Lightweight Cryptography

Sean M Kennedy[1,*], Darius E Cepulis[1], Abhimanyu Chadha[1], Frazier N Baker[1,2]

1 Department of Electrical Engineering and Computer Science, University of Cincinnati
812 Rhodes Hall, 2851 Woodside Drive, Cincinnati, OH 45219

2 Center for Autoimmune Genomics and Etiology, Cincinnati Children's Hospital Medical Center
3333 Burnet Avenue, Cincinnati, Ohio, 45220

* To whom correspondence should be addressed

Abstract. Lightweight cryptographic algorithms are used in cryptosystems that require comparatively less storage, memory, and computing power than traditional cryptosystems. Lightweight cryptography may implement modifications of existing cryptographic algorithms to meet these requirements or might include the design of new algorithms. The need for lightweight cryptography is driven by the increasing prevalence of resource-constrained devices (e.g. devices which could have limited memory, storage, CPU speed or require low energy, etc.) such as RFIDs and Internet of Things (IoT) devices. In this paper we examine the issues in lightweight cryptography, some of the methods used in order to implement lightweight cryptosystems and their applications in resource-constrained devices.

Keywords: lightweight cryptography, lightweight symmetric block ciphers, Bit-slice, ARX algorithms, IoT security, RFID security, resource-constrained devices

Introduction 1

Methods 2

Symmetric Key Encryption: A Brief Summary of Key Concepts 2
The Advanced Encryption Standard 4
LUT 5
Bit-slice 5
ARX Based Algorithms 6
Salsa20 Stream Cipher, and its variant, ChaCha 7
XXTEA 8
The NSA – SPECK and SIMON 9
Other Considerations in Lightweight Cryptography 10
Message Authentication Codes 10

Applications 10

Conclusion: Future Directions 11

Bibliography 15

Introduction

The march of Moore's law in the past decades has led to a modern technological landscape filled with pervasive small computers. Light bulbs and IP cameras and wrist watches all demonstrate some computational power and storage. With these characteristics comes inherent security risks. While not unusual, the rise in computing power and its ability to brute-force older cryptographic algorithms has required that Estonia, a nation that distributes digitally enabled ID cards, publicly declare their awareness of a vulnerability in its IDs and to push an update earlier this year [1]. RFID devices, in general, often contain sensitive information, and yet must communicate quickly and securely over a wireless connection. Internet of Things (IoT) devices particularly are a case where security is of the utmost importance: these devices input from the world—they'd be a prime target for acquiring information, output to the world—a malicious actor could affect the world around these devices, and are highly connected—multiple ingress and egress points must be protected to prevent the introduction and spread of malevolent software. This threat is impending, too: botnets running on our devices are here [2], and researchers have successfully simulated a worm infecting a Philips light bulb with ZigBee communication and spreading catastrophically through a densely populated urban area, granting control to thousands of people's' homes [3].

In light of these threats, modern ambient computing devices need a lightweight solution to cryptography that is compatible with its resource constraints. As Sklavos & Zaharakis discuss in a survey [4], popular cryptographic algorithms are based on significant processing power, good memory resources, and power availability. The need for lighter cryptography is acknowledged, too, by, among others, US Congress requiring a standard of IoT security [5], and NIST addressing the standardization of lightweight methods. While reducing heavier algorithms to resource-constrained IoT is an area of research, better results may be achieved through dedicated algorithms. These resource-constrained algorithms are

collectively called “lightweight cryptography”, and are considered compromises in security, power consumption, or latency, among other factors [6].

Methods

Symmetric Key Encryption: A Brief Summary of Key Concepts

For the purposes of explaining some concepts in lightweight cryptography it is necessary to understand some key concepts in general cryptography. *Symmetric key* cryptosystems use the same key to decrypt and encrypt data. The requirement is for that key to be kept private for the assurance of the encrypted data’s security (hence, the key is called the *private key*). A *block cipher* encrypts plaintext one block at a time (for example, AES encrypts in block sizes of 128 bits), as opposed to encrypting individual bits. In this survey, we will focus on *iterated* symmetric block ciphers which normally come in two flavors, *Feistel Networks* and *Substitution-Permutation Networks*.

A common design strategy in modern block ciphers is to use the *iterated* cipher design. *Iterated ciphers* need a *round function* and a *key schedule* to encrypt plaintext over a number of *rounds*. *Rounds* in the context of iterated ciphers is the number of times a cryptographic algorithm (the *round function*) is to be performed before outputting a resulting ciphertext (for example, the Salsa**20** cipher mentioned later is composed of **20** rounds). The *key schedule* is constructed from an initial key K (by using a fixed key scheduling algorithm) and is composed of *subkeys (a.k.a. round keys)* (K^1, , K^{total_rounds}) [7]. The aim of applying an algorithm over multiple rounds is to produce a ciphertext that has sufficient confusion and diffusion properties (properties that ensure the ciphertext is resilient to *cryptanalysis* – techniques used to analyze and break cryptosystems) [8].

A *Feistel network* is a class of iterated block ciphers that include the now obsolete Data Encryption Standard (DES), and a multitude of ciphers that utilize ARX-based algorithms (which

are described in a later section). A Feistel network consists of states, where each state u^i is separated into halves of the same length L^i and R^i. The *round function* g is in the form: $g(L^{i-1}, R^{i-1}, K^i) = (L^i, R^i)$, where $L^i = R^{i-1}$ and $R^i = L^{i-1} \oplus f(R^{i-1}, K^i)$ [7]. One advantage of using a Feistel cipher is the encryption and decryption operations are similar. This is especially useful for lightweight purposes as this quality generally reduces the number of operations, and improves throughput. Another advantage of using a Feistel cipher is that the *round function* does not have to be injective, because given the *round key* the round function is always invertible [7].

A *substitution-permutation network* (or SPN) is a type of iterated cipher built from two components, a type of permutation called a substitution box (a.k.a. S-box, see the next section), (π_s) and permutation used to permute a length of bits, (π_p). Let *l* and *m* be two positive vectors, where *lm* is the block length of the cipher, then π_s: {0, 1}^{l} {0, 1}^{l} and π_p = {1, , *lm*} {1, , *lm* }. In each round of a SPN, (except for the last one) *m* substitutions are performed using the S-box, and one permutation will be performed using π_p. Before each substitution is performed the round key bits are incorporated using an XOR [7].

Substitution boxes are sets of rules indicating replacement bit strings for use in multiple subsequent steps of replacement. Substitution boxes are utilized as part of symmetric encryption, so there are some constraints on their implementation. They must be reversible and they should aim to be nonlinear and as complex as possible to decrease vulnerability to differential and linear cryptanalysis and algebraic attacks. S-boxes are theoretically represented using matrix algebra, but can be implemented using a number of data structures and algorithms [9].

The Advanced Encryption Standard

In 1997, the Advanced Encryption Standard (AES), a proposed new NIST approved block cipher, was sought as a replacement for the aging Data Encryption Standard (DES). After

several rounds of competition between 21 different candidate algorithms designed by groups from 15 different countries, the Rijndael cipher was chosen to be the AES algorithm. A combination of security, efficiency, flexibility and performance is what cinched the vote for Rijndael. AES has a block length of 128 bits, and key lengths of 128 bits (with 10 rounds), 192 bits (with 12 rounds) or 256 bits (with 14 rounds). AES is a type of SPN that encrypts plaintext using the following steps:

1) Initialize *State* equal to the plaintext *X*. Perform operation *AddRoundKey* which is the *RoundKey* *State*
2) In every round, except for the last, perform operation *SubBytes* which is a substitution on *State* using an *S-Box.* Perform operation *ShiftRows* on *State* which is a type of permutation, perform operation *MixColumns* which is a type of linear transformation, perform *AddRoundKey*
3) Perform operations *SubBytes*, *ShiftRow*s, and *AddRoundKey*
4) Set Y (the ciphertext) to the resulting *State*.

AES is a very resilient cryptosystem, and in theory, no known attacks are faster than the naïve exhaustive search. Unfortunately for lightweight cryptographic purposes, modifying AES to include less than 10 rounds (which is what AES-128 uses) makes it more vulnerable to attack than using the NIST approved versions of AES [7].

LUT

Look Up Tables (LUT) can be used as an implementation of S-Boxes for symmetric encryption, provided that all possible outputs can be stored on the machine. The benefit of storing these values in such a way comes with the speed of the lookup operation. Substitutions can be computed in one lookup, which is a very fast [10]. Many lightweight cryptographic systems, such as PRESENT, SKINNY, and Piccolo, implement look-up tables at a hardware level [10], [11]. PRESENT's S-Box is shown in hexadecimal associated arrays in Figure 1 [11].

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

Figure 1. S-box for PRESENT from Bogdanov et al. [11].

Bit-slice

Substitution boxes can also be implemented using bit-slices. Instead of building a lookup table and having single-operation access to substitutions, substitution boxes can be computed using a number of bitwise operations. This number can vary with the number of bits for the substitution box. Bit-slice substitution boxes have been popular in recent years because they are simple from both a software and a hardware perspective [10]. Scream is a lightweight cryptographic method that can use bit-slicing for substitution boxes [12]. Figure 2 shows the implementation for Scream's bit-slice S-box implementation [12].

Require: 8 16-bit words $(W_0, \ldots, W_7)$

First 5-bit S-box

$W_2 = W_2 \oplus (W_0 \wedge W_1)$
$W_1 = W_1 \oplus W_2$
$W_3 = W_3 \oplus (W_0 \wedge W_4)$
$W_2 = W_2 \oplus W_3$
$W_0 = W_0 \oplus (W_3 \wedge W_1)$
$W_4 = W_4 \oplus W_1$
$W_1 = W_1 \oplus (W_2 \wedge W_4)$
$W_1 = W_1 \oplus W_0$

Extend-Xor

$W_0 = W_0 \oplus W_5$
$W_1 = W_1 \oplus W_6$
$W_2 = W_2 \oplus W_7$

Constant

$W_3 = \neg(W_3)$
$W_4 = \neg(W_4)$

First 3-bit S-box

$(t_5, t_6, t_7) = (W_5, W_6, W_7)$
$W_5 = W_5 \oplus \neg(t_6) \wedge t_7$
$W_6 = W_6 \oplus \neg(t_7) \wedge t_5$
$W_7 = W_7 \oplus \neg(t_5) \wedge t_6$

Truncate-Xor

$W_5 = W_0 \oplus W_5$
$W_6 = W_1 \oplus W_6$
$W_7 = W_2 \oplus W_7$

Second 5-bit S-box

$W_2 = W_2 \oplus (W_0 \wedge W_1)$
$W_1 = W_1 \oplus W_2$
$W_3 = W_3 \oplus (W_0 \wedge W_4)$
$W_2 = W_2 \oplus W_3$
$W_0 = W_0 \oplus (W_3 \wedge W_1)$
$W_4 = W_4 \oplus W_1$
$W_1 = W_1 \oplus (W_2 \wedge W_4)$
$W_1 = W_1 \oplus W_0$

Figure 2. Bit slice s-box implementation in Scream[12] as denoted by Grosso et al [12].

ARX Based Algorithms

ARX based encryption algorithms, named for the type of operations performed in their round function (modular **A**ddition, bitwise **R**otation and e**X**clusive-or), are used in many

lightweight cryptosystems because of their resilience to timing based attacks (i.e., a side channel attack based on the analysis of the running time of a cryptographic algorithm) and their implementation is compact and efficient in software and hardware. ARX based algorithms have the advantage that the ARX set of operations have been proven to be functionally complete. Consequently, they can be used to obtain the same level of security of any other series of operations [13]. One of the most widely used symmetric key block ciphers is the AES family of block ciphers. ARX algorithms have several advantages over AES (or other S-box based cryptosystems) for use in lightweight cryptography.

Compared to S-box designs, ARX algorithms use fewer operations making their implementations faster and more suitable for resource-constrained devices [14]. The absence of lookup tables in ARX algorithms, which are utilized in S-box designs, reduces the risk of a certain type of side channel attack. This particular side channel attack involves the calculation of the time it takes for the S-box to handle many inputs, and by taking the maximum of these values the adversary can determine the key k[i] of a given input n[i], a detailed description of this attack can be found in Bernstein's cache-timing cryptanalysis of AES [15]. Another benefit to using ARX algorithms is the length of code is comparatively less than other symmetric key algorithms (i.e., less storage space required on the device) [14].

Lightweight ciphers that utilize ARX-based designs include the Salsa20 Stream Cipher and its variant ChaCha designed by Daniel J. Bernstein, a security researcher at the Eindhoven University of Technology [16], the XXTEA block cipher designed by the David Wheeler and Roger Needham of the Cambridge Computer Laboratory [17], and the SIMON and SPECK block ciphers designed by the National Security Agency (NSA) [18].

Salsa20 Stream Cipher, and its variant, ChaCha

The Salsa20 stream cipher family includes variants for 20 12 and 8 rounds. For lightweight cryptography, the 12 and 8 round versions are preferred for their improved speed. To obtain the keystream the Salsa20 core needs a eight words of key, two words of block counter, and two words nonce. The core then creates an array containing the eight words of key, two words of block counter, two words of nonce plus an additional four constant words. It then performs a series of additions mod 32, rotations of varying bit sizes and XORing (the ARX operations) in each round. Each round of Salsa20 includes 16 additions, 16 rotations, and 16 XORs. One advantage of this algorithm is the keystream is composed of 64-byte blocks each of which is independent from the given key, nonce and 64-bit block number. This allows for the output stream to be accessed randomly and for parallel computation of blocks [16]. The ChaCha variant of Salsa20 improves the time per round function by changing the how each operation is applied, updating each word twice, as opposed to only once in Salsa20 [19].

XXTEA

XXTEA is the corrected extended version of the Tiny Encryption Algorithm (TEA), and is the most recent addition in the TEA series of lightweight block ciphers by Roger Needham and David Wheeler. This algorithm, like the other TEA algorithms, is noteworthy for the compact size of its code. For example, a C implementation of XXTEA encryption can be as little as 15 lines [17]. The block size is a variable-length multiple of 32 with a minimum size of 64 bits and the key size is 128 bits. The algorithm loops through the block words and adds to each one a function of its neighbors, full cycle number (the full cycle is n rounds long, n = the number of words of block) and the key . The number of times to perform a full cycle on a given block is 6 + 52/n [17].

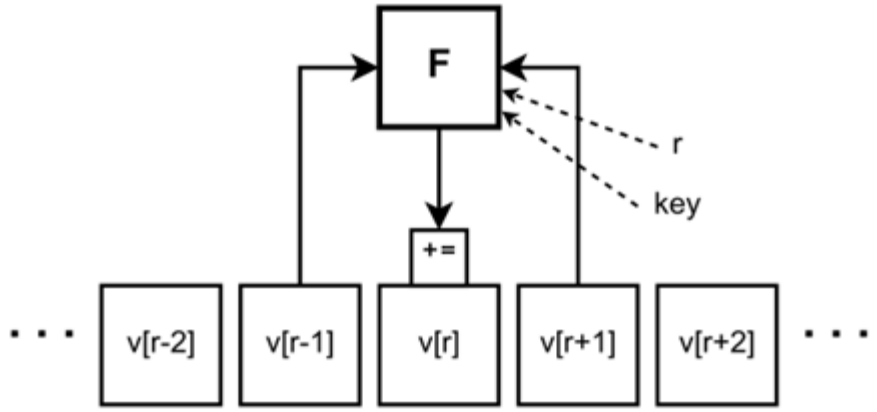

Figure 1. The structure of XXTEA encryption from Yarrkov [17]

The NSA – SPECK and SIMON

The National Security Agency (NSA) has publicly released two families of block ciphers designed for devices that require lightweight cryptosystems, SPECK and its sister algorithm SIMON. SPECK utilizes ARX based algorithms and is designed to be implemented in software, while SIMON is optimized for use in hardware. SIMON and SPECK were designed specifically for use in resource-constrained devices with an emphasis on flexibility and portability. The NSA claims "Speck has the highest throughput on 64- bit processors of any block cipher implemented in software, and amongst block ciphers in the literature, Simon appears to have the highest throughput per unit area for pipelined ASIC implementations" [18]. They both support block sizes of 32, 48, 64, 96, and 128 bits and key sizes ranging from 64 to 256 bits. SIMON and SPECK use a round function that takes two blocks x and y and the round key k and maps them in the following way (x, y) (y $\oplus$f(x)$\oplus$k, x), where f (x) = (Sx & S8x) $\oplus$ S2x (with S^j being a left circular shift by j bits, and and S^{-j} being a right circular shift by j bits). Decryption in SIMON is achieved "by swapping cipher- text words, reading round keys in reverse order, and then swapping the resulting plaintext words" [18] whereas SPECK decryption utilizes modular subtraction and reversed rotations. The major difference between SPECK and SIMON is that SPECK reuses the round function for the purpose of key scheduling, which allows for code

reuse and improved efficiency, whereas SIMON - which, because it is optimized for hardware - does not [18].

Other Considerations in Lightweight Cryptography

Message Authentication Codes

Message Authentication Codes (or MACs) have the potential to be particularly efficient, as they provide both message security and a check for message integrity. They generate tags from a message and a key. According to the NIST standardizing body, these tags should be at least 64 bits in most cases[6]. Though they are beyond the scope of this paper, interested parties can explore the Chaskey, TuLP, and LightMAC algorithms, which are dedicated to lightweight systems. Notably, Chaskey leans heavily on the ARX methodology heretofore discussed [20].

Applications

These technologies show particular promise in the areas of RFID and IoT. RFID applications have extreme resource, latency, and side-channel security constraints. RFID chips often contain extremely sensitive personal information. That being said, as Biryukov and Perrin discuss in their survey of Lightweight Symmetric Cryptography [10], RFID applications also don't encrypt large amounts of data, and have low throughput. Therefore, attacks requiring large amounts of data can be disregarded. Furthermore, the impact of an attack is limited geographically to the location of the card, Bearing in mind these allowances, "weak" cryptography (on the order of 80-bit keys) with high side-channel protection to prevent over-the-air attacks would be desirable. To this end, Yu et. al. [21] propose an XTEA-based algorithm, earlier discussed in this paper.

IoT devices present a large number of unique constraints. Sklavos & Zaharakis discuss

in their survey constraints and allowances for IoT [4]. Most notably, IoT devices must have high usability (read: low latency) without compromising user anonymity and data protection. IoT, too, has the potential to deal with highly sensitive data, like biometric data from upcoming Body Area Networks. Biryukov and Perrin's survey also notes that IoT devices will likely be running the algorithms on their current microprocessor rather than on dedicated hardware [10]. Therefore, software rather than hardware algorithms are to be considered (e.g. Chaskey, Fly, Lea, Pride, Sparx) These software algorithms can be lightened, however, as the smaller messages of IoT allow smaller internal states and smaller key sizes [10]. Without these precautions, actions (as discussed in the introduction), involving catastrophic spread of malware and compromise of user security and physical environment, are readily accessible tools to malicious actors .

Conclusion: Future Directions

In 2013, the National Institute of Standards and Technology (NIST) initiated the Lightweight Cryptography Project tasked with investigating the need for new cryptographic algorithms for use on resource-constrained devices. To achieve this task NIST holds workshops on a regular basis, the first being the NIST Lightweight Cryptography Workshop held in 2015. NIST most recently held a workshop in October 2016 (several of the research papers mentioned in this overview were presented at these NIST conferences). In March 2017 NIST published their Report on Lightweight Cryptography, a summary of their Lightweight Cryptography Project's efforts to date and plans to standardize lightweight cryptographic algorithms. In that report, they give some general metrics on resource-constrained devices in terms of software and hardware limitations. The report also briefly summarizes the current block ciphers designed to improve performance over NIST's AES-128 including the previously mentioned XXTEA, SIMON and SPECK. The report lists some general reasons why these

algorithms have performance benefits over AES including smaller block sizes, key sizes, simpler round functions and key scheduling algorithms, and smaller implementations in code [6].

The authors go on to summarize the state of lightweight hash functions, message authentication codes and stream ciphers. Additionally, they discuss how NIST-approved algorithms perform on resource-constrained devices, and conclude in the case of AES (one of NIST's approved block ciphers), when the performance of AES is acceptable it should be used. In the case of NIST approved hash functions, they find the SHA family of cryptographic hash functions are not suitable for use in resource-constrained devices as a result of the their size requirements [6]. The authors' description of the scope of the NIST Lightweight Cryptography Project includes the desire to focus on several general categories of symmetric cryptosystems (i.e., block ciphers, stream ciphers and message authentication codes among others) [6] and stresses the need for long-term security, which should aim for post-quantum resilience. NIST evaluates the lightweight cryptographic algorithms submitted to them based on profiles, "which consist of a set of design goals, physical characteristics of target devices, performance characteristics imposed by the applications, and security characteristics" [6].

Table 1 Profile Characteristics

Physical characteristics	Performance characteristics	Security characteristics
Area (in GEs, logic blocks, or mm^2) Memory (RAM/ROM) Implementation type (hardware, software, or both) Energy (J)	Latency (in clock cycles or time period) Throughput (cycles per byte) Power (W)	Minimum security strength (bits) Attack models (e.g., related key, multi-key) Side channel resistance requirements

NIST profiles for evaluating lightweight cryptosystems from NIST's Report on Lightweight Cryptography [6]

The IACR (International Association for Cryptologic Research) organizes a conference, the International Workshop on Lightweight Cryptography for Security and Privacy (a.k.a. LightSec) on an annual basis, the first being held in 2011. The goal of the LightSec conferences to promote research on 'lightweight security'. This workshop provides a platform for concerns about lightweight security to be tackled and to evaluate proposed solutions. Selected papers from the most recent LightSec at the time of this survey can be found in the post-refereed proceedings of LightSec '16 [22].

CryptoLUX is a cryptography research group based out of the University of Luxembourg. They are actively conducting research on lightweight cryptographic algorithms and have produced some new designs in the past few years. Their most recent research was presented at the most recent NIST Lightweight Cryptography Workshop and documented the design of a family of lightweight block ciphers called SPARX. SPARX combines the resilience to timing attacks from ARX-Based algorithms with the benefits provided by S-boxes (i.e., the resilience to cryptanalysis) [23].

One important consideration to the design of new cryptosystems is their strength against post-quantum attacks. For example, the difficulty to factor a large number (the public key) into two large primes is the basis for RSA (an asymmetric cryptosystem). The modern RSA cryptosystem achieves its level of resilience to current factorization methods by recommending the use of a 3072-bit key size. A post-quantum RSA key size to provide sufficient rigidity to quantum factorization (see Shor's algorithm) would need to be over 1-terabyte [24].

Over the last few years several algorithms have been designed with post-quantum security in mind. At the last NIST Lightweight Cryptography Workshop (held in 2016), 2 out of 17 total papers accepted concerned post-quantum security for the IoT. The Walnut Digital Signature Algorithm designed by security researchers at the SecureRF Corporation is an

asymmetric signature validation algorithm, that the designers claim “verifies signatures significantly faster than [elliptic-curve cryptography] in both software and hardware, even in small, constrained environments and is resistant to all known attacks, both conventional and quantum” [25].

A combination of the proliferation of devices that cannot efficiently handle modern cryptographic standards and a lack of implementation of lightweight cryptography has led to a series of troubling events notably including an embarrassing fiasco for the government of Estonia, pressure from the U.S. congress to improve IoT security and semi-regular news stories about IoT botnets causing record setting DDoS attacks (see Mirai, IoT Reaper, etc.). Fortunately, in recent years there has been a push by academia and industry to develop lightweight cryptographic solutions. We have discussed a subset of those solutions in this paper. It is now in the hands of the producers of resource-constrained devices whether they will implement these solutions or continue to risk distributing devices with dubious security.

Bibliography

1. Possible security risk affects 750,000 Estonian ID-cards. Estonian World. 2017 Sep 5 [accessed 2017 Nov 15]. http://estonianworld.com/technology/possible-security-risk-affects-750000-estonian-id-cards/

2. Greenberg A, Greenberg A, Newman LH, Newman LH, Barrett B, Lapowsky I, Greenberg A, Jarvis B. The Reaper IoT Botnet Has Already Infected a Million Networks. Wired. 2017 Oct 20 [accessed 2017 Nov 15]. https://www.wired.com/story/reaper-iot-botnet-infected-million-networks/

3. Ronen E, O'Flynn C, Shamir A, Weingarten A-O. IoT Goes Nuclear: Creating a ZigBee Chain Reaction. 2016.

4. Sklavos N, Zaharakis ID. Cryptography and Security in Internet of Things (IoTs): Models, Schemes, and Implementations. IEEE; 2016. p. 1–2.

5. Warner M. Internet of Things (IoT) Cybersecurity Improvement Act of 2017. 2017. https://www.congress.gov/bill/115th-congress/senate-bill/1691/text

6. McKay KA, Bassham L, Turan MS, Mouha N. Report on lightweight cryptography. Gaithersburg, MD: National Institute of Standards and Technology; 2017. doi:10.6028/NIST.IR.8114

7. Stinson D. Cryptography: Theory and Practice,Second Edition. 2nd ed. CRC/C&H; 2002.

8. Gray RM. Entropy and Information Theory. 2nd ed. Springer Publishing Company, Incorporated; 2011.

9. Cryptography in C and C++: See what it takes to build an industrial-strength crypto API, Second Edition.

10. Biryukov A, Perrin LP. State of the Art in Lightweight Symmetric Cryptography. 2017.

11. Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJ, Seurin Y, Vikkelsoe C. PRESENT: An ultra-lightweight block cipher. CHES. 2007;4727:450–466.

12. Grosso V, Leurent G, Standaert F, Varici K, Journault A, Durvaux F, Gaspar L, Kerckhof S. SCREAM Side-Channel Resistant Authenticated Encryption with Masking. CAESAR submission. 2015.

13. Khovratovich D, Nikoli I. Rotational Cryptanalysis of ARX. FSE. 2010;6147:333–346.

14. Dinu D, Perrin L, Udovenko A, Velichkov V, Großschädl J, Biryukov A. Design Strategies for ARX with Provable Bounds: SPARX and LAX. In: Advances in Cryptology–ASIACRYPT 2016: 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I 22. Springer Berlin Heidelberg; 2016.

15. Bernstein DJ. Cache-timing attacks on AES. 2005.

16. Bernstein DJ. The Salsa20 family of stream ciphers. Springer. 2008.

17. Yarrkov E. Cryptanalysis of XXTEA. IACR Cryptology ePrint Archive. 2010:254.

18. Beaulieu R, Shors D, Smith J, Treatman-Clark S, Weeks B, Wingers L. The SIMON and SPECK lightweight block ciphers. In: Proceedings of the 52nd Annual Design Automation Conference. ACM; 2015. p. 175.

19. Bernstein DJ. ChaCha, a variant of Salsa20. Workshop Record of SASC. 2008;8:3–5.

20. Mouha N, Mennink B, Van Herrewege A, Watanabe D, Preneel B, Verbauwhede I. Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers. In: Selected Areas in Cryptography -- SAC 2014. Springer, Cham; 2014. p. 306–323. (Lecture Notes in Computer Science).

21. Yu J, Khan G, Yuan F. XTEA encryption based novel RFID security protocol. In: 2011 24th Canadian Conference on Electrical and Computer Engineering(CCECE). 2011. p. 000058–000062.

22. Bogdanov A. Lightweight Cryptography for Security and Privacy: 5th International Workshop, LightSec 2016, Aksaray, Turkey, September 21-22, 2016, Revised Selected Papers. Springer International Publishing; 2017.

23. Bache F, Schneider T, Moradi A, Giineysu T. SPARX: A side-channel protected processor for ARX-based cryptography. In: Design, Automation Test in Europe Conference Exhibition (DATE), 2017. 2017. p. 990–995.

24. Bernstein DJ, Heninger N, Lou P, Valenta L. Post-quantum RSA. In: Post-Quantum Cryptography. Springer, Cham; 2017. p. 311–329. (Lecture Notes in Computer Science).

25. Atkins D. Walnut Digital Signature Algorithm: A lightweight, quantum-resistant signature scheme for use in passive, low-power, and IoT devices. SecureRF.