMON and SPECK lightweight block ciphers. In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*, pages 1–6. IEEE, 2015.

[Can06] Christophe De Cannière. Trivium: A stream cipher construction inspired by block cipher design principles. In Sokratis K. Katsikas, Javier Lopez, Michael Backes, Stefanos Gritzalis, and Bart Preneel, editors, *ISC 2006: 9th International Conference on Information Security*, volume 4176 of *Lecture Notes in Computer Science*, pages 171–186. Springer, Heidelberg, August / September 2006.

[CDK09] Christophe De Cannière, Orr Dunkelman, and Miroslav Knežević. KATAN and KTANTAN - a family of small and efficient hardware-oriented block ciphers. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems – CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, Heidelberg, September 2009.

[CNO08] Nicolas T. Courtois, Karsten Nohl, and Sean O'Neil. Algebraic attacks on the crypto-1 stream cipher in mifare classic and oyster cards. Cryptology ePrint Archive, Report 2008/166, 2008. http://eprint.iacr.org/2008/166.

[Cop94] Don Coppersmith. The Data Encryption Standard (DES) and its strength against attacks. *IBM journal of research and development*, 38(3):243–250, 1994.

[CY04] Scott Contini and Yiqun Lisa Yin. Fast software-based attacks on SecurID. In Roy and Meier [RM04], pages 454–471.

[CYK+12] Jiali Choy, Huihui Yap, Khoongming Khoo, Jian Guo, Thomas Peyrin, Axel Poschmann, and Chik How Tan. SPN-hash: Improving the provable resistance against differential collision attacks. In Mitrokotsa and Vaudenay [MV12], pages 270–286.

[DBG+15] Dumitru-Daniel Dinu, Alex Biryukov, Johann Großschädl, Dmitry Khovratovich, Yann Le Corre, and Léo Perrin. FELICS – Fair Evaluation of Lightweight Cryptographic Systems. In *NIST Workshop on Lightweight Cryptography 2015*. National Institute of Standards and Technology (NIST), 2015.

[DEMS16] C. Dobraunig, M. Eichlseder, F. Mendel, and M Schläffer. Ascon v1.2. Candidate for the CAESAR Competition. See also http://ascon.iaik.tugraz.at/, 2016.

[DGV94] Joan Daemen, René Govaerts, and Joos Vandewalle. A new approach to block cipher design. In Ross J. Anderson, editor, *Fast Software Encryption – FSE'93*, volume 809 of *Lecture Notes in Computer Science*, pages 18–32. Springer, Heidelberg, December 1994.

[DHW+12] B. Driessen, R. Hund, C. Willems, C. Paar, and T. Holz. Don't trust satellite phones: A security analysis of two satphone standards. In *2012 IEEE Symposium on Security and Privacy*, pages 128–142, May 2012.

[DPU+16] Daniel Dinu, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, Johann Großschädl, and Alex Biryukov. Design strategies for ARX with provable bounds: Sparx and LAX. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology – ASIACRYPT 2016, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 484–513. Springer, Heidelberg, December 2016.

[DPVAR00] Joan Daemen, Michaël Peeters, Gilles Van Assche, and Vincent Rijmen. Nessie proposal: NOEKEON. First Open NESSIE Workshop, see also http://gva.noekeon.org/papers/2000-NESSIE-Noekeon-Spec.pdf, 2000.

[DR98] Joan Daemen and Vincent Rijmen. AES proposal: Rijndael. AES submission. See also http://csrc.nist.gov/archive/aes/rijndael/, 1998.

[DR00] Joan Daemen and Vincent Rijmen. The block cipher BKSQ. In Jean-Jacques Quisquater and Bruce Schneier, editors, *Smart Card Research and Applications: Third International Conference, CARDIS'98, Louvain-la-Neuve, Belgium, September 14-16, 1998. Proceedings*, volume 1820 of *Lecture Notes in Computer Science*, pages 236–245, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[DRL11] M. David, D. C. Ranasinghe, and T. Larsen. A2U2: A stream cipher for printed electronics RFID tags. In *2011 IEEE International Conference on RFID*, pages 176–183, April 2011.

[EJ03] Patrik Ekdahl and Thomas Johansson. A new version of the stream cipher SNOW. In Kaisa Nyberg and Howard M. Heys, editors, *SAC 2002: 9th Annual International Workshop on Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 47–61. Springer, Heidelberg, August 2003.

[EKM+08] Thomas Eisenbarth, Timo Kasper, Amir Moradi, Christof Paar, Mahmoud Salmasizadeh, and Mohammad T. Manzuri Shalmani. On the power of power analysis in the real world: A complete break of the keeloqcode hopping scheme. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 203–220. Springer, Heidelberg, August 2008.

[EM97] Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. *Journal of Cryptology*, 10(3):151–162, 1997.

[ESSS12] Daniel Engels, Markku-Juhani O. Saarinen, Peter Schweitzer, and Eric M. Smith. The Hummingbird-2 lightweight authenticated encryption algorithm. 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 19–31, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[ETS06] ETSI/SAGE. Specification of the 3GPP confidentiality and integrity algorithms UEA2 & UIA2. Document 2: SNOW 3G specification. Technical report, ETSI/Sage, September 2006. Available at http://www.gsma.com/aboutus/wp-content/uploads/2014/12/snow3gspec.doc (Microsoft Word document).

[ETS11] ETSI/Sage. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 4 : Design and Evaluation Report. Technical report, ETSI/Sage, September 2011. Available at http://www.gsma.com/aboutus/wp-content/uploads/2014/12/EEA3_EIA3_Design_Evaluation_v2_0.pdf.

[Fed15] Federal Agency on Technical Regulation and Metrology (GOST). (GOST R 34.12–2015) information technology – cryptographic data security – block ciphers, 2015. http://tc26.ru/en/standard/gost/GOST_R_34_12_2015_ENG.pdf.

[FL01] Scott R. Fluhrer and Stefan Lucks. Analysis of the E0 encryption system. In Serge Vaudenay and Amr M. Youssef, editors, *SAC 2001: 8th Annual International Workshop on Selected Areas in Cryptography*, volume 2259 of *Lecture Notes in Computer Science*, pages 38–48. Springer, Heidelberg, August 2001.

[FWS+03] Niels Ferguson, Doug Whiting, Bruce Schneier, John Kelsey, Stefan Lucks, and Tadayoshi Kohno. Helix: Fast encryption and authentication in a single cryptographic primitive. In Thomas Johansson, editor, *Fast Software Encryption – FSE 2003*, volume 2887 of *Lecture Notes in Computer Science*, pages 330–346. Springer, Heidelberg, February 2003.

[GdKGV14] Flavio D. Garcia, Gerhard de Koning Gans, and Roel Verdult. Wirelessly lockpicking a smart card reader. *International Journal of Information Security*, 13(5):403–420, 2014.

[GGNS13] Benoît Gérard, Vincent Grosso, María Naya-Plasencia, and François-Xavier Standaert. Block ciphers that are easier to mask: How far can we go? In Bertoni and Coron [BC13], pages 383–399.

[GLS+14] Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, Kerem Varici, François Durvaux Anthony Journault, Lubos Gaspar, and Stéphanie Kerckhof. SCREAM & iSCREAM Side-Channel Resistant Authenticated Encryption with Masking. Candidate for the CAESAR Competition. See also http://perso.uclouvain.be/fstandae/SCREAM/, 2014.

[GLSV15] Vincent Grosso, Gaëtan Leurent, François-Xavier Standaert, and Kerem Varici. LS-designs: Bitslice encryption for efficient masked software implementations. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption – FSE 2014*, volume 8540 of *Lecture Notes in Computer Science*, pages 18–37. Springer, Heidelberg, March 2015.

[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*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.

[Gol97] Jovan Dj. Golic. Cryptanalysis of alleged A5 stream cipher. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 239–255. Springer, Heidelberg, May 1997.

[Gol13] Jovan Dj. Golic. Cryptanalytic attacks on MIFARE classic protocol. In Ed Dawson, editor, *Topics in Cryptology – CT-RSA 2013*, volume 7779 of *Lecture Notes in Computer Science*, pages 239–258. Springer, Heidelberg, February / March 2013.

[GPP11] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON family of lightweight hash functions. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 222–239. Springer, Heidelberg, August 2011.

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED block cipher. In Preneel and Takagi [PT11], pages 326–341.

[GvRVWS10] Flavio D. Garcia, Peter van Rossum, Roel Verdult, and Ronny Wichers Schreur. Dismantling SecureMemory, CryptoMemory and CryptoRF. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 250–259, New York, NY, USA, 2010. ACM.

[HIK+11] Shoichi Hirose, Kota Ideguchi, Hidenori Kuwakado, Toru Owada, Bart Preneel, and Hirotaka Yoshida. A lightweight 256-bit hash function for hardware and low-end devices: Lesamnta-LW. In Kyung Hyune Rhee and DaeHun Nyang, editors, *ICISC 10: 13th International Conference on Information Security and Cryptology*, volume 6829 of *Lecture Notes in Computer Science*, pages 151–168. Springer, Heidelberg, December 2011.

[HJM07] Martin Hell, Thomas Johansson, and Willi Meier. Grain: a stream cipher for constrained environments. *Int. J. Wire. Mob. Comput.*, 2(1):86–93, May 2007.

[HKM17] Matthias Hamann, Matthias Krause, and Willi Meier. LIZARD – A lightweight stream cipher for power-constrained devices. *IACR Transactions on Symmetric Cryptology*, 2017(1):45–79, 2017.

[HLK+14] Deukjo Hong, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu, and Dong-Geon Lee. LEA: A 128-bit block cipher for fast encryption on common processors. In Yongdae Kim, Heejo Lee, and Adrian Perrig,

editors, *WISA 13: 14th International Workshop on Information Security Applications*, volume 8267 of *Lecture Notes in Computer Science*, pages 3–27. Springer, Heidelberg, August 2014.

[HomBC] Homer. *Odyssey*. 800-700BC.

[HSH+06] Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bon-Seok Koo, Changhoon Lee, Donghoon Chang, Jesang Lee, Kitae Jeong, Hyun Kim, Jongsung Kim, and Seongtaek Chee. HIGHT: A new block cipher suitable for low-resource device. In Louis Goubin and Mitsuru Matsui, editors, *Cryptographic Hardware and Embedded Systems – CHES 2006*, volume 4249 of *Lecture Notes in Computer Science*, pages 46–59. Springer, Heidelberg, October 2006.

[IKD+08] Sebastiaan Indesteege, Nathan Keller, Orr Dunkelman, Eli Biham, and Bart Preneel. A practical attack on KeeLoq. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 1–18. Springer, Heidelberg, April 2008.

[ISSK09] Maryam Izadi, Babak Sadeghiyan, Seyed Saeed Sadeghian, and Hossein Arabnezhad Khanooki. MIBS: A new lightweight block cipher. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *CANS 09: 8th International Conference on Cryptology and Network Security*, volume 5888 of *Lecture Notes in Computer Science*, pages 334–348. Springer, Heidelberg, December 2009.

[JK12] Goce Jakimoski and Samant Khajuria. ASC-1: An authenticated encryption stream cipher. In Ali Miri and Serge Vaudenay, editors, *SAC 2011: 18th Annual International Workshop on Selected Areas in Cryptography*, volume 7118 of *Lecture Notes in Computer Science*, pages 356–372. Springer, Heidelberg, August 2012.

[JNP14] Jérémy Jean, Ivica Nikolìc, and Thomas Peyrin. Joltik v1. Candidate for the CAESAR Competition. See also http://www1.spms.ntu.edu.sg/~syllab/m/index.php/Joltik, 2014.

[JSV17] Anthony Journault, François-Xavier Standaert, and Kerem Varici. Improving the security and efficiency of block ciphers based on ls-designs. *Designs, Codes and Cryptography*, 82(1):495–509, 2017.

[KDH13] Ferhat Karakoç, Hüseyin Demirci, and A. Emre Harmancı. ITUbee: A software oriented lightweight block cipher. In Gildas Avoine and Orhun Kara, editors, *Lightweight Cryptography for Security and Privacy: LightSec 2013*, volume 8162 of *Lecture Notes in Computer Science*, pages 16–27, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[Ker83] Auguste Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires*, IX:5–38, January 1883. Available online on the website of Fabien Petitcolas: http://www.petitcolas.net/kerckhoffs/crypto_militaire_1.pdf (the article was written in French).

[KG16] Pierre Karpman and Benjamin Grégoire. The LITTLUN S-box and the FLY block cipher. In *Lightweight Cryptography Workshop 2016, October 17-18 (informal proceedings)*. National Institute of Standards and Technology, 2016.

[KLPR10] Lars R. Knudsen, Gregor Leander, Axel Poschmann, and Matthew J. B. Robshaw. PRINTcipher: A block cipher for IC-printing. In Mangard and Standaert [MS10], pages 16–32.

[KRW99] Lars R. Knudsen, Matthew J. B. Robshaw, and David Wagner. Truncated differentials and Skipjack. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 165–180. Springer, Heidelberg, August 1999.

[KSW04] Chris Karlof, Naveen Sastry, and David Wagner. TinySec: a link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 162–175. ACM, 2004.

[KW01] Lars Knudsen and David Wagner. On the structure of Skipjack. *Discrete Applied Mathematics*, 111(1–2):103 – 116, 2001. Coding and Cryptology.

[KW13] Lars R. Knudsen and Huapeng Wu, editors. *SAC 2012: 19th Annual International Workshop on Selected Areas in Cryptography*, volume 7707 of *Lecture Notes in Computer Science*. Springer, Heidelberg, August 2013.

[Lea14] Gregor Leander. Lightweight block cipher design. Presentation slides available at https://summerschool-croatia.cs.ru.nl/2014/slides/Lightweight%20Block%20Cipher%20Design.pdf, 2014.

[LK06] Chae Hoon Lim and Tymur Korkishko. mCrypton - a lightweight block cipher for security of low-cost RFID tags and sensors. In Jooseok Song, Taekyoung Kwon, and Moti Yung, editors, *WISA 05: 6th International Workshop on Information Security Applications*, volume 3786 of *Lecture Notes in Computer Science*, pages 243–258. Springer, Heidelberg, August 2006.

[LLLS14] Ruilin Li, Heng Li, Chao Li, and Bing Sun. A low data complexity attack on the GMR-2 cipher used in the satellite phones. In Moriai [Mor14], pages 485–501.

[LM91] Xuejia Lai and James L. Massey. A proposal for a new block encryption standard. In Ivan Damgård, editor, *Advances in Cryptology – EUROCRYPT'90*, volume 473 of *Lecture Notes in Computer Science*, pages 389–404. Springer, Heidelberg, May 1991.

[LMV05] Yi Lu, Willi Meier, and Serge Vaudenay. The conditional correlation attack: A practical attack on bluetooth encryption. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 97–117. Springer, Heidelberg, August 2005.

[LPPS07] Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm. New lightweight DES variants. In Biryukov [Bir07], pages 196–210.

[LPTY16] Atul Luykx, Bart Preneel, Elmar Tischhauser, and Kan Yasuda. A MAC mode for lightweight block ciphers. In Thomas Peyrin, editor, *Fast Software Encryption – FSE 2016*, volume 9783 of *Lecture Notes in Computer Science*, pages 43–59. Springer, Heidelberg, March 2016.

[LRW02] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 31–46. Springer, Heidelberg, August 2002.

[LST+09] Stefan Lucks, Andreas Schuler, Erik Tews, Ralf-Philipp Weinmann, and Matthias Wenzel. Attacks on the DECT authentication mechanisms. In Marc Fischlin, editor, *Topics in Cryptology – CT-RSA 2009*, volume 5473 of *Lecture Notes in Computer Science*, pages 48–65. Springer, Heidelberg, April 2009.

[LT11] Javier Lopez and Gene Tsudik, editors. *ACNS 11: 9th International Conference on Applied Cryptography and Network Security*, volume 6715 of *Lecture Notes in Computer Science*. Springer, Heidelberg, June 2011.

[LV04] Yi Lu and Serge Vaudenay. Faster correlation attack on Bluetooth keystream generator E0. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 407–425. Springer, Heidelberg, August 2004.

[MAM17] Vasily Mikhalev, Frederik Armknecht, and Christian Müller. On ciphers that continuously access the non-volatile key. *IACR Transactions on Symmetric Cryptology*, 2016(2):52–79, 2017.

[Mat97] Mitsuru Matsui. New block encryption algorithm MISTY. In Eli Biham, editor, *Fast Software Encryption – FSE'97*, volume 1267 of *Lecture Notes in Computer Science*, pages 54–68. Springer, Heidelberg, January 1997.

[MBTM16] Kerry A. McKay, Larry Bassham, Meltem Sönmez Turan, and Nicky Mouha. NISTIR 8114 – DRAFT report on lightweight cryptography. Available on the NIST website: http://csrc.nist.gov/publications/drafts/nistir-8114/nistir_8114_draft.pdf, 2016.

[MMH+14] Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In Antoine Joux and Amr M. Youssef, editors, *SAC 2014: 21st Annual International Workshop on Selected Areas in Cryptography*, volume 8781 of *Lecture Notes in Computer Science*, pages 306–323. Springer, Heidelberg, August 2014.

[Mor14] Shiho Moriai, editor. *Fast Software Encryption – FSE 2013*, volume 8424 of *Lecture Notes in Computer Science*. Springer, Heidelberg, March 2014.

[MS10] Stefan Mangard and François-Xavier Standaert, editors. *Cryptographic Hardware and Embedded Systems – CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*. Springer, Heidelberg, August 2010.

[MV12] Aikaterini Mitrokotsa and Serge Vaudenay, editors. *AFRICACRYPT 12: 5th International Conference on Cryptology in Africa*, volume 7374 of *Lecture Notes in Computer Science*. Springer, Heidelberg, July 2012.

[NESP08] Karsten Nohl, David Evans, Starbug Starbug, and Henryk Plötz. Reverse-engineering a cryptographic RFID tag. In *USENIX security symposium*, volume 28, 2008.

[Nob94] Nobody. Thank you Bob Anderson. Mail to the cypherpunk mailing list from nobody@jpunix.com, available at https://web.archive.org/web/20010722163902/http://cypherpunks.venona.com/date/1994/09/msg00304.html, September 1994.

[NTW10] Karsten Nohl, Erik Tews, and Ralf-Philipp Weinmann. Cryptanalysis of the DECT standard cipher. In Seokhie Hong and Tetsu Iwata, editors, *Fast Software Encryption – FSE 2010*, volume 6147 of *Lecture Notes in Computer Science*, pages 1–18. Springer, Heidelberg, February 2010.

[NW97] R. M. Needham and D. J. Wheeler. Tea extensions. Technical report, Cambridge University, Cambridge, UK, October 1997.

[OC16] Colin O'Flynn and Zhizhang Chen. Power analysis attacks against IEEE 802.15.4 nodes. In *Constructive Side-Channel Analysis and Secure Design – COSADE 2016*, volume 9689 of *Lecture Notes in Computer Science*, Cham, 2016. Springer International Publishing.

[Per17] Léo Perrin. More reverse-engineered S-boxes. Presentation at the rump session of ESC'2017. Slides available at https://www.cryptolux.org/mediawiki-esc2017/images/2/2e/Rump.pdf, 2017.

[PLW10] Axel Poschmann, San Ling, and Huaxiong Wang. 256 bit standardized crypto for 650 GE - GOST revisited. In Mangard and Standaert [MS10], pages 219–233.

[PMA07] Lea Troels Møller Pedersen, Carsten Valdemar Munk, and Lisbet Møller Andersen. Cryptography – the rise and fall of DVD encryption. Available online at http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=3672D97255B2446765DA47DA97960CDF?doi=10.1.1.118.6103&rep=rep1&type=pdf, 2007.

[PRC12] Gilles Piret, Thomas Roche, and Claude Carlet. PICARO - a block cipher allowing efficient higher-order side-channel resistance. In Feng Bao, Pierangela Samarati, and Jianying Zhou, editors, *ACNS 12: 10th International Conference on Applied Cryptography and Network Security*, volume 7341 of *Lecture Notes in Computer Science*, pages 311–328. Springer, Heidelberg, June 2012.

[PT11] Bart Preneel and Tsuyoshi Takagi, editors. *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*. Springer, Heidelberg, September / October 2011.

[Riv95] Ronald L. Rivest. The RC5 encryption algorithm. In Bart Preneel, editor, *Fast Software Encryption – FSE'94*, volume 1008 of *Lecture Notes in Computer Science*, pages 86–96. Springer, Heidelberg, December 1995.

[RM04] Bimal K. Roy and Willi Meier, editors. *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*. Springer, Heidelberg, February 2004.

[Rob16] Matt Robshaw. Lightweight cryptography and RAIN RFID. Invited presentation at the NIST Workshop on Lightweight Cryptography 2016. Slides available at https://www.nist.gov/sites/default/files/documents/2016/10/17/robshaw-presentation-lwc2016.pdf, 2016.

[ROSW16] Eyal Ronen, Colin O'Flynn, Adi Shamir, and Achi-Or Weingarten. IoT goes nuclear: Creating a Zigbee chain reaction. Cryptology ePrint Archive, Report 2016/1047, 2016. http://eprint.iacr.org/2016/1047.

[SIH+11] Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An ultra-lightweight blockcipher. In Preneel and Takagi [PT11], pages 342–357.

[SMMK13] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. TWINE : A lightweight block cipher for multiple platforms. In Knudsen and Wu [KW13], pages 339–354.

[SPGQ06] François-Xavier Standaert, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. Sea: A scalable encryption algorithm for small embedded applications. In Josep Domingo-Ferrer, Joachim Posegga, and Daniel Schreckling, editors, *Smart Card Research and Advanced Applications: CARDIS 2006*, volume 3928 of *Lecture Notes in Computer Science*, pages 222–236, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[SPR+04] François-Xavier Standaert, Gilles Piret, Gaël Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. ICEBERG: An involutional cipher efficient for block encryption in reconfigurable hardware. In Roy and Meier [RM04], pages 279–299.

[SS16] Peter Schwabe and Ko Stoffelen. All the AES you need on cortex-M3 and M4. In *Selected Areas in Cryptography – SAC 2016*, volume To appear of *Lecture Notes in Computer Science*, 2016.

[SSA+07] Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-bit blockcipher CLEFIA (extended abstract). In Biryukov [Bir07], pages 181–195.

[UMHA16] Rei Ueno, Sumio Morioka, Naofumi Homma, and Takafumi Aoki. A high throughput/gate AES hardware architecture by compressing encryption and decryption datapaths - toward efficient CBC-mode implementation. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems – CHES 2016*, volume 9813 of *Lecture Notes in Computer Science*, pages 538–558. Springer, Heidelberg, August 2016.

[U.S98] U.S. Department Of Commerce/National Institute of Standards and Technology. Skipjack and KEA algorithms specifications, v2.0, 1998. http://csrc.nist.gov/groups/ST/toolkit/documents/skipjack/skipjack.pdf.

[U.S99] U.S. Department Of Commerce/National Institute of Standards and Technology. Data Encryption Standard. *Federal Information Processing Standards Publication (FIPS)*, 1999. Available on the NIST website: http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.

[VGB12] Roel Verdult, Flavio D. Garcia, and Josep Balasch. Gone in 360 seconds: Hijacking with hitag2. In *Proceedings of the 21st USENIX Conference on Security Symposium*, Security'12, pages 37–37, Berkeley, CA, USA, 2012. USENIX Association.

[VGE13] Roel Verdult, Flavio D Garcia, and Baris Ege. Dismantling Megamos crypto: Wirelessly lockpicking a vehicle immobilizer. In *Supplement to the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 703–718. USENIX Association, August 2013.

[WIK+08] Dai Watanabe, Kota Ideguchi, Jun Kitahar, Kenichiro Muto, Hiroki Furuichi, and Toshinobu Kaneko. Enocoro-80: A hardware oriented stream cipher. In *The Third International Conference on Availability, Reliability and Security — ARES 08*, pages 1294–1300, 2008.

[WSD+99] David Wagner, Leone Simpson, Ed Dawson, John Kelsey, William Millan, and Bruce Schneier. Cryptanalysis of ORYX. In Stafford E. Tavares and Henk Meijer, editors, *SAC 1998: 5th Annual International Workshop on Selected Areas in Cryptography*, volume 1556 of *Lecture Notes in Computer Science*, pages 296–305. Springer, Heidelberg, August 1999.

[WSK97] David Wagner, Bruce Schneier, and John Kelsey. Cryptoanalysis of the cellular encryption algorithm. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 526–537. Springer, Heidelberg, August 1997.

[Wu16] Hongjun Wu. ACORN: A lighweight authenticated cipher (v3). Candidate for the CAESAR Competition. See also https://competitions.cr.yp.to/round3/acornv3.pdf, 2016.

[WW05] Ralf-Philipp Weinmann and Kai Wirt. Analysis of the DVB Common Scrambling Algorithm. In David Chadwick and Bart Preneel, editors, *Communications and Multimedia Security: 8th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, Sept. 15–18, 2004, Windermere, The Lake District, United Kingdom*, volume 175 of *IFIP – The International Federation for Information Processing*, Boston, MA, 2005. Springer US.

[WZ11] Wenling Wu and Lei Zhang. LBlock: A lightweight block cipher. In Lopez and Tsudik [LT11], pages 327–344.

[YKPH11] Huihui Yap, Khoongming Khoo, Axel Poschmann, and Matt Henricksen. EPCBC - a block cipher suitable for electronic product code encryption. In Dongdai Lin, Gene Tsudik, and Xiaoyun Wang, editors, *CANS 11: 10th International Conference on Cryptology and Network Security*, volume 7092 of *Lecture Notes in Computer Science*, pages 76–97. Springer, Heidelberg, December 2011.

[YZS+15] Gangqiang Yang, Bo Zhu, Valentin Suder, Mark D. Aagaard, and Guang Gong. The simeck family of lightweight block ciphers. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems – CHES 2015*, volume 9293 of *Lecture Notes in Computer Science*, pages 307–329. Springer, Heidelberg, September 2015.

[ZBL+15] Wentao Zhang, Zhenzhen Bao, Dongdai Lin, Vincent Rijmen, Bohan Yang, and Ingrid Verbauwhede. RECTANGLE: A bit-slice lightweight block cipher suitable for multiple platforms. *Science China Information Sciences*, 58(12):1–15, 2015.

[ZSX+14] Bin Zhang, Zhenqing Shi, Chao Xu, Yuan Yao, and Zhenqi Li. Sablier v1. Candidate for the CAESAR Competition. See also https://competitions.cr.yp.to/round1/sablierv1.pdf, 2014.

[ZWW+14] Lei Zhang, Wenling Wu, Yanfeng Wang, Shengbao Wu, and Jian Zhang. LAC: A lightweight authenticated encryption cipher. Candidate for the CAESAR Competition, 2014.

A List of Ligthweight Ciphers from the Industry

A.1 Industry-Designed Stream Ciphers

A5/1. The exact design date of this algorithm is unclear but a first approximation of its inner workings was published in 1994 [And94]. It generates a keystream from a 22-bit

IV along with a 64-bit key using three different LFSRs whose lengths add up to 64 bits. Practical attacks have been implemented using time-memory trade-offs exploiting the fact that the update function of the internal state is not bijective [Gol97, BSW01]. The most time efficient of those needs only 2^{24} simple steps provided that a significant (but practical) pre-computation was performed. Furthermore, 10 bits of the key were always set to 0 in many implementations. The 2G GSM protocol still uses this algorithm.

A5/2. A cipher somewhat similar similar to A5/1 but even weaker was intended to be used in countries targeted by American export restrictions. It is called A5/2. It is vulnerable to ciphertexts only attacks with complexity 2^{16} using redundancy introduced by error correcting codes. It requires a one-time pre-computation of practical complexity. Unfortunately, interoperability imposed the implementation of this algorithm on devices supposed to run A5/1 instead, thus making downgrade attacks possible [BBK03, BBK08].

A5-GMR-1 and A5-GMR-2. Satellite phones have their own protocols and, therefore, use their own cryptographic algorithms. The two algorithms used, A5-GMR-1 and A5-GMR-2, were reverse-engineered by Driessen *et al.* in [DHW+12]. Those are very different from one another but both are easily attacked.

- A5-GMR-1 is a variant of A5/2 with an internal state consisting in 4 LFSRs with a total size of 82 bits. Those are clocked irregularly, much like in A5/2. It can be attacked using only known ciphertexts by inverting 2^{21} triangular matrices of size 532×532, which requires roughly $2^{21} \times 532^2/2 \approx 2^{38.1}$ simple operations. A significant but practical pre-computation step is necessary.
- A5-GMR-2 is a byte oriented stream cipher with a much more sophisticated structure based on 3 different components denoted $\mathcal{F}, \mathcal{G}$ and $\mathcal{H}$ by Driessen *et al.*. Surprisingly, $\mathcal{H}$ uses the S-Box S_2 and S_6 of the DES. A practical attack with very low data and time complexity is presented in [LLLS14]. It requires guessing at most 32 bits using only 1 frame of 15 bytes for an average complexity of 2^{28}.

Atmel Ciphers. The stream ciphers used by the SecureMemory, CryptoMemory and CryptoRF families of products from Atmel are similar to one another. They are proprietary algorithms which were reverse-engineered and attacked by Garcia *et al.* in [GvRVWS10]. Other more powerful attacks were later proposed by Biryukov *et al.* [BKZ11] breaking the cipher of SecureMemory in time $2^{29.8}$ using 1 frame and the cipher of CryptoMemory in time 2^{50} using 30 frames and about 530 Mb of memory. The ciphers rely on 3 NLFSRs with a total size of a bit more than 100 bits. The attacks found by both Garcia *et al.* and Biryukov *et al.* were successfully implemented.

Crypto-1. It is a stream cipher used by the *Mifare classic* line of smartcards of NXP. It was reverse-engineered by Nohl *et al.* in [NESP08] and was subsequently attacked by many teams [CNO08, Gol13] with a time complexity as low as 2^{32}. It has been used at least since 1998 but the exact date of its design is unclear. It is based on a 48-bit LFSR combined with several non-linear Boolean functions.

Content Scrambling System (Css). In order to implement Digital Rights Managements (DRMs), the content of DVD discs is encrypted. This encryption used to be performed with a stream cipher called CSS. It uses two 17- and 25-bit long LFSRs to generate two 8-bit words in parallel. These are afterwards added modulo 2^8 to obtain a byte of keystream. However, unlike in most stream ciphers, this key stream is not simply XORed with the plaintext. Instead, the plaintext first goes through an 8-bit bijective S-Box whose result is added to the keystream to obtain the ciphertext. This operation is sometimes called the

mangling step. A full description is available in [BD04] and in [PMA07]. Several powerful attacks target the protocol using this stream cipher. However, given its key length of 40 bits, the cipher alone is vulnerable to a brute-force search of time complexity 2^{40}.

Common Scrambling Algorithm (Csa-SC). The Common Scrambling Algorithm is used to secure digital television broadcast. It cascades two ciphers, as described in [WW05]. The first is a block cipher which we call Csa-BC and which is described below. The second is a stream cipher which we call Csa-SC. The stream cipher is based on two FSRs consisting of twenty 4-bit cells each and a combiner with memory. The feedback function of the registers involves, among other things, several 5×2 S-Boxes. The combiner uses addition modulo 2^4 to extract 2 bits of keystream from its internal state and the two shift registers at each clock cycle. In [WW05], several undesirable properties are presented. For example, the keystream often has very short cycles. It is also possible to recover the secret key by solving about 2^{28} systems of 60 linear equations with 40 unknowns which must take at most $2^28 \times 60^3 \approx 2^{45.7}$.

Dsc. The Dect[11] Standard Cipher, usually abbreviated into Dsc, is a stream cipher used to encrypt the communications of cordless phones. First, attacks targeting the protocol using it and its flawed implementation were presented in [LST+09]. It was subsequently reverse-engineered and its attackers found practical attacks requiring only about 2^{15} samples of keystream and 2^{34} trial encryptions which take a couple of hours on a standard computer to recover the key [NTW10]. It is described by the authors of this paper as being "an asynchronous stream cipher with low gate complexity that takes a 64-bit secret key and a 35-bit initialization vector." Its structure, based on irregularly clocked LFSRs, is reminiscent of that of A5/1.

E0. The privacy of the Bluetooth protocol is now based on the Aes but it used to rely on a custom stream cipher called E0. Its 128-bit internal state is divided into 4 LFSRs and its filter function has its own 2-bit memory. A description of E0 can be found in the papers presenting attacks against it such as [FL01, LV04, LMV05]. Lu *et al.* found an attack which recovers the secret key using the first 24 bits of $2^{23.8}$ frames and with 2^{38} computations.

Hitag2 ; Megamos. These stream ciphers are used in the *car immobilizers* implemented by different car manufacturers. These devices prevent a car engine from starting unless a specific transponder is close to them. While initially kept secret, the first was published by Wiener[12] and the second was reverse-engineered by Verdult *et al.* [VGE13]. They are both stream ciphers with a small internal state of 48 and 57 bits respectively. These small sizes and other weaknesses in the ciphers themselves and in the protocols using them lead to practical attacks against the devices relying on these algorithms for security. For example, it is possible to attack a car key using Hitag2 using 1 min of communication between the key and the car and about 2^{35} encryptions. The secret key of Megamos can be recovered in time 2^{48} but more powerful attacks are possible when we take into account the key management method of the devices using it.

iClass. Formally, iClass is family of smartcards introduced in 2002. The stream cipher it uses was reverse-engineered and attacked by Garcia *et al.* in [GdKGV14]. It has a 40-bit internal state. The cryptanalysts who reverse-engineered it presented attacks against

[11] Dect stands for "Digital Enhanced Cordless Telecommunications".

[12] While this first publication is mentioned for example by Verdult *et al.* in [VGB12], we were not able to find a copy of it. Nevertheless, the specification of Hitag2 can be found in [VGB12].

this cipher in the same paper. By recording 2^{22} authentication attempts, the key can be recovered using 2^{40} trial encryptions.

Kindle Cipher (PC1). This stream cipher was first published on Usenet by Alexander Pukall in 1997, meaning that this algorithm was not technically designed in the industry. However, it was not designed by academics and Amazon used it at least up until 2012 for the DRM scheme "protecting" its e-book using the MOBI file format. It uses a 128-bit key and a separate 24-bit internal state updated using different operations, including modular multiplications. The keystream is generated byte by byte. It has been broken by Biryukov *et al.* [BLR13] using e.g. 2^{20} known plaintexts and a time of 2^{31}. Even practical known-ciphertexts attacks are possible in some contexts.

Oryx. While A5/1 "secures" GSM communications in Europe, the stream cipher ORYX was chosen by the Telecom Industry Association Standard (TIA) to secure phone communications in north America. A description of the algorithm can be found in [WSD+99] where practical attacks are presented. It uses a 96-bit key, a 96-bit internal state consisting of three 32-bit LFSRs, and an 8-bit S-Box which changes every time. It is possible to attack it in time 2^{16} using 25 bytes of known plaintext.

RC4. First designed by Ron Rivest in 1987, this stream cipher was intended to remain a trade secret of the RSA company. However, it was leaked to the cypherpunk mailing list in 1994 [Nob94] and turned out to be a remarkably simple algorithm. Unfortunately it has several issues, in particular when its first outputs are not discarded. The attack from [ABP+13] was successfully implemented: using between 2^{28} and 2^{32} encryptions of the same message, it is possible to recover it using biases in the keystream.

The now deprecated WEP protocol for wireless communications used it for encryption. It lead to practical attacks implemented e.g. by the `aircrack`[13] tool allowing an attacker to recover the password protecting WiFi access. It uses a 256-byte internal state containing all numbers in $\{0, ..., 255\}$ which is updated using a very simple rule each time an 8-bit output is generated. It supports all key sizes between 40 and 2048 bits, although it usually uses 128-bit keys.

A.2 Industry-Designed Block Ciphers

Cmea This block cipher was used by the TIA to secure the transmission of phone numbers across telephone lines. A good description of this algorithm is provided in [WSK97] which, incidentally, describes an attack against the full cipher. It encrypts a block of an arbitrary number of bytes — although in practice those were usually 2 to 6 bytes long — using a 64-bit key. It is vulnerable to a known plaintext attacks requiring only 40–80 blocks of data and taking a time between 2^{24} and 2^{32} encryptions. Its 8-bit S-Box seems to contain a hidden structure [Per17].

Cryptomeria. It is a block cipher nicknamed "C2" in the literature. It shares the same structure as the DES: it encrypts 64-bit blocks using a 56-bit key and uses a 32-bit Feistel function. It works by mixing in a 32-bit subkey with a modular addition, then use one 8-bit S-Box call followed by a 32-bit linear permutation. The S-Box is secret, so an S-Box recovery attack has been proposed [BKLM09]. The same paper presents a key recovery with time complexity 2^{48}. This algorithm was intended from the start to be used by "things", namely DVD players (in which case it can be seen as a successor of CSS) and some SD cards. In total, 10 rounds are used; which means that only 10 S-Box calls are needed

[13] Its successor, `aircrack-ng`, is described on its official website http://www.aircrack-ng.org/.

to encrypt one 64-bit block compared to, say, the 160 calls needed to encrypt one 128-bit block using AES-128.

Common Scrambling Algorithm (Csa-BC). The Common Scrambling Algorithm uses a stream cipher (described above) and a block cipher which we call CSA-BC. It encrypts a 64-bit block using a 64-bit key. Its structure is reminiscent of a generalised Feistel network using eight 8-bit branches. The Feistel functions are based on a unique random-looking 8-bit S-Box B and a variant defined as $\sigma \circ B$, where σ is a simple bit permutation. An encryption consists in 56 rounds. A full specification is given in [WW05]. To the best of our knowledge, there is no attack other than brute-force against this cipher.

Dst40. This algorithm was reverse-engineered from partial information disclosed in a patent and from a physical device implementing this block cipher [BGS+05]. It was used by RFID transponders sold by Texas Instrument. They were used in car immobilizers and for electronic payment. The cipher itself encrypts a 40-bit block with a 40-bit key using 200 rounds of an unbalanced Feistel network. The Feistel function maps 38 bits of internal state and a 40-bit subkey to a 2-bit output by nesting several Boolean functions. Due to its key size of 40 bits, a brute-force search is practical.

Keeloq. It is a so-called "code-hopping encoder". A US patent was filed in 1993 and eventually granted in 1996 [BSK96] but it was designed earlier, around 1985 [Lea14]. Using modern terminology, it is a 32-bit block cipher which uses a 64-bit key. It was first kept secret but its specification was leaked in 2006. Using this information, several teams presented practical attacks against devices using this algorithm [IKD+08]. For example, the key be recovered using 2^{16} known plaintexts and $2^{44.5}$ encryptions. Far more powerful side-channel attacks have also been proposed against commercial implementations of the cipher [EKM+08]. This ciphers was still in use when these attacks were found, 20 years after its design.

A.3 Industry-Designed Macs

SecurID mac. A SecurID is small hardware token used for authentication and designed by SDTI (which was later bought by RSA Security). It displays a 6 digit number which changes every minute. It is based on a 64-bit MAC described for example in [BLP04]. This paper also presents attacks against the algorithm. The details of the MAC were initially kept secret but were eventually leaked which lead to the attacks of Biryukov *et al.*. These were later sped up by Contini *et al.* [CY04] to obtain a time complexity of about 2^{44} MAC computations.