IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2021, No. 3, pp. 137–169. DOI:10.46586/tosc.v2021.i3.137-169

Boomeyong: Embedding Yoyo within Boomerang and its Applications to Key Recovery Attacks on AES and Pholkos

Mostafizar Rahman[1], Dhiman Saha[2] and Goutam Paul[1]

[1] Cryptology and Security Research Unit (CSRU), Indian Statistical Institute, Kolkata, India
mrahman454@gmail.com,goutam.paul@isical.ac.in
[2] de.ci.phe.red Lab, Department of Electrical Engineering and Computer Science, Indian Institute of Technology, Bhilai, Raipur, India
dhiman@iitbhilai.ac.in

Abstract. This work investigates a generic way of combining two very effective and well-studied cryptanalytic tools, proposed almost 18 years apart, namely the boomerang attack introduced by Wagner in FSE 1999 and the yoyo attack by Ronjom *et al*. in Asiacrypt 2017. In doing so, the *s-box switch* and *ladder switch* techniques are leveraged to embed a yoyo trail inside a boomerang trail. As an immediate application, a 6-round key recovery attack on `AES`-128 is mounted with time complexity of 2^{78}. A 10-round key recovery attack on recently introduced `AES`-based tweakable block cipher Pholkos is also furnished to demonstrate the applicability of the new technique on `AES`-like constructions. The results on `AES` are experimentally verified by applying and implementing them on a small scale variant of `AES`. We provide arguments that draw a relation between the proposed strategy with the retracing boomerang attack devised in Eurocrypt 2020. To the best of our knowledge, this is the first attempt to merge the yoyo and boomerang techniques to analyze SPN ciphers and warrants further attention as it has the potential of becoming an important cryptanalysis tool.

Keywords: AES · Boomerang · Distinguisher · Key Recovery · Pholkos · Symmetric-Key Cryptanalysis · Yoyo

1 Introduction

Cryptanalysis is one of the most important ways of determining the strength of a cryptosystem. Ever since the introduction of differential cryptanalysis by Biham and Shamir [BS91], a multitude of cryptanalytic techniques that build upon the basic idea of differential cryptanalysis has been proposed. Among these, a certain class of attacks particularly aims to divide a cipher into multiple sub-ciphers and study the sub-ciphers individually often analyzing the interactions between them. These methods find high probability trails (primarily due to the lesser number of rounds) for the sub-ciphers and compose them efficiently to mount an attack on the complete cipher. Some of the prominent candidates of this class are the boomerang attack [Wag99], amplified boomerang attack (rectangle attack) [KKS01], impossible differential attack [BBS99], rebound attack [MRST09]. These techniques have been widely applied to several ciphers: like the rectangle attack on Serpent [BDK01, BDK02], Kasumi [BDK05]; impossible differential attacks on `AES` [LDKK08, ZWF07, BDK06], CLEFIA, Camellia, LBlock, Simon, ARIA [WZF07, WZZ09, BNPS14, BMNPS14], Rijndael-160 and Rijndael-224 [Min17], rebound attack on Whirlpool and Grøstl [MRST09, MRST10], KECCAK [DGPW12] and boomerang attack on `AES` in single-key setting [Bir05] and in the

Received: 2021-06-01 Accepted: 2021-08-01 Published: 2021-09-17

Table 1: Comparisons of key recovery attacks on AES and Pholkos. Note that, time complexity is measured in terms of one AES and Pholkos encryption respectively (where no unit is mentioned). Memory complexity is measured in terms of memory required to store a single state of the primitive. All the attacks tabulated for AES are key recovery attacks. For 10-round Pholkos, key recovery attack using boomeyong is compared with the distinguishing attack given by the designers. CP and ACC are Chosen Plaintext and Adaptively Chosen Ciphertext respectively. Mix. Diff., Ret. Boom., and ETD refers to Mixture Differential, Retracing Boomerang and Extended Truncated Differential respectively.

Primitive	Attack Type	Complexity			Ref.
		Data	Time	Mem.	
5-round AES-128	Improved Square	2^{33} CP	2^{35}	Negl.	[FKL+01]
	Mix. Diff.	2^{24} CP	2^{24}	$2^{21.5}$	[BDK+18]
	Mix. Diff.	2^{32} CP	2^{34}	2^{32}	[Gra18]
	Yoyo Attack	$2^{13.3}$ ACC	2^{33}	Negl.	[RBH17]
	Partial Sum	2^{8} CP	2^{40}	Negl.	[Tun12]
	Ret. Boom.	2^{9} ACC	2^{23}	2^{9}	[DKRS20]
	Ret. Boom.	2^{15} ACC	$2^{16.5}$	2^{9}	[DKRS20]
	Boomeyong	**2^{49} ACC**	**2^{48} XOR**	**2^{23}**	**Section 4.1**
6-round AES-128	Yoyo Attack	$2^{122.8}$ ACC	$2^{121.8}$ XOR	Negl.	[RBH17]
	Exchange Attack	$2^{88.2}$ CP	$2^{88.2}$	Negl.	[BR19]
	Ret. Boom.	2^{26} ACC	2^{80}	2^{35}	[DKRS20]
	Partial Sum	$2^{34.5}$ CP	2^{44}	2^{32}	[FKL+01]
	ETD	$2^{71.3}$ CP	$2^{78.7}$	-	[BGL20]
	Boomeyong	**$2^{79.72}$ ACC**	**2^{78}**	**2^{28}**	**Section 4.2**
10-round Pholkos	Boomerang	2^{260} ACC	2^{260}	2^{32}	[BLLS20]
	Boomeyong	**$2^{189.8}$** ACC	**$2^{188.8}$** XOR	**2^{122}**	**Section 5.2**

related-key setting [BK09, BKN09, GL08, SSA10, FGL09]. A recent addition to the class include the retracing boomerang attack [DKRS20] and the extended truncated differential attack [BGL20] on AES. The retracing boomerang attack has been proposed in Eurocrypt 2020 by Dunkelman *et al.* and at the outset it tries to additionally spatially divide the sub-ciphers. The extended truncated differential attack mounts distinguishing and key-recovery attack on 5-round and 6-round AES by prepending a round that starts from the diagonal subspaces as proposed in [GRR17].

In particular, the boomerang attack is the center of interest concerning this work as the techniques developed here extensively rely on it. Boomerang attack, introduced by Wagner, makes use of two differentials to construct a distinguisher spanning over a large number of rounds when it is not possible to devise a single differential. As stated earlier, it conceptually divides a cipher into two sub-ciphers where each differential corresponds to each sub-cipher. Though initially thought to be independent, it has been shown that the differentials can rely on each other based on their interaction at the boundary of the sub-ciphers. The dependency can either lead to an incompatibility as shown by Murphy [Mur11] or can be exploited to improve the number of rounds as shown later by the idea of *s-box switch*, *ladder switch* [BKN09, BK09] and further generalized by the sandwich attack [DKS10, DKS14]. This also leads to the introduction of new tools like the boomerang connectivity table (BCT) [CHP+18], Feistel BCT [BHL+20] and the boomerang

distribution table (BDT) [WP19]. Another interesting cryptanalytic technique that is structurally similar to boomerang (though it does not divide the cipher into sub-ciphers) is the yoyo game which was introduced by Biham *et al.* to analyze Skipjack [BBD+98]. In Asiacrypt 2017, it has been used to devise a deterministic distinguisher for generic 2-round Substitution-Permutation Network (SPN) [RBH17] which leads to key recovery attacks on 5-round AES. The concept of yoyo game is further extended and applied to AES in known-key setting [SRP18] and on ForkAES [BBJ+19] in secret-key setting. Table 1 lists the complexities of the attacks presented in this paper on 5-round AES-128 (variant of AES with key length of 128 bits), 6-round AES-128 and 10-round Pholkos respectively along with the other attacks.

Our Contribution

The work investigates the yoyo technique further to essentially extend the number of rounds that it can penetrate. The new approach can be visualized like an embedding of the yoyo game inside a boomerang trail, where the upper trail of the boomerang essentially conforms to the yoyo while the lower trail is a standard but specially crafted differential trail. It applies the concept of the s-box switch and the ladder switch in the boundary of the upper and lower trail. The primary motivation is to construct the lower trail in such a way that the difference added to the ciphertexts leads to a yoyo *word-swap* in the boundary of upper and lower trails. This in turn satisfies the essential criteria of the yoyo and leads to return of the yoyo with probability 1 which can be verified at the top, like the classical yoyo trick. We prove how the s-box switch and ladder switch help achieve the required word-swap (see Fig. 5). The proof idea stems from the fact that the words that swap can be mapped to an equivalent s-box switch while the words that remain unchanged make a ladder switch. The price we pay is the construction of a truncated differential trail superimposed on the yoyo which behaves like the upper trail of the boomerang. This is the motivation for using the term *embedding* while visualizing this setting. So in classical boomerang terms if the truncated upper trail has a complexity p and the lower trail has a complexity q, owing to the word-swap happening at the boundary, the complexity of the complete boomerang distinguisher is pq^2. We save a factor of p while going up due to the yoyo property.

As a natural application, first of all, $5(= 4 + 1)$-round AES is considered, where the yoyo covers the first 4 rounds constituting the upper trail and the lower trail covers 1 round. By embedding yoyo within boomerang, first a distinguisher is reported at the expense of 2^{47} oracle queries contributing to the data complexity and 2^{46} XOR operations contributing to the time complexity. The distinguisher is used further to correctly recover the secret key of AES-128 (variant of AES with key length of 128 bits) with the time complexity of 2^{48} XOR operations. The next result is the application to 6-rounds which is achieved by sandwiching the yoyo in-between a classical 1-round differential on top and the lower boomerang trail developed in the 5-round attack. The result is a key recovery attack on 6-round AES-128 with the time complexity of 2^{78} AES encryptions and the data complexity of $2^{79.72}$ adaptive chosen ciphertexts. Note that, the distinguishing attack on the AES is independent of the key size whereas the key recovery attacks described in the paper are applicable on AES-128. However, the key recovery attacks can be further extended to recover the key of 6/7-round of a variant of AES with 256-bit key. In the rest of the paper, unless otherwise mentioned, AES-128 is referred to as AES.

Finally, to show the versatility of the strategy a 10-round key recovery attack is mounted on a very recently proposed AES based tweakable block cipher Pholkos. We support all our claims with theoretical arguments. The combination of the two strategies seems to be an interesting proposition and may lead to improved results for other SPN ciphers as well thereby providing better insights. One can appreciate the fact that the proposed technique bears structural similarity with some of the well-known results. For instance, the 6-round

Table 2: Key recovery attacks reported in this work. ACC is adaptive chosen ciphertexts.

Attack	Complexity			Ref.
	Data	Time	Memory	
5-round AES	2^{49} ACC	2^{48} XOR	2^{23}	Section 4.1
6-round AES	$2^{79.72}$ ACC	2^{78}	2^{28}	Section 4.2
10-round Pholkos †	$2^{189.8}$ ACC	$2^{188.8}$ XOR	2^{122}	Section 5.2

† 512-bit key

attack can easily be seen in the framework of the sandwich attack where 4 rounds of AES form the middle layer. In that sense, this work reports the first result where the middle layer consists of 4 rounds of AES. On the other hand, the attack can also be shown to have a close relation to the retracing attack, a discussion on which is furnished later (See Section 6). The contributions of the current work are summarized in Table 2.

Organization of the paper

The rest of the paper is organized as follows. In Section 2, we briefly discuss about boomerang attack, yoyo attack, AES and the significance of signal-to-noise ratio in differential cryptanalysis. The notion of embedding yoyo within boomerang is introduced and thoroughly illustrated in Section 3. In Section 4, the developed cryptanalytic technique is applied on 5-round and 6-round AES to mount key recovery attacks. As an additional application of the developed techniques, a 10-round attack on Pholkos [BLLS20] is shown in Section 5. Section 6 illustrates the close relation between retracing boomerang attack and the attacks presented in this paper. Finally, the concluding remarks are furnished in Section 7. In Appendix C, the key recovery attacks on 5-round and 6-round AES-128 are extended to recover the key of a variant of AES with key size of 256 bits.

2 Preliminaries

This section describes the pre-requisites for this paper. First, a brief description of AES is provided. Then, the boomerang attack and the yoyo attack are described briefly with necessary results. Finally, a short discussion on signal-to-noise ratio in the context of differential cryptanalysis is provided.

2.1 AES: The Advanced Encryption Standard

AES, designed by Joan Daemen and Vincent Rijmen, is an iterated block cipher with 128-bit data blocks [AES01, DR02]. Depending on the key length, it has three variants- i) AES-128- it uses a 128-bit key, ii) AES-192- it uses a 192-bit key and iii) AES-256- it uses a 256-bit key. The number of rounds in AES-128, AES-192 and AES-256 are 10, 12 and 14 respectively.

128-bit plaintext in AES is represented by a 4×4 byte matrix called state. The rows and columns of the state are both numbered from 0 to 3. In each round, four transformations are applied to an AES state. They are-

- SubBytes (SB)- It is a non-linear substitution operation applied to each byte of AES state in parallel.
- ShiftRows (SR)- It cyclically shifts left different rows of the state by different offsets. In general, for $0 \leq i \leq 3$, i-th row is cyclically shifted left by i bytes.

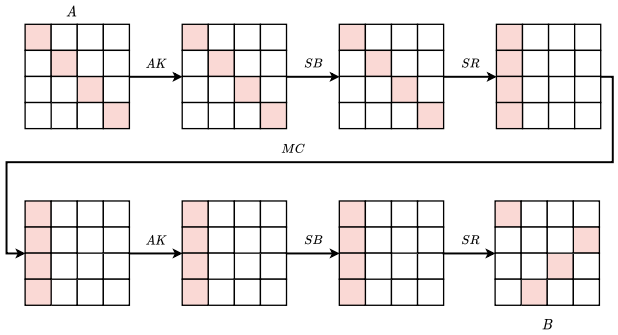


Figure 1: AES Super-Sbox

- MixColumns (MC)- It is column-mix operation. For applying this operation, a 4×4 constant maximum distance separable (MDS) matrix is used. Note that, in the context of differential cryptanalysis, in the input and output of mixcolumns, out of 8 bytes at least 5 bytes should be active. So, if there are 4 active bytes in the input of MC, then there must be at least one active byte in the output. In the rest of the paper, a term *4-to-1* property is used which denotes the transition from 4 active bytes to 1 active byte via MC. *4-to-1* property occurs with probability $4 \times 2^{-24} = 2^{-22}$.

- AddRoundKey (AK)- This operation is the XOR-ing of subkey with the AES state. The subkeys for each round are generated by key scheduling algorithm.

All the operations discussed above are invertible. In the last round, MC is omitted and before the start of the first round, AK is applied to the state. In this paper, a special construction named Super-Sbox [DR06] is used for applying the attacks. Details regarding this are now discussed.

Super-SBox. Super-Sbox [DR06] was introduced and first studied by Daemen and Rijmen in 2006. Refer to Fig. 1 for Super-Sbox construction in AES. Consider the diagonal in A (four red-colored bytes). After the application of AK, SB and SR, those four bytes aligns in a column. The following MC affects only that column. As AK and SB are byte-wise operations, those four bytes remain independent of the other 12 bytes. The last SR operation aligns the bytes to an inverse diagonal in B. These four bytes in B are dependent on the four bytes in A only through the 1.5 rounds. This is conceptualized as Super-Sbox with 32-bit input and 32-bit output. In general, an inverse diagonal in B is uniquely determined by a diagonal in A. There are four Super-Sbox in AES state. Fig. 2 depicts the four parallel Super-Sbox.

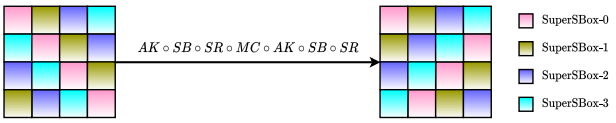


Figure 2: 4 Parallel AES Super-Sbox

2.2 Boomerang Attack

Boomerang attack which was given by Wagner is an extension of differential cryptanalysis [Wag99]. It attempts to construct a trail for a cipher by a quartet of plaintexts combining two differential trails. It decomposes the cipher into two parts- the upper and lower; likewise, the differential trails corresponding to these parts are called the upper trail and the lower trail. Consider a cipher E which is decomposed into E_0 and E_1, with E_0 being the upper part. Let $Pr[\alpha \xrightarrow{E_0} \beta] = p$ and $Pr[\gamma \xrightarrow{E_1} \delta] = q$ and initially assume $p = q = 1$. The boomerang attack works in the following way.

1. Choose a pair of plaintext P^1, P^2 such that $P^1 \oplus P^2 = \alpha$. Encrypt them using E to obtain C^1, C^2 respectively.
2. Construct C^3, C^4 such that $C^1 \oplus \delta = C^3$ and $C^2 \oplus \delta = C^4$. Decrypt them using E to obtain P^3, P^4 respectively. The value $P^3 \oplus P^4$ should be α.

As $Pr[\alpha \xrightarrow{E_0} \beta] = 1$, $E_0(P^1) \oplus E_0(P^2) = \beta$. Also, $E_1^{-1}(C^1) \oplus E_1^{-1}(C^3) = E_1^{-1}(C^2) \oplus E_1^{-1}(C^4) = \gamma$ as $Pr[\gamma \xrightarrow{E_1} \delta] = 1$. Note that, $E_0(P^i) = E_1^{-1}(C^i)$.

Let E encrypts P^1, P^2, P^3 and P^4 to obtain C^1, C^2, C^3 and C^4. Q^1, Q^2, Q^3 and Q^4 are intermediate encrypted values of P^1, P^2, P^3 and P^4 $\big(Q^i = E_0(P^i) = E_1^{-1}(C^i)\big)$. Then $Q^1 \oplus Q^2 = \beta$ and $Q^1 \oplus Q^3 = Q^2 \oplus Q^4 = \gamma$. Now,

$$\begin{aligned} Q^3 \oplus Q^4 &= E_0(P^3) \oplus E_0(P^4) \\ &= \big(E_0(P^1) \oplus E_0(P^2)\big) \oplus \big(E_0(P^1) \oplus E_0(P^3)\big) \oplus \big(E_0(P^2) \oplus E_0(P^4)\big) \\ &= \big(Q^1 \oplus Q^2\big) \oplus \big(Q^1 \oplus Q^3\big) \oplus \big(Q^2 \oplus Q^4\big) \\ &= \beta \end{aligned}$$

As $p = 1$, so $P^3 \oplus P^4 = \alpha$. Note that, for any arbitary p, q, $P^3 \oplus P^4 = \alpha$ with probability p^2q^2 under the assumption that the upper and the lower trail are independent. Biryukov and Khovratovich further improved the boomerang attack by introducing the concept of *s-box switch* and *ladder switch* [BKN09, BK09]. These notions add dependency between the upper and the lower trail.

S-box Switch and Ladder Switch. Assume, that the last substitution layer in E_0 partitions the state into t parts, *i. e.*, $Q^1 = Q_0^1||Q_1^1 \cdots Q_{t-1}^1$ and $Q^2 = Q_0^2||Q_1^2 \cdots Q_{t-1}^2$. In similar way, β and γ can also be partitioned. Let the last substitution layer in E_0 be S and $S^{-1}(Q_i^1) \oplus S^{-1}(Q_i^2) = \phi$. Consider the i-th partition.

$$\begin{aligned} Q_i^3 &= Q_i^1 \oplus \gamma_i \\ Q_i^4 &= Q_i^2 \oplus \gamma_i \end{aligned}$$

For satisfying the E_0 trail, $S^{-1}(Q_i^3) \oplus S^{-1}(Q_i^4) = \phi$ must hold. If $Pr[\phi \xrightarrow{S} \beta_i] = q'$, then q'^2 probability needs to be paid for satisfying this trail. Now, analyze two special cases.

- Case 1 [$\gamma_i = \beta_i$]: In such cases, $Q_i^3 = Q_i^2$ and $Q_i^4 = Q_i^1$. Now,

  $$S^{-1}(Q_i^3) \oplus S^{-1}(Q_i^4) = S^{-1}(Q_i^2) \oplus S^{-1}(Q_i^1) = \phi$$

  Therefore, in such cases probability for one side needs to be paid and other side occurs deterministically; which improves the overall probability by a factor of q'. This is known as *s-box switch*.

- Case 2 [$\gamma_i = 0$]: In such cases, $Q_i^3 = Q_i^1$ and $Q_i^4 = Q_i^2$. Observe that, irrespective of the value of β_i, $S^{-1}(Q_i^3) \oplus S^{-1}(Q_i^4) = \phi$ always holds. The trail probability is improved by a factor of q'^2. This is referred to as *ladder switch*.

Figure 3: Illustration of Zero Difference Pattern (ZDP) for an AES state. The gray colored bytes denotes the active ones, whereas the white ones denote the inactive bytes. In this case, if each diagonal is considered as a word, then the ZDP of Fig. 3a and Fig. 3b are $(1, 0, 1, 0)$ and $(0, 1, 0, 1)$ respectively; whereas if inverse diagonals are considered as a word, then the ZDP becomes $(1, 1, 1, 0)$ and $(0, 1, 0, 1)$. Note that, the diagonal (inverse diagonal) containing the i-th byte $(0 \leq i \leq 3)$ of the first row is considered as i-th diagonal (inverse diagonal).

In [BKN09, BK09], these two notions were exploited to mount related key boomerang attacks on AES-192 and AES-256. Later on, Dunkelman *et al.* formalized these notions by dividing the cipher into E_0, E_m and E_1 [DKS10]. Cid *et al.* considered the E_m as a S-box layer and developed a tool *Boomerang Connectivity Table* in order to unify the *s-box switch* and the *ladder switch* [CHP+18]. Further, to realize the switching effect on multiple rounds *Boomerang Difference Table* was proposed [WP19].

2.3 Yoyo Attack

Yoyo game is a cryptanalytic technique that was first introduced by Biham *et al.* [BBD+98] and was applied to analyze Skipjack [NSA98]. It is an adaptive chosen ciphertext/plaintext based strategy that is used to identify pairs of texts which satisfy a certain invariant property. Later, yoyo-based technique was applied on Feistel ciphers [BLP15]. In Asiacrypt 2017, Rønjom *et al.* used this strategy to devise a deterministic distinguisher for generic 2-Substitution Permutation rounds [RBH17]. Further, this result is applied and extended to mount attacks on 5-round and 6-round AES. As the existing result on the generic 2-round substitution-permutation networks are reused, so the existing definitions and notations in [RBH17] are reviewed.

Definition 1. Zero Difference Pattern (ZDP) [RBH17] Let $\alpha \in \mathbb{F}_q^n$ for $q = 2^k$ and $z_i = 1$ if $\alpha_i = 0$ or $z_i = 0$ otherwise. Then the Zero Difference Pattern for α is $\nu(\alpha) \in \mathbb{F}_2^n$ and is defined as

$$\nu(\alpha) = (z_0, z_1, ..., z_{n-1}).$$

Fig. 3 illustrates the notion of ZDP for AES state. Next, the procedure of exchanging words between two states are defined.

Definition 2. Swapping of Words [RBH17] Given a vector $v \in \mathbb{F}_2^n$ and $\alpha, \beta \in \mathbb{F}_q^n$ be two states, then a new state $\rho^v(\alpha, \beta) \in \mathbb{F}_q^n$ is created from α, β by exchanging components among them. The i^{th} component of $\rho^v(\alpha, \beta)$ is defined as

$$\rho^v(\alpha, \beta)_i = \begin{cases} \alpha_i, & \text{if } v_i = 1; \\ \beta_i, & \text{if } v_i = 0. \end{cases} \tag{1}$$

Note that, the weight of a vector v is denoted by $wt(v)$. In other words, it can be said that while constructing $\rho^v(\alpha, \beta)$ from α, $(n - wt(v))$ words of α are swapped with β.

Now, the following proposition dictates the process of devising a deterministic distinguisher for generic 2-round substitution-permutation network (G_2).

Proposition 1. *[RBH17] Let p^0, $p^1 \in \mathbb{F}_q^n$ and they are encrypted using G_2 to obtain c^0, c^1, i. e., $c^0 = G_2(p^0)$ and $c^1 = G_2(p^1)$. Two new states c'^0, c'^1 are constructed using a vector $v \in \mathbb{F}_2^n$, i. e., $c'^0 = \rho^v(c^0, c^1)$ and $c'^1 = \rho^v(c^1, c^0)$. Then*

$$\nu(G_2^{-1}(c'^0) \oplus G_2^{-1}(c'^1)) = \nu(p'^0 \oplus p'^1) = \nu(p^0 \oplus p^1).$$

Proposition 1 can be applied directly to devise a deterministic distinguisher for 4-round `AES` [RBH17]. 4-round `AES` can be considered as $S \circ L \circ S$ layer where S corresponds to the Super-Sbox layer and L corresponds to mixcolumns operation. Suppose, two plaintexts p^0, p^1, where $\nu(p^0 \oplus p^1) = (0, 1, 1, 1)$ are encrypted using 4-round `AES` to obtain c^0, c^1. Consider a vector $v = (0, 1, 1, 1)$ and two new states c'^0, c'^1 are constructed using v as stated in Proposition 1 (the inverse diagonal containing the byte (0,0) is swapped between c^0, c^1). Now, c'^0, c'^1 are decrypted using 4-round `AES` to obtain p'^0, p'^1. According to Proposition 1, $\nu(p'^0 \oplus p'^1) = (0, 1, 1, 1)$ must holds.

2.4 Signal-to-Noise Ratio

While mounting a key recovery attack, situations may arrive when it is not possible to distinguish the right pair from the wrong ones. In such cases, the notion of *signal-to-noise* ratio is used. *Signal-to-noise* ratio (S/N) is used to determine the number of right pairs required to recover the right key. The right key can be suggested by both right pairs or wrong pairs. The ones that are suggested by the right pair are called signal whereas the ones that are suggested by the wrong pair are called noise. Let M be the number of pairs queried by the adversary and p be the probability of the characteristic. Then the number of right pair is Mp. As each right pair suggests the right key one time, hence the amount of signal is Mp. Now the number of wrong pairs is $M(1-p)$. Let a filtering technique is used and a wrong pair survive the filtering with probability β. Therefore, after filtering the remaining number of wrong keys is $M(1-p)\beta$. Consider η be the average number of key candidates suggested by the wrong pair. Note that, such suggestions consist of both right and wrong key candidates. So, the total number of keys suggested by wrong pairs is $M(1-p)\beta\eta$. Under the assumption that the keys suggested by wrong pairs are uniformly distributed, the amount of noise is $M(1-p)\beta\eta 2^{-k}$, where k is the length of the guessed key in bits. Therefore *signal-to-noise* ratio is $\frac{Mp}{M(1-p)\beta\eta 2^{-k}} = \frac{2^k p}{(1-p)\beta\eta}$.

A counter is maintained for each key suggested by either right or wrong pairs. The value of the counter for each key depends on the *signal-to-noise* ratio. If $S/N > 1$ then the right key is suggested more than the other keys whereas for $S/N < 1$ the right key is suggested fewer times than the wrong ones. By analysing the counters, the right key can be detected. More details regarding the *signal-to-noise* ratio is provided in [KR11, SSL15]. For $S/N > 1$, generally the candidate key with highest counter value is considered as the right candidate. But cases may arrive when the counter value for the right candidate is not maximum. Thus, as stated in [SSL15], several key candidates whose counter value is close the highest one is considered as the candidate key. This method is known as ranking test.

In the context of differential cryptanalysis, the relation between the number of right pairs required to identify the unique key, the number of key candidates whose counter value is close to the highest one and the success probability was given by Selçuk in [Sel08]. Let M be the number of pairs queried to the oracle, p be the probability of the characteristic and let k be the length of the guessed key in bits. Without loss of generality, let the right key be denoted by $\mathcal{K}_0$ and $\mathcal{K}_1, \cdots, \mathcal{K}_{2^k-1}$ denote the wrong keys. A plaintext pair suggests $\mathcal{K}_i$ as key candidate with probability p_i and the counter value for each $\mathcal{K}_i$ is T_i. Under the assumption that T_i's are independent and identically distributed (i.i.d.), the probability of any of the wrong keys being suggested as the right one is the same and is denoted by

p_w. For $1 \leq i \leq 2^k - 1$, T_i follows the binomial distribution $\mathcal{B}(M, p_w)$ and T_0 follows the binomial distribution $\mathcal{B}(M, p_0)$. For large values of M, these binomial distributions can be approximated by normal distribution $\mathcal{N}(\mu_w, \sigma_w^2)$ and $\mathcal{N}(\mu_0, \sigma_0^2)$ where

$$\mu_0 = p_0 M, \qquad \sigma_0^2 = p_0(1-p_0)M \approx p_0 M$$
$$\mu_w = p_w M, \qquad \sigma_w^2 = p_w(1-p_w)M \approx p_w M.$$

The right keys are deterministically suggested by the right pairs and probabilistically suggested by the wrong pairs; whereas wrong keys are probabilistically suggested by both the right and the wrong pairs. If a certain key is suggested as the right key candidate by a random pair with probability p_r, then

$$p_0 = p + (1-p)p_r \approx p + p_r$$
$$p_w = p_r.$$

The attack is successfully performed if $\mathcal{K}_0$ is ranked among the top r candidates on the basis of the counter values. Let ϕ be the probability density function and Φ be the cumulative distribution function. Then the success probability P_s can be given by

$$P_s = \int_{-\frac{\mu_0 - \mu_q}{\sqrt{\sigma_0^2 + \sigma_q^2}}}^{\infty} \phi(x)dx,$$

where $\sigma_q = \frac{\sigma_w}{\phi(\Phi^{-1}(1-2^{log_2 r - k}))} 2^{-\frac{2k - log_2 r}{2}}$ and $\mu_q = \mu_w + \sigma_w \Phi^{-1}(1 - 2^{log_2 r - k})$ [Sel08]. Based on this, the following propositions connect success probability, data complexity and the number of top ranked values that should be considered as right key candidate.

Proposition 2. *[Sel08] Let the correct key $\mathcal{K}_0$ of length k is among the top r values of key counters with probability P_s when a differential attack with characteristic probability p is mounted using M plaintext-ciphertext pairs and signal-to-noise ratio of S_N. Under the assumptions that the counters corresponding to the wrong keys are independent and follows an identical distribution and the value of k and M is too large, then P_s can be expressed as a function of the other variables by the following equation:*

$$P_s = \Phi\left(\frac{\sqrt{pMS_N} - \Phi^{-1}(1 - 2^{log_2 r - k})}{\sqrt{S_N + 1}}\right)$$

Proposition 3. *[Sel08] Let the correct key $\mathcal{K}_0$ of length k is among the top r values of key counters with probability P_s when a differential attack with characteristic probability p is mounted using M plaintext-ciphertext pairs and signal-to-noise ratio of S_N. Under the assumptions that the counters corresponding to the wrong keys are independent and follows an identical distribution, the value of k and M is too large, then M can be expressed as a function of the other variables by the following equation:*

$$M = \frac{(\sqrt{S_N + 1}\Phi^{-1}(P_s) + \Phi^{-1}(1 - 2^{log_2 r - k}))^2}{S_N} p^{-1}.$$

These two propositions are used to estimate the success probability of the boomeyong attacks on 6-round `AES` and 10-round Pholkos.

Now, the details regarding the process of embedding yoyo within boomerang to devise a new cryptanalytic tool boomeyong is discussed.

3 Boomeyong: Embedding Yoyo within Boomerang

The central notion of this work is to devise a cryptanalytic technique by combining two powerful techniques: yoyo and boomerang. The same conceptual division as used in the boomerang attack is considered for embedding yoyo within boomerang leading to a new strategy which we call *boomeyong*. Proposition 1 states that there is a deterministic distinguisher for $S \circ L \circ S$ construction irrespective of the internal structure of S and L layer (Here, S corresponds to the substitution layer and L corresponds to linear layer). The trick is to use this $S \circ L \circ S$ layer as the upper trail in devising the boomerang trail. The problem of embedding yoyo game within boomerang is that the previous is based on classical differential whereas for the latter one truncated forms are considered. Refer

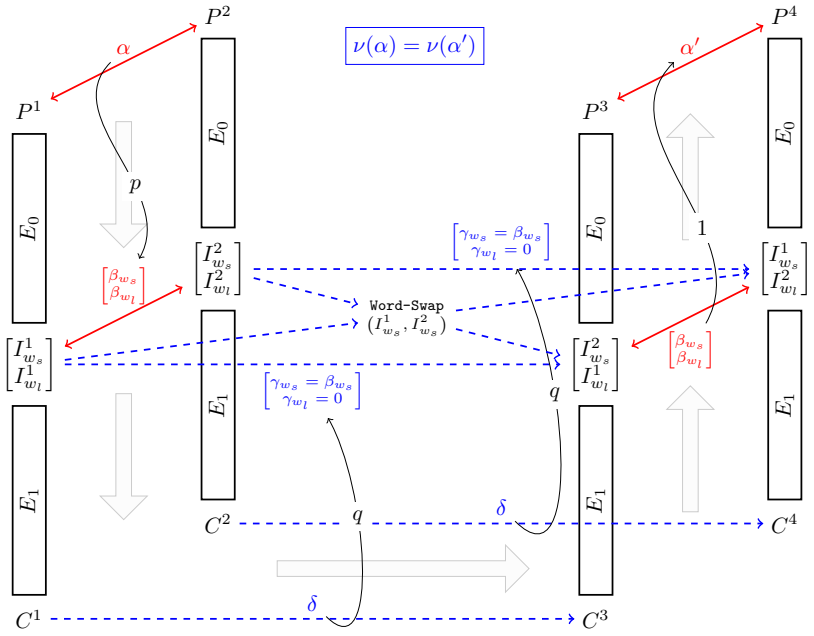


Figure 4: Embedding yoyo within boomerang. Note that, for the yoyo game E_0 corresponds to $S \circ L \circ S$ layer, whereas for the boomerang there is no such constraints. Here, the trail superimposed on yoyo is $\alpha \to \beta$ with a probability p. The words of β that are intended to be swapped are denoted by β_{w_s}. These words will be switched using corresponding words in the lower trail $\delta \to \gamma$ which holds with probability q using the idea of s-box switch. The remaining words γ i.e. γ_{w_l} in the lower trail are zero thereby leading to a ladder switch of the corresponding words in β i.e. β_{w_l}. Note that $\Pr[\beta \to \alpha'] = 1$ due to the yoyo trick.

to Fig. 4 for the attack. Let $E : \mathbb{F}^n_{2^k} \mapsto \mathbb{F}^n_{2^k}$ be a cipher which is divided into two parts: E_0 (upper) and E_1 (lower). E_0 is comprised of initial $S \circ L \circ S$ layers and the remaining parts of the cipher is considered as E_1. Now, P^1, P^2, P^3 and P^4 be four plaintexts which are encrypted by E to obtain C^1, C^2, C^3 and C^4 respectively. Aim is to simulate yoyo game in the upper trail E_0. Let $Q^i = E_0(P^i)$ for $1 \leq i \leq 4$. Therefore, if P^1, P^2 is considered as initial pair, then by virtue of yoyo game $\nu(P^1 \oplus P^2) = \nu(P^3 \oplus P^4)$. This also implies that Q^3, Q^4 can be obtained by swapping words between Q^1, Q^2. Therefore, $Q^1 \oplus Q^2 = Q^3 \oplus Q^4 = \beta$ (say). Consider, $P^1 \oplus P^2 = \alpha$, $P^3 \oplus P^4 = \alpha'$. The difference of boomerang with the attack developed in this work is that for the former one $\alpha = \alpha'$, whereas for the latter one $\alpha = \alpha'$ does not hold always; instead $\nu(\alpha) = \nu(\alpha')$ must hold.

Constructing the lower trail is quite similar to the construction of the lower trail in the

boomerang attack. Let $Q^1 \oplus Q^3 = \gamma$, which gives $Q^2 \oplus Q^4 = Q^1 \oplus Q^3 = \gamma$. Now, for the lower half a trail $\delta \xrightarrow{E_1^{-1}} \gamma$ needs to be constructed. For realizing the 'Swapping of Words' in the middle (the boundary of E_0 and E_1), a special kind of relationship must exist between β and γ.

Theorem 1. *Let Q^1, $Q^2, \gamma \in \mathbb{F}_{2^k}^n$ and $Q^1 \oplus Q^2 = \beta$. Consider, $Q^i = Q_1^i || Q_2^i || \cdots || Q_n^i$, $\beta = \beta_1 || \cdots || \beta_n$ and $\gamma = \gamma_1 || \cdots || \gamma_n$. If $J \subset \{1, \cdots, n\}$, $J \neq \emptyset$ and for all $j \in J$, $\gamma_j = \beta_j$; otherwise, $\gamma_j = 0$, then, $Q^1 \oplus \gamma$, $Q^2 \oplus \gamma$ can be formed by swapping words between Q^1, Q^2.*

Proof. Construct $v \in \mathbb{F}_2^n$ as

$$v_j = \begin{cases} 0, & \text{if } j \in J; \\ 1, & \text{Otherwise.} \end{cases}$$

for $(1 \leq j \leq n)$. Now, following definition 2 construct $\rho^v(Q^1, Q^2)$. For $(1 \leq j \leq n)$

$$\rho^v(Q^1, Q^2)_j = \begin{cases} Q_j^1, & \text{if } v_j = 1; \\ Q_j^2, & \text{if } v_j = 0. \end{cases}$$

$$\implies \rho^v(Q^1, Q^2)_j = \begin{cases} Q_j^1, & \text{if } \gamma_j = 0; \\ Q_j^2, & \text{if } \gamma_j = \beta_j. \end{cases}$$

$$\implies \rho^v(Q^1, Q^2)_j = \begin{cases} Q_j^1, & \text{if } \gamma_j = 0; \\ Q_j^1 \oplus \beta_j, & \text{if } \gamma_j = \beta_j. \end{cases}$$

$$\implies \rho^v(Q^1, Q^2)_j = Q_j^1 \oplus \gamma_j$$

Therefore, $\rho^v(Q^1, Q^2) = Q^1 \oplus \gamma$. In similar way, it can be proved that $\rho^v(Q^2, Q^1) = Q^2 \oplus \gamma$. □

Theorem 1 states that the words in γ either should be zero or equal the value of the same word in β. This ensures that in the middle swapping of words has taken place between the initial pair and thus for E_0 yoyo game is run. Fig. 5 shows the swapping mechanism in the middle.

For the upper trail E_0, α is not fixed; instead $\nu(\alpha)$ is fixed. Let $Pr[\{\alpha | \nu(\alpha) = t\} \xrightarrow{E_0} \beta] = p$ and $Pr[\delta \xrightarrow{E_1^{-1}} \gamma] = q$. Therefore, at the cost of pq^2 probability Q^3, Q^4 are formed by swapping words between Q^1, Q^2 and thus with the same probability it is expected that $\nu(P^3 \oplus P^4) = \nu(P^1 \oplus P^2)$. Let $wt(\nu(P^1 \oplus P^2)) = t$. If E is a random permutation, then this event would occur with probability 2^{-tk}. While embedding yoyo within the boomerang distinguishers, such upper and lower trails should be considered for which $pq^2 > 2^{-tk}$.

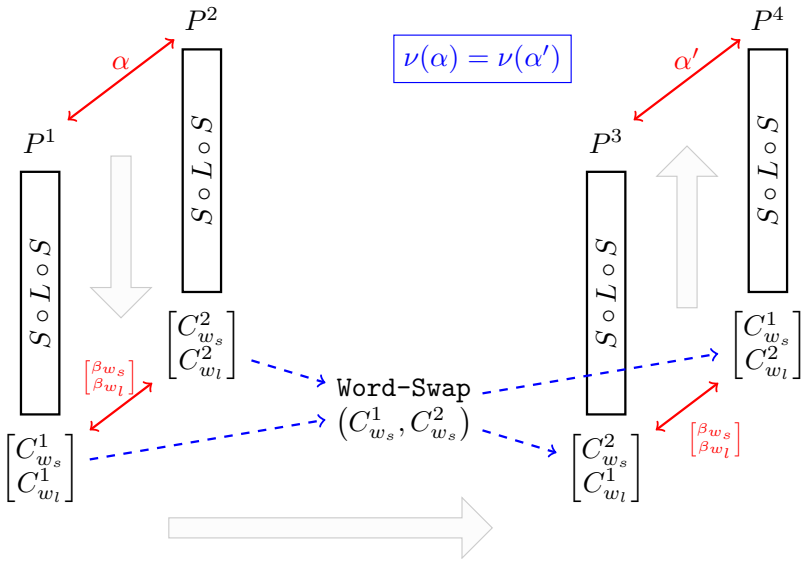
Attack Idea. Based on the analysis, the following are the steps of devising a distinguisher by embedding yoyo within boomerang. Suppose, access to oracle **O** is given and the distinguisher tries to distinguish that whether **O** is E or a random permutation.

1. Choose two plaintext P^1, P^2 such that $wt\big(\nu(P^1 \oplus P^2)\big) = t$. Encrypt them using **O** to obtain C^1, C^2 respectively.
2. Prepare $C^3 = C^1 \oplus \delta$, $C^4 = C^2 \oplus \delta$ and decrypt them by **O** to obtain P^3, P^4.
3. Check whether $\nu(P^1 \oplus P^2) = \nu(P^3 \oplus P^4)$ or not.
4. If $\nu(P^1 \oplus P^2) = \nu(P^3 \oplus P^4)$, then distinguish **O** as E; otherwise, repeat step 1 to step 3 $\frac{1}{pq^2}$ times. Even after repeating $\frac{1}{pq^2}$ if distinguisher fails to find a quartet (P^1, P^2, P^3, P^4) such that $\nu(P^1 \oplus P^2) = \nu(P^3 \oplus P^4)$, then distinguish **O** as a random permutation.

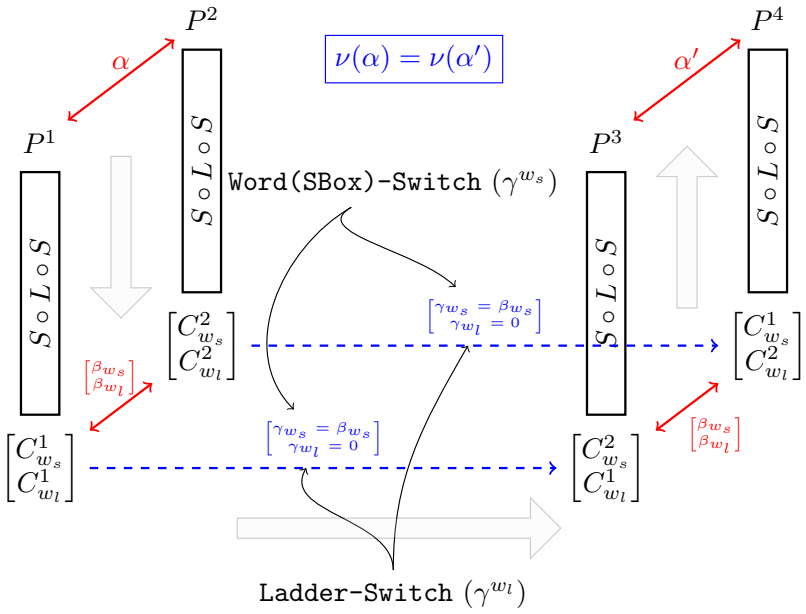
Now, the techniques developed here are extensively applied to 5-round and 6-round AES and 10-round Pholkos.

4 Boomeyong Attacks on AES

In the previous section, it is shown how to embed yoyo within a boomerang. The first application of this technique is mounting attacks on 5-round and 6-round AES. The main



(a) Yoyo Word-Swap



(b) Equivalent S-box and Ladder switch

Figure 5: Visualizing Yoyo Word-Swap as a combination of S-box switch and Ladder switch operations

disadvantage of appending a boomerang trail under the yoyo is that it is no longer possible to swap words between ciphertexts deterministically. In this regard, first of all, a probabilistic yoyo game needs to be devised. The next two definitions define the diagonals, inverse diagonals and columns of an AES state. The notation $\subset_\phi$ is used to denote non-null proper subset.

Definition 3. [GRR17] For a set $I \subset_\phi \{0,1,2,3\}$, let $\mathcal{C}_I = \{(i,j) : i \in S, j \in I\}$. For an AES state X, a set of columns I is represented by $\mathcal{C}_I(X)$.

Definition 4. [GRR17] For a set $I \subset_\phi \{0,1,2,3\}$, let $\mathcal{D}_I = \{(i, j+i \mod 4) : i \in S, j \in I\}$. For an AES state X, a set of diagonals I is represented by $\mathcal{D}_I(X)$.

Definition 5. [GRR17] For a set $I \subset_\phi \{0,1,2,3\}$, let $\mathcal{ID}_I = \{(i, j-i \mod 4) : i \in S, j \in I\}$. For an AES state X, a set of inverse diagonals I is represented by $\mathcal{ID}_I(X)$.

The following two lemmas provide the basis of devising a probabilistic yoyo game for AES. The main motivation of devising such a game is to penetrate more rounds at the expense of probability. For 5-round AES, the aim is to add such a difference in the ciphertext so that in the fourth round before mixcolumns swapping of inverse diagonals is realized.

Lemma 1. *Let $I, J \subset_\phi \{0,1,2,3\}$ and $p^1, p^2 \in \mathbb{F}_{2^8}^{4\times 4}$. Then the probability that a set of inverse diagonals J are swapped between p^1, p^2, given that a set of columns I are swapped is given by $P_{ID}(|I|,|J|) = 2^{-8\times\left(4(|I|+|J|)-2|I||J|\right)}$.*

Proof. Note that, $|\mathcal{C}_I \cap \mathcal{ID}_J| = |I||J|$. So, $|\mathcal{C}_I \cup \mathcal{ID}_J| = 4(|I|+|J|) - |I||J|$. Among all these $\mathcal{C}_I \cup \mathcal{ID}_J$ bytes, if the bytes only in $\mathcal{C}_I \cap \mathcal{ID}_J$ are active between p^1, p^2 then column swap is equivalent to inverse diagonal swap. Therefore, bytes in $(\mathcal{C}_I \cup \mathcal{ID}_J) \setminus (\mathcal{C}_I \cap \mathcal{ID}_J)$ needs to be inactive. $|(\mathcal{C}_I \cup \mathcal{ID}_J) \setminus (\mathcal{C}_I \cap \mathcal{ID}_J)| = 4(|I|+|J|) - 2|I||J|$. Hence, the required probability $P_{ID}(|I|,|J|) = 2^{-8\times\left(4(|I|+|J|)-2|I||J|\right)}$ is achieved. □

Lemma 2. *Let $p^1, p^2 \in \mathbb{F}_{2^8}^{4\times 4}$ and $c^1 = f(p^1), c^2 = f(p^2)$, where f is $MC \circ AK \circ SB \circ SR \circ AK$. Then the probability of occurence of certain p^1, p^2, such that swapping of a set of inverse diagonals $I \subset_\phi \{0,1,2,3\}$ between c^1, c^2 is equivalent to swapping of inverse diagonals between p^1, p^2 is given by $P_{swap}(|I|) = \sum_{j=1}^{3} \binom{4}{j} P_{ID}(|I|, j)$.*

Proof. It is easy to visualize that due to SR, swapping of $\mathcal{ID}_I$ between c^1, c^2 is equivalent to swapping of $\mathcal{C}_I$ between p^1, p^2. Lemma 1 states that swapping of $\mathcal{C}_I$ is equivalent to swapping of $\mathcal{ID}_J$ (where $J \subset_\phi \{0,1,2,3\}$) when bytes in $(\mathcal{C}_I \cup \mathcal{ID}_J) \setminus (\mathcal{C}_I \cap \mathcal{ID}_J)$ of $(p^1 \oplus p^2)$ are inactive. Such p^1, p^2 occur with probability $P_{ID}(|I|,|J|)$. By taking sum over all possible choices of J, $P_{swap}(|I|) = \sum_{j=1}^{3} \binom{4}{j} P_{ID}(|I|, j)$. □

For $|I| = 1$, $P_{swap} \approx 2^{-46}$, which is its maximum value. For a better visualization of Lemma 1 consider the case when $I = \{3\}$ and $J = \{2,3\}$. In Fig. 6 only the bytes in $\mathcal{C}_{\{3\}} \cap \mathcal{ID}_{\{2,3\}}$ are active. So, swapping the last column between p^1, p^2 can also be considered as swapping of last two inverse diagonals. An example regarding Lemma 2 is described in Appendix A. Next, we apply these results to devise a yoyo game embedded within a boomerang for 5-round AES.

4.1 Distinguishing and Key Recovery Attacks on 5-round AES

The attack strategy discussed above is applied to devise a 5-round AES distinguisher, which is subsequently converted into a key recovery attack. First of all, 5-round AES is divided into two parts- before MC of the 4-th round is termed as E_0 and the remaining part of the cipher is termed as E_1. Note that, E_0 is comprised of $S \circ L \circ S$ layer where S and

L corresponds to AES Super-Sbox and MC respectively. Fig. 7 depicts the E_0 and E_1 partition in AES. Now, Proposition 1 and Lemma 2 are combined to design a 5-round AES distinguisher by devising a probabilistic yoyo game by embedding yoyo within boomerang.

Definition 6. Let $\alpha \in \mathbb{F}_{2^8}^{4\times4}$ be a state and $v \in \mathbb{F}_2^4$ be a vector. Then a state $\tau^v(\alpha) \in \mathbb{F}_{2^8}^{4\times4}$ is constructed from α such that for $0 \leq i \leq 3$

$$\mathcal{ID}_{\{i\}}(\tau^v(\alpha)) = \begin{cases} \mathcal{ID}_{\{i\}}(\alpha), & \text{if } v_i = 0; \\ 0, & \text{Otherwise.} \end{cases}$$

.

Lemma 3. *Let p^1, $p^2 \in \mathbb{F}_{2^8}^{4\times4}$ and $c^1 = R_5(p^1)$, $c^2 = R_5(p^2)$ where R_5 is 5-round AES or alternatively $R_5 = E_1 \circ E_0$. For any vector $v \in \mathbb{F}_2^4$ such that $1 \leq wt(v) \leq 3$, let $c'^1 = c^1 \oplus \tau^v(c^1 \oplus c^2)$, $c'^2 = c^2 \oplus \tau^v(c^2 \oplus c^1)$ and $p'^1 = R_5^{-1}(c'^1)$, $p'^2 = R_5^{-1}(c'^2)$. Then $\nu(p^1 \oplus p^2) = \nu(p'^1 \oplus p'^2)$ occurs with probability $P_{swap}\big(4 - wt(v)\big)$.*

Proof. Let $s^1 = E_0(p^1)$ and $s^2 = E_0(p^2)$. Due to Lemma 2, the probability of occurence of certain s^1, s^2 such that swapping of $\mathcal{ID}_I$ between c^1, c^2 is equivalent to swapping of $\mathcal{ID}_J$ (where $I, J \subset_\phi \{0,1,2,3\}$) between s^1, s^2 is $P_{swap}(|I|)$. Let $s'^1 = E_1^{-1}(c'^1)$ and $s'^2 = E_1^{-1}(c'^2)$. Due to the existence of Super-Sbox in E_1, the intermediate pair s'^1, s'^2 can be considered as constructed from s^1, s^2 as follows. $s'^1 = s^1 \oplus \gamma$ and $s'^2 = s^2 \oplus \gamma$, where some inverse diagonals in γ are zero and some of them are exactly equal to the same inverse diagonal in $s^1 \oplus s^2$. Thus by Theorem 1, s'^1, s'^2 is constructed from s^1, s^2 using word swap. Then by Proposition 1, this new pair should preserve the zero difference property. So, the zero difference property over $E_1 \circ E_0$ ($E_1 \circ E_0$ is R_5) can be preserved at the expense of $P_{swap}(|I|)$ probability. From Definition 6 it can be concluded that $|I| = 4 - wt(v)$. □

Note that, in Lemma 3, the value of $P_{swap}\big(4 - wt(v)\big)$ is maximum ($\approx 2^{-46}$) when $wt(v) = 3$. Next, the upper trail and the lower trail are constructed for 5-round AES distinguisher by leveraging on Lemma 3. For lower trail, $v = 1110$ ($I = \{3\}$) is considered and for better understanding of the upper trail, $J = \{3\}$ is shown in Fig. 8.

Constructing the Upper Trail. Refer to Fig. 8 for the upper trail. For α, pair of plaintexts p^1, p^2 are chosen such that $wt\big(\nu(p^1 \oplus p^2)\big) = 1$. In β, at the cost of 2^{-48}, 6 bytes in $(\mathcal{C}_{\{3\}} \cup \mathcal{ID}_{\{3\}}) \setminus (\mathcal{C}_{\{3\}} \cap \mathcal{ID}_{\{3\}})$ remain inactive. By considering the cases when

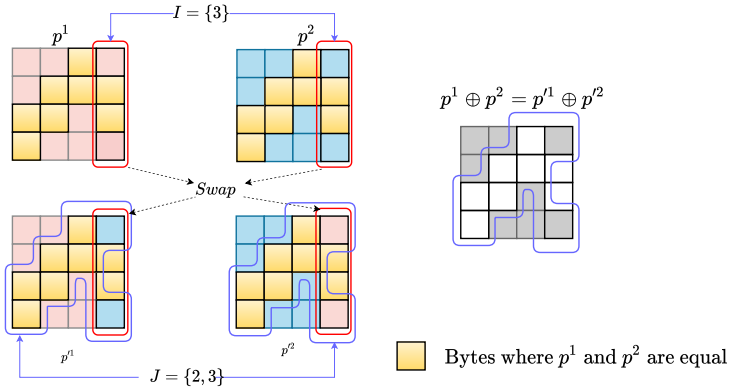


Figure 6: Visualization of Lemma 1 when $I = \{3\}$ and $J = \{2,3\}$. As $I = \{3\}$ the last column between p^1 and p^2 is swapped, which is equivalent to swapping of the third and fourth inverse diagonals between p^1 and p^2 because of the positions of inactive bytes in $p^1 \oplus p^2$. Note that, in the last column of p'^1 and p'^2 there are two swapped bytes.

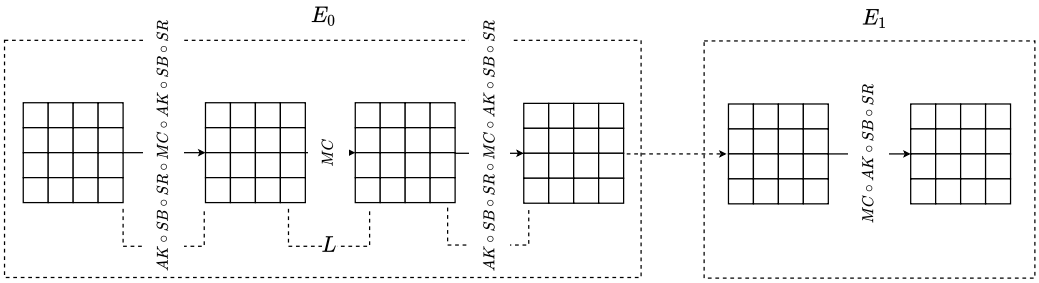


Figure 7: Partitioning 5-round AES in E_0 and E_1

$J = \{0\}, \{1\}$ or $\{2\}$, the probability is increased to 2^{-46}. We ignore the cases when $|J| > 1$, as it has a negligible effect on the cumulative probability.

Constructing the Lower Trail. For 5-round AES, the construction of the lower trail partially depends on the upper trail. At least one word of γ should be equal to a word in the same position of β. In Fig. 8, β_3 is equal to γ_3; $\gamma_0 = \gamma_1 = \gamma_2 = 0$. To generate such γ, dependency on the upper trail is required while constructing δ. Let p^1, p^2 are encrypted to obtain c^1, c^2. For $0 \leq i \leq 3$, δ is constructed as follows-

$$\delta_i = \begin{cases} c_3^1 \oplus c_3^2, & \text{if } i = 3; \\ 0, & \text{Otherwise.} \end{cases}$$

Note that, by Definition 6, $\delta = \tau^v(c^1 \oplus c^2)$. In the upper trail, β occurs with probability 2^{-46}. In the lower trail, one may think that $\delta \xrightarrow{E_1^{-1}} \gamma$ occurs probabilistically. But assuming that β has occured, $\delta \xrightarrow{E_1^{-1}} \gamma$ occurs deterministically. This determines that the overall complexity of the attack is 2^{-46}.

Attack Overview.

1. Prepare a structure of 2^{23} plaintexts p^i, $i \in \{1, 2, \cdots 2^{23}\}$ such that all bytes are constant except the bytes in principal diagonal (0^{th} diagonal), which are different for each p^i.

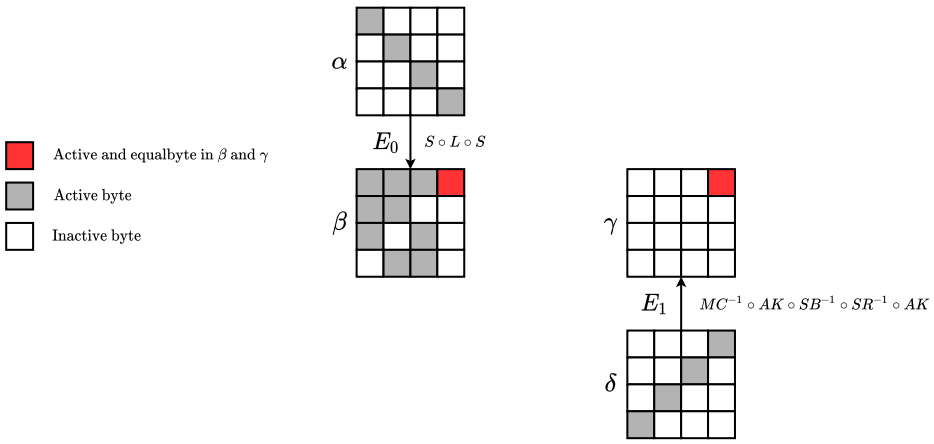


Figure 8: Upper and Lower Trail of 5-round AES. In this trail, the red-colored byte should be equal in β and γ in order to realize the inverse diagonal swap in the boundary of E_0 and E_1.

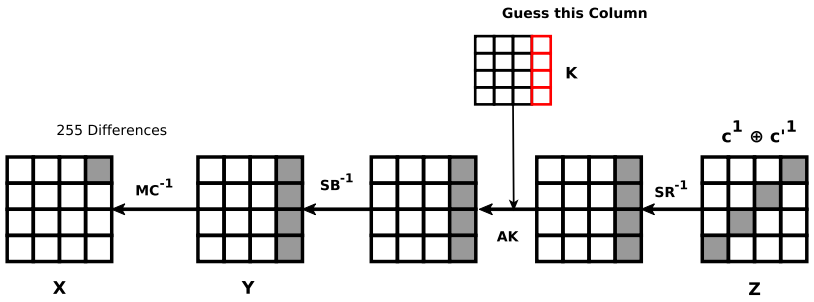


Figure 9: Key Recovery Attack on 5-round AES

2. For $0 \leq i \leq 2^{23}$, query encryption oracle with each p^i to obtain c^i.
3. For $0 \leq i \leq 2^{23} - 1$ and for $i + 1 \leq j \leq 2^{23}$,
   (a) Construct δ as-

$$\delta_m = \begin{cases} c_3^i \oplus c_3^j, & \text{if } m = 3; \\ 0, & \text{Otherwise.} \end{cases}$$

   for $0 \leq m \leq 3$.
   (b) Prepare $c'^i = c^i \oplus \delta$ and $c'^j = c^j \oplus \delta$. Query decryption oracle with c'^i, c'^j to obtain p'^i, p'^j.
   (c) Check whether $\nu(p^i \oplus p^j) = \nu(p'^i \oplus p'^j)$. If yes, distinguish oracle as 5-round AES and refer to (p^i, p^j, p'^i, p'^j) as a quartet.
4. If no quartet is found, distinguish oracle as random permutation.

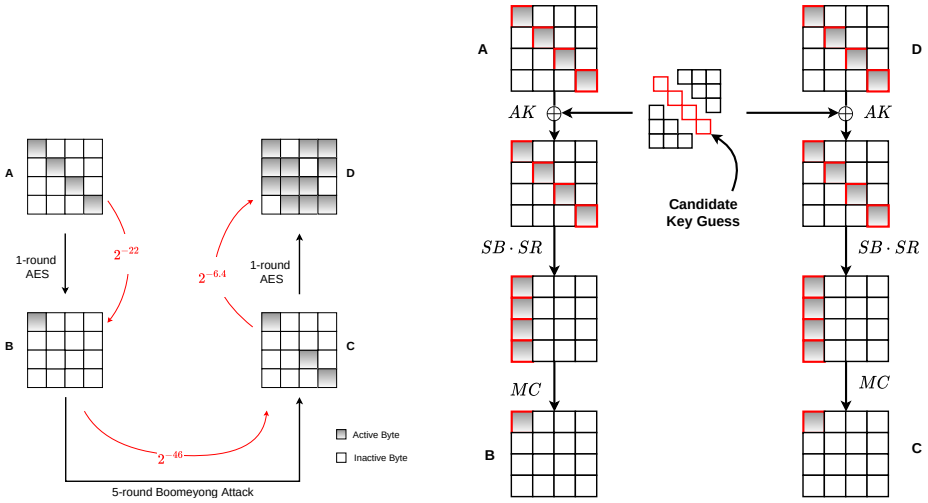
Analysis. The data complexity of the attack is (2^{23} encryption queries + 2^{47} decryption queries) $\approx 2^{47}$ decryption queries. The time complexity of the attack is 2^{46} XOR operations. The memory complexity is 2^{23} AES state which is used to store the encrypted plaintexts.

Experimental Verification. Due to high data complexity, it is quite difficult to run the complete attack. Instead, an experiment is run to verify the existence of such claimed trails. One such trail is listed in Appendix B. In addition, an experiment for the distinguishing attack is run on 64-bit AES whose details are provided in Section 4.3.

Key Recovery Attack.

The key recovery attack is an extension of the distinguishing attack. Refer to Fig. 9 for the attack. Let's assume distinguisher has successfully found a quartet (p^1, p^2, p'^1, p'^2) and its corresponding ciphertexts (c^1, c^2, c'^1, c'^2). Consider the active bytes of $(c^1 \oplus c'^1)$ in Z. SR^{-1} aligns the bytes in the last column. Guess the last column of K, invert the bytes using SB^{-1}. Consider the differential in Y, apply MC^{-1} to it and check whether it transits to a single byte or not in X. The guesses for which only a single active byte is obtained in X are right guesses. The active byte in X can have 255 different values; thus for the active diagonal $255 \approx 2^8$ right key candidates are obtained. The process is repeated for the remaining three diagonals which gives a total 2^{32} right key candidates. An exhaustive search is done over these 2^{32} candidates to recover the right key.

Analysis. For guessing each column, four different right pairs are required. So, the data complexity and the memory complexity is $4 \times 2^{47} = 2^{49}$ adaptive chosen plaintexts



(a) Candidate pair for Boomeyong Attack on 6-round AES

(b) Candidate Key for Boomeyong Attack on 6-round AES

Figure 10: Key Recovery Attack on 6-round AES

and ciphertexts and 2^{23} AES states respectively. Once a right pair is found using the distinguisher, 2^8 key candidates for a column can be retrieved by doing $2^{32} \times 2$ one round AES encryption for a column. To retrieve key candidates for all the columns, $2^{32} \times 2 = 2^{33}$ one round AES encryption needs to be done. Considering five such operations as 5-round AES, $2^{33}/5 = 2^{30.5}$ AES encryptions are required. For exhaustive search, 2^{32} more encryptions are required. So, the total time complexity is $2^{32} + 2^{30.5} \approx 2^{32.4}$ AES encryptions and $4 \times 2^{46} = 2^{48}$ XOR operations.

4.2 Key Recovery Attack on 6-round AES

The 6-round key recovery attack on AES is the extension of the 5-round attack described in this paper. The 6-round attack extensively uses the *4-to-1* property of the AES in the initial round. One round is prepended to the 5-round boomeyong attack. As shown in Fig. 10a, if a diagonal is inactive in **D** for a pair then it is included in the candidate set. The main problem is that the candidate set contains right and wrong pairs as for a random pair, any one of the diagonals is inactive with probability $4 \times 2^{-32} = 2^{-30}$. However, using the boomeyong attack such a pair can be obtained with much lesser probability. Hence, to retrieve the right key candidate using the candidate pairs the notion of the signal-to-noise ratio is applied.

Attack Idea. Refer to Fig. 10a for the attack. Choose pairs of plaintexts such that only 4 bytes of a diagonal of the pairs are active; the remaining bytes are inactive. Query the pairs to the encryption oracle to obtain corresponding ciphertext pairs. An inverse diagonal is swapped between the ciphertexts to obtain new pair of texts which are queried to the decryption oracle to obtain new pair of plaintexts. As already stated in Section 4.1, with probability 2^{-46} swapping an inverse diagonal between the ciphertexts is equivalent to swapping an inverse diagonal between the intermediate states in the previous round. This is a base condition for the yoyo property, under which it is expected that there is one inactive Super-Sbox between the intermediate state before one round decryption (Position **C** in Fig. 10a). Now in **C**, out of 3 active bytes, one byte becomes inactive with probability $3 \times 2^{-8} = 2^{-6.4}$ (The inactive diagonal in **D** should not be the same as the active diagonal in **A**, otherwise the number of candidate keys increases significantly.

So, the corresponding byte in **C** should be active). The transition **A**→ **B** occurs with probability $4 \times 2^{-24} = 2^{-22}$. Hence, the cumulative probability of obtaining an inactive diagonal is $2^{-22} \times 2^{-46} \times 2^{-6.4}$=$2^{-74.4}$. For a random pair, a pair of texts with such an inactive diagonal can be obtained with probability $3 \times 2^{-32} = 2^{30.4}$. Therefore, by this attack, a set of right and wrong pairs can be obtained and there is no way to distinguish the right ones from the wrong ones. If $2^{74.4}$ pairs are queried then it is expected that there are around $2^{74.4} \times 2^{-30.4} = 2^{44}$ wrong pairs and one right pair. The right key candidate is suggested by the right pair whereas the wrong pairs can suggest both right and wrong key candidates. The diagonal of the key corresponding to the active diagonal of the initial plaintext pairs are guessed (refer to Fig. 10b). So, the size of the guessed key space is 32 bits and thus a counter for each of the 2^{32} keys is maintained to count the key suggestions. To determine the required number of right pairs, the notion of the signal-to-noise ratio is applied.

Algorithm 1: Algorithm for Key Recovery Attack on 6-round `AES`

Result: The secret key
$v \leftarrow 1110$
for $0 \leq m \leq 1$ *initialize* $K_m = \phi$ **do**
  for $0 \leq i < 2^{32}$ **do**
    $ctr[i] \leftarrow 0$
  end
  for $0 \leq i < 2^{77.72}$ **do**
    Choose 2 `AES` state P_i^1, P_i^2 such that only the 4-bytes in $\mathcal{D}_{\{m\}}(P_i^1 \oplus P_i^2)$ are active
    $C_i^1 = Enc(P_i^1)$, $C_i^2 = Enc(P_i^2)$
    $C_i^3 = C_i^1 \oplus \tau^v(C_i^1 \oplus C_i^2)$ and $C_i^4 = C_i^2 \oplus \tau^v(C_i^1 \oplus C_i^2)$
    $P_i^3 = Dec(C_i^3)$ and $P_i^4 = Dec(C_i^4)$
    if $\mathcal{D}_{\{m\}}(P_i^3 \oplus P_i^4)$ *is inactive* **then**
      Discard P_i^1, P_i^2, P_i^3, P_i^4
      Go to step 7
    end
    if $\nexists j \in \{0,1,2,3\}$ *and* $j \neq m$ *such that* $\mathcal{D}_{\{j\}}(P_i^3 \oplus P_i^4)$ *is inactive* **then**
      Discard P_i^1, P_i^2, P_i^3, P_i^4
      Go to step 7
    end
    for $0 \leq j < 2^{32}$ **do**
      $X \leftarrow MC^m \circ s^4\left(\mathcal{D}_{\{m\}}(P_i^1) \oplus j\right) \oplus MC^m \circ s^4\left(\mathcal{D}_{\{m\}}(P_i^2) \oplus j\right)$
      $Y \leftarrow MC^m \circ s^4\left(\mathcal{D}_{\{m\}}(P_i^3) \oplus j\right) \oplus MC^m \circ s^4\left(\mathcal{D}_{\{m\}}(P_i^4) \oplus j\right)$
      if *There is only one active byte in* X *and* Y *and its position is same in both* X *and* Y **then**
        $ctr[j] \leftarrow ctr[j] + 1$
      end
    end
  end
  Include the first 2^7 key candidates with highest counter value in K_m
end
K_2 and K_3 are populated with all 2^{32} candidates
Exhaustively search for the right subkey in $K_0 \times K_1 \times K_2 \times K_3$
Finds the secret key from the subkey

Table 3: Required number of plaintext-ciphertext pairs versus the success probability for key recovery attack on 6-round AES. The value of r is considered 2^7 for all the cases.

Pairs Required	Success Probability
$2^{76.95}$	0.65
$2^{77.14}$	0.7
$2^{77.31}$	0.75
$2^{77.51}$	0.8
$2^{77.72}$	0.85
$2^{77.96}$	0.9

Determining the required number of right pairs. By referring to Section 2.4, the values of p and k are $2^{-74.4}$ and 32 respectively. Now, the number of keys (right and wrong) suggested by each wrong pair needs to be determined. Consider P^1, P^2 be a pair of texts which are encrypted, diagonals are swapped between their corresponding ciphertexts and decrypted to obtain P^3, P^4. Let the first diagonal of the key be guessed. So, the first diagonal of P^1, P^2 is partially encrypted for one round using the guessed key and checked whether *4-to-1* transition occurred or not. Similar experiment is done with P^3, P^4. If *4-to-1* occurs for both the cases, then the value of the counter corresponding to the key is incremented. After *4-to-1* the position of the active byte should be same for both cases. Hence, for a fixed wrong pair and a fixed guessed key, the counter value is incremented with probability $4 \times 2^{-24} \times 2^{-24} = 2^{-46}$. So, the average number of keys suggested by a wrong pair is $2^{-46} \times 2^{32} = 2^{-14}$ ($\eta = 2^{-14}$). Note that, if the first diagonal of P^3, P^4 is inactive, then the pair needs to be discarded as this pair suggests $4 \times 2^{-24} \times 2^{32} = 2^{10}$ keys and to recover the correct key the data complexity may need to be increased. Hence, with probability $3 \times 2^{-32} = 2^{-30.4}$ a wrong pair survives. Therefore, $S/N = \frac{2^{32} \times 2^{-74.4}}{(1-2^{-74}) \times 2^{-30.4} \times 2^{-14}} \approx 2^2$. Plugging in the values of r as 2^7 in Proposition 3, the number of plaintexts-ciphertexts pairs required to recover a diagonal of the correct key for various success probabilities are listed in Table 3. From Table 3, the number of plaintexts-ciphertexts pairs required for key recovery with success probability 0.85 is $2^{77.72}$. As p is $2^{-74.4}$, $2^{77.72} \times 2^{-74.4} = 9.98$ right pairs are required to recover four bytes of the right key. The process is repeated one more time for another diagonal. The remaining part of the key is recovered using exhaustive search. In order to minimize the cost of the exhaustive search, the value of r is considered as 2^7. Hence, the cumulative success probability is $0.85 \times 0.85 \approx 0.72$. Details regarding key recovery attack on 6-round AES are given in Algorithm 1. Note that, in Step 20 and Step 21 in Algorithm 1, s^4 is four parallel application of subBytes on four bytes and MC^m is application of MC on m-th column.

Analysis. With reference to the Step 6, $2^{77.72}$ pairs are required to be queried to both the encryption and the decryption oracle for the first and second diagonal. Hence, the data complexity is $2 \times 2 \times 2 \times 2^{77.72} = 2^{80.72}$ encryption/decryption queries. Time complexity involves $2^{79.72}$ XOR operations, computations of $MC \circ SB \circ AK$ operations for a single column in Step 20 and Step 21 and exhaustive search for finding the right key. After filtering, the remaining number of pairs is $2^{77.72} \times 2^{-30.4} = 2^{47.32}$. So, the total number of $MC \circ SB \circ AK$ operations is $2^{47.32} \times 4 \times 2 = 2^{50.32}$. As four such operations approximately constitute one round AES encryption, it is assumed that 24 such operations are equivalent to one AES (6-round) encryption. So, the total number of such operations are $2^{50.32}/24 \approx 2^{45.73}$ AES encryptions. In Step 30, $|K_0|=|K_1|=2^7$ and $|K_2|=|K_3| = 2^{32}$.

Therefore, $2^7 \times 2^7 \times 2^{32} \times 2^{32} = 2^{78}$ offline computations of AES encryptions are required to recover the right key. The cost of $2^{79.72}$ XOR operations is lesser in comparison to the 2^{78} AES encryptions (even if 6 XOR operations are considered as one encryption of 6-round AES, then the $2^{79.72}$ XOR operations are equivalent to $2^{77.14}$ AES encryptions). Memory requirement for this attack is the memory used for storing the counter. As a byte is sufficient for storing the value for each index of the counter, 2^{32} bytes are required which is equivalent to $2^{32}/16 = 2^{28}$ AES states that constitutes the memory complexity.

Reducing the Encryption Queries. Refer to β in the upper trail in Fig. 8. Only four combinations corresponding to the position of the active byte in the last column are considered. But similar events can occur for the other columns also. Thus, instead of swapping only the last inverse diagonal, if all the inverse diagonals are swapped then it is possible to reduce the number of encryption queries by $\frac{3}{4}$; as for each pair of initial plaintexts, four different pairs of ciphertexts after swapping can be constructed. Thus the number of encryption queries can be reduced. The number of decryption queries can not be decreased as all the swapped pairs need to be queried to the decryption oracle. The number of encryption queries can be further reduced by using the structure technique. Hence the modified data complexity is approximately $2^{79.72}$.

One may be tempted to think that instead of repeating the algorithm for two diagonals independently, reusing the set of plaintext-ciphertext pairs that suggest the top key candidates to recover the second diagonal of the key may lead to a significant reduction in the number of wrong pairs while keeping the number of right pairs the same. However, our investigation suggests that in the above modified strategy the number of right pairs corresponding to the second diagonal also reduces. It happens because the pairs whose second diagonal is inactive need to be discarded while recovering the second diagonal of the key. This claim about the ineffectiveness of the above mentioned strategy has also been supported by our experimental results.

Moreover, it can be noted that the key recovery attacks on 5/6-round AES-128 can be extended to mount key recovery attacks on 6/7-round AES-256 respectively. The details of those attacks are provided in Appendix C.

4.3 Experimental Verification on 64-bit AES

To show the validity of the attacks presented in this paper, experimental verification of the attacks are carried out on a small-scale variant of AES proposed by Cid *et al.* [CMR05]. The variant that is considered has a block length of 64 bits and thus referred here as 64-bit AES. The bytes in the original AES are replaced with nibbles (4 bits). The round operations - SubBytes, ShiftRows, MixColumns and AddRoundKey are redefined to comply with the 64-bit version. As the design of 64-bit AES is quite similar to the original version, the analysis on AES presented in this paper applies to it. Thus it provides a framework for verifying the validity of the attacks.

Distinguishing Attack on 5-round 64-bit AES

Recall the attack in Section 4.1. In this case, the modified probability of the occurrence of β is $4 \times 2^{-24} = 2^{-22}$. Hence, by checking 2^{22} pairs of plaintexts the validity of the attack can be established. Hence, a structure with 2^{11} plaintexts are constructed such that only the bytes in principal diagonal differ; the remaining bytes are the same for all plaintexts. Using these states, the experiment for the 5-round attack is carried out on 64-bit AES. As expected, a pair of states with the same zero difference pattern as the initial pair of states is obtained. The code for the 5-round attack on 64-bit AES is available online[1].

[1] https://github.com/de-ci-phe-red-LABS/Boomeyong-codes-ToSC_2021_3

Key Recovery Attack on 6-round 64-bit AES

To validate the theoretical claims, experiments have been conducted on the 6-round 64-bit AES [CMR05] where key recovery attacks could successfully recover a diagonal. Here, we detail the experimental results of the attack. One can recall from Section 4.2 that swapping an inverse diagonal between ciphertexts is equivalent to swapping an inverse diagonal between the intermediate states in the previous round with probability $6 \times 2^{-24} = 2^{-22}$. For the rest of the discussion refer to Fig. 10a. For 64-bit AES it can be seen that the transition from $\mathbf{A} \rightarrow \mathbf{B}$ occurs with probability $4 \times 2^{-12} = 2^{-10}$. In $\mathbf{C}$, out of three active bytes one becomes inactive with probability $3 \times 2^{-4} = 2^{-2.4}$. Hence, the total probability of the characteristic is $2^{-10} \times 2^{-22} \times 2^{-2.4} = 2^{-34.4}$. For any random pair, any one of the three diagonals become inactive with probability $3 \times 2^{-16} = 2^{-14.4}$ (this is the filtering probability). Average number of keys suggested by each wrong pair is $2^{16} \times 4 \times 2^{-12} \times 2^{-12} = 2^{-6}$. Hence, $S/N = \frac{2^{16} \times 2^{-34.4}}{(1-2^{-34.4}) \times 2^{-14.4} \times 2^{-6}} \approx 2^2$. With reference to Proposition 3, if the values of r and P_s are set to 2^7 and 0.75 respectively, then the number of plaintext-ciphertext pairs required to recover the correct key is $2^{37.4}$ (8 right pairs are required). After the filtering, expected number of pairs (both right and wrong) is $2^{37.4} \times 2^{-14.4} = 2^{23}$. As described in Section 2.4, the counter values corresponding to each key follows the normal distribution. Hence, the counter with the highest value may not be the right key (if the counter values would have followed uniform distribution, then the candidate key having the highest counter value could have been considered as the right key).

The experiment is initiated by randomly choosing a 64-bit key. The experiment is conducted to recover the nibbles corresponding to the first diagonal of the key. As expected, after the filtering 6749861 pairs ($\approx 2^{22.69}$) survive. After the experiment, the counter value corresponding to the first diagonal is 15; whereas the highest value for the counter is 17. The counter value corresponding to the right key is among the top 128 values (number of key candidates corresponding to the counter value 17, 16 and 15 are 4, 2 and 10 respectively).

To further validate the success probability of the proposed attack, the partial key recovery corresponding to a diagonal has been repeated 55 times. Out of which, 43 times the diagonal corresponding to the right key rank among the top 2^7 candidates. Hence, the practical success probability of the attack is $43/55 = 0.78$ which is close to the theoretical value of 0.75.

5 Boomeyong Attack on Pholkos

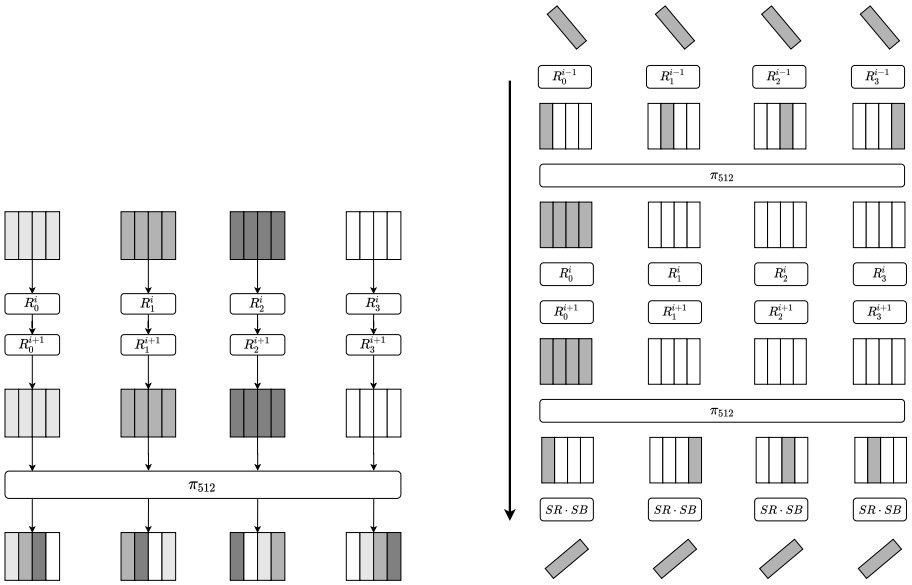
Next, the boomeyong technique is applied on a tweakable block cipher Pholkos [BLLS20]. Attack strategy quite similar to the 6-round attack on AES is used to mount an key recovery attack on 10-round Pholkos with the data, time and memory complexity of $2^{189.8}$, $2^{188.8}$ and 2^{122}. Till now, there is a distinguishing attack on 10-round Pholkos block cipher by the designers whose data, time and memory complexity is 2^{260}, 2^{260} and 2^{32} respectively.

5.1 Specification of Pholkos

Pholkos is a recently proposed family of tweakable block cipher which is based on AES round functions. It follows the design strategy of AESQ [BK14] and Haraka [KLMR16]. An instance of Pholkos with a block size of n bits and a key size of k bits is denoted by Pholkos-n-k. The tweak size in Pholkos is 128 bits for all variants. The secret key variants of Pholkos are Pholkos-256-256, Pholkos-512-256 and Pholkos-512-512. n-bit Pholkos state is considered as $n/128$ parallel AES substates where each substate goes through 2 rounds of AES operations followed by a columnwise permutation of words between substates.

The substates are indexed from 0 to $\frac{n}{128} - 1$ with the leftmost substate indexed as 0. The `AddRoundKey (AK)` operation in `AES` is substituted by `AddRoundTweakey (ATK)` in Pholkos. Like `AES`, `MC` is also omitted in the last round of Pholkos. The total number of rounds in Pholkos variants with a block size of 256 and 512 are 16 and 20 respectively. The details regarding key expansion and tweakey generation is omitted here; for more details refer to [BLLS20]. The notations are reviewed here.

- $P_i[j]$: Denotes the j-th substate in the i^{th} round of Pholkos state P.
- $\mathcal{X}_i^P$: Denotes the state before MC in the i^{th} round for an initial state P.
- $\mathcal{X}_i^P[j]$: Denotes the j^{th} substate before MC in the i^{th} round for an initial state P.



(a) Two Rounds of Pholkos-512 **(b)** `MegaSbox` in Pholkos-512

Figure 11: Pholkos-512 and `MegaSbox`

As the attacks discussed here are independent of the key size, an instance of Pholkos *with block size b is denoted by* Pholkos-*b*. Fig. 11a shows round operations for Pholkos-512 and Pholkos-256. In Pholkos, there is a group of 128 bits which is independent of other bits in the Pholkos state over a certain number of rounds. This is called `MegaSbox` (cf. in [DLP+09]) and details regarding this are now discussed.

MegaSbox. Refer to Fig. 11b for the `MegaSbox` construction in Pholkos. Four diagonals in four `AES` substates are aligned to a column in each substate due to the effect of R_j^{i-1} for $0 \leq j \leq 3$ in $i-1$ round. The subsequent π_{512} combines these columns in a single substate where they go through two rounds of `AES`. The following π_{512} breaks the substate by moving the columns to different substates and $SR \circ SB$ aligns the bytes in inverse diagonals. The `MegaSbox` in Pholkos-512 spans over 3.5 rounds. 3.5 rounds Pholkos-512 can be considered as four parallel operations of `MegaSbox`. This `MegaSbox` is exploited while mounting the key recovery attack on Pholkos.

5.2 Key Recovery Attack on 10-round Pholkos

The key recovery attack on 10-round Pholkos is similar to the 6-round key recovery attack on `AES`. For the upper trail, the $S \circ L \circ S$ layer needs to be identified. Here, S and L refers

to the MegaSbox and MC respectively. As MegaSbox spans over 3.5 rounds, $S \circ L \circ S$ layer starting from round 2 covers 7.5 rounds in total. The strategy remains the same- at the end of 10-round, such a δ to be added so that the inverse diagonals are swapped between the intermediate states in the previous round. Contrary to AES, here four different inverse diagonals in four substates need to be swapped and they should be a part of the same MegaSbox. Suppose, P^1, P^2 be two Pholkos states which are encrypted to obtain C^1, C^2 respectively. By Lemma 2, swapping of $\mathcal{ID}_{\{3\}}$ between $C^1[3]$ and $C^2[3]$ is equivalent to swapping of $\mathcal{ID}_J$ for $J \subset_\phi \{0,1,2,3\}$, between $\mathcal{X}_9^{P^1}[3]$ and $\mathcal{X}_9^{P^2}[3]$ with probability 2^{-46} (approx). If all other inverse diagonals corresponding to a MegaSbox in the remaining substates are inactive in the difference $\mathcal{X}_9^{P^1} \oplus \mathcal{X}_9^{P^2}$, then swapping of $\mathcal{ID}_{\{3\}}$ between $C^1[3]$ and $C^2[3]$ is equivalent to swapping of MegaSbox in $\mathcal{X}_9^{P^1} \oplus \mathcal{X}_9^{P^2}$ with probability $2^{-46} \times 2^{-32\times 3} = 2^{-142}$. Thus for the lower trail of the boomerang, δ is constructed by taking $\mathcal{ID}_{\{3\}}$ from the last substate of $C^1 \oplus C^2$ and setting all other bytes to zero. Then, with probability 2^{-142} it is known that swapping of MegaSbox has occurred in the middle.

Attack Idea. Choose a pair of plaintext P^1, P^2 such that only the four bytes in $\mathcal{D}_{\{0\}}(P^1[0] \oplus P^2[0])$ are active. P^1, P^2 are queried to the encryption oracle to obtain C^1, C^2. After one round of partial encryption only one byte becomes active with probability 2^{-22} (i. e. in $P_1^1[0] \oplus P_1^2[0]$ only one byte is active) which implies that only one MegaSbox is active. Now, a inverse diagonal is swapped between C^1, C^2 and the new states are queried to the decryption oracle to obtain P^3, P^4. Now, with probability 2^{-142} only one MegaSbox should be active in $P_1^3[0] \oplus P_1^4[0]$. t bytes out of the 16 bytes of the active MegaSbox are inactive with probability $\binom{16}{t} \times 2^{-8t}$. Hence, with probability $2^{-22} \times 2^{-142} \times \binom{16}{t} \times 2^{-8t} = 2^{-164-8t} \times \binom{16}{t}$, t diagonals are inactive in $P^3 \oplus P^4$. For a random pair of texts, t diagonals are inactive with probability $\binom{16}{t} \times 2^{-32t}$. Note that, for $7 \leq t \leq 16$, $2^{-164-8t} \times \binom{16}{t} > \binom{16}{t} \times 2^{-32t}$ and thus a right pair can be uniquely distinguished; but it requires a data complexity around $2^{206.5}$. To reduce the data complexity, instead of using a unique right pair, a set of right and wrong pairs are used and then by using the ranking test the right key candidate is guessed. Note that, as t diagonals are inactive in $P^3 \oplus P^4$, a wrong pair survives the filtering with probability $\binom{16}{t} \times 2^{-32t}$. Now, 128 bits of the key are guessed corresponding to four active diagonals in $P^3 \oplus P^4$. For a wrong pair, out of 2^{32} key guesses for each diagonal, $2^{32} \times 2^{-22} = 2^{10}$ guesses conforms to the *4-to-1* transition. So, a wrong pair suggests 2^{40} key guesses. Therefore,

$$S/N = \frac{2^{128} \times 2^{-164-8t} \times \binom{16}{t}}{\binom{16}{t} \times 2^{-32t} \times 2^{40}} = 2^{-76+24t}$$

Now, $S/N > 1$ when $t \geq 4$. As the probability of the trail is $2^{-164-8t} \times \binom{16}{t}$, so with the increasing value of t, the trail probability decreases significantly. Hence, $t = 4$ is considered. For $t = 4$, the trail probability is $2^{-185.2}$, $S/N = 2^{20}$. With reference to Proposition 3, if $r = 2^7$ is considered, then the success probability is 0.92 if $2^{186.2}$ plaintext-ciphertext pairs are used for recovering a diagonal of the key. As in Algorithm 2, this step is repeated three times, so the overall success probability of recovering the correct key is 0.78. Therefore, collecting two right pairs is enough for guessing the right key. As 128 bits of the key are guessed at a time, the size of the counter is 2^{128}. Algorithm 2 in Appendix D gives the details of the key recovery mechanism.

Analysis. Referring to Step 8 in Algorithm 2, $2^{186.2}$ pairs are required for encryption and decryption queries in each iteration. So, total data complexity is $3 \times 2^{186.2} \times 4 = 2^{189.8}$ encryption/decryption queries. Time complexity involves $3 \times 2^{186.2} \times 2 = 2^{188.8}$ XOR operations, computations of partial encryptions and cost of exhaustive search in Step 37. Out of $2^{186.2}$ pairs, after the filtering $2^{58.2}$ pairs remain. Each pair suggests around 2^{40} candidate keys. So, for $2^{58.2}$ pairs, $2 \times 2^{40} \times 2^{58.2} = 2^{99.2}$ partial encryptions are computed. As this process is repeated for three different sets of diagonals, total number of

partial encryptions is $2^{100.8}$. By assuming four such partial encryptions as one round of Pholkos encryption, the total computation is $2^{100.8}/40 = 2^{95.5}$ Pholkos encryptions. As $|K_0| = |K_1| = |K_2| = 2^7$ and $|K_3| = 2^{128}$, Step 37 requires computations of 2^{149} Pholkos encryptions. Memory requirement for this attack is the memory used for storing the counter. As a byte is sufficient for storing the value for each index of the counter, 2^{128} bytes are required which is equivalent to $2^{128}/64 = 2^{122}$ Pholkos states and that constitutes the memory complexity.

Next, a brief discussion about the close relationship between attacks presented in this paper and recently proposed retracing boomerang framework [DKRS20] is given.

6 Relation with Retracing Boomerang Attack

Dunkelman *et al*. recently proposed the retracing boomerang attack [DKRS20] in Eurocrypt 2020. With some restrictions, those attacks can also be visualised using the boomeyong attack framework. Before discussing further, a brief description of retracing boomerang is provided. In retracing boomerang attack, a cipher is divided into $E_{12} \circ E_{11} \circ E_0$. The E_{12} is further divided into two parts: E_{12}^L and E_{12}^R where E_{12}^L operates on b bits on the left and E_{12}^R operates on the remaining $(n-b)$ bits on the right. Let consider $Pr[\alpha \xrightarrow{E_0} \beta] = p$, $Pr[\gamma \xrightarrow{E_{11}} (\mu_L, \mu_R)] = q_1$, $Pr[\mu_L \xrightarrow{E_{12}^L} \delta_L] = q_2^L$ and $Pr[\mu_R \xrightarrow{E_{12}^R} \delta_R] = q_2^R$. In general, the probability of a boomerang distinguisher satisfying these trails is $(pq_1q_2^Rq_2^L)^2$. There are two variants of retracing boomerang attack- shifting retracing boomerang attack and mixing retracing boomerang attack. Suppose, E encrypts P^1, P^2 to C^1, C^2. In the shifting retracing boomerang attack, if $C_R^1 \oplus C_R^2 = 0$ or δ_R, only then a new pair of ciphertexts are formed by performing $C^1 \oplus \delta$ and $C^2 \oplus \delta$. This increases the probability of the boomerang distinguisher to $(pq_1q_2^L)^2q_2^R$ as in the return path the differential over E_{12}^R is deterministically satisfied. In the mixing variant, δ is constructed as $(0, C_R^1 \oplus C_R^2)$. This improves the probability of boomerang distinguisher by a factor of $(q_2^L)^{-2}(q_2^R)^{-1}$.

In particular, the mixing variant can be redefined using the boomeyong framework. Consider the last $AK \circ SR \circ SB \circ AK \circ MC$ operations of 5-round `AES` on first three columns as E_{12}^R and on the last column as E_{12}^L. In the boomeyong attack, δ is constructed by taking the difference of two ciphertexts in one inverse diagonal; the other inverse diagonals are set to zero. By following this strategy, it is possible to swap a column one round inside without incurring any probability and when the required differential in the upper trail occurs, this strategy essentially swaps inverse diagonals one round inside, which is a necessary condition for the yoyo game to occur in the upper trail. In the mixing retracing boomerang attack, δ is constructed by following a similar kind of strategy. However, the main advantage of the boomeyong attack over mixing retracing boomerang attack is that no extra cost is incurred for the return path of the upper trail as it occurs deterministically. Fig. 12 shows the relation between the mixing retracing boomerang attack and the boomeyong attack on `AES`.

7 Conclusion

In the current work, we concentrated on devising a generic strategy for embedding the yoyo trick inside a boomerang trail. In doing so, we take a fresh look at the word-swap operation of the yoyo trick that is fundamental to the deterministic nature of the basic yoyo game. Our investigations lead to proving that the word-swap operation is a *combination* of s-box switch and ladder switch if we geometrically visualize the yoyo to be on top of the lower boomerang trail. The core idea here is to devise the lower boomerang trail in such a way that the intended s-box and ladder switches happen at the boundary thereby fulfilling the condition of the yoyo game which then leads to a deterministic transition

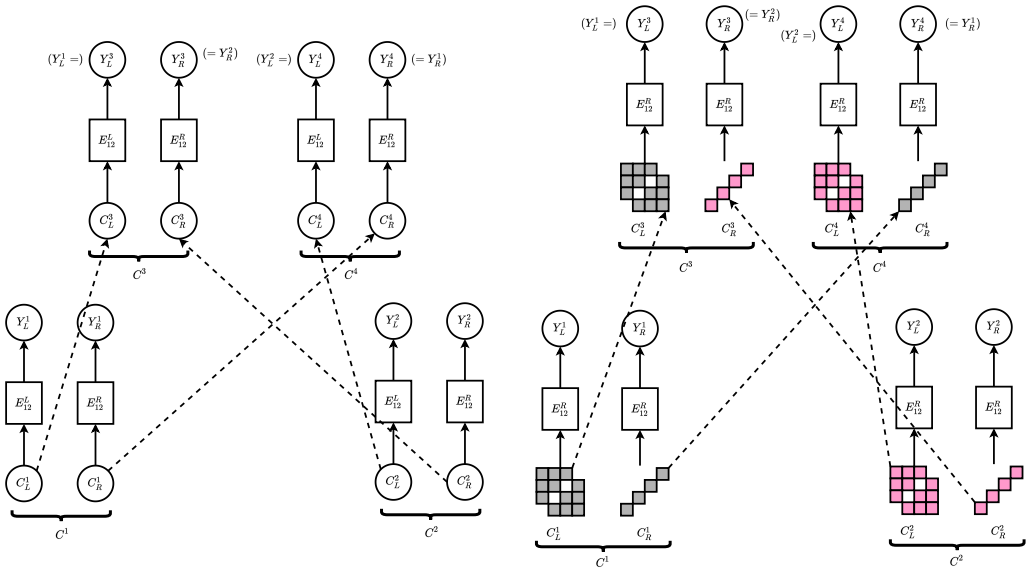


Figure 12: Relationship of boomeyong on `AES` with mixing retracing boomerang attack [DKRS20]. In the left, framework for mixing retracing boomerang attack is shown; whereas on the right, lower trail of the boomeyong attack on 5-round `AES` is shown. In both the attacks, a part of the state is exchanged between the ciphertexts. Here, $Y_L^i \leftarrow {E_{12}^L}^{-1}(C_L^i)$ and $Y_R^i \leftarrow {E_{12}^R}^{-1}(C_R^i)$.

on the way back to the top. The proposed strategy leads to new key recovery attacks on `AES` reduced to 5 and 6 rounds. The 5-round attack has a time complexity of 2^{48} XOR operations. The 6-round attack reaches a time complexity of 2^{78} `AES` encryptions. The attack is further adapted on 10 out of 20 rounds of `Pholkos`-512 showcasing its versatility. To the best of our knowledge, this is the first-ever third-party cryptanalysis of Pholkos. While mounting the key recovery attacks, the notion of *signal-to-noise* ratio is employed. The attacks on `AES` are experimentally verified by employing them on a 64-bit variant of `AES` (code is available online[2]). We also establish a relation of the proposed strategy with the retracing boomerang attack. It is worth mentioning that the boomeyong strategy performs better than most of the recent attacks reported on 6-round `AES` like extended truncated differential attack, exchange attack, yoyo attack in time/data complexity or both. Finally, the embedded yoyo-boomerang strategy helps to increase the understanding of `AES` and other `AES`-like designs and may be used as an effective cryptanalysis tool for other SPN and non-SPN ciphers as well.

Acknowledgments

The authors are thankful to Lorenzo Grassi and the anonymous reviewers of ToSC for their valuable comments and suggestions.

References

[AES01] Specification for the Advanced Encryption Standard (AES). Federal Information Processing Standards Publication 197, 2001.

[2] https://github.com/de-ci-phe-red-LABS/Boomeyong-codes-ToSC_2021_3

[BBD+98] Eli Biham, Alex Biryukov, Orr Dunkelman, Eran Richardson, and Adi Shamir. Initial Observations on Skipjack: Cryptanalysis of Skipjack-3XOR. In Stafford E. Tavares and Henk Meijer, editors, *SAC*, volume 1556 of *LNCS*, pages 362–376. Springer, 1998.

[BBJ+19] Subhadeep Banik, Jannis Bossert, Amit Jana, Eik List, Stefan Lucks, Willi Meier, Mostafizar Rahman, Dhiman Saha, and Yu Sasaki. Cryptanalysis of ForkAES. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *Applied Cryptography and Network Security*, pages 43–63, Cham, 2019. Springer International Publishing.

[BBS99] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In Jacques Stern, editor, *EUROCRYPT*, volume 1592 of *LNCS*, pages 12–23. Springer, 1999.

[BDK01] Eli Biham, Orr Dunkelman, and Nathan Keller. The Rectangle Attack — Rectangling the Serpent. In Birgit Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, pages 340–357, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[BDK02] Eli Biham, Orr Dunkelman, and Nathan Keller. New Results on Boomerang and Rectangle Attacks. In Joan Daemen and Vincent Rijmen, editors, *Fast Software Encryption*, pages 1–16, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[BDK05] Eli Biham, Orr Dunkelman, and Nathan Keller. A Related-Key Rectangle Attack on the Full KASUMI. In Bimal Roy, editor, *Advances in Cryptology - ASIACRYPT 2005*, pages 443–461, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[BDK06] Eli Biham, Orr Dunkelman, and Nathan Keller. Related-Key Impossible Differential Attacks on 8-Round AES-192. In David Pointcheval, editor, *Topics in Cryptology – CT-RSA 2006*, pages 21–33, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[BDK+18] Achiya Bar-On, Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. Improved Key Recovery Attacks on Reduced-Round AES with Practical Data and Memory Complexities. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part II*, volume 10992 of *LNCS*, pages 185–212. Springer, 2018.

[BGL20] Zhenzhen Bao, Jian Guo, and Eik List. Extended Truncated-differential Distinguishers on Round-reduced AES. *IACR Transactions on Symmetric Cryptology*, 2020(3):197–261, Sep. 2020.

[BHL+20] Hamid Boukerrou, Paul Huynh, Virginie Lallemand, Bimal Mandal, and Marine Minier. On the Feistel Counterpart of the Boomerang Connectivity Table: Introduction and Analysis of the FBCT. *IACR Transactions on Symmetric Cryptology*, 2020(1):331–362, May 2020.

[Bir05] Alex Biryukov. The Boomerang Attack on 5 and 6-Round Reduced AES. In Hans Dobbertin, Vincent Rijmen, and Aleksandra Sowa, editors, *Advanced Encryption Standard – AES*, pages 11–15, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[BK09] Alex Biryukov and Dmitry Khovratovich. Related-Key Cryptanalysis of the Full AES-192 and AES-256. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, pages 1–18, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[BK14] Alex Biryukov and Dmitry Khovratovich. PAEQ: Parallelizable Permutation-Based Authenticated Encryption. In Sherman S. M. Chow, Jan Camenisch, Lucas C. K. Hui, and Siu Ming Yiu, editors, *Information Security*, pages 72–89, Cham, 2014. Springer International Publishing.

[BKN09] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolić. Distinguisher and Related-Key Attack on the Full AES-256. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, pages 231–249, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[BLLS20] Jannis Bossert, Eik List, Stefan Lucks, and Sebastian Schmitz. Pholkos – Efficient Large-state Tweakable Block Ciphers from the AES Round Function. Cryptology ePrint Archive, Report 2020/275, 2020. https://eprint.iacr.org/2020/275.

[BLP15] Alex Biryukov, Gaëtan Leurent, and Léo Perrin. Cryptanalysis of Feistel Networks with Secret Round Functions. In Orr Dunkelman and Liam Keliher, editors, *SAC*, volume 9566 of *LNCS*, pages 102–121. Springer, 2015.

[BMNPS14] Christina Boura, Marine Minier, María Naya-Plasencia, and Valentin Suder. Improved Impossible Differential Attacks against Round-Reduced LBlock. Research Report 2014/279, IACR Cryptology ePrint Archive, April 2014.

[BNPS14] Christina Boura, María Naya-Plasencia, and Valentin Suder. Scrutinizing and Improving Impossible Differential Attacks: Applications to CLEFIA, Camellia, LBlock and Simon. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014*, pages 179–199, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[BR19] Navid Ghaedi Bardeh and Sondre Rønjom. The Exchange Attack: How to Distinguish Six Rounds of AES with 2ˆ88.2 Chosen Plaintexts. In Steven D. Galbraith and Shiho Moriai, editors, *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part III*, volume 11923 of *LNCS*, pages 347–370. Springer, 2019.

[BS91] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In Alfred J. Menezes and Scott A. Vanstone, editors, *Advances in Cryptology-CRYPTO' 90*, pages 2–21, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.

[CHP+18] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang Connectivity Table: A New Cryptanalysis Tool. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT II*, volume 10821 of *LNCS*, pages 683–714. Springer, 2018.

[CMR05] C. Cid, S. Murphy, and M. J. B. Robshaw. Small Scale Variants of the AES. In Henri Gilbert and Helena Handschuh, editors, *Fast Software Encryption*, pages 145–162, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[DGPW12] Alexandre Duc, Jian Guo, Thomas Peyrin, and Lei Wei. Unaligned Rebound Attack: Application to Keccak. In Anne Canteaut, editor, *Fast Software Encryption*, pages 402–421, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[DKRS20] Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir. The Retracing Boomerang Attack. In *EUROCRYPT (1)*, volume 12105 of *LNCS*, pages 280–309. Springer, 2020.

[DKS10] Orr Dunkelman, Nathan Keller, and Adi Shamir. A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, pages 393–410, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[DKS14] Orr Dunkelman, Nathan Keller, and Adi Shamir. A Practical-Time Related-Key Attack on the KASUMI Cryptosystem Used in GSM and 3G Telephony. *J. Cryptology*, 27(4):824–849, 2014.

[DLP+09] Joan Daemen, Mario Lamberger, Norbert Pramstaller, Vincent Rijmen, and Frederik Vercauteren. Computational Aspects of the Expected Differential Probability of 4-Round AES and AES-like Ciphers. *Computing*, 85(1âĂŞ2):85âĂŞ104, June 2009.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES — the Advanced Encryption Standard*. Springer-Verlag, 2002.

[DR06] Joan Daemen and Vincent Rijmen. Understanding Two-Round Differentials in AES. In Roberto De Prisco and Moti Yung, editors, *SCN*, volume 4116 of *LNCS*, pages 78–94. Springer, 2006.

[FGL09] Ewan Fleischmann, Michael Gorski, and Stefan Lucks. Attacking 9 and 10 Rounds of AES-256. In Colin Boyd and Juan González Nieto, editors, *Information Security and Privacy*, pages 60–72, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[FKL+01] Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Mike Stay, David Wagner, and Doug Whiting. Improved Cryptanalysis of Rijndael. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, and Bruce Schneier, editors, *Fast Software Encryption*, pages 213–230, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[GL08] Michael Gorski and Stefan Lucks. New Related-Key Boomerang Attacks on AES . In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *Progress in Cryptology - INDOCRYPT 2008*, pages 266–278, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[Gra18] Lorenzo Grassi. Mixture Differential Cryptanalysis: A New Approach to Distinguishers and Attacks on round-reduced AES. *IACR Trans. Symmetric Cryptol.*, 2018(2):133–160, 2018.

[GRR17] Lorenzo Grassi, Christian Rechberger, and Sondre Rønjom. Subspace Trail Cryptanalysis and its Applications to AES. *IACR Transactions on Symmetric Cryptology*, 2016(2):192–225, Feb. 2017.

[KKS01] John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified Boomerang Attacks Against Reduced-Round MARS and Serpent. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, and Bruce Schneier, editors, *Fast Software Encryption*, pages 75–93, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[KLMR16] Stefan Kölbl, Martin M. Lauridsen, Florian Mendel, and Christian Rechberger. Haraka v2 - Efficient Short-Input Hashing for Post-Quantum Applications. Cryptology ePrint Archive, Report 2016/098, 2016. https://eprint.iacr.org/2016/098.

[KR11] Lars R. Knudsen and Matthew J. B. Robshaw. *The Block Cipher Companion*. Springer Publishing Company, Incorporated, 2011.

[LDKK08] Jiqiang Lu, Orr Dunkelman, Nathan Keller, and Jongsung Kim. New Impossible Differential Attacks on AES. In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *Progress in Cryptology - INDOCRYPT 2008*, pages 279–293, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[Min17] Marine Minier. Improving Impossible-Differential Attacks against Rijndael-160 and Rijndael-224. *Designs, Codes and Cryptography*, 82(1):117–129, Jan 2017.

[MRST09] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In Orr Dunkelman, editor, *Fast Software Encryption*, pages 260–276, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[MRST10] Florian Mendel, Christian Rechberger, Martin Schläffer, and Søren S. Thomsen. Rebound Attacks on the Reduced Grøstl Hash Function. In Josef Pieprzyk, editor, *Topics in Cryptology - CT-RSA 2010*, pages 350–365, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[Mur11] Sean Murphy. The Return of the Cryptographic Boomerang. *IEEE Trans. Information Theory*, 57(4):2517–2521, 2011.

[NSA98] N.S.A. National Security Agency. SKIPJACK and KEA Algorithm Specifications, 1998.

[RBH17] Sondre Rønjom, Navid Ghaedi Bardeh, and Tor Helleseth. Yoyo Tricks with AES. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT I*, volume 10624 of *LNCS*, pages 217–243. Springer, 2017.

[Sel08] Ali Aydın Selçuk. On Probability of Success in Linear and Differential Cryptanalysis. *Journal of Cryptology*, 21(1):131–147, Jan 2008.

[SRP18] Dhiman Saha, Mostafizar Rahman, and Goutam Paul. New Yoyo Tricks with AES-based Permutations. *IACR Transactions on Symmetric Cryptology*, 2018(4):102–127, Dec. 2018.

[SSA10] H. Soleimany, A. Sharifi, and M. Aref. Improved Related-Key Boomerang Cryptanalysis of AES-256. In *2010 International Conference on Information Science and Applications*, pages 1–7, April 2010.

[SSL15] Kazuo Sakiyama, Yu Sasaki, and Yang Li. *Security of Block Ciphers: From Algorithm Design to Hardware Implementation*. Wiley Publishing, 1st edition, 2015.

[Tun12] Michael Tunstall. Improved "Partial Sums"-based Square Attack on AES. In Pierangela Samarati, Wenjing Lou, and Jianying Zhou, editors, *SECRYPT 2012 - Proceedings of the International Conference on Security and Cryptography, Rome, Italy, 24-27 July, 2012, SECRYPT is part of ICETE - The International Joint Conference on e-Business and Telecommunications*, pages 25–34. SciTePress, 2012.

[Wag99] David Wagner. The Boomerang Attack. In Lars R. Knudsen, editor, *FSE*, volume 1636 of *LNCS*, pages 156–170. Springer, 1999.

[WP19] Haoyang Wang and Thomas Peyrin. Boomerang Switch in Multiple Rounds. Application to AES Variants and Deoxys. *IACR Trans. Symmetric Cryptol.*, 2019(1):142–169, 2019.

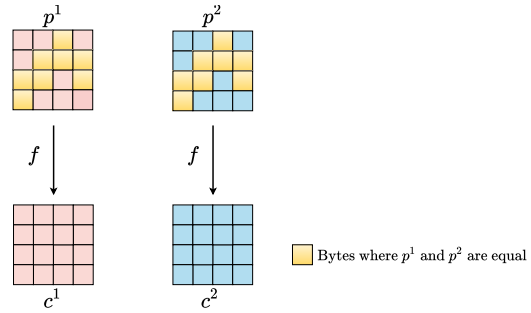
[WZF07] Wen-Ling Wu, Wen-Tao Zhang, and Deng-Guo Feng. Impossible Differential Cryptanalysis of Reduced-Round ARIA and Camellia. *J. Comput. Sci. Technol.*, 22(3):449âĂŞ456, May 2007.

[WZZ09] Wenling Wu, Lei Zhang, and Wentao Zhang. Improved Impossible Differential Cryptanalysis of Reduced-Round Camellia. In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *Selected Areas in Cryptography*, pages 442–456, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

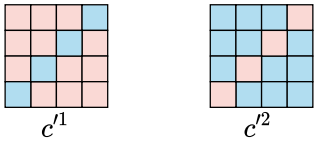
[ZWF07] Wentao Zhang, Wenling Wu, and Dengguo Feng. New Results on Impossible Differential Cryptanalysis of Reduced AES. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *Information Security and Cryptology - ICISC 2007*, pages 239–250, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

A Example Related to Lemma 2

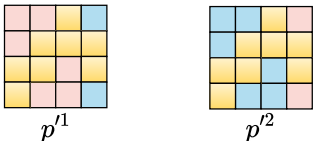
Let p^1, p^2, c^1 and c^2 be four `AES` states as shown in Fig. 13a where $c^1 = f(p^1)$ and $c^2 = f(p^2)$ (Here, f is the same function as described in Lemma 2). $\mathcal{ID}_{\{3\}}$ is swapped between c^1 and c^2 to obtain c'^1 and c'^2 as shown in Fig. 13b. Let $p'^1 = f^{-1}(c'^1)$ and $p'^2 = f^{-1}(c'^2)$. Now, swapping of $\mathcal{ID}_{\{3\}}$ between c^1 and c^2 can be equivalently considered as swapping of $\mathcal{C}_{\{3\}}$ between p'^1 and p'^2 due to the function f. Due to the bytes that are equal in p^1 and p^2, this can also be considered as swapping of $\mathcal{ID}_{\{2,3\}}$ between p^1 and p^2 (Fig. 13c).



(a) Depiction of p^1, p^2, c^1 and c^2.



(b) c'^1 and c'^2 which is obtained by swapping $\mathcal{ID}_{\{3\}}$ between c^1 and c^2.



(c) p'^1 and p'^2 which can be equivalently obtained by swapping $\mathcal{ID}_{\{2,3\}}$ between p^1 and p^2.

Figure 13: An example elaborating a case described in Lemma 2.

Note that, in this example a case is shown when $J = \{2, 3\}$; but, in Lemma 2 all the cases are considered where $J \subset_\phi \{0, 1, 2, 3\}$.

B Sample Trail for 5-round AES-128

Here, a trail for 5-round AES-128 as claimed in Section 4.1 is provided as an illustration. Note that, the trail is searched using only 2^{23} encryptions and checking whether the six specific bytes in the intermediate state after 4 rounds of encryption are inactive or not. The existence of such trails strengthens the validity of the attacks on 5-round and 6-round AES discussed in this paper. The pair of plaintexts p_1, p_2, the *key* and other intermediate states are stipulated in hexadecimal form.

$$p_1 = \begin{bmatrix} \texttt{E8} & \texttt{0} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{77} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{91} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{0} & \texttt{BF} \end{bmatrix} \qquad p_2 = \begin{bmatrix} \texttt{AC} & \texttt{0} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{7D} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{18} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{0} & \texttt{3F} \end{bmatrix}$$

$$key = \begin{bmatrix} \texttt{A0} & \texttt{85} & \texttt{AF} & \texttt{1B} \\ \texttt{39} & \texttt{9C} & \texttt{95} & \texttt{4B} \\ \texttt{79} & \texttt{29} & \texttt{EB} & \texttt{60} \\ \texttt{34} & \texttt{7E} & \texttt{D7} & \texttt{8A} \end{bmatrix}$$

- Initial difference of p_1 and p_2.

$$\begin{bmatrix} \texttt{44} & \texttt{0} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{0A} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{89} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{0} & \texttt{80} \end{bmatrix}$$

- Difference of intermediate states after 4 rounds of encryption (excluding the last mixcolumns operation).

$$\begin{bmatrix} \texttt{61} & \texttt{B5} & \texttt{EB} & \texttt{16} \\ \texttt{E7} & \texttt{7E} & \texttt{0} & \texttt{0} \\ \texttt{37} & \texttt{0} & \texttt{2C} & \texttt{0} \\ \texttt{0} & \texttt{79} & \texttt{17} & \texttt{0} \end{bmatrix}$$

- Difference of ciphertexts after 5 rounds of encryption.

$$\begin{bmatrix} \texttt{79} & \texttt{96} & \texttt{BE} & \texttt{76} \\ \texttt{B1} & \texttt{96} & \texttt{DE} & \texttt{F7} \\ \texttt{E3} & \texttt{4B} & \texttt{1A} & \texttt{64} \\ \texttt{68} & \texttt{11} & \texttt{02} & \texttt{56} \end{bmatrix}$$

- Difference of states after swapping the last column between ciphertexts and subsequent 5 rounds of decryption.

$$\begin{bmatrix} \texttt{6A} & \texttt{0} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{5D} & \texttt{0} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{A4} & \texttt{0} \\ \texttt{0} & \texttt{0} & \texttt{0} & \texttt{12} \end{bmatrix}$$

C Attacks on AES-256

The key recovery attacks on 5-round and 6-round AES-128 can be extended to mount attacks on 6-round and 7-round AES-256. This variant of AES is composed of 14 rounds and 15 subkeys are used where the first two subkeys are part of the master key. The remaining keys are derived from the master key using a key scheduling algorithm. Let K_0 and K_1 denote the first two subkeys. The attack idea is that if K_0 is correctly guessed then K_1 for 6/7-round AES-256 can be recovered by following strategies similar to the one proposed in this work for AES-128.

To mount an attack on 6/7-round AES-256, first K_0 needs to be guessed. Then intermediate states similar to the ones used for attacking AES-128 are constructed. These states are inverted one round by using the guessed value of key K_0. These inverted states form the input of AES-256, which are then queried to the encryption oracle. The states that are obtained from the decryption oracle are encrypted one round using the same guessed value of K_0 to obtain the intermediate states. It can be noted that for 6/7-round AES-256, these intermediate states along with the initially constructed ones reduce the attack to recover K_1 to a setting analogous to 5/6-round AES-128 respectively (as described in Section 4).

The attack depends on the guess of the key K_0. Brute-force attack is applied to recover the 128-bit key K_0 and thus 2^{128} try-outs are required. Hence, the data complexity is 2^{128} encryption queries and the time complexity is 2^{128} times of the corresponding values of the attacks on AES-128. However, the memory complexity remains the same as two independent key guesses has no effect on one another, i. e., the data obtained for one key guess have no use for another key guess. Therefore, the data, time and memory complexity of the key recovery attack on 6-round AES-256 are $2^{49} \times 2^{128} = 2^{177}$ adaptive chosen ciphertexts, $2^{48} \times 2^{128} = 2^{176}$ XOR operations and 2^{23} AES states respectively. The corresponding complexities of the attack on 7-round AES-256 are $2^{79.72} \times 2^{128} = 2^{207.72}$ adaptive chosen ciphertexts, $2^{78} \times 2^{128} = 2^{208}$ AES encryptions and 2^{28} AES states respectively.

D Algorithm for Key Recovery Attack on Pholkos

Algorithm 2: Algorithm for Key Recovery Attack on 10-round Pholkos

Result: The secret key

$v \leftarrow 1110$

Initialize a pholkos state δ by setting all bytes to 0

for $0 \leq m \leq 2$ **do**

  Initialize $K_m = \phi$

  for $0 \leq i < 2^{128}$ **do**

    $ctr[i] \leftarrow 0$

  end

  for $0 \leq i < 2^{186.2}$ **do**

    Choose 2 pholkos state $P^{1,i}$, $P^{2,i}$ such that only the 4-bytes in $\mathcal{D}_{\{m\}}(P^{1,i}[0] \oplus P^{2,i}[0])$ are active

    $C^{1,i} = Enc(P^{1,i})$, $C^{2,i} = Enc(P^{2,i})$

    $\delta[3] = \tau^v(C^{1,i}[3] \oplus C^{2,i}[3])$

    $C^{3,i} = C^{1,i} \oplus \delta$

    $C^{4,i} = C^{2,i} \oplus \delta$

    $P^{3,i} = Dec(C^{3,i})$ and $P^{4,i} = Dec(C^{4,i})$

    $DiaCount \leftarrow 0$

    for $0 \leq j \leq 3$ **do**

      for $0 \leq k \leq 3$ **do**

        if $\mathcal{D}_{\{k\}}(P^{3,i}[j] \oplus P^{4,i}[j])$ *is inactive* **then**

          $DiaCount \leftarrow DiaCount + 1$

        end

      end

    end

    if $DiaCount < 4$ **then**

      Discard $P^{1,i}$, $P^{2,i}$, $P^{3,i}$, $P^{4,i}$

      Go to Step 9

    end

    for $0 \leq j < 2^{128}$ **do**

      Use the 128-bit key to partially encrypt m-th diagonal of the 4 substates

      if *If there is one active byte corresponding to each of the four diagonals* **then**

        $ctr[j] \leftarrow ctr[j] + 1$

      end

    end

  end

  Include the top 2^7 key candidates with highest counter value in K_m.

end

K_3 is populated with all 2^{128} candidates

Exhaustively search for the right subkey in $K_0 \times K_1 \times K_2 \times K_3$

Finds the secret key from the subkey