Des. Codes Cryptogr. (2017) 82:495–509
DOI 10.1007/s10623-016-0193-8

Improving the security and efficiency of block ciphers based on LS-designs

Anthony Journault[1] · **François-Xavier Standaert**[1] · **Kerem Varici**[1]

Received: 16 October 2015 / Accepted: 16 February 2016 / Published online: 9 March 2016

Abstract LS-designs are a family of bitslice ciphers aiming at efficient masked implementations against side-channel analysis. This paper discusses their security against invariant subspace attacks, and describes an alternative family of eXtended LS-designs (XLS-designs), that enables additional options to prevent such attacks. LS- and XLS-designs provide a large family of ciphers from which efficient implementations can be obtained, possibly enhanced with countermeasures against physical attacks. We argue that they are interesting primitives in order to discuss the general question of "how simple can block ciphers be?".

Keywords Block cipher · Physical security · Side channel analysis · Design

Mathematics Subject Classification 94A60

1 Introduction

LS-designs are a family of block ciphers proposed at FSE 2014, aimed for efficient bitslice implementations [18]. They essentially combine linear diffusion L-boxes with non-linear bitslice S-boxes. The instances proposed so far (namely the involutive cipher Robin and the non-involutive cipher Fantomas) have additionally been selected to minimize the total number of AND gates, in order to allow efficient masked implementations against side-channel attacks [8], which is also beneficial to multiparty computation and fully homomorphic encryption [2]. In a more recent work by Leander et al., it has been shown that the involutive instance Robin was susceptible to an invariant subspace attack, leading to a weak keys set of density 2^{-32} [25]. This raised questions regarding the origin of the attack and the possibility to prevent it for involutive LS-designs.

This is one of several papers published in *Designs, Codes and Cryptography* comprising the "Special Issue on Coding and Cryptography".

✉ Kerem Varici
kerem.varici@uclouvain.be

[1] ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Louvain-la-Neuve, Belgium

In this paper, we complement these works with two main contributions.

First, we analyze the invariant subspace attack against Robin and show that it can be prevented with simple heuristics, e.g. a better choice of round constants. For this purpose, we exploit the fact that these constants should have all their bits varying (in bitslice representation), in order to avoid invariant subspaces for the S-boxes or L-boxes to be trivially propagated through the rounds.

Second we question the possibility to improve the efficiency of LS-designs with a better choice (and different sizes) of components. In particular, Robin and Fantomas are based on 8-bit S-boxes and 16-bit L-boxes. While it is very convenient from an implementation point-of-view, the selection of these components was partially heuristic (since, e.g. an exhaustive analysis of 8-bit S-boxes is computationally out of reach). As a result, we investigate an alternative approach in two steps. First, we design 32-bit "Super S-Boxes" taking advantage of optimal components, i.e. 4-bit S-boxes and 32-bit L-boxes based on a maximum distance separable (MDS) code. Second, we combine these Super S-boxes with an additional ShiftColumns operation. Both the use of Super S-boxes and their combination with a ShiftColumns operation are naturally reminiscent from an AES-like cipher [11,17], but with a bitslice rather than block-oriented structure. Interestingly, we show that the resulting eXtended LS-designs (XLS-designs) can also be implemented very efficiently on various platforms, e.g. based only on table lookups and word-oriented operations, yet leading to slightly more complex tradeoffs than LS-designs, due to their slightly more involved structure. For concreteness and further investigations, we additionally specify an instance of such XLS-design, denoted as Mysterion, with 128-bit or 256-bit block size.

2 The invariant subspace attacks against Robin

2.1 LS-design, Robin and Fantomas

LS-designs are a family of block ciphers that are composed of a combination of lookup table-based L-boxes and bitslice S-boxes. The definition of s-bit S-boxes and l-bit L-boxes directly gives rise to an instance of $n = s \cdot l$-bit cipher. One advantage of LS-designs is their inherent simplicity, as illustrated with the short specifications given in Algorithm 1. The cipher takes n-bit plaintext and key blocks as input, and follows substitution permutation network (SPN) approach. Namely, the inputs and state are represented as $s \cdot l$ arrays of bits, with s the number of rows and l is the number of columns. In each round, the S-box operation acts on the columns, and the L-box operation acts on the rows. These two components combined with constant and key addition define the round function of LS-designs, that is iterated N_r times in order to obtain the ciphertext.

Algorithm 1 LS-design with l-bit L-boxes and s-bit S-boxes

$x \leftarrow P \oplus K$; ▷ x is an $s \cdot l$-bit matrix
for $0 \leq r < N_r$ **do**
  for $0 \leq i < l$ **do** ▷ S-box Layer
    $x[\star, i] = \mathsf{S}[x[\star, i]]$;
  for $0 \leq j < s$ **do** ▷ L-box Layer
    $x[j, \star] = \mathsf{L}[x[j, \star]]$;
  $x \leftarrow x \oplus K \oplus C(r)$; ▷ Key addition and round constant
return x

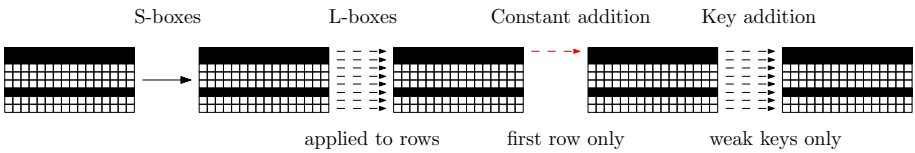

Fig. 1 An example of invariant subspace attack against one round of Robin

Concretely, both Robin and Fantomas were based on 8-bit S-boxes and 16-bit L-boxes. For the former one, these components are involutive, in order to improve the performances of the cipher when decryption has to be implemented.

2.2 Invariant subspace attacks and results on Robin

The invariant subspace attack was first introduced at CRYPTO 2011 [23] and applied to the lightweight "PRINTcipher" [21]. We can summarize the attack as follows. Let us consider an n-bit iterative block cipher, with round function $\mathsf{R}_k : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n$, such that $\mathsf{R}_k(x) = \mathsf{E}(x + k)$, with E an n-bit permutation. If there exists a subspace $\mathcal{S} \subseteq \mathbb{F}_2^n$ and two constants $a, b \in \mathbb{F}_2^n$ such that $\mathsf{E}(\mathcal{S} + a) = \mathcal{S} + b$, then for a round key $k = s + a + b$ with $s \in \mathcal{S}$, the following holds:

$$\mathsf{R}_k(\mathcal{S} + b) = \mathsf{E}\big((\mathcal{S} + b) + (s + a + b)\big) = \mathsf{E}(\mathcal{S} + a) = \mathcal{S} + b.$$

That is, the round function maps the affine subspace $\mathcal{S} + a$ onto $\mathcal{S} + b$. Furthermore, if all round keys are in $\mathcal{S} + (a + b)$, then this property is iterative. This is the case for some key-alternating ciphers [5], where the same master key is used as subkey through the whole cipher, e.g. LED [19], Zorro [16], Noekeon [13], Fantomas and Robin [18], which are therefore natural targets for invariant subspace attacks. The Eurocrypt 2015 paper [25] that exhibits a weak keys set of density 2^{-32} for Robin is based on this property, and takes advantage of the involutive nature of its components together with weak round constants.

More precisely, the involutive building blocks of Robin help finding self-similarities within the cipher—a new type of such self-similarities (for the L-boxes) is actually given in the Eurocrypt paper. Besides, and as mentioned above, LS-designs are such that S-boxes act through columns and the linear layer acts through rows. Hence, if there exists an invariant subspace (e.g.) for the S-box layer and all inputs to the S-boxes are chosen from it, then the linear layer will not change this subspace.[1] That is, if we call the bits which form the invariant subspace active and other bits passive (as in differential cryptanalysis), then the linear layer does not mix active and passive bits. Combined with the fact that the round constants of Robin are sparse, and only apply to one state row, this allowed the propagation of invariant subspaces through the cipher. An illustration of the attack based on an invariant subspace for the S-box layer is given in Fig. 1, where black boxes represent bits that form an invariant subspace.

3 A simple tweak: modifying the round constants

Based on the previous description, a couple of ways to fix the invariant subspace attack could be considered for Robin, e.g. changing its components (linear layer & S-box), applying a key

[1] As just mentioned, this attack can be applied by finding an invariant subspace for the linear layer as well, in which case the S-box layer will not change the subspace.

scheduling, or changing the round constants. Among those, changing the round constants is the easiest one, since it implies minimum changes on the design. Concretely, one suggestion is to use a dense set of round constants, applied to all the rows rather than a single one. For example, a linear feedback shift register (LFSR) with 16-bit state size (and e.g. primitive polynomial $P(X) = X^{16} + X^5 + X^3 + X^2 + 1$) could be used for this purpose. Eight consecutive states can then be combined together to form each round constant. We verified with the same generic algorithm as described in [24] that this choice was sufficient to remove the invariant subspace from the Robin rounds (up to the computational limits of the algorithm). We also checked exhaustively that no invariant subspaces can propagate through the rounds of reduced (32-bit) LS-designs using such dense constants. Note that despite no invariant subspace attack has been exhibited against the non-involutive cipher Fantomas, it has a similar structure as Robin, and its round constants are sparse as well. Therefore, tweaking this cipher (e.g. with stronger round constants) could be advisable.

3.1 Concrete proposal for Robin*

While conceptually simple, the tweak in the previous section is still quite expensive, since it requires generating 8×16 pseudorandom bits per round. Yet, adding a full round constant seems to be the only simple way to avoid the invariant subspaces in Robin. In the following, we suggest a simple intermediate path and a concrete proposal for Robin*. Namely, instead of generating (and keeping in memory) 8×16 bits at each round with an LFSR, we generate 16-bit constants that we then rotate before addition to the state. Concretely, the 16-bit round constants in Robin* are defined as:

$$T(\rho) = 2199 \cdot \rho \bmod 2^{16}.$$

We consider constants of the form $T(\rho) = T \cdot \rho \bmod 2^{16}$ because they can implemented by incrementing a counter by steps of T. However, we would rather avoid the trivial choice $T = 1$ because this implies simple linear relations between the constants, such as $T(2\rho+1) = T(2\rho) \oplus 1$. Following, we built 16×16 matrices with the binary representation of $T(1)$, $T(2)$, …, $T(16)$, and computed the rank of these matrices. There are a few values that give a full rank matrix, and we decided to use the smallest value with this property: 2199. Next, Algorithm 2 describes how the 16-bit round constants will be extended to 128 bits, where $RotL(x, y)$ stands for the left rotation of x by y bits, and T_i is the 16-bit constant of round i. The heuristic tool of [25] did not exhibit any difference between this solution and the more (memory) expensive one in the previous paragraph. Eventually, Robin* mostly follows the specifications in Algorithm 1, with only modification that the 8-bit constants C are replaced by 128-bit constants C^*.

Algorithm 2 n-bit constants $C^\star(T_i, s)$ from l-bit constants T_i

$C^\star \leftarrow 0$; ▷ $C^\star$ is an $s \cdot l$-bit matrix
for $0 \leq j < s$ **do**
  $C^\star[j, \star] = RotL(T_i, j)$
return x

4 eXtended LS-designs and Mysterion

The previous sections highlighted that invariant subspace attacks against (involutive) LS-designs exploit the structural simplicity of these ciphers. While this simplicity is highly beneficial to implementation efficiency, it also leads to the question whether a slightly more involved structure could provide better security margins. In this section, we investigate this option and, motivated by the efficient masking goal of LS-designs, combine it with a further improvement of the balance between linear and non-linear operations within the cipher. The rationale behind this tweaked approach is twofold. First, for the linear part, we observe that from the security point-of-view it would be interesting to take advantage of a (non-binary) MDS code to build the diffusion layer. Second, for the non-linear part, S-boxes with smaller bit sizes are chosen since it is known how to construct optimal ones. For example, Ullrich et al. found an optimal (from the linear and differential cryptanalysis points-of-views) 4-bit S-box requiring only 4 AND gates (later denoted as Class 13) [30]. Based on these observations, we propose new instances of ciphers where an optimal 4-bit S-box with an MDS diffusion matrix are combined, which results in 32-bit Super S-boxes, and then combine these Super S-boxes with a ShiftColumns operation to obtain 128- and 256-bit ciphers. Admittedly, this approach does not strictly follow the LS-design specifications, since (i) its diffusion layer is not based on binary matrices anymore, and (ii) it requires an additional ShiftColumns operations. So it primarily aims to improve the security margins of LS-designs, e.g. against linear and differential cryptanalysis and invariant subspace attacks (see Sect. 4.2). Yet, and quite interestingly, we will show in Sect. 4.3 that the resulting XLS-designs can still be implemented efficiently, taking advantage of the linearity of the MDS diffusion and ShiftColumns operations. So intuitively, the main price to pay for the latter approach is slightly more complex specifications (although they can be viewed as a bitslice counterpart to AES-like ciphers and have a concise description), which are interesting to compare with the extreme simplicity of LS-designs, both from the implementation efficiency and the physical security points-of-view.

4.1 Specifications

XLS-designs can be described as combination of b LS-designs of $s \cdot l$ bits, where s is the size of the S-box (in bits, as in LS-designs), and l is the size of the underlying MDS matrix of the L-box (and no longer the bit size of the L-box as in LS-designs), resulting in a $n = b \cdot s \cdot l$-bit cipher. Note that the change on l notation is necessary to keep notations consistent with LS-designs, since a binary matrix cannot be MDS. Concretely, the internal state of an XLS-design can be written as $X[\star, \star, \star]$, such that $X[i, \star, \star]$ is an $s \cdot l$-bit block (with $1 \leq i \leq b$), $X[i, j, \star]$ is block i's jth l-bit row (with $1 \leq j \leq s$) and $X[i, \star, j]$ is block i's jth s-bit column (with $1 \leq j \leq l$). As illustrated in Fig. 2, the S-box layer of XLS-designs is strictly the same as in Algorithm 1. Their L-box layer slightly changes compared to LS-designs, since it is applied to all the rows of each block at once (rather than to row by row in LS-designs). And the main difference is the additional ShiftColumns layer, that can be viewed as the bitslice dual to the ShiftRows operation in the AES Rijndael, and will be defined next.

XLS-designs are succinctly described in Algorithm 3. We now describe the different components that give rise to the Mysterion-128 (4×32-bit blocks), and the Mysterion-256 (8×32-bit blocks), that both exploit 4-bit S-boxes.

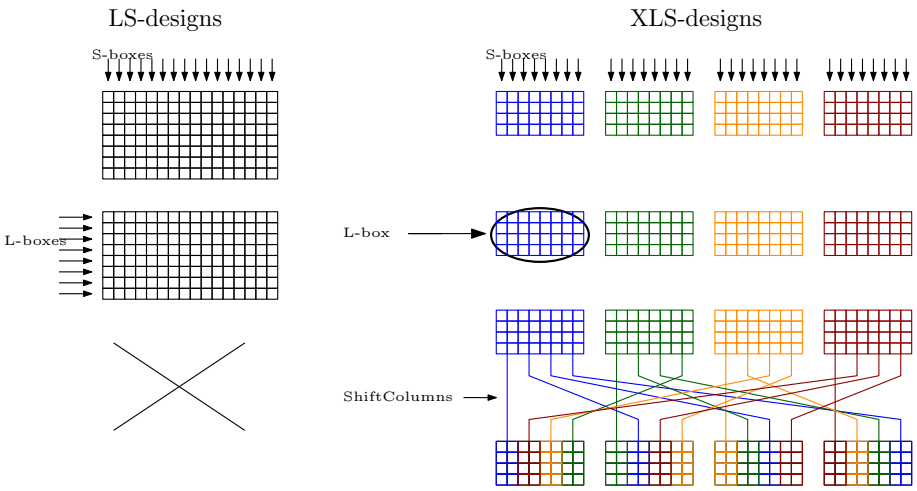

Fig. 2 128-bit LS-design versus 128-bit XLS-design

Algorithm 3 XLS-design with $l \cdot s$-bit L-boxes, s-bit S-boxes and b blocks

1: $x \leftarrow P \oplus K$ ▷ x is a $s \cdot (l.b)$ bits matrix
2: **for** $0 \leq r < N_r$ **do**
3: **for** $0 \leq j < b$ **do**
4: **for** $0 \leq i < l$ **do**
5: $x[j, \star, i] = \mathsf{S}[x[j, \star, i]]$; ▷ S-box layer
6: **for** $0 \leq j < b$ **do**
7: $x[j, \star, \star] = \mathsf{L}[x[j, \star, \star]]$; ▷ L-box layer
8: **for** $0 \leq k < s$ **do**
9: $x[\star, k, \star] = \mathsf{ShiftColumns}[x[\star, k, \star]]$; ▷ ShiftColumns layer
10: $x \leftarrow x \oplus K \oplus C(r)$ ▷ Key and round constant addition
return x

The S-box Mysterion uses the Class13 S-box [30], that has a bitslice representation with four AND[2] and four XOR gates (see Appendix 1), algebraic degree three, differential probability of 2^{-2}, and linear probability of 2^{-1}.

The L-box Mysterion uses a linear transformation derived from the recent paper by Augot and Finiasz [3], in which an algorithm that allows to find recursive MDS diffusion layers using shortened BCH codes is described. (Recursive MDS matrices can be expressed as a power of the companion matrix of a polynomial.) This algorithm uses the degree of the polynomial k (hence the size of the companion matrices), and the field size $q = 2^s$ as parameters, and provides all the polynomials of degree k over $\mathbb{F}_{2^s}$ such as their companion matrices raised to the power k gives MDS diffusion layers. We ran it with parameters $k = 8$ and $s = 4$ using Magma, in order to obtain an 8×8 MDS matrix over $\mathbb{F}_{2^4}$. The selected degree-8 polynomial with coefficients in $\mathbb{F}_{2^4} \cong \mathbb{F}_2[\alpha]/(\alpha^4 + \alpha + 1)$, is $\mathsf{P}(\mathsf{X}) = \mathsf{X}^8 + \alpha^3 \cdot \mathsf{X}^7 + \alpha^4 \cdot \mathsf{X}^6 + \alpha^{12} \cdot \mathsf{X}^5 + \alpha^8 \cdot \mathsf{X}^4 + \alpha^{12} \cdot \mathsf{X}^3 + \alpha^4 \cdot \mathsf{X}^2 + \alpha^3 \cdot \mathsf{X} + 1$. The resulting diffusion layer is coming from an MDS code $[16, 8, 9]_{\mathbb{F}_{2^4}}$ and therefore has both its differential and linear branch number equal to 9.

[2] More precisely, three ANDs and one OR, which can be masked at the same cost.

ShiftColumns For Mysterion-128, ShiftColumns acts on columns two by two. The first two columns of each block are not moved, the second two columns are moved by one block, the third two columns are moved by two blocks, and the fourth two columns are moved by three blocks. This operation can also be described as a bit permutation of a 32-bit word, with logic operations: $\mathtt{X} = (\mathtt{X}\ \&\ \mathtt{0xC0C0C0C0}) \vee \mathtt{ROL}(\mathtt{X}\ \&\ \mathtt{0x03030303}, 8) \vee \mathtt{ROL}(\mathtt{X}\ \&\ \mathtt{0x0C0C0C0C}, 16) \vee \mathtt{ROL}(\mathtt{X}\ \&\ \mathtt{0x30303030}, 24)$, where $\vee$ and & stand for logic OR and AND, and $\mathtt{ROL}(\mathtt{X}, \mathtt{n})$ stands for the left rotation of X by n bits. For Mysterion-256, ShiftColumns acts on columns one by one. The first columns of each block are not moved, the second columns are moved by one block, . . ., and the eighth columns are moved by seven blocks. See Appendix 1 for the alternative description.

These components directly define our two instances Mysterion-128, with parameters $b = 4, s = 4, l = 8$, and Mysterion-256, with parameters $b = 8, s = 4, l = 8$. As for round constants, we suggest to use simpler ones as in the original Robin and Fantomas ciphers. This will be further justified in the next section.

4.2 Security analysis

We now exhibit the good cryptanalytic properties of Mysterion with two main goals. On the one hand, we show that simple 4-round bounds against linear and differential cryptanalyses can be obtained for XLS-designs, inheriting from their AES-like structure. On the other hand, we argue why its more complex structure also improves resistance against invariant subspace attacks. We also (briefly discuss) a couple of additional standard cryptanalyses against block ciphers. Note that as for LS-designs, no related-key security is claimed for Mysterion.

Security against linear and differential cryptanalyses A straightforward application of the wide-trail strategy [10] leads to the following theorems.

Theorem 1 *Four rounds of* Mysterion-128 *has at least 45 active S-boxes.*

Theorem 2 *Four rounds of* Mysterion-256 *has at least 81 active S-boxes.*

A sketch of the proofs is given in Appendix 1. As a result, we have the next bounds for the probabilities $\mathrm{Pr}_{lin}(4\mathrm{R})$ (resp. $\mathrm{Pr}_{diff}(4\mathrm{R})$) of linear (resp. differential) characteristics over 4 rounds of Mysterion-128, where $\mathrm{Pr}^{max}_{lin}(\text{S-box})$ (resp. $\mathrm{Pr}^{max}_{diff}(\text{S-box})$) stands for the linear (resp. differential) probability of the S-box:

$$\mathrm{Pr}_{lin}(4\mathrm{R}) \leq \mathrm{Pr}^{max}_{lin}(\text{S-box})^{45} = 2^{-45},\ \mathrm{Pr}_{diff}(4\mathrm{R}) \leq \mathrm{Pr}^{max}_{diff}(\text{S-box})^{45} = 2^{-90}.$$

And similarly, for the Mysterion-256, we have:

$$\mathrm{Pr}_{lin}(4\mathrm{R}) \leq \mathrm{Pr}^{max}_{lin}(\text{S-box})^{81} = 2^{-81},\ \mathrm{Pr}_{diff}(4\mathrm{R}) \leq \mathrm{Pr}^{max}_{diff}(\text{S-box})^{81} = 2^{-162}.$$

Table 1 compares the upper bounds for the maximum probabilities of differential characteristics for Robin, Fantomas and Mysterion. Setting the number of rounds to 12 for Mysterion-128 and 16 for Mysterion-256 leads to very comfortable security margins, and better bounds than for Robin and Fantomas. Linear characteristics behaves in the same way, leading to similar recommendations.

Security against invariant subspace attacks As discussed in Sect. 2.2, invariant subspace attacks can be of two types. A (simpler) one taking advantage of invariant subspaces in the S-box and (a more intricate) one using equality spaces in the L-box (that is highly structured in the case of Robin). The first one is easy to bypass with a good choice of S-box, e.g.

Table 1 Maximum probability of differential characteristics for LS- and XLS-designs

Number of rounds	8	12	16
Prob. diff char. for Robin	2^{-128}	2^{-192}	$\mathbf{2^{-256}}$
Prob. diff char. for Fantomas	2^{-160}	$\mathbf{2^{-256}}$	2^{-344}
Prob. diff char. for Mysterion-128	2^{-180}	$\mathbf{2^{-270}}$	2^{-360}
Prob. diff char. for Mysterion-256	2^{-324}	2^{-486}	$\mathbf{2^{-648}}$

the Class13 S-box has no trivial invariant subspaces.[3] The second one is more difficult to analyze. So far, results of [25] only describe a heuristic tool allowing to look for such invariant subspaces. Hence, running this tool (with the available computational resources) on full cipher instances, and exhaustively searching on reduced cipher instances, is the best that one can currently do. For example, invariant subspaces against Fantomas (and Robin⋆) could not be spotted by using this approach. In the case of Mysterion, we first note that the use of a 32-bit L-box is not sufficient to prevent the existence of invariant subspaces within the rounds (as revealed by an exhaustive analysis performed on a 32-bit block). However, the addition of a ShiftColumns operation will break the propagation of any subspace found for the L-box with high probability. This was confirmed by a computationally-bounded analysis performed on Mysterion-128. We therefore conclude that XLS-designs can withstand invariant subspace attacks even with sparse round constants (as usually used in block cipher designs, to limit their memory requirements).

Algebraic attacks Algebraic attacks on block ciphers were introduced by Courtois and Pieprzyk [9]. They essentially represent a block cipher as a system of non-linear equations and look for solution using some specialized solver. Since block ciphers are defined as iterations of a complex round function, the number of equations and variables grows rapidly and solving them is expected to be a hard problem. Exactly determining the security level against such attacks is difficult. Yet, one usually evaluates the number of variables and quadratic equations of the cipher for this purpose [4]. In Mysterion, we used 4-bit S-boxes with algebraic degree $d = 3$ and every 4-bit S-box has at least $e = 21$ quadratic equations for the $v = 8$ input/output variables. This means $(d^2 \cdot N_r \cdot e)$ quadratic equations in $(d^2 \cdot N_r \cdot v)$ variables for N_r rounds. In the case of Mysterion-128, we end up with 4032 equations in 1536 variables. These numbers are increased to 5376 equations in 2048 variables for Mysterion-256. In comparison, the AES has 6296 equations in 3296 variables [4]. We expect these numbers to be sufficient for both instances of Mysterion to be secure against algebraic attacks.

Higher-order differential attacks/cube attacks Higher-order differential cryptanalysis [20] and Cube attacks [14] are powerful cryptanalytic tools based on differential derivatives of high orders. One recent extension of these attacks is the zero-sum distinguisher described in [6]. The usual strategy to prevent such cryptanalyses is to guarantee a high algebraic degree after some cipher rounds. We used the tools from [7] to compute this number of rounds. As reported in Table 2, the algebraic degree reaches its maximum after 7 and 9 rounds, respectively for Mysterion-128 and Mysterion-256. In these cases, a partition of size 2^{127} and 2^{255} would be required to construct zero-sum distinguishers.

Integral attacks / division property Integral cryptanalysis was proposed in [12,22]. The attack considers a collection of m-bytes of plaintexts and their corresponding ciphertext values and

[3] Any S-box has (small dimensional) subspaces that gets mapped to subspaces.

Table 2 Estimated algebraic degree for Mysterion in function of the number of rounds

No of rounds	1	2	3	4	5	6	7	8	9	Reference
Mysterion-128	3	9	27	81	112	123	127	–	–	[7]
	–	–	12	28	84	113	124	–	–	[29]
Mysterion-256	3	9	27	81	198	237	250	254	255	[7]
	–	–	12	28	84	199	237	250	254	[29]

aims at extracting key information by observing the sum of ciphertext values for this collection of chosen plaintexts. It can be efficiently applied to block ciphers based on SPNs like the AES or LED. Since the Mysterion design also fits into this category, we briefly discuss its susceptibility to such attacks. For AES, the best integral property can be found up to four rounds, and then this property can be used to mount an attack from seven rounds to nine rounds depending on key sizes [15,26]. For Mysterion, a similar result can be obtained for four rounds, but that leaves comfortable security margin for the full cipher since we have 12 and 16 rounds for Mysterion-128 and Mysterion-256. We further mention a new type of integral property, namely the division property, introduced recently at EUROCRYPT 2015 [29]. It allows to construct more efficient integral distinguishers exploiting the limited algebraic degree of reduced ciphers. We complement the results of [7] with new bounds from this reference in Table 2 (which suggest sufficient security margins).

Boomerang attacks The boomerang attack [31] is a special type of differential cryptanalysis, where the main idea is to divide a cipher E into two sub-ciphers E_0 and E_1 such that $E = E_0 \circ E_1$. The attacker then constructs two relatively short differentials for E_0 and E_1 instead of finding a long differential for the cipher E. This may improve the results since shorter differentials usually have better probabilities. We know from Theorem 1 that four rounds of Mysterion-128 has at least 45 active S-boxes. If we use two four-round characteristics for E_0 and E_1, then the best differential probability of a boomerang distinguisher becomes $2^{-45\times 2} \times 2^{-45\times 2} = 2^{-180}$, which is smaller than $2^{-n} = 2^{-128}$. Therefore, we can deduce that any boomerang distinguisher with eight rounds or more will not work against Mysterion-128 (a similar conclusion can be reached for Mysterion-256, for which an eight-round boomerang distinguisher will have the best differential probability equal to $2^{-81\times 2} \times 2^{-81\times 2} = 2^{-324} \gg 2^{-256}$).

4.3 Performances

One of the goals of LS-designs (hence, by extension, XLS-designs) is to allow efficient masked implementations. In this respect, a natural problem is to find out whether the slightly more complex structure of XLS-designs, using (non-binary) MDS matrices and an additional ShiftColumns transformation, leads to a loss of efficiency. In this section, we briefly discuss this issue and detail how efficient table lookup-based implementations of Mysterion-128 can be obtained.

In general, the implementation of a 32-bit Super S-box can be directly implemented with logic operations (which is more time consuming), or with table lookups as in the case of the AES Rijndael (which is faster, but requires 16 tables of 256 bytes, rather than 4 such tables for a 128-bit LS-design). Next, the ShiftColumns operation mixes bits of different blocks, which can exploit the logic representation given in Sect. 4.1, or be implemented with

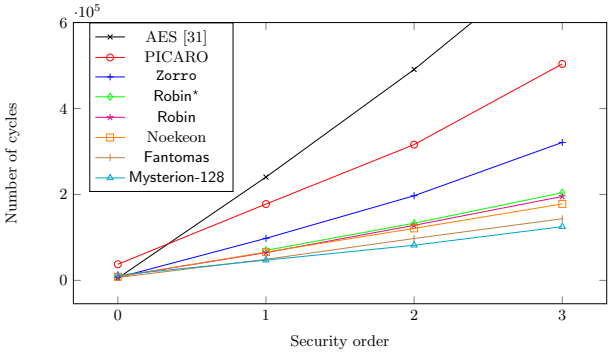

Fig. 3 Encryption times for different 128-bit block ciphers in an Atmel AtMega644p

table lookups. This leads to interesting tradeoffs from the physical security point-of-view. On the one hand, the logic representation of ShiftColumns requires less memory than its table-based execution, and acts at the row level. On the other hand, performing ANDs with constants including some bits set to zero can be viewed as a bit manipulation that may be harder to prevent leakages (as argued in [18]).

We implemented Mysterion-128 on a 8-bit microcontroller (Atmel AVR, AtMega644p), based on a mixed approach, namely table lookups for the L-boxes and logic operations for ShiftColumns. We also implemented Robin* for which we use a look-up table for the 16 round constants.[4] Our reference code is written in *C* and used the *avr-libc* library with headers $\#include < avr/pgmspace.h >$ and $\#include < avr/io.h >$. The *PROGMEM* attribute is used to save RAM. Results were obtained with the *avr-gcc* compiler and optimization option *-O2*. The execution time of the implementations are simulated using the *Atmel AVR Studio 6* software. Performances are reported for an unrolled version of the code.

Figure 3 summarizes our results in terms of number of cycles for Mysterion-128, together with natural competitors, i.e. Robin and Fantomas [18], Zorro [16], Noekeon [13], PICARO [27] and the AES [28]. Security order 0 means unprotected implementation *i.e.* no mask, security order 1 means two shares or one mask, and so on. The main conclusion of these evaluations is that such an XLS-design has excellent performances, except for unprotected implementations (for which Mysterion-128 is slightly less efficient than its competitors, Robin being the best one). More precisely, the reduced amount of non-linear operations in Mysterion-128 allows its implementations to compare favorably with its competitors already for first-order security. As previously mentioned, the price to pay for these excellent performances are potentially more leaky operations, which can be avoided using table lookups, but would then lead to larger memory requirements. Eventually we observe that Robin* has a only limited cycles' overhead compared to Robin, due to its XORs on the full state and rotations for the constants.

5 Conclusions

This work extends the block cipher design space from LS-designs to XLS-designs. We believe this is an interesting step forward, since it is in line with the general question of "how simple can block ciphers be?", in a context—i.e. considering the risk of side-channel analysis—

[4] For more constrained space, they could be computed as suggested in Sect. 3.1.

where simplicity is usually correlated with security. Indeed, simple and very structured ciphers are generally easier to protect against physical attacks. In this respect, our first contribution is to show that LS-designs are not inherently susceptible to invariant subspace attacks, but that their instantiation should be carefully considered. And our second contribution is to show that XLS-designs can indeed be implemented efficiently (and lead to better security bounds against linear and differential cryptanalysis), but that their best implementation requires informed decisions (e.g. on whether the use of bit manipulations can be critical). These questions lead to many open problems regarding the best cipher instances for different block/key sizes. For example, instances with 3-bit S-boxes could be considered to minimize the AND depth as in [2]; instances with 5-bit S-boxes could lead to even reduced round requirements (for linear and differential attacks); even for the 4-bit instances, it could be interesting to investigate the use of L-boxes based circulant matrices such as advertised in [1], which would allow alternative implementations to prevent cache-based timing attacks (although SSSE3 instructions can also be used for this),… And should we use LS- or XLS-designs for any of those instances? Whether a key scheduling has to be included and how, especially for cipher instances claiming some related key security (contrary to this work), but also to prevent invariant subspace attacks, is another interesting question.

Acknowledgements François-Xavier Standaert Standaert is a research associate of the Belgian Fund for Scientific Research (FNRS-F.R.S.). This work has been funded in parts by the Brussels Region INNOVIRIS project SCAUT, and by the European Commission through the ERC Project 280141 (CRASH).

Appendix: Specifications of Mysterion's components

Mysterion S-box

Algorithm 4 Class13 S-box, bitslice representation

Require: 4 input bits (A, B, C, D)
Ensure: 4 output bits such as $(a, b, c, d) = \mathsf{S}(A, B, C, D)$
1: $a = A\&B$;
2: $a = a \oplus C$;
3: $c = B \mid C$;
4: $c = c \oplus D$;
5: $d = a\&D$;
6: $d = d \oplus A$;
7: $b = c\&A$;
8: $b = b \oplus B$;
9: **return** (a, b, c, d)

Mysterion L-box

$$C = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & \alpha^3 & \alpha^4 & \alpha^{12} & \alpha^8 & \alpha^{12} & \alpha^4 & \alpha^3 \end{pmatrix}, C^8 = \begin{pmatrix} 1 & \alpha^3 & \alpha^4 & \alpha^{12} & \alpha^8 & \alpha^{12} & \alpha^4 & \alpha^3 \\ \alpha^3 & \alpha^{13} & \alpha^4 & \alpha & 1 & \alpha^2 & \alpha^2 & \alpha^{12} \\ \alpha^{12} & \alpha^{14} & \alpha^{12} & \alpha^{14} & \alpha^2 & \alpha^7 & \alpha^5 & \alpha^8 \\ \alpha^8 & 1 & \alpha^5 & \alpha^{14} & \alpha^7 & \alpha & \alpha^2 & \alpha^3 \\ \alpha^3 & \alpha^{14} & \alpha^9 & \alpha^{10} & \alpha^{10} & \alpha^9 & \alpha^{14} & \alpha^3 \\ \alpha^3 & \alpha^2 & \alpha & \alpha^7 & \alpha^{14} & \alpha^5 & 1 & \alpha^8 \\ \alpha^8 & \alpha^5 & \alpha^7 & \alpha^2 & \alpha^{14} & \alpha^{12} & \alpha^{14} & \alpha^{12} \\ \alpha^{12} & \alpha^2 & \alpha^2 & 1 & \alpha & \alpha^4 & \alpha^{13} & \alpha^3 \end{pmatrix}.$$

C is the companion matrix of the polynomial defined in Sect. 4.1. C^8 is the underlying matrix of the Mysterion L-box.

ShiftColumns of Mysterion-256

Proofs for the bound of the number of active S-boxes

Proof of Theorem 2 The internal state of Mysterion-256 can be seen as a square, since the number of blocks is equal to the number of columns in a block in this case. Therefore, the proof directly results from the Four-Round Propagation Theorem of the AES Rijndael given in [10]. That is, the number of active S-boxes over four rounds of Mysterion-256 is lower bounded by the square of the branch number of the L-box, which corresponds to $9^2 = 81$ active S-boxes (See Fig. 4). □

Proof of Theorem 1 Contrary to Mysterion-256, we cannot directly use the Four-Round Propagation Theorem of the AES to lower bound the number of active S-boxes in Mysterion-128, since its number of columns is larger than its number of blocks (*i.e.* the state is no longer a square). However, similar bounds can be deduced from a modified version of the theorem proven in [10]. We show in the following how Mysterion-128 can fulfill the hypotheses of this theorem with a simple rearrangement of its operations. For this purpose, we first need to set some definitions and notations. First, a bundle is a 4-bit word and corresponds to a column in the representation of the internal state of Mysterion-128. We denote by $\mathcal{L}$ the application of the L-box on each block of the state, which are divided into four independent parts of eight bundles each. We call this partition of the bundles Ξ. Mysterion-128 is a key-alternating block cipher and iteratively applies the same round function, composed of an S-box layer, an L-box layer, a ShiftColumns layer, and key and round constant additions. As the latter do not influence the number of active S-boxes, we will omit them in the following. Based on this, four rounds of Mysterion-128 can then be written as:

$$\begin{aligned}&\mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{S} \circ \mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{S} \circ \mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{S} \circ \\ &\quad \mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{S}.\end{aligned}$$

We next reorganise these operations in order to highlight a particular structure of the linear transformation for 4 rounds, which allows a simpler analysis. More precisely, since ShiftColumns commutes with S, we have the following equivalent definition of four rounds of Mysterion-128:

$$\begin{aligned}&\mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{ShiftColumns} \circ \mathsf{S} \circ \mathcal{L} \circ \mathsf{S} \circ \\ &\quad \mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{ShiftColumns} \circ \mathsf{S} \circ \mathcal{L} \circ \mathsf{S}.\end{aligned}$$

Thanks to this representation, we easily identify two different transformations $\tau^a = \mathcal{L} \circ S$ and $\tau^b = \mathbf{L} \circ S$, where $\mathbf{L} = \mathsf{ShiftColumns} \circ \mathcal{L} \circ \mathsf{ShiftColumns}$. Then four rounds of Mysterion-128 are the alternation of τ^a and τ^b:

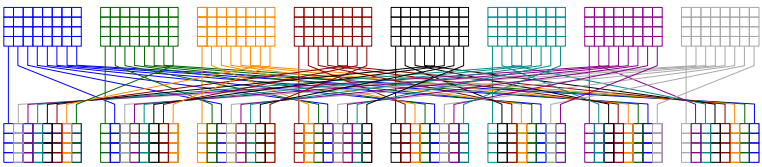

Fig. 4 ShiftColumns of Mysterion-256

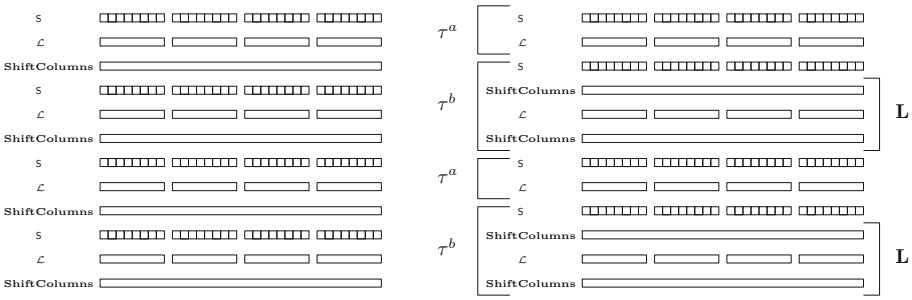

Fig. 5 Two equivalent representations of four rounds of Mysterion-128

$$\tau^a \circ \tau^b \circ \tau^a \circ \tau^b.$$

Figure 5 summarizes our notations and modified representation of Mysterion-128. □

We finally exploit the following theorem from [10]:

Theorem 3 *For a key alternating block cipher with round tranformations τ^a and τ^b, the number of active S-boxes of any trail over*

$$\tau^b \circ \tau^a \circ \tau^b \circ \tau^a$$

is lower bounded by $\mathcal{B}(\mathcal{L}) \times \mathcal{B}(\mathbf{L}, \Xi)$, where $\mathcal{B}(\mathcal{L})$ is the branch number of the linear transformation $\mathcal{L}$ and $\mathcal{B}(\mathbf{L}, \Xi)$ is the branch number of the linear transformation $\mathbf{L}$ with respect to the partition of the bundles Ξ.

The branch number of $\mathcal{L}$ is 9 (the L-box of Mysterion is an MDS code $[16, 8, 9]_{\mathbb{F}_{24}}$). The partition Ξ divide the state into the 4 blocks. We say a block is active when it has at least one active (*i.e* non zero) bundle. As the ShiftColumns operation spreads two bundles of each block into other blocks, and as $\mathcal{L}$ is MDS, we have that the minimum number of input/output active blocks, therefore the branch number of $\mathbf{L}$ with respect to the partition Ξ, is 5. As a result, the number of active S-boxes over four rounds is lower bounded by $9 \times 5 = 45$.

References

1. Albrecht M.R., Driessen B., Kavun, E.B., Leander G., Paar C., Yalçin T.: Proceedings on Block ciphers—focus on the linear layer (feat. PRIDE) Part I. In: Garay J.A., Gennaro R. (eds.) Advances in Cryptology—CRYPTO 2014—34th Annual Cryptology Conference, Santa Barbara, 17–21 Aug, 2014. Lecture Notes in Computer Science, vol. 8616, pp. 57–76. Springer, Berlin (2014).
2. Albrecht M.R., Rechberger C., Schneider T., Tiessen T., Zohner M. Ciphers for MPC and FHE. In: Oswald, E., Fischlin M (eds.): Proceedings on Advances in Cryptology—EUROCRYPT 2015—34th Annual International Conference on the Theory and Applications of Cryptographic Techniques Part I, Sofia, 26–30 Apr 2015, pp. 430–454. Lecture Notes in Computer Science, vol. 9056. Springer, Berlin (2015)
3. Augot D., Finiasz M.: Direct construction of recursive MDS diffusion layers using shortened BCH codes. In: Cid C., Rechberger C. (eds.): Fast Software Encryption-21st International Workshop, FSE 2014, 3–5 London, 2014, Revised Selected Papers, pp. 3–17. Lecture Notes in Computer Science, vol. 8540, Springer, Berlin (2015).
4. Biryukov A., De Cannière C.: Block ciphers and systems of quadratic equations. In: Johansson T. (ed.) Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, 24–26 Feb, 2003, Revised Papers. Lecture Notes in Computer Science, vol. 2887, pp. 274–289. Springer, Berlin (2003).

5. Bogdanov A., Knudsen L.R., Leander G., Standaert F.-X., Steinberger, J.P., Tischhauser E.: Key-alternating ciphers in a provable setting: encryption using a small number of public permutations—(extended abstract). In: Pointcheval D., Johansson T. (eds.) Proceedings on Advances in Cryptology—EUROCRYPT 2012—31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, 15–19 Apr 2012. Lecture Notes in Computer Science, vol. 7237, pp. 45–62. Springer, Berlin (2012).
6. Boura C., Canteaut A.: Zero-sum distinguishers for iterated permutations and application to Keccak-f and Hamsi-256. In: Biryukov A., Gong G., Stinson D.R. (eds.) Selected Areas in Cryptography—17th International Workshop, SAC 2010, Waterloo, 12–13 Aug 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6544, pp. 1–17. Springer, Berlin (2010).
7. Boura C., Canteaut A. De Cannière C.: Higher-order differential properties of Keccak and Luffa. In: Joux A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 252–269. Springer, Berlin (2011).
8. Chari S., Jutla C.S., Rao J.R., Rohatgi P.: Towards sound approaches to counteract power-analysis attacks. In: Wiener M.J. (ed.) Proceedings on Advances in Cryptology—CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, 15–19 Aug 1999. Lecture Notes in Computer Science, vol. 1666, pp. 398–412. Springer, Berlin (1999).
9. Courtois N., Pieprzyk J.: Cryptanalysis of block ciphers with overdefined systems of equations. In: Zheng Y. (ed.) Proceedings on Advances in Cryptology—ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, 1–5 Dec 2002. Lecture Notes in Computer Science, vol. 2501, pp. 267–287. Springer, Berlin (2002).
10. Daemen J., Rijmen V.: The wide trail design strategy. In: Honary B. (ed.) Proceedings on Cryptography and Coding, 8th IMA International Conference, Cirencester, 17–19 Dec 2001. Lecture Notes in Computer Science, vol. 2260, pp. 222–238. Springer, Berlin (2001).
11. Daemen J., Rijmen V.: Information Security and Cryptography. The Design of Rijndael: AES—The Advanced Encryption Standard. Springer, Berlin (2002)
12. Daemen J., Knudsen L.R., Rijmen V.: The block cipher Square. In: Biham E. (ed.) Proceedings on Fast Software Encryption, 4th International Workshop, FSE '97, Haifa, 2–22 Jan 1997. Lecture Notes in Computer Science, vol. 1267, pp. 149–165. Springer, Berlin (1997).
13. Daemen J., Peeters M., Van Assche G., Rijmen V.: Nessie proposal: the block cipher NOEKEON. Nessie submission (2000). http://gro.noekeon.org/.
14. Dinur I., Shamir A.: Cube attacks on tweakable black box polynomials. In: Joux A. (ed.) Proceedings on Advances in Cryptology—EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, 26–30 Apr 2009. Lecture Notes in Computer Science, vol. 5479, pp. 278–299, Springer, Berlin (2009).
15. Galice S., Minier M.: Improving integral attacks against Rijndael-256 up to 9 rounds. In: Vaudenay S. (ed.) Proceedings on Progress in Cryptology—AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, 11–14 Jun 2008. Lecture Notes in Computer Science, vol. 5023, pp. 1–15. Springer, Berlin (2008).
16. Gérard B., Grosso V., Naya-Plasencia M., Standaert, F.-X.: Block ciphers that are easier to mask: How far can we go? In: Bertoni G., Coron J.-S. (ed.) Proceedings on Cryptographic Hardware and Embedded Systems—CHES 2013—15th International Workshop, Santa Barbara, 20–23 Aug 2013. Lecture Notes in Computer Science, vol. 8086, pp. 383–399. Springer, Berlin (2013).
17. Gilbert H., Peyrin T.: Super-sbox cryptanalysis: improved attacks for AES-like permutations. In: Hong S., Iwata T. (eds.) Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, 7–10 Feb 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6147, pp. 365–383. Springer, Berlin (2010).
18. Grosso V., Leurent G., Standaert F.-X., Varici K.: LS-designs: Bitslice encryption for efficient masked software implementations. In: Cid C., Rechberger C. (eds.): Fast Software Encryption—21st International Workshop, FSE 2014, 3–5 London, 2014, Revised Selected Papers, pp. 18–37. Lecture Notes in Computer Science, vol. 8540, Springer, Berlin (2015).
19. Guo J., Peyrin T., Poschmann A. Robshaw M.: The LED block cipher. In: Preneel B., Takagi T. (eds.) Proceedings on Cryptographic Hardware and Embedded Systems—CHES 2011—13th International Workshop, Nara, Sep 28–1 Oct 2011. Lecture Notes in Computer Science, vol. 6917, pp. 326–341. Springer, Berlin (2011).
20. Knudsen L.R.: Truncated and higher order differentials. In: Preneel, B. (ed.) Proceedings on Fast Software Encryption: Second International Workshop, Leuven, 14–16 Dec 1994. Lecture Notes in Computer Science, vol. 1008, pp. 196–211. Springer, Berlin (1994).
21. Knudsen L.R., Leander G., Poschmann A., Robshaw M.J.B.: Printcipher: a block cipher for ic-printing. In: Mangard S., Standaert F.-X. (eds.): Proceedings on Cryptographic Hardware and Embedded Systems,

CHES 2010, 12th International Workshop, Santa Barbara, 17–20 Aug 2010, pp. 16–32. Lecture Notes in Computer Science, vol. 6225, Springer, Berin (2010).
22. Knudsen L.R., Wagner D.: Integral cryptanalysis. In: Daemen J., Rijmen V. (eds.) Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, 4–6 Feb 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2365, pp. 112–127. Springer, Berlin (2002).
23. Leander G., Abdelraheem M.A., AlKhzaimi H., Zenner E.: A cryptanalysis of PRINTcipher: the invariant subspace attack. In: Rogaway P. (ed.) Proceedings on Advances in Cryptology—CRYPTO 2011—31st Annual Cryptology Conference, Santa Barbara, 14–18 Aug 2011. Lecture Notes in Computer Science, vol. 6841, pp. 206–221. Springer, Berlin (2011).
24. Leander G., Minaud B, Rønjom S.: A generic approach to invariant subspace attacks cryptanalysis of Robin, iSCREAM and Zorro. To Appear in the Proceedings of EUROCRYPT 2015 (2015).
25. Leander G., Minaud B., Rønjom S.: A generic approach to invariant subspace attacks: cryptanalysis of Robin, iSCREAM and zorro. In: swald, E., Fischlin M (eds.): Proceedings on Advances in Cryptology—EUROCRYPT 2015—34th Annual International Conference on the Theory and Applications of Cryptographic Techniques Part I, Sofia, 26–30 Apr 2015, pp. 254–283. Lecture Notes in Computer Science, vol. 9056, Springer, Berlin (2015).
26. Minier M., Phan R.C.-W., Pousse B.: On integral distinguishers of Rijndael family of ciphers. Cryptologia **36**(2), 104–118 (2012).
27. Piret G., Roche T., Carlet C.: PICARO—a block cipher allowing efficient higher-order side-channel resistance—extended version-IACR Cryptology ePrint Archive 2012, 358 (2012)
28. Rivain M., Prouff E.: Provably secure higher-order masking of AES. In: Mangard S., Standaert F.-X. (eds.): Proceedings on Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, 17–20 Aug 2010, pp. 413–427. Lecture Notes in Computer Science, vol. 6225, Springer, Berin (2010).
29. Todo Y.: Structural evaluation by generalized integral property. In: Oswald, E., Fischlin M (eds.): Proceedings on Advances in Cryptology—EUROCRYPT 2015—34th Annual International Conference on the Theory and Applications of Cryptographic Techniques Part I, Sofia, 26–30 Apr 2015, pp. 287–314. Lecture Notes in Computer Science, vol. 9056, Springer, Berlin (2015).
30. Ullrich M., De Cannière C., Indesteege S., Küçük Ö., Mouha N., Preneel B.: Finding optimal bitsliced implementations of 4×4-bit s-boxes. In: SKEW 2011 Symmetric Key Encryption Workshop, Copenhagen, pp. 16–17 (2011)
31. Wagner D.: The boomerang attack. In: Lars R.K. (ed.) Proceedings on Fast Software Encryption, 6th International Workshop, FSE '99, Rome, 24–26 Mar 1999. Lecture Notes in Computer Science, vol. 1636, pp. 156–170. Springer, Berlin (1999).