On Self-Equivalence Encodings in White-Box Implementations

Adrián Ranea[1] and Bart Preneel[2]

imec-COSIC, KU Leuven, Belgium
firstname.lastname@esat.kuleuven.be

Abstract. All academic methods to secure software implementations of block ciphers against adversaries with full control of the device have been broken. Despite the huge progress in the cryptanalysis of these white-box implementations, no recent progress has been made on the design side. Most of the white-box designs follow the CEJO framework, where each round is encoded by composing it with small random permutations. While several generic attacks have been proposed on the CEJO framework, no generic analysis has been performed on self-equivalence encodings, a different design where only the affine layer of each round is encoded with random self-equivalences of the S-box layer, that is, affine permutations commuting with the non-linear layer.
In this work, we analyse the security of white-box implementations based on self-equivalence encodings for a broad class of SPN ciphers. First, we characterize the self-equivalence groups of S-box layers, and we prove that all the self-equivalences of a cryptographically strong S-box layer have a diagonal shape. Then, we propose the first generic attack on self-equivalence encodings. Our attack, based on affine equivalence problems, identifies the connection between the security of self-equivalence encodings and the self-equivalence structure of the cipher components. While we show that traditional SPN ciphers with cryptographically strong S-box layers cannot be secured with self-equivalence encodings, our analysis shows that self-equivalence encodings resist the generic attack if the cipher components satisfy several conditions, revealing the potential of self-equivalence encodings to secure other types of ciphers.

Keywords: White-box cryptography · Self-Equivalence · SPN

1 Introduction

Traditional black-box cryptography assumes that the endpoints of the communication are secured. However, in some real-word scenarios, the adversaries can access the endpoints and tamper with the cryptographic device. While several hardware-based countermeasures have been proposed to preclude grey-box attacks, such as side-channel and fault attacks, some applications (e.g., DRM or mobile banking) demand software-only solutions against adversaries with full control of the cryptographic device.

To model this powerful adversary from a cryptographic point of view, Chow et al. proposed the white-box model [15], where the adversary has full control of the cryptographic device. In particular, the adversary can observe and modify at will the intermediate values in the execution of the cryptographic algorithm.

In the same seminal work, Chow et al. proposed the first white-box implementation of a block cipher, a software implementation of AES designed to resist key-extraction attacks in the white-box model. Although their method only focused on key-extraction resistance, omitting other white-box threats such as code lifting orattacks inverting the functionality, it was the first practical approach considering a cryptographic adversary, as opposed to other heuristic countermeasures such as software obfuscation or software tamper-resistance.

The method by Chow et al., also called the CEJO framework, represents the cryptographic computation as a network of look-up tables and randomizes each look-up table by composing it with random encodings. To preserve the functionality, the output encoding of one step cancels the input encoding of the next step. Thus, the resulting white-box implementation is functionally equivalent to the original cipher up to the first and last encodings, which are not cancelled. Without these external encodings, the method is trivially insecure.

The white-box implementation by Chow et al. was broken by the BGE attack, and all the subsequent CEJO implementations [14, 31, 38, 26, 32, 36, 3] have also been broken [25, 37, 23, 34, 17, 28, 3, 13, 18, 30]. Lastly, McMillion et al. considered a white-box implementation of AES based on self-equivalences encodings and showed it was insecure [33]. As opposed to CEJO implementations, self-equivalence encodings randomize only the affine part of each round, by composing with random self-equivalences of the S-box layer SL, that is, affine permutations (A, B) such that $SL = B \circ SL \circ A$.

Due to the complexity of designing secure white-box implementations and the practical limitations introduced by external encodings, several commercial white-box implementations have been designed without external encodings but whose security relies on the secrecy of their design. Apart from the drawbacks of security by obscurity, several white-box attacks based on grey-box techniques have been proposed that do not require knowledge of the design, such as [10, 5, 35]. The effectiveness of these attacks was shown in the WhibOx Contest [12], a competition where participants submitted white-box implementations of AES and broke other participants' submissions. In the first edition all the submissions were broken, and in the second edition 3 implementations stayed unbroken during the attack period of the competition, but they were broken afterwards [24].

Apart from white-box implementations of existing block ciphers, other white-box constructions have been published, such as white-box implementations of ciphers with secret S-boxes [11, 4] or incompressible white-box ciphers [7, 9]. The latter constructions are dedicated ciphers designed to preclude code lifting attacks by relying on a huge implementation such that an adversary with partial access cannot find an equivalent implementation with significantly less size.

In this work we address the challenging problem of designing white-box implementations of block ciphers in the white-box model proposed by Chow

et al., that is, where the specifications of the design are public and the security relies on the secrecy of the internal and external encodings. While there has been outstanding progress in adapting grey-box techniques to attack white-box implementations relying on secret designs, these techniques do not pose a threat to white-box implementations relying on external encodings yet, as long as the external encodings are chosen carefully [2]. Thus, grey-box techniques are outside the scope of this paper, and we focus instead on the design side of white-box implementations, for which no significant progress has been made recently.

Apart from the CEJO implementations and the white-box AES implementation based on self-equivalence encodings, no more white-box implementations following this model have been published. Some generic attacks have been proposed on the CEJO framework [34, 3, 18], showing that CEJO implementations are not suitable for a broad class of SPN ciphers. However, self-equivalence encodings have only been considered for AES, and the security of this type of encodings has not been analysed for other ciphers.

Contributions. In this paper, we analyse the security of self-equivalence encodings in white-box implementations of SPN ciphers. We first formalize self-equivalence implementations, a class of white-box implementations that hide the round key material in random-looking affine permutations built from self-equivalences of the S-box layer. We also illustrate how CEJO implementations can be efficiently transformed into self-equivalence implementations.

Then, we study the self-equivalence group of an S-box layer, on which the security of self-equivalence encodings is based. To this end, we introduce diagonal self-equivalences; given a permutation F built from the concatenation of smaller permutations, a diagonal self-equivalence of F is a self-equivalence that can also be decomposed as the concatenation of smaller permutations. We prove that in order to have non-diagonal self-equivalences, F needs to have additive self-equivalences, linear components and several linear structures, or equivalently, differential and linear approximations of probability one. As a result, S-box layers of SPN ciphers employing cryptographically strong S-boxes have only diagonal self-equivalences. Our characterization of diagonal self-equivalences can be of independent interest. For example, it can be applied to count the number of solutions (A, B) of an affine equivalence problem $G = B \circ F \circ A$ where the central map F is given as the concatenation of smaller permutations.

Finally, we propose the first generic attack on self-equivalence implementations of SPN ciphers with cryptographically strong S-boxes. Our attack partially recovers the self-equivalence encodings up to some unknown affine permutations belonging to a small subgroup of the self-equivalence group; the key is then recovered by a brute force search over the small subgroup. The attack is based on affine and linear equivalence problems and its complexity depends on the self-equivalence groups of the linear layer blocks and the S-boxes. Our analysis shows that if these self-equivalence groups satisfy some properties, a self-equivalence implementation is secure against the generic attack. While traditional SPN ciphers such as AES do not satisfy these properties, our analysis provides the foundations to secure a different class of ciphers with self-equivalence encodings.

Outline. In Sect. 2, the notation and the preliminaries are introduced. The CEJO framework is described Sect. 3 and self-equivalence implementations are introduced in Sect. 4. Diagonal self-equivalences are characterized in Sect. 5, and the generic attack on self-equivalence encodings is described in Sect. 6. Section 7 presents the conclusions and the future work.

2 Preliminaries

2.1 Basics

In this document, capital letters are used for functions (e.g., F, G), lower letters for values (e.g., x, y), and calligraphic letters for sets of functions (e.g., $\mathcal{F}, \mathcal{G}$).

Let $\mathbb{F}_q$ be the finite field with q elements. The vector space of n-bit values is denoted by $\mathbb{F}_2^n$, and the addition in $\mathbb{F}_2^n$ by $\oplus$. A function $F : \mathbb{F}_2^n \mapsto \mathbb{F}_2^m$ is called an (n, m)-bit function, or just n-bit function if $n = m$. Given two functions F and G, their composition is denoted by $F \circ G$ and their concatenation by $F \parallel G$, that is, $(F \parallel G)(x, y) = (F(x), G(y))$. The n-bit identity function is denoted by $\mathrm{Id}_n(x) = x$, and the addition by a constant is denoted by $\oplus_a(x) = x \oplus a$.

Given an affine function A, we denote its linear part by $\mathrm{Lin}(A)$, that is, $A = \oplus_a \circ \mathrm{Lin}(A)$ for some constant a. If $L : (\mathbb{F}_2^n)^m \to (\mathbb{F}_2^n)^m$ is a linear function, we denote by $\{L_{i,j} : 1 \le i, j \le m\}$ the n-bit linear functions associated with the blocks of L seen as a block matrix, that is, in matrix notation

$$L \times \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} = \begin{pmatrix} L_{1,1} & \cdots & L_{1,m} \\ \vdots & \ddots & \\ L_{m,1} & & L_{m,m} \end{pmatrix} \times \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} .$$

The general linear group consisting of all n-bit linear permutations is denoted by GL_n, and the affine general linear group of $\mathbb{F}_2^n$ is denoted by AGL_n.

The cardinality of a set $\mathcal{F}$ is denoted by $|\mathcal{F}|$ and the Cartesian product of sets $\mathcal{F}$ and $\mathcal{G}$ is denoted by $\mathcal{F} \times \mathcal{G}$. We translate the inversion, concatenation and composition of functions to set operations as follows:

$$\begin{aligned} \mathcal{F}^{-1} &= \{F^{-1} : \ F \in \mathcal{F}\} , \\ \mathcal{F} \parallel \mathcal{G} &= \{F \parallel G : \ F \in \mathcal{F}, G \in \mathcal{G}\} , \\ \mathcal{F} \circ \mathcal{G} &= \{F \circ G : \ F \in \mathcal{F}, G \in \mathcal{G}\} . \end{aligned}$$

Moreover, given a set of affine permutations $\mathcal{A}$, we denote its linear part by $\mathrm{Lin}(\mathcal{A}) = \{\mathrm{Lin}(A) : A \in \mathcal{A}\}$. When a set contains a single element $\{F\}$, we denote $F \circ \mathcal{G} = \{F\} \circ \mathcal{G}$ and $F \parallel \mathcal{G} = \{F\} \parallel \mathcal{G}$ with a slight abuse of notation.

SPN ciphers. Given an n-bit iterated block cipher, we denote the encryption function for a fixed key k by $E_k = E^{(n_r)}_{k^{(n_r)}} \circ \cdots \circ E^{(1)}_{k^{(1)}}$, where $E^{(r)}_{k^{(r)}}$ denotes the rth round function and $k^{(r)}$ denotes the rth round key. For ease of notation,

we omit the round-key subscript of the round functions. An SPN (Substitution-Permutation Network) cipher is an iterated block cipher where the rth round function is defined by

$$E^{(r)} = LL \circ SL \circ \oplus_{k^{(r)}} ,$$

where LL is the n-bit linear layer, $SL = S_1 \parallel \cdots \parallel S_\rho$ is the S-box layer composed of m-bit S-boxes S_i, and $k^{(r)}$ is the rth round key. In some cases, the first and last rounds are defined in a different way. Figure 1 depicts the round function of an SPN cipher.

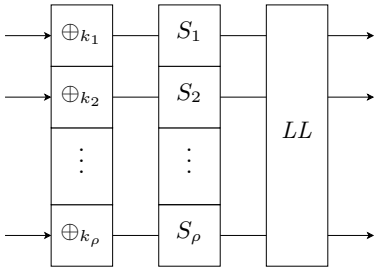

Fig. 1. The round function of an SPN cipher.

2.2 Self-Equivalences

In this section, we introduce the notion of self-equivalence [8] and some of its properties for n-bit functions.

Definition 1. *Let F be an n-bit function. A pair of n-bit affine permutations (A, B) such that $F = B \circ F \circ A$ is called an (affine) self-equivalence of F.*

We also say A (resp. B) is a right (resp. left) self-equivalence of F. If A and B are linear, (A, B) is called a linear self-equivalence, and if $(A, B) = (\oplus_a, \oplus_b)$ are constants, (A, B) is called an additive self-equivalence.

Definition 2. *Two n-bit functions F and G are said to be affine (resp. linear) equivalent if there exists a pair of n-bit affine (resp. linear) permutations (A, B) such that $G = B \circ F \circ A$.*

The notion of (affine) self-equivalence arises from the affine equivalence relation. This equivalence relation induces a partition on the set of n-bit functions, where a function belongs to the affine class containing all its affine equivalent functions.

We denote the set of self-equivalences of F by $\mathrm{SE}(F)$, the set of left self-equivalences by $\mathrm{LSE}(F)$, and the set of right self-equivalences by $\mathrm{RSE}(F)$, i.e.,

$$\begin{aligned}
\mathrm{SE}(F) &= \{(A, B) \in \mathrm{AGL}_n \times \mathrm{AGL}_n : F = B \circ F \circ A\} , \\
\mathrm{RSE}(F) &= \{A : \exists B \text{ s.t. } (A, B) \in \mathrm{SE}(F)\} , \\
\mathrm{LSE}(F) &= \{B : \exists A \text{ s.t. } (A, B) \in \mathrm{SE}(F)\} .
\end{aligned}$$

The next lemma recalls the well-known fact that the self-equivalences form a group, which can be easily proved by noticing that $\mathrm{SE}(F)$ is the stabilizer of a group action where $\mathrm{AGL}_n \times \mathrm{AGL}_n$ acts over the set of n-bit functions [21, 29].

Lemma 1. *$\mathrm{SE}(F)$ is a subgroup of $\mathrm{AGL}_n \times \mathrm{AGL}_n$, and $\mathrm{LSE}(F)$ and $\mathrm{RSE}(F)$ are subgroups of AGL_n. Moreover, if G is a function affine equivalent to F, i.e., $G = B \circ F \circ A$, then the self-equivalence groups of F and G are conjugates, that is, $\mathrm{RSE}(G) = A^{-1} \circ \mathrm{RSE}(F) \circ A$ and $\mathrm{LSE}(G) = B \circ \mathrm{LSE}(F) \circ B^{-1}$.*

If F is a permutation, each right self-equivalence corresponds to a unique left self-equivalence. In this case, $\mathrm{SE}(F), \mathrm{LSE}(F)$ and $\mathrm{RSE}(F)$ have the same cardinality, which is a divisor of the number of affine permutations. In particular, $|\mathrm{SE}(F)| = |\mathrm{AGL}_n|$ if F is an n-bit affine permutation. For a non-invertible function, a right self-equivalence can correspond to several left self-equivalences, and vice-versa. If F is non-invertible but affine, the cardinality of $\mathrm{SE}(F)$ can be computed, which depends on the rank of its matrix representation over $\mathbb{F}_2$ [29].

2.3 Affine Equivalence Problems

Self-equivalences are strongly related to the solutions of affine and linear equivalence problems, on which many white-box implementations are based.

Definition 3. *An affine (resp. linear) equivalence problem is a functional equation of the form $G = Y \circ F \circ X$ where F and G are known n-bit functions and (X, Y) are unknown affine (resp. linear) permutations.*

Given any solution (X_0, Y_0) of the affine equivalence problem $G = Y \circ F \circ X$, the solution set can be seen as the subgroup $\mathcal{S} = \{(A \circ X_0, Y_0 \circ B) : (A, B) \in \mathrm{SE}(F)\}$ of $\mathrm{AGL}_n \times \mathrm{AGL}_n$. Thus, the number of solutions is equal to the number of self-equivalences of F. If the unknowns (X, Y) are restricted to given subgroups $(\mathcal{A}, \mathcal{B})$, then the solution set can be seen as the subgroup

$$\mathcal{S} = \{(A \circ X_0, Y_0 \circ B) : (A, B) \in \mathrm{SE}(F) \cap (\mathcal{A} \times \mathcal{B})\}.$$

Several algorithms have been published for finding a solution of an equivalence problem. For n-bit permutations, Biryukov et al. proposed an $\mathcal{O}(n^3 2^n)$ algorithm to solve the linear case and an $\mathcal{O}(n^3 2^{2n})$ algorithm to solve the affine case [8], whereas Dinur recently proposed an $\mathcal{O}(n^3 2^n)$ algorithm to solve the affine case for random permutations [19]. Since we will mostly consider affine equivalence problems involving permutations with several self-equivalences, which is fairly rare in the random case, in this document we will assume $O(n^3 2^{2n})$ to be the complexity of solving an affine equivalence problem with n-bit permutations.

For n-bit functions $F = F_1 \parallel \cdots \parallel F_\rho$ composed of m-bit cryptographic S-boxes[1] F_i, Derbez et al. obtained an algorithm to solve the affine equivalence

[1] The algorithm by Derbez et al. assumes that the S-boxes do not have non-trivial linear components, i.e., there does not exist a non-zero $(m, 1)$-bit linear function B such that $(B \circ S)(x) = \sum b_i S(x)_i$ is a linear function. Most SPN ciphers employ cryptographically strong S-boxes satisfying this requirement.

problem with time complexity $O(2^m n^3 + n^4/m + 2^{2m} mn^2)$ [18]. For n-bit affine functions F, the affine equivalence problem can be represented as a linear system of $n+1$ equations and solved with Gaussian elimination in time $\mathcal{O}((n+1)^\omega)$, where $2 < \omega < 3$ is the matrix multiplication constant.

While several efficient algorithms have been proposed for finding *a* solution of an equivalence problem, no efficient techniques have been proposed to count its number of solutions. In general, the algorithm by Biryukov et al. can be applied to find all the solutions by repeating the algorithm for each possible initial guess. The work factor of this approach is at least 2^{2n} for the linear case and 2^{3n} for the affine case, and it is also lower bounded by the number of solutions.

A particular class of problems for which we can characterize the solution set is the class of univariate linear equivalence problem $G = X^{-1} \circ M_\alpha \circ X$ where M_α is the n-bit linear permutation corresponding to the multiplication by the finite field element $\alpha \in \mathbb{F}_{2^n}$. For an n-bit function F, let $\mathrm{C}(F) = \{A \in \mathrm{GL}_n : A \circ F = F \circ A\}$ be the centralizer of F, that is, the subgroup of linear permutations commuting with F. Then, the solution set of $G = X^{-1} \circ M_\alpha \circ X$ is $\mathcal{S} = \mathrm{C}(M_\alpha) \circ X_0$, for any particular solution X_0. If α has multiplicative order d (i.e., the minimum exponent such that $\alpha^d = 1$), then $\mathrm{C}(M_\alpha)$ is isomorphic to the set of invertible $\frac{n}{d} \times \frac{n}{d}$ matrices over $\mathbb{F}_{2^d}$ [22]. In particular, $\mathrm{C}(M_\alpha) = \{M_\gamma : 0 \neq \gamma \in \mathbb{F}_{2^n}\}$ if α is a primitive element.

3 CEJO Implementations

In this section we introduce CEJO implementations and the generic cryptanalysis of the CEJO framework. CEJO implementations are a particular class of white-box implementations based on encoded round functions.

Definition 4 ([15]). *Let F be an (n,m)-bit function and let (I,O) be a pair of n-bit and m-bit permutations, respectively. The function $\overline{F} = O \circ F \circ I$ is called an encoded F, and I and O are called the input and output encoding, respectively.*

Definition 5. *Let $E_k = E^{(n_r)} \circ \cdots E^{(1)}$ be the encryption[2] function of an iterated n-bit cipher with fixed key k. An encoded implementation of E_k is an encoded E_k composed of encoded round functions, that is,*

$$\overline{E_k} = \overline{E^{(n_r)}} \circ \cdots \circ \overline{E^{(1)}} = \left(O^{(n_r)} \circ E^{(r)} \circ I^{(n_r)}\right) \circ \cdots \circ \left(O^{(1)} \circ E^{(1)} \circ I^{(1)}\right),$$

for some n-bit permutations $(I^{(r)}, O^{(r)})$ satisfying $I^{(r+1)} = \left(O^{(r)}\right)^{-1}$.

In other words, an encoded implementation is the composition of encoded round functions where the intermediate encodings are cancelled out, that is, $\overline{E_k} = O^{(n_r)} \circ E_k \circ I^{(1)}$. The intermediate encodings are also called the round encodings, and the encodings $(I^{(1)}, O^{(n_r)})$ are called the external encodings. The

[2] In this paper we focus on white-box implementations of encryption functions; the definitions for decryption functions are similar.

round encodings are sampled at random from a group $\mathcal{E}$ herein called the round encoding space, which varies from implementation to implementation.

All the white-box implementations of block ciphers published in the literature are encoded implementations following the so-called CEJO framework [14]. CEJO implementations employ round encoding spaces composed of small non-linear permutations and wider linear functions.

Definition 6. *Let $\mathcal{N}_{m_n}$ be the set of m_n-bit non-linear permutations. A CEJO implementation with m_n-bit non-linear encodings and m_l-bit linear encodings is an encoded implementation of an SPN cipher with round encoding space*

$$\mathcal{E} = (\mathcal{N}_{m_n} \parallel \cdots \parallel \mathcal{N}_{m_n}) \circ (\mathrm{GL}_{m_l} \parallel \cdots \parallel \mathrm{GL}_{m_l}) .$$

Figure 2 depicts an CEJO encoded round function with the usual parameters $2m_n = m_l = m$, where m is the bit-size of the S-box. For example, the encoded AES implementation proposed by Chow et al. employs 4-bit non-linear encodings and 8-bit linear encodings.

Internally, a CEJO encoded round function is implemented in software as a network of look-up tables, requiring roughly $\mathcal{O}(2^{\max(2m_n, m_l, m)})$ bits of memory. Since white-box attacks to CEJO implementations only require black-box access to the encoded round functions, we will omit the internal description of a CEJO encoded round function and refer to [14, 3] for more details.

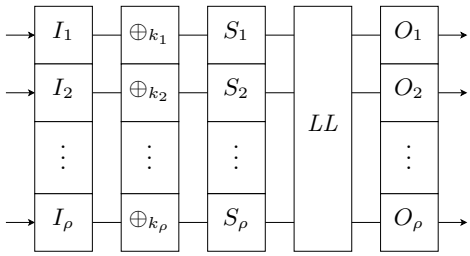

Fig. 2. A CEJO encoded round function with $\frac{m}{2}$-bit non-linear encodings and m-bit linear encodings. Each m-bit function I_i and O_i consists of the concatenation of two $\frac{m}{2}$-bit non-linear permutations composed with an m-bit linear permutation.

3.1 Security of CEJO Implementations

The white-box implementation by Chow et al. and the subsequent CEJO implementations were designed to prevent key-extraction attacks in the white-box model, that is, an attacker with full control on the implementation should not be able to recover the key; the specifications of the cipher and the implementation are public, but the key and the encodings are unknown to the adversary.

CEJO implementations do not aim to prevent attacks inverting the implementation. In some scenarios (e.g., DRM), inverting the functionality is not a threat.

Note that designing a white-box implementation preventing both key-extraction and inversion attacks is a much harder problem, since it would turn a symmetric cipher into a public-key encryption scheme.

Informally, the security of CEJO implementations relies on a disambiguation problem; for a given encoded round function, many pairs of round keys and round encodings result in the same encoded round function. While encoded round functions are individually secure, all CEJO implementations have been broken by analysing few consecutive rounds.

Key-extraction attacks on CEJO implementations typically consists of two main steps: reducing the round encoding space and guessing the round keys. First, the adversary obtains new encoded round functions with the same underlying round keys but with round encodings restricted to a smaller encoding space. Then, the adversary performs an exhaustive search over the reduced round encoding space to recover the round keys. For most SPN ciphers, the key schedule can be inverted and few round keys can uniquely determine the master key.

The crucial step is the reduction of the round encoding space. While several reduction attacks to CEJO implementations have been published, only three generic reduction attacks (i.e., considering a broad class of ciphers) have been proposed. Michiels et al. proposed the first reduction attack [34], composed of one algorithm to remove the non-linear part of the encodings and another algorithm to partially recover the remaining affine encodings. Baek et al. generalized the algorithm to remove the non-linear part of the encodings [3] and Derbez et al. improved the recovering of the affine encodings by an efficient algorithm to solve affine equivalence problems of S-box layers [18]. The following proposition illustrates the reach of current generic reduction attacks, proved in Appendix A.

Proposition 1. *Let E_k be the encryption function of an n-bit SPN cipher with linear layer LL and S-box layer SL consisting of m-bit cryptographic S-boxes, and let $\widetilde{E_k}$ be a CEJO implementation of E_k with m_n-bit non-linear encodings and m_l-bit linear encodings. Given black-box access to three consecutive encoded round functions $\widetilde{E^{(r-2)}}, \widetilde{E^{(r-1)}}$ and $\widetilde{E^{(r)}}$, one can find another encoded $E^{(r)}$ with round encoding space $LL \circ \mathrm{LSE}(SL)$, i.e.,*

$$\widehat{E^{(r)}} = \widehat{O} \circ E^{(r)} \circ \widehat{I}\,, \quad \widehat{I}, \widehat{O^{-1}} \in LL \circ \mathrm{LSE}(SL)\,,$$

in time $\mathcal{O}\left((n/m_n)2^{3m_n} + 2^m n^3 + n^4/m + 2^{2m}mn^2\right)$.

As a result, given a CEJO implementation of an SPN cipher, we can find another encoded implementation where the round encoding space has been reduced to $LL \circ \mathrm{LSE}(SL)$. This reduction attack is efficient for practical CEJO implementations, since the complexity of Proposition 1 is similar to the memory complexity of a CEJO implementation.

For white-box implementations of AES, several reduction attacks have been proposed that reduced the round encoding space to a single element [6, 28, 18]. However, no generic attack has been proposed reducing the round encoding space further than Proposition 1. This round encoding space is strongly related to the

self-equivalence set of the S-box layer. In the next section, we introduce a class of white-box implementations where the round encodings are sampled at random from the self-equivalences set of the S-box layer and analyse their security.

4 Self-Equivalence Implementations

In [33], McMillion et al. considered a white-box implementation of AES where the round encodings were self-equivalences of the S-box layer and showed it was insecure. However, this type of encodings has not been considered for other ciphers, nor their generic security have been addressed. In this section, we introduce and analyse self-equivalence implementations, a particular class of white-box implementations based on self-equivalence encodings.

As opposed to encoded implementations, self-equivalence implementations only encode the affine part of the round function. Given an encryption function $E_k = E^{(n_r)} \circ \cdots E^{(1)}$ of an SPN cipher with round function $E^{(r)} = LL \circ SL \circ \oplus_{k^{(r)}}$, we define the intermediate affine layers by

$$AL^{(r)} = \oplus_{k^{(r)}} \circ LL\,, \quad r = 2, \ldots, n_r$$

and the first and last layer affine layers by $AL^{(1)} = \oplus_{k^{(1)}}$ and $AL^{(n_r+1)} = LL$. Although affine layers depends on the round keys, we omit the round-key subscript for ease of notation. Note that the rth affine layer includes the round key of the rth round but the linear layer of the previous round. The first and last affine layers might be defined differently if the SPN cipher defines the first and last round in a different way.

Definition 7. *Let E_k be the encryption function of an SPN cipher. A self-equivalence implementation is an encoded E_k given by*

$$\overline{E_k} = AL^{(n_r+1)} \circ SL \circ \overline{AL^{(n_r)}} \circ SL \circ \overline{AL^{(n_r-1)}} \circ \cdots \circ \overline{AL^{(2)}} \circ SL \circ \overline{AL^{(1)}}\,,$$

where the intermediate encodings of the affine layers are self-equivalences of the S-box layer, i.e.,

$$\overline{AL^{(r)}} = O^{(r)} \circ AL^{(r)} \circ I^{(r)}\,, \quad (O^{(r)}, I^{(r+1)}) \in \mathrm{SE}(SL)\,,$$

and the external encodings $(I^{(1)}, O^{(n_r)})$ are affine permutations.

The last affine layer is not encoded since it is not key-dependent. The intermediate encodings are cancelled due to the self-equivalence property, i.e.,

$$\begin{aligned}\overline{AL^{(r+1)}} \circ SL \circ \overline{AL^{(r)}} &= O^{(r+1)} \circ AL^{(r+1)} \circ (I^{(r+1)} \circ SL \circ O^{(r)}) \circ AL^{(r)} \circ I^{(r)} \\ &= O^{(r+1)} \circ AL^{(r+1)} \circ SL \circ AL^{(r)} \circ I^{(r)}\,.\end{aligned}$$

Similar to encoded implementations, the intermediate encodings are called the round encodings, and the subgroup of self-equivalences from which the round encodings are sampled at random is called the round encoding space $\mathcal{E} \leq \mathrm{SE}(SL)$.

Self-equivalence implementations can be implemented in software in a simple and efficient way. An encoded affine layer can be implemented by a single $n \times n$ binary matrix and an n-bit constant, requiring $\mathcal{O}(n^2 + n)$ bits of memory and $\mathcal{O}(n^2/\log_2 n)$ bit operations to evaluate [1]. In addition, the S-box layer does not need to be protected, and there are no restrictions on the bit-size of the S-boxes.

A CEJO implementation can be efficiently transformed into a self-equivalence implementation following Proposition 1, and a self-equivalence implementation can be considered as a CEJO implementation with n-bit linear encodings by defining the encoded round function as

$$\overline{E^{(r)}} = SL \circ \overline{AL^{(r)}} = \left(LL \circ I^{(r+1)}\right)^{-1} \circ E^{(r)} \circ (LL \circ I^{(r)})$$

While the memory complexity of a CEJO implementation is exponential in terms of the bit-size of the encodings, self-equivalence implementations can be efficiently implemented in all cases. Therefore, practical CEJO implementations can be efficiently translated to self-equivalence implementations, but the latter ones cannot be efficiently transformed to practical CEJO implementations if the self-equivalences of the S-box layer are not composed of smaller affine permutations. In the next section, we study the self-equivalences of S-box layers and analyse the case when all the self-equivalences are composed of smaller permutations.

5 Diagonal Self-Equivalences

Let $F = F_1 \parallel \cdots \parallel F_\rho$ be an arbitrary n-bit permutation composed of smaller m-bit permutations F_i. If (A_i, B_i) is a self-equivalence of F_i for $i = 1, \ldots, \rho$, then $(A_1 \parallel \cdots \parallel A_\rho, B_1 \parallel \cdots \parallel B_\rho)$ is a self-equivalence of F. In addition, if two of the smaller permutations (F_i, F_j) are the same, then the transposition (i, j) is also a self-equivalence of the S-box layer.

Definition 8. *A $(m \times \rho)$-bit linear permutation P is called an $[m, \rho]$-block permutation if $P(x_1, \ldots, x_\rho) = (x_{\pi(1)}, \ldots, x_{\pi(\rho)})$ for some permutation π of $\{1, \ldots, \rho\}$. We denote the set of block permutations by $\mathcal{P}_{m,\rho}$.*

In particular, if all the small permutations F_i are the same, then the set

$$\mathcal{D} = \left\{ \left((A_1 \parallel \cdots \parallel A_\rho) \circ P, P^{-1} \circ (B_1 \parallel \cdots \parallel B_\rho)\right) : \begin{array}{c} (A_i, B_i) \in \mathrm{SE}(F_i) \\ P \in \mathcal{P}_{m,\rho} \end{array} \right\}$$

is a subgroup of $\mathrm{SE}(F)$. In some cases, the set $\mathcal{D}$ contains all the self-equivalences of F. For example, De Mulder et al. [17] computed all the linear self-equivalences of $S \parallel S$, where S is the AES S-box, and found that all the right (resp. left) linear self-equivalences have a diagonal shape of the form

$$\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}, \quad \begin{pmatrix} 0 & A_1 \\ A_2 & 0 \end{pmatrix},$$

where A_1 and A_2 are right (resp. left) linear self-equivalence of S.

Unfortunately, it is not known in which cases the subgroup $\mathcal{D}$ is the total self-equivalence group $\mathrm{SE}(F)$. To consider also functions $F = F_1 \circ \cdots \circ F_\rho$ with different permutations F_i, we define diagonal self-equivalences as follows.

Definition 9. *Let $F = F_1 \parallel \cdots \parallel F_\rho$ be an n-bit permutation composed of m-bit permutations F_i. An (affine) diagonal self-equivalence (A, B) of F is a pair of n-bit affine permutations $(A, B) \in \mathrm{SE}(F)$ such that*

$$A = (A_1 \parallel \cdots \parallel A_\rho) \circ P\,, \qquad B = P^{-1} \circ (B_1 \parallel \cdots \parallel B_\rho)\,,$$

for some m-bit functions A_i and B_i and some $[m, \rho]$-block permutation P.

The set of diagonal self-equivalences is a subgroup of $\mathrm{SE}(F)$, and it includes $\mathcal{D}$ when all F_i are the same. Moreover, if (A, B) is a diagonal self-equivalence, then A_i and B_i are permutations and the matrices corresponding to $\mathrm{Lin}(A)$ and $\mathrm{Lin}(B)$ are diagonal block matrices up to some permutation of the blocks, i.e.,

$$\mathrm{Lin}(A) = \begin{pmatrix} \mathrm{Lin}(A_1) & & \\ & \ddots & \\ & & \mathrm{Lin}(A_\rho) \end{pmatrix} \times P\,.$$

The property of having non-diagonal self-equivalences is not invariant in the affine class. However, if we consider the subgroup of affine permutations $\mathcal{Q} = \{Q_1 \parallel \cdots \parallel Q_\rho : Q_i \in \mathrm{AGL}_m\}$, then F has non-diagonal self-equivalences if and only if $Q' \circ F \circ Q$ has non-diagonal self-equivalences, for all $Q, Q' \in \mathcal{Q}$.

The main result of this section is Theorem 1, which shows that in order to have non-diagonal self-equivalences a function F must have additive self-equivalences, linear components and several linear structures. We first introduce these concepts before stating the theorem.

Given an (n, m)-bit function G, we denote by $(G_1, \ldots, G_m)$ the canonical or coordinate components of G, that is, $G(x) = (G_1(x), \ldots, G_m(x))$. The set of (Boolean) components of G can be defined as the set of n-bit Boolean functions given by the linear combinations of the canonical components of G, i.e.,

$$\left\{ \sum_{i=1}^{m} a_i G_i \mid (a_1, \ldots, a_m) \in \mathbb{F}_2^m \right\}.$$

Excluding the trivial component defined by $(a_1, \ldots, a_m) = (0, \ldots, 0)$, we say that r components $\sum_i a_i^{(1)} G_i, \ldots, \sum_i a_i^{(r)} G_i$ are linear independent if the $\mathbb{F}_2$-vectors $(a^{(1)}, \ldots, a^{(r)})$ are linear independent.

An additive self-equivalence of an (n, m)-bit function G is a pair of functions $(\oplus_a, \oplus_b)$, where a is an n-bit constant and b is an m-bit constant, such that $G = \oplus_b \circ G \circ \oplus_a$. For Boolean functions, this notation coincides with the notion of linear structure [20], while for vectorial Boolean functions the notion of linear structure is weaker.

Definition 10. *Let $G : \mathbb{F}_2^n \to \mathbb{F}_2$ be a Boolean function. An n-bit vector a is called a linear structure of G if there exists a bit b such that $\oplus_b \circ G = G \circ \oplus_a$.*

Definition 11. *Let G be an (n, m)-bit function. An n-bit vector a is a linear structure of G if a is a linear structure of some canonical component of G.*

Any (vectorial) Boolean function has the trivial linear structure $a = 0$ and the trivial additive self-equivalence $(\oplus_0, \oplus_0)$. We are now ready to state the main result of this section.

Theorem 1. *Let $F = F_1 \parallel \cdots \parallel F_\rho$ be an n-bit permutation, where each F_i is an m-bit permutation. If F has a non-diagonal self-equivalence, then F contains three permutations (F_i, F_j, F_h), $j \neq h$, such that*

(a) F_i has a non-trivial additive self-equivalence.
(b) F_j has a non-trivial linear component.
(c) F_h has 2^{m-r} linear structures that are common to r linear independent components, for some $0 < r < m$.

The proof of Theorem 1 is presented in Appendix B. Note that if $F_i = F_h$, then the condition (c) is redundant since (a) would imply (c). However, if $F_i = F_j$ neither (a) implies (b) nor vice-versa.

We checked for each 4-bit affine class $\mathcal{C}_i$ whether $\mathcal{C}_i \parallel \mathcal{C}_i$ has a non-diagonal self-equivalence. Using the list of 4-bit affine classes by De Cannière [16], we found that only the last 8 classes $\mathcal{C}_{293}, \mathcal{C}_{294}, \ldots, \mathcal{C}_{301}$ have non-diagonal self-equivalences when concatenated with themselves. These classes are actually the only ones that have both additive self-equivalences and linear components, showing that the necessary conditions of Theorem 1 are also sufficient for this case.

Cryptographically strong S-boxes do not have non-trivial linear components or additive self-equivalences, since they correspond to linear approximations and differentials with probability one, respectively. Most SPN ciphers employ S-box layers SL given by the concatenation of a single cryptographically strong S-box S, and for this case we can characterize the self-equivalence group of SL as follows.

Corollary 1. *Let S be an m-bit permutation without non-trivial additive self-equivalences or linear components, and consider the $(m \times \rho)$-bit permutation $SL = S \circ \cdots \circ S$. Then, the self-equivalence group of SL is given by*

$$\mathrm{SE}(SL) = \left\{ \big((A_1 \parallel \cdots \parallel A_\rho) \circ P, P^{-1} \circ (B_1 \parallel \cdots \parallel B_\rho)\big) : \begin{array}{c} (A_i, B_i) \in \mathrm{SE}(S) \\ P \in \mathcal{P}_{m,\rho} \end{array} \right\}.$$

In addition, $|\mathrm{SE}(SL)| = \rho! \times |\mathrm{SE}(S)|^\rho$.

While we apply the study of diagonal self-equivalences and Theorem 1 for the security analysis of self-equivalence implementations, they can also be of independent interest. For example, Theorem 1 can be used to characterize and count the number of solutions of affine equivalence problems $G = Y \circ F \circ X$ where F is given by the concatenation of cryptographically strong S-boxes.

6 Security of Self-Equivalence Implementations

In this section, we analyse the security of self-equivalence implementations against key-extraction attacks in the white-box model. Similar to CEJO implementations, self-equivalence implementations do not aim to prevent attacks inverting the implementation. Moreover, it is assumed that the specifications of the block cipher and the self-equivalence implementation are public, but the key and the encodings are unknown to the adversary.

In a self-equivalence implementation the round key material is hidden in the encoded affine layer, given as a binary constant and a binary matrix; the binary constant masks the round key with the round encodings and the binary matrix masks the linear part of the input and output encodings with each other. If many pairs of round keys and self-equivalences lead to the same encoded affine layer, each encoded affine layer is individually secure and the adversary is forced to solve a disambiguation problem involving several rounds.

Since CEJO encodings can be reduced to self-equivalence encodings, several white-box attacks [6, 28, 18] on CEJO implementations of AES contain steps to disambiguate diagonal self-equivalence encodings. However, they exploit specific properties of AES, and no generic attack on self-equivalence encodings, or even diagonal encodings, have been proposed this far.

To analyse the security of self-equivalence encodings, we describe the first generic attack on self-equivalence implementations. Since this is the first generic analysis, we focus on the class of SPN ciphers where the S-box layer does not have both a non-trivial additive self-equivalence and a linear component. Otherwise, the S-box layer will have a differential and a linear approximation with probability one, which are usually avoided in most SPN designs.

As most white-box attacks, our generic attack is a reduction attack. In other words, we describe a method to obtain new encoded affine layers for which the round encodings are restricted to a smaller encoding space. Afterwards, the key can be recovered by performing an exhaustive search over the reduced encoding space and filtering wrong candidates using the key schedule. Thus, the complexity of the key-recovery attack is given by the complexity to reduce the round encoding space and the complexity to brute force the reduced encoding space, where the latter step is lower bounded by the cardinality of the reduced encoding space.

The reduction step is based on equivalence problems involving the linear part of the round encodings with small solution sets, inspired from the white-box attacks on AES [6, 28, 18]. These equivalence problems consider unknowns restricted to some self-equivalence subgroup of the S-boxes, and estimating the complexity of these problems highly depends on the structure of the subgroup. Thus, our complexity metric is the cardinality of the reduced round encoding space, which provides a lower bound of the complexity of the whole attack.

Before we describe the attack, we recall the notation and the components of a self-equivalence implementation. Let E_k be the encryption function for a fixed key k of an arbitrary SPN cipher whose S-box layer SL does not have a non-trivial additive self-equivalence or a linear component. Let $\overline{E_k}$ be a self-equivalence implementation of E_k with encoded affine layers $\overline{AL^{(1)}}, \ldots, \overline{AL^{(n_r)}}$. Recall the rth

intermediate encoded affine layer is defined by $\overline{AL^{(r)}} = O^{(r)} \circ (\oplus_{k^{(r)}} \circ LL) \circ I^{(r)}$, where $k^{(r)}$ is the rth round key, LL is the linear layer and $(I^{(r)}, O^{(r)})$ are the round encodings satisfying $(O^{(r)}, I^{(r+1)}) \in \mathrm{SE}(SL)$.

For this class of SPN ciphers all the self-equivalences of the S-box layer are diagonal, following Theorem 1. For ease of explanation, we will consider diagonal self-equivalences without block permutations, since they do not significantly impact the reduction step. In this case, the round encoding space of the self-equivalence implementation is $\mathrm{SE}(S_1) \parallel \cdots \parallel \mathrm{SE}(S_\rho)$. Figure 3 depicts an intermediate encoded affine layer with this type of round encoding space.

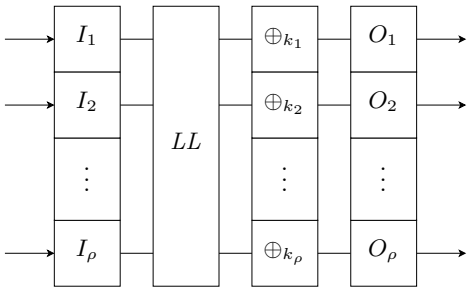

Fig. 3. An encoded affine layer with diagonal self-equivalence encodings.

Given an intermediate[3] encoded affine layer $\overline{AL^{(r)}}$, the core step in the reduction attack is to obtain another encoded affine layer differing only in the ith block of the output encoding, i.e.,

$$\widehat{AL^{(r)}} = O'^{(r)} \circ AL^{(r)} \circ I^{(r)}, \quad O'^{(r)}_h = O^{(r)}_h \ \forall h \neq i,$$

such that $\mathrm{Lin}(O'^{(r)}_i)$ belongs to a proper subgroup of $\mathrm{Lin}(\mathrm{RSE}(S_i))$. The encoding space of the whole round encoding $(I^{(r)}, O^{(r)})$ can be then reduced by applying this step for all output encoding blocks of $\overline{AL^{(r)}}$ and $\overline{AL^{(r-1)}}$ and using the self-equivalence property $SL = I^{(r)} \circ SL \circ O^{(r-1)}$.

The reduction of the linear part of the output encoding is based on equivalence problems, and we do not reduce the constant part of the round encodings to avoid equivalence problems dependent on the round key. Recall an affine (resp. linear) equivalence problem is a functional equation $G = Y \circ F \circ X$ where F and G are given n-bit functions and (X, Y) are unknown n-bit affine (resp. linear) permutations. In our case, the unknowns (X, Y) are restricted to some given subgroup $\mathcal{A} \times \mathcal{B}$; given any solution (X_0, Y_0) the solution set is the group

$$\mathcal{S} = \{(A \circ X_0, Y_0 \circ B) : (A, B) \in \mathrm{SE}(F) \cap (\mathcal{A} \times \mathcal{B})\}.$$

[3] The attack does not target the external encodings, which are assumed to be random affine permutations. Note that a self-equivalence implementation without external encodings is trivially insecure.

We will describe two generic classes of equivalence problems that contain the output round encoding as a particular solution, and we will reduce the round encoding space to their solution sets. These two classes of equivalence problems, the centralizer and the asymmetric problems, can be combined and do not assume any particular structure in the self-equivalences groups of the S-boxes. However, other equivalence problems can be considered by exploiting specific properties of the self-equivalence groups. As an example, we will also describe a class of linear equivalence problems for S-boxes with only linear self-equivalences.

6.1 The Centralizer Problems

The centralizer problems are a class of univariate linear equivalence problems that allow reducing the round encoding space to the centralizer of linear layer blocks. Since these equivalence problems only involve the encoded affine layer of one round, we omit the round superscript for these problems.

Given an $(m \times \rho)$-bit linear function A, recall that we denote by $A_{i,j}$ the (i,j) m-bit block of A seen as a block matrix. Moreover, the centralizer of an n-bit function F is defined as $\mathrm{C}(F) = \{A \in \mathrm{GL}_n : A \circ F = F \circ A\}$. The centralizer problems are based on the following proposition, which is proved in Appendix C.

Proposition 2. *Let $\overline{AL} = O \circ AL \circ I$ be the encoded affine layer of an intermediate round and consider the linear layer block $L = LL_{i,j} \circ \left(LL^{-1}\right)_{j,i}$. Then, the univariate linear equivalence problem,*

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} \circ \mathrm{Lin}\left(\overline{AL}^{-1}\right)_{j,i} = X \circ L \circ X^{-1}\,, \quad X \in \mathrm{Lin}(\mathrm{RSE}(S_i))\,, \qquad (1)$$

contains $X_0 = \mathrm{Lin}(O_i)$ as a solution. In particular, its solution set is given by

$$\mathcal{S} = \mathrm{Lin}(O_i) \circ \left(\mathrm{C}(L) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))\right).$$

Without the restriction $X \in \mathrm{Lin}(\mathrm{RSE}(S_i))$, a solution of Eqn. (1) could be easily obtained with complexity $\mathcal{O}(m^\omega)$, for $2 < w < 3$. However, the restriction makes the complexity harder to estimate, which strongly depends on the structure of $\mathrm{Lin}(\mathrm{RSE}(S_i))$. In general, one can perform an exhaustive search over $\mathrm{Lin}(\mathrm{RSE}(S_i))$ using the algorithm of Biryukov et al. [8]; the complexity of this approach is at least 2^{3m} and is also lower bounded by $|\mathrm{Lin}(\mathrm{RSE}(S_i))|$.

Let $X_0 = \mathrm{Lin}(O_i) \circ H$ be an arbitrary solution of Eqn. (1), for some unknown $H \in \mathrm{C}(L) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))$. Since $X_0 \in \mathrm{Lin}(\mathrm{RSE}(S_i))$, there exists an m-bit constant x_0 such that $\oplus_{x_0} \circ X_0 \in \mathrm{RSE}(S_i)$. Let $C = (C_1 \parallel \cdots \parallel C_\rho)$ be the right self-equivalence of SL where $C_i = (\oplus_{x_0} \circ X_0)^{-1}$ and $C_h = \mathrm{Id}\ \ \forall h \neq i$. Then, we can obtain another encoded affine layer

$$\widehat{AL} = C \circ \overline{AL} = O' \circ AL \circ I\,, \quad O'_h = O_h\ \forall h \neq i\,,$$

with (unknown) self-equivalences as round encodings and that only differs in the ith block of the output encoding. In addition, the linear part of O'_i is restricted to the subgroup $\mathrm{Lin}(\mathrm{RSE}(S_i)) \cap \mathrm{C}(L)$.

In other words, the centralizer problem given by Eqn. (1) allows reducing the encoding space associated to the linear part of the ith output encoding block, $\mathrm{Lin}(\mathrm{RSE}(S_i))$, to the subgroup $\mathrm{Lin}(\mathrm{RSE}(S_i)) \cap \mathrm{C}(L)$. Note that this subgroup is related to the self-equivalences of the linear layer blocks, showing that the security of self-equivalence implementations not only depends on the self-equivalences of the S-box layer but also on the self-equivalence structure of the linear layer.

If L is the identity or its matrix representation over $\mathbb{F}_2$ has low rank, its centralizer contains too many elements and the reduction is not significant. Nevertheless, we can consider similar equivalence problems involving other blocks of $\overline{AL}$ and LL. In general, any sequence of the form

$$\begin{aligned} A &= A_{i_1,j_1} A_{j_1,i_2} \cdots A_{i_s,j_s} A_{j_s,i_{s+1}}\,, \quad i_1 = i_{s+1} \\ A_{i_h,j_h} &\in \left\{ \mathrm{Lin}(AL)_{i_h,j_h}, \left(\mathrm{Lin}\left(AL^{-1}\right)_{i_h,j_h}\right)^{-1} \right\} \\ A_{j_h,i_{h+1}} &\in \left\{ \mathrm{Lin}\left(AL^{-1}\right)_{j_h,i_{h+1}}, \left(\mathrm{Lin}(AL)_{i_{h+1},j_h}\right)^{-1} \right\} \end{aligned}$$

leads to a centralizer problem of the form $A = \mathrm{Lin}(O_{i_1}) \circ L \circ \mathrm{Lin}(O_{i_1})^{-1}$, for some L composed of blocks of LL. In Appendix E we provide an example of a centralizer problem to reduce the round encoding space of a self-equivalence implementation of AES.

6.2 The Asymmetric Problems

The asymmetric problems are a class of affine equivalence problems that allow reducing the round encoding space to the intersection between the right and left self-equivalence groups of the S-boxes. As in the centralizer problems, we omit the round superscript for ease of notation. The asymmetric problems are based on the following proposition, whose proof is given in Appendix D.

Proposition 3. *Let $\overline{AL} = O \circ AL \circ I$ be the encoded affine layer of an intermediate round. Then, the affine*[4] *equivalence problem*

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} \circ S_j = X \circ LL_{i,j} \circ S_j \circ Y\,, \quad \mathrm{Lin}(X) \in \mathrm{Lin}(\mathrm{RSE}(S_i)) \tag{2}$$

contains a solution (X_0, Y_0) such that $\mathrm{Lin}(X_0) = \mathrm{Lin}(O_i)$. In particular, the linear part of the set of X-solutions $\mathcal{S}_X$ is given by the GL_n-subgroup

$$\mathrm{Lin}\left(\mathcal{S}_X\right) = \mathrm{Lin}\left(O_i\right) \circ \left(\mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j)) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))\right).$$

As in the previous case, the complexity to obtain a solution of Eqn. (2) is harder to estimate and strongly depends on $\mathrm{Lin}(\mathrm{RSE}(S_i))$. In general, the algorithm by Biryukov et al. can be used to perform an exhaustive search over $\mathrm{Lin}(\mathrm{RSE}(S_i))$ with complexity lower bounded by $\max(2^{3m}, |\mathrm{Lin}(\mathrm{RSE}(S_i))|)$.

Let (X_0, Y_0) be an arbitrary solution of Eqn. (2). Then, there exists an $H \in \mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j)) \cap \mathrm{Lin}(\mathrm{RSE}(S_i)$ satisfying $\mathrm{Lin}(X_0) = \mathrm{Lin}\left(O_i\right) \circ H$. Let

[4] If I_j is linear and $S_j(0) = 0$, then Eqn. (2) is a linear equivalence problem.

x_0 be an m-bit constant such that $\oplus_{x_0} \circ \mathrm{Lin}(X_0) \in \mathrm{RSE}(S_i)$ and consider $C = (C_1 \parallel \cdots \parallel C_\rho) \in \mathrm{SE}(SL)$ such that $C_i = (\oplus_{x_0} \circ \mathrm{Lin}(X_0))^{-1}$ and $C_h = \mathrm{Id},\ \forall h \neq i$. Then, we can obtain another encoded affine layer with self-equivalence encodings,

$$\widehat{AL} = C \circ \overline{AL} = O' \circ AL \circ I\,, \quad O'_h = O_h\ \forall h \neq i\,,$$

that only differs in the ith block of the output encoding. Moreover, while $\mathrm{Lin}(\mathrm{RSE}(S_i))$ was the encoding space of $\mathrm{Lin}(O_i)$, the encoding space of $\mathrm{Lin}(O'_i)$ has been reduced to $\mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j)) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))$.

If $LL_{i,j}$ is invertible, note that $\mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j))$ is the conjugate of $\mathrm{Lin}(\mathrm{LSE}(S_j))$ by $LL_{i,j}$, i.e., $\mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j)) = LL_{i,j} \circ \mathrm{Lin}(\mathrm{LSE}(S_j)) \circ LL_{i,j}^{-1}$. Thus, this equivalence problem allows reducing the encoding space of the linear part of the ith output encoding block to the intersection between $\mathrm{Lin}(\mathrm{RSE}(S_i))$ and a conjugate group of $\mathrm{LSE}(S_j)$ by a linear layer block. In Appendix F we show how an asymmetric problem can completely break a self-equivalence implementation of AES by reducing the encoding space to a set of one element.

Other equivalence problems. The centralizer and the asymmetric problems are generic equivalence problems that can be considered for any self-equivalence implementation. However, other equivalence problems can be considered, either by combining these two or by exploiting specific properties of the self-equivalence structure of the cipher components.

For example, if all the self-equivalences of the S-box layer are linear, we can combine the centralizer and the asymmetric problems to obtain

$$A_{i,i}^{(r)} \circ S_i \circ A_{i,j}^{(r-1)} \circ B_{j,i}^{(r-1)} \circ S_i^{-1} \circ B_{i,i}^{(r)} =$$
$$O_i^{(r)} \circ LL_{i,i} \circ S_i \circ LL_{i,j} \circ \left(LL^{-1}\right)_{j,i} \circ S_i^{-1} \circ \left(LL^{-1}\right)_{i,i} \circ \left(O_i^{(r)}\right)^{-1}\,,$$

where $A^{(r)} = \mathrm{Lin}\left(\overline{AL^{(r)}}\right)$ and $B^{(r)} = \mathrm{Lin}\left(\overline{AL^{(r)}}\right)^{-1}$. By considering $X = O_i^{(r)}$ as the unknown, we obtain an univariate linear equivalence problem that reduces the encoding space associated to $O_i^{(r)}$ to

$$\mathrm{RSE}(S_i) \cap \mathrm{C}(LL_{i,i} \circ S_i \circ LL_{i,j} \circ (LL^{-1})_{j,i} \circ S_i^{-1} \circ (LL^{-1})_{i,i})\,.$$

This class of linear equivalence problems can be extended to any number of rounds, leading to many potential significant reductions.

7 Conclusion

In this document, we analysed the security of self-equivalence encodings. We described how a CEJO implementation can be transformed into a self-equivalence implementation, we characterized the self-equivalences of an S-box layer, and we propose a generic attack on a self-equivalence implementation of an SPN cipher with a cryptographically strong S-box layer.

Our generic attack shows that if the cipher components satisfy some properties, namely linear layer blocks with low rank or with big centralizers and S-boxes with plenty of affine self-equivalences and with similar left and right self-equivalences, a self-equivalence implementation resists the generic attack.

While traditional SPN ciphers with cryptographically strong S-box layers, such as AES, do not satisfy these strong requirements, our analysis reveals the potential of self-equivalence encodings to secure other types of ciphers. In particular, our analysis identifies future research directions that can lead to secure white-box implementations, namely non-diagonal self-equivalence encodings and self-equivalences implementations of ciphers with weaker round functions.

Acknowledgements. Adrián Ranea is supported by a PhD Fellowship from the Research Foundation – Flanders (FWO). The authors would like to thank the anonymous reviewers for their comments and suggestions.

References

1. Albrecht, M.R., Bard, G.V., and Hart, W.: Algorithm 898: Efficient multiplication of dense matrices over GF(2). ACM Trans. Math. Softw. 37(1) (2010)
2. Amadori, A., Michiels, W., and Roelse, P.: A DFA Attack on White-Box Implementations of AES with External Encodings. In: SAC. LNCS, vol. 11959. Springer, Heidelberg (2019)
3. Baek, C.H., Cheon, J.H., and Hong, H.: White-box AES implementation revisited. J. Commun. Networks 18(3) (2016)
4. Bai, K., and Wu, C.: An AES-Like Cipher and Its White-Box Implementation. Comput. J. 59(7) (2016)
5. Banik, S., Bogdanov, A., Isobe, T., and Jepsen, M.B.: Analysis of Software Countermeasures for Whitebox Encryption. IACR Trans. Symmetric Cryptol. 2017(1) (2017)
6. Billet, O., Gilbert, H., and Ech-Chatbi, C.: Cryptanalysis of a White Box AES Implementation. In: Selected Areas in Cryptography. LNCS, vol. 3357. Springer, Heidelberg (2004)
7. Biryukov, A., Bouillaguet, C., and Khovratovich, D.: Cryptographic Schemes Based on the ASASA Structure: Black-Box, White-Box, and Public-Key (Extended Abstract). In: ASIACRYPT (1). LNCS, vol. 8873. Springer, Heidelberg (2014)
8. Biryukov, A., Cannière, C.D., Braeken, A., and Preneel, B.: A Toolbox for Cryptanalysis: Linear and Affine Equivalence Algorithms. In: EUROCRYPT. LNCS, vol. 2656. Springer, Heidelberg (2003)
9. Bogdanov, A., and Isobe, T.: White-Box Cryptography Revisited: Space-Hard Ciphers. In: ACM Conference on Computer and Communications Security. ACM (2015)
10. Bos, J.W., Hubain, C., Michiels, W., and Teuwen, P.: Differential Computation Analysis: Hiding Your White-Box Designs is Not Enough. In: CHES. LNCS, vol. 9813. Springer, Heidelberg (2016)
11. Bringer, J., Chabanne, H., and Dottax, E.: White Box Cryptography: Another Attempt. IACR Cryptology ePrint Archive 2006 (2006)
12. Capture the Flag Challenge - The WhibOx Contest, https://whibox-contest.github.io/

13. Cheon, J.H., Hong, H., Lee, J., and Lee, J.: An Efficient Affine Equivalence Algorithm for Multiple S-Boxes and a Structured Affine Layer. In: SAC. LNCS, vol. 10532. Springer, Heidelberg (2016)
14. Chow, S., Eisen, P.A., Johnson, H., and Oorschot, P.C. van: A White-Box DES Implementation for DRM Applications. In: Digital Rights Management Workshop. LNCS, vol. 2696. Springer, Heidelberg (2002)
15. Chow, S., Eisen, P.A., Johnson, H., and Oorschot, P.C. van: White-Box Cryptography and an AES Implementation. In: Selected Areas in Cryptography. LNCS, vol. 2595. Springer, Heidelberg (2002)
16. De Cannière, C.: Analysis and design of symmetric encryption algorithms. Doctoral Dissertaion, KULeuven (2007)
17. De Mulder, Y., Roelse, P., and Preneel, B.: Cryptanalysis of the Xiao - Lai White-Box AES Implementation. In: Selected Areas in Cryptography. LNCS, vol. 7707. Springer, Heidelberg (2012)
18. Derbez, P., Fouque, P.-A., Lambin, B., and Minaud, B.: On Recovering Affine Encodings in White-Box Implementations. IACR Trans. Cryptogr. Hardw. Embed. Syst. 2018(3) (2018)
19. Dinur, I.: An Improved Affine Equivalence Algorithm for Random Permutations. In: EUROCRYPT (1). LNCS, vol. 10820. Springer, Heidelberg (2018)
20. Evertse, J.-H.: Linear Structures in Blockciphers. In: EUROCRYPT. LNCS, vol. 304. Springer, Heidelberg (1987)
21. Faugère, J.-C., and Perret, L.: Polynomial Equivalence Problems: Algorithmic and Theoretical Aspects. In: EUROCRYPT. LNCS, vol. 4004. Springer, Heidelberg (2006)
22. Gordon, N.A., Jarvis, T.M., and Shaw, R.: Some aspects of the linear groups GL (n, q). Tech. rep., (2003)
23. Goubin, L., Masereel, J.-M., and Quisquater, M.: Cryptanalysis of White Box DES Implementations. In: Selected Areas in Cryptography. LNCS, vol. 4876. Springer, Heidelberg (2007)
24. Goubin, L., Rivain, M., and Wang, J.: Defeating State-of-the-Art White-Box Countermeasures with Advanced Gray-Box Attacks. IACR Trans. Cryptogr. Hardw. Embed. Syst. 2020(3) (2020)
25. Jacob, M., Boneh, D., and Felten, E.W.: Attacking an Obfuscated Cipher by Injecting Faults. In: Digital Rights Management Workshop. LNCS, vol. 2696. Springer, Heidelberg (2002)
26. Karroumi, M.: Protecting White-Box AES with Dual Ciphers. In: ICISC. LNCS, vol. 6829. Springer, Heidelberg (2010)
27. Lai, X.: Additive and Linear Structures of Cryptographic Functions. In: FSE. LNCS, vol. 1008. Springer, Heidelberg (1994)
28. Lepoint, T., Rivain, M., Mulder, Y.D., Roelse, P., and Preneel, B.: Two Attacks on a White-Box AES Implementation. In: Selected Areas in Cryptography. LNCS, vol. 8282. Springer, Heidelberg (2013)
29. Lin, D., Faugère, J.-C., Perret, L., and Wang, T.: On enumeration of polynomial equivalence classes and their application to MPKC. Finite Fields and Their Applications 18(2) (2012)
30. Lin, T., Yan, H., Lai, X., Zhong, Y., and Jia, Y.: Security Evaluation and Improvement of a White-Box SMS4 Implementation Based on Affine Equivalence Algorithm. Comput. J. 61(12) (2018)
31. Link, H.E., and Neumann, W.D.: Clarifying Obfuscation: Improving the Security of White-Box DES. In: ITCC (1). IEEE Computer Society (2005)

32. Luo, R., Lai, X., and You, R.: A new attempt of white-box AES implementation. In: SPAC. IEEE (2014)
33. McMillion, B., and Sullivan, N.: Attacking White-Box AES Constructions. In: SPRO@CCS. ACM (2016)
34. Michiels, W., Gorissen, P., and Hollmann, H.D.L.: Cryptanalysis of a Generic Class of White-Box Implementations. In: Selected Areas in Cryptography. LNCS, vol. 5381. Springer, Heidelberg (2008)
35. Rivain, M., and Wang, J.: Analysis and Improvement of Differential Computation Attacks against Internally-Encoded White-Box Implementations. IACR Trans. Cryptogr. Hardw. Embed. Syst. 2019(2) (2019)
36. Shi, Y., Wei, W., and He, Z.: A Lightweight White-Box Symmetric Encryption Algorithm against Node Capture for WSNs. Sensors 15(5) (2015)
37. Wyseur, B., Michiels, W., Gorissen, P., and Preneel, B.: Cryptanalysis of White-Box DES Implementations with Arbitrary External Encodings. In: Selected Areas in Cryptography. LNCS, vol. 4876. Springer, Heidelberg (2007)
38. Xiao, Y., and Lai, X.: A secure implementation of white-box AES. In: 2009 2nd International Conference on Computer Science and its Applications (2009)

A Proof of Proposition 1

Proposition 1. *Let E_k be the encryption function of an n-bit SPN cipher with linear layer LL and S-box layer SL consisting of m-bit cryptographic S-boxes, and let $\overline{E_k}$ be a CEJO implementation of E_k with m_n-bit non-linear encodings and m_l-bit linear encodings. Given black-box access to three consecutive encoded round functions $\overline{E^{(r-2)}}, \overline{E^{(r-1)}}$ and $\overline{E^{(r)}}$, one can find another encoded $E^{(r)}$ with round encoding space $LL \circ \mathrm{LSE}(SL)$, i.e.,*

$$\widehat{E^{(r)}} = \widehat{O} \circ E^{(r)} \circ \widehat{I}\,, \quad \widehat{I}, \widehat{O^{-1}} \in LL \circ \mathrm{LSE}(SL)\,,$$

in time $\mathcal{O}\left((n/m_n)2^{3m_n} + 2^m n^3 + n^4/m + 2^{2m} mn^2\right)$.

Proof. We apply first the algorithm by Baek et al. [3, Theorem 1 and 2] to recover the non-linear part of the output encodings of $\overline{E^{(i)}}$, for $i \in \{r-2, r-1, r\}$. Each m_n-bit non-linear function is recovered up to an affine transformation in time $\mathcal{O}\left((n/m_n)2^{3m_n}\right)$. As a result, new encoded round functions are obtained,

$$\widetilde{E^{(i)}} = \widetilde{O^{(i)}} \circ E^{(i)} \circ \widetilde{I^{(i)}}\,, \quad i \in \{r-1, r\}\,, \tag{3}$$

where the input and output round encodings are unknown affine permutations.

Since $\widetilde{E^{(r-1)}}$ and $\widetilde{E^{(r)}}$ are affine equivalent to the S-box layer SL, the corresponding affine equivalence problems can be solved with the algorithm by Derbez et al. in time $O(2^m n^3 + n^4/m + 2^{2m} mn^2)$ [18, Algorithm 1]. Thus, two pairs of affine permutations, $(A^{(r-1)}, B^{(r-1)})$ and $(A^{(r)}, B^{(r)})$, are obtained such that

$$\widetilde{E^{(i)}} = B^{(i)} \circ SL \circ A^{(i)}\,, \quad i \in \{r-1, r\}\,. \tag{4}$$

Combining Eqns. (3) and (4) leads to the functional equation

$$B^{(i)} \circ SL \circ A^{(i)} = (\widetilde{O^{(i)}} \circ LL) \circ SL \circ (\oplus_{k^{(i)}} \circ \widetilde{I^{(i)}}) .$$

Note that $(A^{(i)}, B^{(i)})$ is not necessarily equal to $(\oplus_{k^{(i)}} \circ \widetilde{I^{(i)}}, \widetilde{O^{(i)}} \circ LL)$. Nevertheless, there exists a self-equivalence $(C^{(i)}, D^{(i)}) \in \mathrm{SE}(SL)$ such that

$$A^{(i)} = C^{(i)} \circ \oplus_{k^{(i)}} \circ \widetilde{I^{(i)}} ,$$
$$B^{(i)} = \widetilde{O^{(i)}} \circ LL \circ D^{(i)} .$$

Finally, by composing $\widetilde{E^{(r)}}$ with the key-independent permutations $B^{(r-1)}$ and $B^{(r)}$, we can obtain an encoded $E^{(r)}$ with round encoding space $LL \circ \mathrm{LSE}(SL)$,

$$\widehat{E^{(r)}} = \left(B^{(r)}\right)^{-1} \circ \widetilde{E^{(r)}} \circ B^{(r-1)} = (LL \circ D^{(r)})^{-1} \circ E^{(r)} \circ (LL \circ D^{(r-1)}) .$$

□

B Proof of Theorem 1

First, we will introduce some concepts and lemmas connecting the notions of additive self-equivalences, linear structures and linear components.

Lemma 2. *Let G be an n-bit function. Then $(\oplus_a, \oplus_b)$ is an additive self-equivalence of G if and only if a is a linear structure of all canonical components of G.*

Proof. The lemma immediately follows from the definition of additive self-equivalence and linear structure. If $(\oplus_a, \oplus_b) \in \mathrm{SE}(G)$, then $\oplus_b \circ G = G \circ \oplus_a$, which is equivalent to $\oplus_{b_i} \circ G_i = G_i \circ \oplus_a$ for $i = 1, \ldots, n$. □

Given an n-bit Boolean function G, we say that G is independent of the jth input x_j if x_j does not appear in the Algebraic Normal Form (ANF) of G, i.e., the unique representation of G as an n-variable polynomial over $\mathbb{F}_2$. Similarly, we say that G is linear in x_j if all the ANF terms that contains x_j are of degree one. The next lemma shows the connection between independent variables and additive self-equivalences, previous analysed in [27].

Lemma 3. *Let G be an n-bit function and consider the jth canonical n-bit vector $e^{(j)}$, where $e_i^{(j)} = 0$ if and only if $i \neq j$. If each canonical component of G is linear in the jth input x_j or independent of x_j, then there exists an n-bit value b such that $(\oplus_{e^{(j)}}, \oplus_b)$ is an additive self-equivalence of G.*

Proof. Let G_i be a canonical component of G. If G_i is linear in x_j, note that $G_i(x + e^{(j)}) = G_i(x) \oplus G_i(e^{(j)})$, that is, $\oplus_{G_i(e^{(j)})} \circ G_i = G_i \circ \oplus_{e^{(j)}}$ and $e^{(j)}$ is a linear structure of G_i. Similarly, if G_i is independent of x_j, $G_i(x + e^{(j)}) = G_i(x)$ and $e^{(j)}$ is also a linear structure of G_i. Therefore, $e^{(j)}$ is a linear structure of all canonical components, and by Lemma 2 G has an additive self-equivalence with $e^{(j)}$ as the right equivalence. □

The last lemma required by the proof of Theorem 1 is the follwing technical lemma to obtain a "normal form" of block matrices.

Lemma 4. *Let A_1 and A_2 be two m-bit matrices where $0 < \operatorname{rank}(A_1) \leq \operatorname{rank}(A_2)$. Let $A = (A_1|A_2)$ be the block matrix composed of one block row and two block columns. Then, there exists an invertible (m,m)-bit matrix D_A and a block matrix*

$$C_A = \begin{pmatrix} C_{A,1} & 0 \\ 0 & C_{A,2} \end{pmatrix},$$

where $C_{A,1}$ and $C_{A.2}$ are invertible (m,m)-bit matrices, such that $A' = D_A \times A \times C_A$ is of the form

$$A' = \left(\begin{array}{c:c|c:c} \mathrm{Id}_{\operatorname{rank}(A_1)} & 0 & M_A & \\ \hdashline 0 & & \mathrm{Id}_r \;\; 0 & 0 \\ \hdashline 0 & & 0 & \end{array}\right)$$

where $r = \operatorname{rank}(A) - \operatorname{rank}(A_1)$ and M_A is a $(\operatorname{rank}(A_1), \operatorname{rank}(A_2))$-bit matrix with full rank, i.e., $\operatorname{rank}(M_A) = \operatorname{rank}(A_1)$.

Proof. We will show that we can obtain A' by doing elementary row operations (represented by D_A) and elementary column operations (represented by C_A). Due to the block diagonal shape of C_A, column operations involving both left columns and right columns are not allowed. Thus, we will employ left (resp. right) column operations, i.e., elementary column operations only involving the first (resp. last) m columns. The sequence of elementary operations is the following.

1. With left column operations we can make the last $m - \operatorname{rank}(A_1)$ columns on the left side zero. Similarly, on the right side we can make the last $m - \operatorname{rank}(A_2)$ columns zero with right column operations,

$$\left(\begin{array}{c:c|c:c} * & 0 & * & 0 \\ \hdashline * & 0 & * & 0 \\ \hdashline * & 0 & * & 0 \end{array}\right).$$

2. With row operations we can make the last $m - \operatorname{rank}(A)$ rows zero,

$$\left(\begin{array}{c:c|c:c} * & 0 & * & 0 \\ \hdashline * & 0 & * & 0 \\ \hdashline 0 & 0 & 0 & 0 \end{array}\right).$$

3. With row operations we can make the rows $(\operatorname{rank}(A_1) + 1, \operatorname{rank}(A))$ in the left part zero, obtaining

$$\left(\begin{array}{c:c|c:c} A'_1 & 0 & * & 0 \\ \hdashline 0 & 0 & * & 0 \\ \hdashline 0 & 0 & 0 & 0 \end{array}\right)$$

   where A'_1 is an invertible $\operatorname{rank}(A_1)$-bit matrix.

4. With row operations and left column operations, we can replace A'_1 by $\mathrm{Id}_{\mathrm{rank}(A_1)}$ and obtain
$$\left(\begin{array}{c|c|c|c} \mathrm{Id}_{\mathrm{rank}(A_1)} & 0 & * & 0 \\ \hline 0 & 0 & A'_2 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right)$$
where A'_2 is a $(r, \mathrm{rank}(A_2))$-bit matrix with rank $r = \mathrm{rank}(A) - \mathrm{rank}(A_1)$.
5. With right column operations and row operations involving only the middle rows, we can replace A'_2 by $(\mathrm{Id}_r \,|0)$,
$$\left(\begin{array}{c|c|c|c} \mathrm{Id}_{\mathrm{rank}(A_1)} & 0 & * & 0 \\ \hline 0 & 0 & \mathrm{Id}_r \;\; 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right) .$$
6. Finally, with right column operations and by adding middle rows to the first rows, we can obtain
$$\left(\begin{array}{c|c|c|c} \mathrm{Id}_{\mathrm{rank}(A_1)} & 0 & M_A & 0 \\ \hline 0 & 0 & \mathrm{Id}_r \;\; 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right)$$
where M_A is a $(\mathrm{rank}(A_1), \mathrm{rank}(A_2))$-bit matrix with full rank. □

We are now ready to start with the proof of Theorem 1. The conducting idea of the proof is that F has a non-diagonal self-equivalence (A, B) if and only if there is a function F', $\mathcal{Q}$-affine equivalent to F, such that F' has a simpler non-diagonal self-equivalence (A', B'), where many blocks of A' and B' are the identity. We will extract some $\mathcal{Q}$-invariant necessary conditions to have a non-diagonal self-equivalence from F', and we will translate them to F.

Theorem 1. *Let $F = F_1 \,\|\, \cdots \,\|\, F_\rho$ be an n-bit permutation, where each F_i is an m-bit permutation. If F has a non-diagonal self-equivalence, then F contains three permutations (F_i, F_j, F_h), $j \neq h$, such that*

(a) F_i has a non-trivial additive self-equivalence.
(b) F_j has a non-trivial linear component.
(c) F_h has 2^{m-r} linear structures that are common to r linear independent components, for some $0 < r < m$.

Proof. Let (A, B^{-1}) be an arbitrary non-diagonal self-equivalence of F. Then, $B \circ F = F \circ A$ and $\mathrm{Lin}(A)$ contains two non-zero blocks in the same row, i.e., $\mathrm{Lin}(A)_{i,j}$ and $\mathrm{Lin}(A)_{i,h}$ for some i, j, h. Note that i could be equal to j or h but $j \neq h$.

By evaluating $B \circ F = F \circ A$ at $(x_1, \ldots, x_\rho)$, where $x_k = 0,\ \forall k \neq j, h$, we obtain the following equations for the ith row:

$$c' \oplus \mathrm{Lin}(B)_{i,j}(F_j(x_j)) = F_i(\mathrm{Lin}(A)_{i,j}(x_j) \oplus c)\,, \quad \forall x_j \in \mathbb{F}_2^m\,,$$
$$c''' \oplus \mathrm{Lin}(B)_{i,h}(F_j(x_h)) = F_i(\mathrm{Lin}(A)_{i,h}(x_h) \oplus c'')\,, \quad \forall x_h \in \mathbb{F}_2^m\,,$$

for some constants c, c', c'', c'''. Thus, it is easy to see that $\mathrm{Lin}(B)_{i,j}$ and $\mathrm{Lin}(B)_{i,h}$ have the same rank as $\mathrm{Lin}(A)_{i,j}$ and $\mathrm{Lin}(A)_{i,h}$, respectively.

We will consider two cases depending on whether the blocks $\mathrm{Lin}(A)_{i,j}$ and $\mathrm{Lin}(A)_{i,h}$ have full rank.

Case 1: $\mathrm{Lin}(A)_{i,j}$ *and* $\mathrm{Lin}(A)_{i,h}$ *are invertible.* In this case, consider a diagonal self-equivalence $(C, D^{-1}) = (C_1 \parallel \cdots \parallel C_\rho, D_1^{-1} \parallel \cdots \parallel D_\rho^{-1}) \in \mathrm{SE}(F)$ satisfying

$$\begin{aligned}(\mathrm{Lin}(C_j), \mathrm{Lin}(D_j)) &= (\mathrm{Lin}(A)_{i,j}^{-1}, \mathrm{Lin}(B)_{i,j}^{-1})\,,\\ (\mathrm{Lin}(C_h), \mathrm{Lin}(D_h)) &= (\mathrm{Lin}(A)_{i,h}^{-1}, \mathrm{Lin}(B)_{i,h}^{-1})\,,\\ C_k = D_k &= \mathrm{Id}\,, \quad \forall k \neq j, h\,.\end{aligned}$$

Thus, $(A \circ C, D^{-1} \circ B^{-1})$ is a non-diagonal self-equivalence which satisfies

$$\mathrm{Lin}(A \circ C)_{i,j} = \mathrm{Lin}(A \circ C)_{i,h} = \mathrm{Lin}(B \circ D)_{i,j} = \mathrm{Lin}(B \circ D)_{i,h} = \mathrm{Id}\ .$$

By evaluating $(B \circ D) \circ F = F \circ (A \circ C)$ at $(x_1, \ldots, x_\rho)$, where $x_k = 0,\ \forall k \neq j, h$, we obtain the following equation for the ith row:

$$c' + F_j(x_j) \oplus F_h(x_h) = F_i(x_j + x_h + c)\,, \quad \forall x_j, x_h \in \mathbb{F}_2^m\,, \tag{5}$$

for some constants c, c'. Equation 5 implies that F_i is linear, since the left-hand side of the equation does not contain ANF terms multiplying x_j and x_h bits. But then F_j and F_h are also linear, and the conditions (a), (b), and (c) hold for Case 1.

Case 2: $\mathrm{Lin}(A)_{i,j}$ *or* $\mathrm{Lin}(A)_{i,h}$ *is non-invertible.* Consider the block matrices

$$\begin{aligned}A^\dagger &= (\mathrm{Lin}(A_{i,j}) | \,\mathrm{Lin}(A_{i,h}))\,,\\ B^\dagger &= (\mathrm{Lin}(B_{i,j}) | \,\mathrm{Lin}(B_{i,h}))\,.\end{aligned}$$

Without loss of generality, we assume $0 < \mathrm{rank}(\mathrm{Lin}(A)_{i,j}) \leq \mathrm{rank}(\mathrm{Lin}(A)_{i,h})$ and $\mathrm{rank}(A^\dagger) - \mathrm{Lin}(A)_{i,j} > 0$. Let r_j and r_h be the rank of $A_{i,j}$ and $A_{i,h}$, respectively, and let $r = rank(A^\dagger) - r_j$.

According to Lemma 4 there exist invertible m-bit matrices D_A and D_B and $2m$-bit block matrices C_A and C_B such that

$$A' = D_A \times A^\dagger \times C_A = \left(\begin{array}{c|c|c|c} \mathrm{Id}_{r_j} & 0 & M_A & 0 \\ 0 & 0 & \mathrm{Id}_r\ 0 & 0 \\ 0 & & 0 & \end{array}\right)$$

$$B' = D_B \times B^\dagger \times C_B = \left(\begin{array}{c|c|c|c} \mathrm{Id}_{r_j} & 0 & M_B & 0 \\ 0 & 0 & \mathrm{Id}_r\ 0 & 0 \\ 0 & & 0 & \end{array}\right)$$

where M_A and M_B are (r_j, r_h)-bit matrices with full rank.

By evaluating $B \circ F = F \circ A$ at $(x_1, \ldots, x_\rho)$, where $x_k = 0,\ \forall k \neq j, h$, we obtain the following equation for the ith row:

$$c' + B^\dagger(F_j(x_j), F_h(x_h)) = F_i\big(A^\dagger(x_j, x_h) + c\big)\,, \quad \forall x_j, x_h \in \mathbb{F}_2^m\,,$$

for some constants c and c'. For ease of notation, we will consider the constants c and c' to be zero; the generalization of the proof for arbitrary values of c and c') is straightforward.

By substituting $A^\dagger = D_A^{-1} \times A' \times C_A^{-1}$, $B^\dagger = D_B^{-1} \times B' \times C_B^{-1}$ and $(x_j, x_h) = C_A(x^{(j)}, x^{(h)})$, we obtain

$$\begin{aligned} &B'\big((C_{B,1}^{-1} \circ F_j \circ C_{A,1})(x^{(j)}), (C_{B,2}^{-1} \circ F_h \circ C_{A,2})(x^{(h)})\big) \\ &\quad = (D_B \circ F_i \circ D_A^{-1})(A'(x^{(j)}, x^{(h)})) \end{aligned} \qquad \forall x^{(j)}, x^{(h)} \in \mathbb{F}_2^m\,.$$

Let $F^{(i)}, F^{(j)}$ and $F^{(h)}$ be the m-bit permutations affine equivalent to F_i, F_j and F_j, respectively, given by

$$\begin{aligned} F^{(i)} &= D_B \circ F_i \circ D_A^{-1} \\ F^{(j)} &= C_{B,1}^{-1} \circ F_j \circ C_{A,1} \\ F^{(h)} &= C_{B,2}^{-1} \circ F_h \circ C_{A,2}\,. \end{aligned}$$

Then, we can rewrite the previous equation as follows,

$$B'\big(F^{(j)}(x^{(j)}), F^{(h)}(x^{(h)})\big) = F^{(i)}\big(A'(x^{(j)}, x^{(h)})\big)\,, \quad \forall x^{(j)}, x^{(h)} \in \mathbb{F}_2^m\,. \tag{6}$$

Given an m-bit function G with canonical components $(G_1, G_2, \ldots, G_m)$, let $G_{a,b}$ be the $(m, b-a+1)$-bit function given by $G_{a,b} = (G_a, G_{a+1}, \ldots, G_b)$. Similarly, given an m-bit variable $x = (x_1, \ldots, x_m)$ we denote $x_{a,b} = (x_a, x_{a+1}, \ldots, x_b)$. Then, Eqn. (6) can be decomposed into the following three equations:

$$F_{1,r_j}^{(j)}(x^{(j)}) + M_B\big(F_{1,r_h}^{(h)}(x^{(h)})\big) = F_{1,r_j}^{(i)}(y)\,, \tag{7a}$$

$$0 + F_{1,r}^{(h)}(x^{(h)}) = F_{r_j+1,r_j+r}^{(i)}(y)\,, \tag{7b}$$

$$0 + 0 = F_{r_j+r+1,m}^{(i)}(y)\,. \tag{7c}$$

where $y = A'(x^{(j)}, x^{(h)}) = (x_{1,r_j}^{(j)} + M_A(x_{1,r_h}^{(h)}), x_{1,r}^{(h)}, 0)$. Note that $M_A(x_{1,r_h}^{(h)})$ fully spans $\mathbb{F}_2^{r_j}$ and that the first r_j bits of y contain bits from both $x^{(j)}$ and $x^{(h)}$.

In Eqn. (7a) the left-hand side does not contain ANF terms multiplying $x^{(j)}$ and $x^{(h)}$ bits. Therefore, $F_{1,r_j}^{(i)}$ is linear in the first r_j inputs, which implies that $F_{1,r_j}^{(j)}$ is linear.

From Eqn. (7b) and Eqn. (7c) we can deduce that $F_{r_j+1,m}^{(i)}$ is independent of the first r_j input bits. Therefore, $F^{(i)}$ has at least 2^{r_j} additive self-equivalences by Lemma 3.

Finally, Eqn. (7b) also implies that $F^{(h)}_{1,r}$ is independent of the last $m-r$ inputs. Equivalently, $F^{(h)}$ has at least 2^{m-r} linear structures that are common to r independent components following Lemmas 2 and 3.

As a result, if F has a non-diagonal self-equivalence, we have obtained the following necessary conditions, for $1 \leq r_j, r < m$.

- $F^{(i)}$ has at least 2^{r_j} additive self-equivalences.
- $F^{(j)}$ has at least r_j linear independent components that are linear.
- $F^{(h)}$ has at least 2^{m-r} linear structures that are common to r independent components.

It is easy to see that these three properties are invariant in the class $\mathcal{Q} \circ F \circ \mathcal{Q}$. Therefore, F_i, F_j and F_h verify the same properties, which concludes the proof. □

C Proof of Proposition 2

Proposition 2. *Let $\overline{AL} = O \circ AL \circ I$ be the encoded affine layer of an intermediate round and consider the linear layer block $L = LL_{i,j} \circ \left(LL^{-1}\right)_{j,i}$. Then, the univariate linear equivalence problem,*

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} \circ \mathrm{Lin}\left(\overline{AL}^{-1}\right)_{j,i} = X \circ L \circ X^{-1}\,, \quad X \in \mathrm{Lin}(\mathrm{RSE}(S_i))\,,$$

contains $X_0 = \mathrm{Lin}(O_i)$ as a solution. In particular, its solution set is given by

$$\mathcal{S} = \mathrm{Lin}(O_i) \circ \left(\,\mathrm{C}(L) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))\right).$$

Proof. Note that the linear blocks of the encoded affine layer are defined as

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} = \mathrm{Lin}(O_i) \circ LL_{i,j} \circ \mathrm{Lin}(I_j)\,.$$

The input encoding can be removed by composing with the inverse of the encoded affine layer, obtaining the equation

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} \circ \mathrm{Lin}\left(\overline{AL}^{-1}\right)_{j,i} = \mathrm{Lin}(O_i) \circ LL_{i,j} \circ \left(LL^{-1}\right)_{j,i} \circ \mathrm{Lin}(O_i)^{-1}$$

where the only secret information is $\mathrm{Lin}(O_i)$. Thus, the univariate linear equivalence problem

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} \circ \mathrm{Lin}\left(\overline{AL}^{-1}\right)_{j,i} = X \circ LL_{i,j} \circ \left(LL^{-1}\right)_{j,i} \circ X^{-1}$$

with unknown $X \in \mathrm{Lin}(\mathrm{RSE}(S_i))$ has $\mathrm{Lin}(O_i)$ as particular solution. □

D Proof of Proposition 3

Proposition 3. *Let $\overline{AL} = O \circ AL \circ I$ be the encoded affine layer of an intermediate round. Then, the affine equivalence problem*

$$\mathrm{Lin}\left(\overline{AL}\right)_{i,j} \circ S_j = X \circ LL_{i,j} \circ S_j \circ Y\,, \quad \mathrm{Lin}(X) \in \mathrm{Lin}(\mathrm{RSE}(S_i))$$

contains a solution (X_0, Y_0) such that $\mathrm{Lin}(X_0) = \mathrm{Lin}(O_i)$. In particular, the linear part of the set of X-solutions $\mathcal{S}_X$ is given by the GL_n-subgroup

$$\mathrm{Lin}\left(\mathcal{S}_X\right) = \mathrm{Lin}\left(O_i\right) \circ \left(\mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j)) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))\right).$$

Proof. Let $\overline{AL} = \overline{AL^{(r)}}$ be the encoded affine layer of the rth round. From the definition of encoded affine layer and the self-equivalence property

$$I_j^{(r)} \circ S_j = S_j \circ \left(O_j^{(r-1)}\right)^{-1},$$

it is easy to show that there exists a constant c such that

$$\mathrm{Lin}\left(\overline{AL^{(r)}}\right)_{i,j} \circ S_j = \oplus_c \circ \mathrm{Lin}\left(O_i^{(r)}\right) \circ LL_{i,j} \circ S_j \circ \left(O_j^{(r-1)}\right)^{-1}.$$

Thus, we can consider the affine equivalence problem

$$\mathrm{Lin}\left(\overline{AL^{(r)}}\right)_{i,j} \circ S_j = X \circ LL_{i,j} \circ S_j \circ Y\,.$$

where X and Y are unknown affine permutations such that $\mathrm{Lin}(X) \in \mathrm{Lin}(\mathrm{RSE}(S_i))$. If (X_0, Y_0) is an arbitrary solution, then the solution set S is given by

$$\mathcal{S} = \{(X_0 \circ B, A \circ Y_0) : (A, B) \in \mathrm{SE}(LL_{i,j} \circ S_j),\ \mathrm{Lin}(B) \in \mathrm{Lin}(\mathrm{RSE}(S_i))\}\,.$$

Since $\oplus_c \circ \mathrm{Lin}(O_i^{(r)})$ is a particular X-solution, the linear part of the set of X-solutions is given by the GL_n-subgroup

$$\mathrm{Lin}\left(\mathcal{S}_X\right) = \mathrm{Lin}\left(O_i^{(r)}\right) \circ \left(\mathrm{Lin}(\mathrm{LSE}(LL_{i,j} \circ S_j)) \cap \mathrm{Lin}(\mathrm{RSE}(S_i))\right). \quad \square$$

E Example of the Centralizer Problem

We will show the application of a centralizer problem to reduce the round encoding space of a self-equivalence implementation of AES.

Let $\mathbb{F}_{2^8}$ be the finite field used in AES and let P_d be the 8-bit function corresponding to the power function $x \mapsto x^d$ in $\mathbb{F}_{2^8}$ and M_α be the 8-bit function corresponding to the multiplication $x \mapsto \alpha \times x$ by an $\mathbb{F}_{2^8}$-constant α. The AES S-box is an 8-bit function given by $S(x) = L(P_{254}(x)) + c$, for some linear permutation L and constant c. Since we can push the S-box constant to the

round key, we can redefine the S-box as $S = L \circ P_{254}$, whose self-equivalences are given by [8]

$$\begin{aligned}\mathrm{RSE}(S) &= \{M_\alpha \circ P_{2^i} : 0 \neq \alpha \in \mathbb{F}_{2^8},\ i \in \{0,\dots,7\}\}\,,\\ \mathrm{LSE}(S) &= \{L \circ (M_{\alpha^{254}} \circ P_{2^i})^{-1} \circ L^{-1} : 0 \neq \alpha \in \mathbb{F}_{2^8},\ i \in \{0,\dots,7\}\}\,.\end{aligned}$$

Note that $\mathrm{RSE}(S)$ and $\mathrm{LSE}(S)$ only consist of linear permutations.

For ease of explanation, we consider the round function restricted to a column state and omit the action of ShiftRows. Thus, the linear layer LL corresponds to the action of MixColumns on a column, and its block matrix representation with 8-bit blocks is given by

$$LL = \begin{pmatrix} M_2 & M_3 & M_1 & M_1 \\ M_1 & M_2 & M_3 & M_1 \\ M_1 & M_1 & M_2 & M_3 \\ M_3 & M_1 & M_1 & M_2 \end{pmatrix},$$

where M_1 denotes the 8-bit identity, and M_2 and M_3 corresponds to the multiplication by $\mathtt{2}$ and $\mathtt{3}$ in $\mathbb{F}_{2^8}$, respectively.

Let $\overline{AL} = O \circ \oplus_k \circ LL \circ I$ be an intermediate encoded affine layer, where $O = O_1 \parallel O_2 \parallel O_3 \parallel O_4 \in \mathrm{RSE}(S)^4$. To reduce the encoding space of O_1, we can consider the centralizer problem

$$\mathrm{Lin}\left(\overline{AL}\right)_{1,2} \circ \left(\mathrm{Lin}\left(\overline{AL}\right)_{2,1}\right)^{-1} = X \circ M_3 \circ M_1^{-1} \circ X^{-1}, \quad X \in \mathrm{RSE}(S)$$

with solution set $\mathcal{S} = O_1 \circ (\mathrm{C}(M_3) \cap \mathrm{RSE}(S))$ following Proposition 2. Since $\mathtt{3}$ is a primitive element in $\mathbb{F}_{2^8}$, the centralizer of M_3 is the set of functions corresponding to the multiplication by non-zero $\mathbb{F}_{2^8}$-elements, i.e.,

$$\mathrm{C}(M_3) = \{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\} < \mathrm{RSE}(S)\,.$$

The set $\mathrm{RSE}(S)$ only contains 2040 elements, and we can simply brute force $\mathrm{RSE}(S)$ to obtain an arbitrary solution $X_0 = O_1 \circ M_\gamma$, for some unknown M_γ. Composing $C = (X_0^{-1} \parallel \mathrm{Id} \parallel \mathrm{Id} \parallel \mathrm{Id})$ with $\overline{AL}$ leads to a new encoded AL,

$$\widehat{AL} = C \circ \widehat{AL} = (M_\alpha \parallel O_2 \parallel O_3 \parallel O_4) \circ AL \circ I\,,$$

where the encoding space of the first block of the output encoding has been reduced to $\{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\}$.

While the centralizer problem does not provide a drastic reduction in this case, it shows how the self-equivalence structure of the linear layer also influences the success of the reduction. Thus, even if the AES S-box is replaced with another S-box with more self-equivalences, we can still reduce the encoding space of each block to $\{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\}$ thanks to the centralizers of the MixColumns blocks.

F Example of the Asymmetric Problem

We will show how the asymmetric problem can completely break a self-equivalence implementation of AES by reducing the encoding space to a set of one element. For ease of explanation, we will apply the asymmetric problem to the reduced self-equivalence implementation obtained in the previous example, where we applied the centralizer problem to reduce the encoding space corresponding to the first block of the output encoding to $\{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\}$.

Let $\overline{AL}$ be an intermediate encoded affine layer restricted to a column,

$$\overline{AL} = (O_1 \parallel O_2 \parallel O_3 \parallel O_4) \circ \oplus_k \circ LL \circ (I_1 \parallel I_2 \parallel I_3 \parallel I_4)\,,$$

such that $O_1 \in \{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\} < \mathrm{RSE}(S)$. To reduce the encoding space of O_1, we consider the asymmetric problem

$$\mathrm{Lin}\left(\overline{AL}\right)_{1,3} \circ S = X \circ M_1 \circ S \circ Y, \quad X \in \{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\}$$

that contains the particular X-solution O_1 according to Proposition 3. It is easy to see that this asymmetric problem is a linear equivalence problem, since $S(0) = 0$ and all the right self-equivalences of S are linear. Thus, the X-solution set is given by

$$\mathcal{S}_X = O_1 \circ (\mathrm{LSE}(S) \cap \{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\})\,.$$

Since this intersection is trivial, i.e., $\mathrm{LSE}(S) \cap \{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\} = \{\mathrm{Id}\}$, the asymmetric problem only has one solution.

We can simply perform an exhaustive search over the set $\{M_\alpha : 0 \neq \alpha \in \mathbb{F}_{2^8}\}$ containing 255 elements to obtain the unique solution $X_0 = O_1$, fully recovering the first block of the output encoding. Proceeding similarly with the rest of the output encoding blocks, we can reduce the round encoding space of $\overline{AL}$ to a set of one element. Afterwards, the round encodings and the master key can be recovered easily.