Meet-in-the-Middle Attacks on Reduced Round Piccolo

Mohamed Tolba, Ahmed Abdelkhalek, and Amr M. Youssef

Concordia Institute for Information Systems Engineering,
Concordia University, Montréal, Quebéc, Canada

Abstract. Piccolo is a lightweight block cipher designed by Sony Corporation and published in CHES 2011. It inherits the Generalized Feistel Network (GFN) structure and operates on a 64-bit state. It has two versions; Piccolo-80 and Piccolo-128 with 80-bit and 128-bit keys, respectively. In this paper, we propose meet-in-the-middle attacks on 14-round reduced Piccolo-80 and 16, 17-round reduced Piccolo-128. First, we build a 5-round distinguisher by using specific properties of the linear transformation of Piccolo. This 5-round distinguisher is then used to launch a 14-round attack on Piccolo-80. As Piccolo-128 uses a different key schedule than what is used in Piccolo-80, we utilize the key dependent sieving technique to construct a 7-round distinguisher which is then employed to mount an attack on 16-round reduced Piccolo-128. To extend the attack to 17 rounds, we build a different 6-round distinguisher. For Piccolo-80, the time, data, and memory complexities of the 14-round attack are $2^{75.39}$ encryptions, 2^{48} chosen plaintexts, and $2^{73.49}$ 64-bit blocks, respectively. For Piccolo-128, the data complexity of both the 16-round and 17-round attacks is 2^{48} chosen plaintexts. The time and memory complexities of the 16-round (resp. 17-round) attack are 2^{123} (resp. $2^{126.87}$) encryptions, and $2^{113.49}$ (resp. $2^{125.99}$) 64-bit blocks. To the best of our knowledge, these are currently the best published attacks on both Piccolo-80 and Piccolo-128.

Keywords: Cryptanalysis, Meet-in-the-middle attacks, Generalized type-2 Feistel Structure.

1 Introduction

Recently, there is a huge demand for deploying resource-constrained devices such as RFID tags and wireless sensor nodes. To provide cryptographic security to such resource-constrained devices, new block ciphers of simple round function, and modest, or even no, key schedule are developed. As such, the design and analysis of hardware-oriented lightweight block ciphers have become a hot topic. HIGHT [14], mCrypton [21], DESL/DESXL [19], PRESENT [4], KATAN/KTANTAN [6], PRINTcipher [18], and Piccolo [25] are just few examples of such lightweight block ciphers that are designed to be efficiently deployed on resource-constrained devices.

Piccolo [25] is a hardware-oriented lightweight block cipher designed by Sony Corporation in 2011. It operates on a 64-bit state and has two versions; Piccolo-80 and Piccolo-128 with 80-bit and 128-bit keys, respectively. The differences between the two versions are the key schedule and the number of rounds. The structure of Piccolo inherits the Generalized Feistel Network (GFN) construction and has 4 branches, each of 16-bit length. Piccolo has been analyzed extensively where most of the results are reached using the biclique cryptanalysis technique [16,27,17,26]. The biclique cryptanalysis attack uses weaknesses that exist in the block cipher to accelerate the brute force attack and so it is regarded as a bruteforce-like attack.

Meet-in-the-Middle (MitM) attack is a single-key attack and it is the attack applied to Piccolo-80 and Piccolo-128 in [15] by the authors of Piccolo. The drawback of these two MitM attacks is that they require the full codebook, i.e., 2^{64} plaintexts/ciphertexts pairs and as explicitly mentioned in [15] the number of the attacked rounds would be reduced if the full codebook is not allowed. In addition, under the single-key setting Piccolo-80 and Piccolo-128 have been analyzed using the impossible differential attack [2]. Under the related-key setting, Piccolo has been analyzed using related-key impossible differential attack [23].

The MitM attack was first proposed by Diffie and Hellman in 1977 to be used in the cryptanalysis of Data Encryption Standard (DES) [11]. This attack splits the block cipher into two sub-ciphers such that $E = G_{K_2} \circ F_{K_1}$, where K_1 and K_2 are two distinct key sets which are used in F and G, respectively. Since the MitM attack requires low data complexity, it is considered as one of the major cryptanalysis techniques on block ciphers. However, finding two distinct key sets, K_1 and K_2, that cover a large number of rounds is quite challenging, especially in the block ciphers that use nonlinear key schedule. The three-subset MitM attack proposed by Bogdanov and Rechberger [5] solves this problem by splitting the key into three-subsets K_1, K_2, and K_c such that the key sets K_1 and K_2 may have common key bits that define the set K_c. The attack is then repeated for each possible value of the key bits in K_c. This approach succeeded in attacking the full KTANTAN cipher [5]. In addition to block ciphers, the MitM attack was applied to hash functions to launch preimage or second preimage attacks on Whirlpool [24] and Streebog [1], just to name a few.

Another line of research on the MitM attacks was triggered by Demirci and Selçuk when they were able to attack 8 rounds of both AES-192 and AES-256 [8]. In this attack, the cipher is split into three sub-ciphers, not just two as before, such that $E = E_2 \circ E_{mid} \circ E_1$, where E_{mid} exhibits a distinguishing property that is evaluated offline independently of the middle rounds keys. Then the keys used in E_1 and E_2 are guessed and checked in an online phase whether they verify the distinguishing property or not. The main downside of this attack is the high memory requirement to save a precomputation table. Later on, Dunkelman, Keller and Shamir [12] suggested two techniques to tackle the issue of the high memory requirement; differential enumeration and multisets which helped

Attack	Setting	# Rounds	Pre/Post	Time	Data	Memory	Reference
Impossible Differential	RK	14	None	$2^{68.19}$	$2^{68.19\dagger}$	N.A.	[23]
MitM	SK	14	None	2^{73}	$2^{64\dagger}$	2^{5}	[15]
Impossible Differential	SK	12	Pre	$2^{55.18}$	$2^{36.34}$ CC	2^{63}	[2]
Impossible Differential	SK	13	None	$2^{69.7}$	$2^{43.25}$ CP	2^{62}	[2]
MitM	SK	14	None	$2^{75.39}$	2^{48} CP	$2^{73.49}$	Section 3

Table 1. Summary of the cryptanalysis results on Piccolo-80 (RK: Related-Key Setting Attack, SK: Single-Key Setting Attack, Pre: Pre-whitening Key, Post: Post-whitening Key, CC: Chosen Ciphertext, CP: Chosen Plaintext, †: Requires the full codebook or more)

reduce the memory requirement but not enough to attack AES-128, however. Afterwards, Derbez *et al.* [9] reduced the memory requirement further by using a rebound-like idea and succeeded in attacking AES-128. Finally, Li, Jia and Wang proposed a key-dependent sieve [20] to further reduce the memory complexity of Derbezs attack and presented an attack on 9-round AES-192 and 8-round PRINCE. The MitM attack is not only applied to Substitution Permutation Network (SPN) block ciphers such as AES but also on Feistel Structure as well exemplified by the generic work presented by Guo *et al.* [13] and Lin *et al.* [22]. It is worth noting that despite its high memory requirement, the MitM attack based on Demirci and Selçuk technique proves to be quite successful as represented by the recent work in FSE 2015 against the SPN structure PRINCE [10] and the Feistel construction TWINE [3].

In this paper, we present MitM attacks on 14-round reduced Piccolo-80 and 16, 17-round reduced Piccolo-128. In the attack on Piccolo-80, we first construct a 5-round distinguisher then append 4 rounds at its top and 5 rounds at its bottom. The time, data, and memory complexities of the 14-round attack on Piccolo-80 are $2^{75.39}$ encryptions, 2^{48} chosen plaintexts, and $2^{73.49}$ 64-bit blocks, respectively. To attack 16-round reduced Piccolo-128, we build a 7-round distinguisher then append 3 rounds at its top and 6 rounds at its bottom. Extending the attack by one round using that 7-round distinguisher would require the whole key to be guessed. Hence, we construct a 6-round distinguisher, append 4 rounds at its top and 7 rounds at its bottom. The data complexity of both attacks on 16 and 17-round reduced Piccolo-128 is 2^{48} chosen plaintexts. The time, and memory complexities of the 16-round attack on Piccolo-128 are 2^{123} encryptions, and $2^{113.49}$ 64-bit blocks, respectively. The time, and memory complexities of the 17-round attack on Piccolo-128 are $2^{126.87}$ encryptions, and $2^{125.99}$ 64-bit blocks, respectively. Table 1 and 2 summarize our results and the previous results on Piccolo-80 and Piccolo-128, respectively.

Attack	Setting	# Rounds	Pre/Post	Time	Data	Memory	Reference
Impossible Differential	RK	21	None	$2^{117.77}$	$2^{117.77\dagger}$	N.A.	[23]
MitM	SK	21	None	2^{121}	$2^{64\dagger}$	2^{6}	[15]
Impossible Differential	SK	15	Post	$2^{125.4}$	$2^{58.7}$ CP	2^{61}	[2]
MitM	SK	16	Post	2^{123}	2^{48} CP	$2^{113.49}$	section 3
MitM	SK	17	Post	$2^{126.87}$	2^{48} CP	$2^{125.99}$	section 3

Table 2. Summary of the cryptanalysis results on Piccolo-128 (RK: Related-Key Setting Attack, SK: Single-Key Setting Attack, Pre: Pre-whitening Key, Post: Post-whitening Key, CP: Chosen Plaintext, †: Requires the full codebook or more)

The rest of the paper is organized as follows. In section 2, we provide the notations used throughout the paper and a brief description of Piccolo . Our attacks on 14 rounds of Piccolo-80 and 16, 17 rounds of Piccolo-128 are presented in section 3 and the paper is concluded in section 4.

2 Specifications of Piccolo

2.1 Notations

The following notations are used throughout the rest of the paper:

- $a_{(b)}$: A word a of length b bits.
- $a||b$: Concatenation of the two words a and b.
- a^t: Transposition of the vector or the matrix a.
- a_b: Representation of the word a in base b.
- K: The master key.
- k_i: The i^{th} 16-bit of K from left, where $0 \leq i < 5$ in Piccolo-80 and $0 \leq i < 8$ in Piccolo-128.
- rk_i: The 16-bit key used in round $\lfloor i/2 \rfloor$.
- wk_i: The 16-bit whitening key, where $0 \leq i < 4$.
- X_i: The 64-bit input to round i, where $0 \leq i \leq 26$ in Piccolo-80 and $0 \leq i \leq 32$ in Piccolo-128, X_0 is the plaintext P and X_{26} or X_{32} is the ciphertext C in Piccolo-80 and Piccolo-128, respectively.
- $X_i[j]$: The j^{th} nibble of X_i, where $0 \leq j < 16$.
- $X_i[j:l]$: The nibbles from j to l of X_i, where $j < l$.
- $X_i[j,l]$: The nibbles j and l of X_i.
- $\Delta X_i, \Delta X_i[j]$: The difference at state X_i and nibble $X_i[j]$, respectively.
- X_i^j: The j^{th} state of the 64-bit input to round i.

2.2 Specifications

There are two versions of Piccolo, depending on the key size, Piccolo-80 for 80-bit keys and Piccolo-128 for 128-bit keys. There are two differences between Piccolo-80 and Piccolo-128, the first is the number of rounds. Piccolo-80 iterates over 25 rounds, while Piccolo-128 runs 31 rounds. Piccolo's design employs a Generalized Feistel Network (GFN) structure and its internal state is divided into 4 words each of 16-bit length, i.e., we have 4 branches as shown in Figure 1. Therefore, each round has two Feistel Networks (FN). Each FN has two operations: an F-function (F) and an Add key (AK). The F-function is an unkeyed 16×16-bit function and is applied to the first branch of the FN and, as depicted in the right part of Figure 1, consists of three transformations [25]:

1. First S-box layer: A nonlinear layer that applies the same 4×4-bit bijective S-box S to the 16-bit $X_{(16)} = x_{0(4)}||x_{1(4)}||x_{2(4)}||x_{3(4)}$ data of the first branch of the FN as follows:
$$(x_{0(4)}, x_{1(4)}, x_{2(4)}, x_{3(4)}) \leftarrow (S(x_{0(4)}), S(x_{1(4)}), S(x_{2(4)}), S(x_{3(4)}))$$
2. Diffusion layer: The internal state is multiplied by a matrix M, where the multiplication is performed over $GF(2^4)$ defined by the irreducible polynomial $x^4 + x + 1$. Hence, the output of the first S-box layer is updated as follows:
$$(x_{0(4)}, x_{1(4)}, x_{2(4)}, x_{3(4)})^t \leftarrow M.(x_{0(4)}, x_{1(4)}, x_{2(4)}, x_{3(4)})^t,$$
3. Second S-box layer: It resembles the first S-box layer but applied to the output of the diffusion layer.

Each round of Piccolo contains two round keys used in the two FNs. Moreover, there are two pre-whitening keys wk_0, wk_1 that are xored with the internal state before the first round and two post-whitening keys wk_2, wk_3 that are xored with the internal state after the last round. After applying the two FN operations in each round, a permutation is performed on the byte level, as shown in Figure 1.

The key schedule takes an 80-bit master key K in Piccolo-80 such that $K = k_0||k_1||k_2||k_3||k_4$ or an 128-bit master key K in Piccolo-128 such that $K = k_0||k_1||k_2||k_3||k_4||k_5||k_6||k_7$ and generates the 4 16-bit whitening keys wk_i, $0 \leq i < 4$ and 50 16-bit round keys in Piccolo-80, as per Algorithm 1 or 62 16-bit round keys in Piccolo-128, as per Algorithm 2.

```
Data: Key Scheduling$(k_0, k_1, k_2, k_3, k_4)$
Result: $wk_i, 0 \leqslant i < 4$ and $rk_i, 0 \leqslant i < 50$
$wk_0 \leftarrow k_0^L||k_1^R, wk_1 \leftarrow k_1^L||k_0^R, wk_2 \leftarrow k_4^L||k_3^R, wk_3 \leftarrow k_3^L||k_4^R$;
for $i \leftarrow 0$ to 24 do
    $(rk_{2i}, rk_{2i+1}) \leftarrow (con^{80}_{2i}, con^{80}_{2i+1}) \oplus \begin{cases} (k_2, k_3) & \text{if } i \text{ mod } 5 = 0 \text{ or } 2 \\ (k_0, k_1) & \text{if } i \text{ mod } 5 = 1 \text{ or } 4 \\ (k_4, k_4) & \text{if } i \text{ mod } 5 = 3, \end{cases}$
end
```

Algorithm 1: The Key Schedule employed in Piccolo-80 [25]

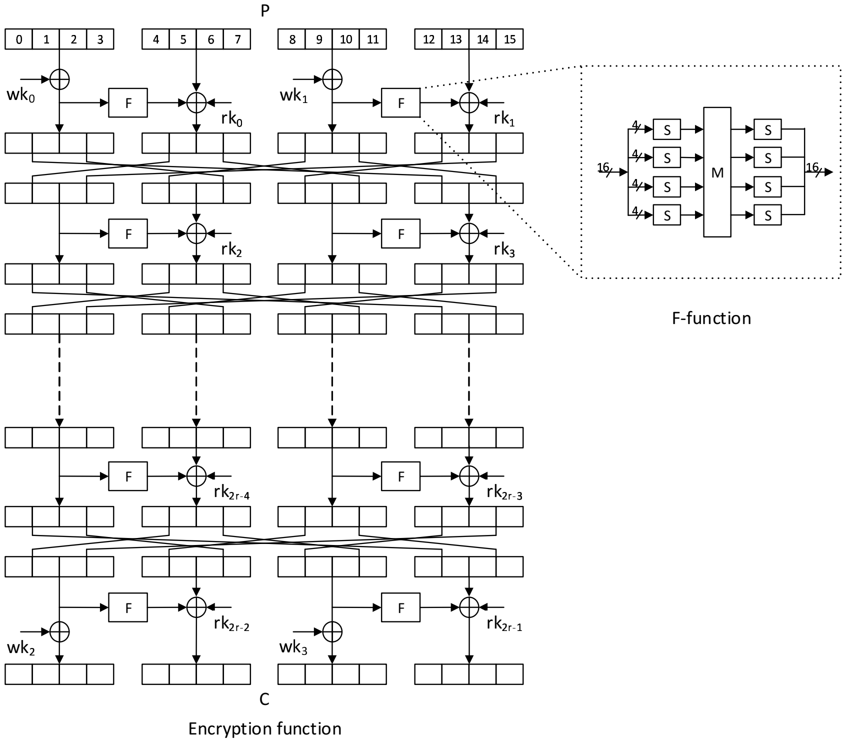

Fig. 1. Structure of Piccolo

In both algorithms, k_i^L and k_i^R are the left and right half byte of k_i. In Algorithm 1, $(con_{2i}^{80}||con_{2i+1}^{80})$ is calcualted as $(con_{2i}^{80}||con_{2i+1}^{80}) \leftarrow (c_{i+1}||c_0||c_{i+1}||00_2 ||c_{i+1}||c_0||c_{i+1}) \oplus 0f1e2d3c_{16}$, where c_i is a 5-bit representation of i. In algorithm 2, we have $(con_{2i}^{128}||con_{2i+1}^{128}) \leftarrow (c_{i+1}||c_0||c_{i+1}||00_2||c_{i+1}||c_0||c_{i+1}) \oplus 6547a98b_{16}$.

```
**Data**: Key Scheduling($k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7$)
**Result**: $wk_i, 0 \leqslant i < 4$ and $rk_i, 0 \leqslant i < 62$
$wk_0 \leftarrow k_0^L||k_1^R, wk_1 \leftarrow k_1^L||k_0^R, wk_2 \leftarrow k_4^L||k_7^R, wk_3 \leftarrow k_7^L||k_4^R$;
**for** $i \leftarrow 0$ *to* 61 **do**
    **if** $(i+2) mod 8 = 0$ **then**
        $(k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7) \leftarrow (k_2, k_1, k_6, k_7, k_0, k_3, k_4, k_5)$;
    **end**
    $rk_i \leftarrow k_{(i+2) mod 8} \oplus con_i^{128}$,;
**end**
```

Algorithm 2: The Key Schedule employed in Piccolo-128 [25]

We measure the memory complexity of our attacks as 64-bit Piccolo blocks and the time complexity in terms of the equivalent number of reduced-round Piccolo encryptions.

3 MitM Attacks on reduced round Piccolo

Generally in the MitM attacks, a reduced round block cipher is split into three sub-ciphers such that $E = E_2 \circ E_{mid} \circ E_1$, where E_{mid} exhibits a distinguishing property. This distinguishing property is evaluated in the offline phase. Then, in the online phase, the keys used in the analysis rounds E_1 and E_2 are guessed and checked whether they verify the distinguishing property or not. If they verify it, they are considered as key candidates, otherwise they are discarded. In our attacks, the distinguishing property is a truncated differential where its input takes a set of possible values and its output is a parameterized function of the input. The values of the output corresponding to the input form an ordered sequence that is used as our property to identify the right key guess. All the ordered sequences resulting from all the possible combinations of the parameters are stored in a precomputation table. As the size of this precomputation table is usually huge, we use two techniques inspired by the MitM attacks on SPN [20,10] to reduce its size and hence we are able to attack more rounds than what we can attack without these techniques.

The δ-*set* concept [7], as captured by Definition 1, is used to build our distinguishers.

Definition 1. *(δ-set, [7]). A δ-set for nibble-oriented cipher is a set of 16 state values that are all different in one nibble (the active nibble) and are all equal in the remaining nibbles (the inactive nibbles).*

The following subsections contain a detailed description of our attacks on 14-round Piccolo-80 and 16, 17-round Piccolo-128, respectively.

3.1 A MitM Attack on 14-Round Piccolo-80

In Piccolo, by noting that when the δ-set is chosen at the second input branch of the FN and the corresponding ordered sequence is evaluated at its first output branch, a distinguisher that minimizes the number of parameters can be constructed. However, such distinguisher does not lead to the best attack on Piccolo-80 since it can be extended upwards in the plaintext direction by two rounds only. If a third round is appended, the full codebook is needed due to the diffusion transformation utilized in Piccolo. Hence, to increase the number of rounds appended on top of the distinguisher, the δ-set is chosen at the first (instead of the second) input branch of the FN which, unfortunately, increases the number of parameters by two additional parameters. Then, in order to reduce the number of parameters, we exploit the properties of the diffusion operation

M. In particular, we choose the δ-set to be after the first S-box layer of the first F-function such that after the diffusion transformation, only two nibbles are active, as shown in Figure 2. By enumerating all the possible values of three active input nibbles of the linear diffusion, it was found that such δ-set that has three active nibbles at the input of the linear transformation, and two active nibbles at its output contains 15 differences. Such δ-set enables us to build a 5-round distinguisher and overcome the problem of the two additional parameters when the δ-set is chosen at the first branch of the FN, as depicted in Figure 2, and captured by the following proposition:

Proposition 1. *Consider the encryption of a δ-set $\{Y^0 = P'^0[0:3]||P^0[4:15], Y^1 = P'^1[0:3]||P^1[4:15], \cdots, Y^{15} = P'^{15}[0:3]||P^{15}[4:15]\}$ through 5 rounds of Piccolo. The ordered sequence $[X_5^0[14:15] \oplus X_5^1[14:15], X_5^0[14:15] \oplus X_5^2[14:15], \cdots, X_5^0\ [14:15] \oplus X_5^{15}[14:15]$ is fully determined by the following 5 16-bit parameters, $X_0^0[0:3]$, $X_1^0[8:11]$, $X_2^0[0:3]$, $X_2^0[8:11]$ and $X_3^0[0:3]$.*

The above proposition means that we have $2^{5\times16} = 2^{80}$ ordered sequences out of the $2^{15\times8} = 2^{120}$ theoretically possible ones.

Proof. The knowledge of the δ-*set* $= \{Y^0, Y^1, \cdots, Y^{15}\}$ allows us to determine $[Y^0 \oplus Y^1, Y^0 \oplus Y^2, \cdots, Y^0 \oplus Y^{15}]$. In the sequel, we show that the ordered sequence at $X_5[14:15]$ can be determined uniquely by the knowledge of the 5 16-bit parameters mentioned in proposition 1. As the δ-set is chosen at the input of the linear transformation M, it has to be propagated forward through M and backward through the first S-box layer to be able to determine the difference $\Delta X_1[6:7, 10:11, 13]$. To do this, we need to know three nibbles after the first S-box layer and two nibbles before the second S-box layer of the first F-function in the first round. However, the knowledge of only 4 nibbles $X_0^0[0:3]$ suffices to bypass the F-function and to compute $\Delta X_1[6:7, 10:11, 13]$. It is to be noted that only two nibbles are active after the F-function due to the restriction we place on the choice of our δ-set. Then, we bypass the second round by the knowledge of $X_1^0[8:11]$ which allows us to compute $\Delta X_2[2:3, 8:11, 14:15]$. By repeating the previous steps and propagating the differences further, $\Delta X_5[14:15]$ is computed. It is worth noting that there are nibbles which should have difference but appear in Figure 2 as if they do not have any difference, because their knowledge do not impact the computation of the ordered sequence at $X_5[14:15]$. For instance, after the third (resp. fourth) round, the difference at $X_3[8:11]$ (resp. $X_4[0:1]$) should be non-zero because the difference at $X_2[2:3, 8:11]$ (resp. $X_3[0:3]$) is non-zero.

In what follows we show how to utilize the above described distinguisher to attack 14-round Piccolo-80 starting from the 5^{th} round (round 4) till the 18^{th} round (round 17) without the pre-whitening or the post-whitening keys. The attack relies on the previous proposition and exploits the linearity of the key schedule to build a 5-round distinguisher and then append 4 rounds above it

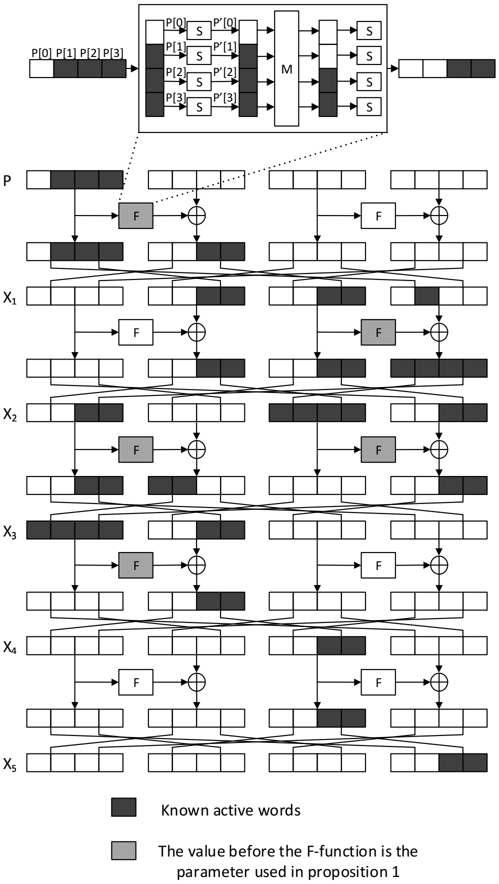

Fig. 2. 5-Round Distinguisher to attack 14-round Piccolo-80

and 5 rounds below it, as seen in Figure 3. The attack has two phases as follows:

Offline Phase. As demonstrated in Proposition 1, we determine all the 2^{80} ordered sequences and store them in a hash table H.

Online Phase. The online phase, as seen in Figure 3, proceeds as follows:

1. A plaintext P^0 is chosen as a reference to all the differences in the δ-set.
2. The δ-set P^0, P^1, $\cdots$, P^{15} is determined by guessing the state variables $X_6^0[8:11]$, $X_6^0[6:7, 12:13]$, $X_6^0[4:5, 14:15]$, and $X_8^0[1:3]$ to decrypt the δ-set differences.
3. The corresponding ciphertexts C^0, C^1, $\cdots$, C^{15} are requested.
4. The ordered sequence differences $[X_{13}^0[14:15] \oplus X_{13}^1[14:15], X_{13}^0[14:15] \oplus X_{13}^2[14:15], \cdots, X_{13}^0[14:15] \oplus X_{13}^{15}[14:15]]$ are determined by guessing the state variables $X_{13}^0[8:11]$, $X_{14}^0[0:3]$, $X_{14}^0[8:11]$, $X_{15}^0[0:3]$, $X_{15}^0[8:11]$,

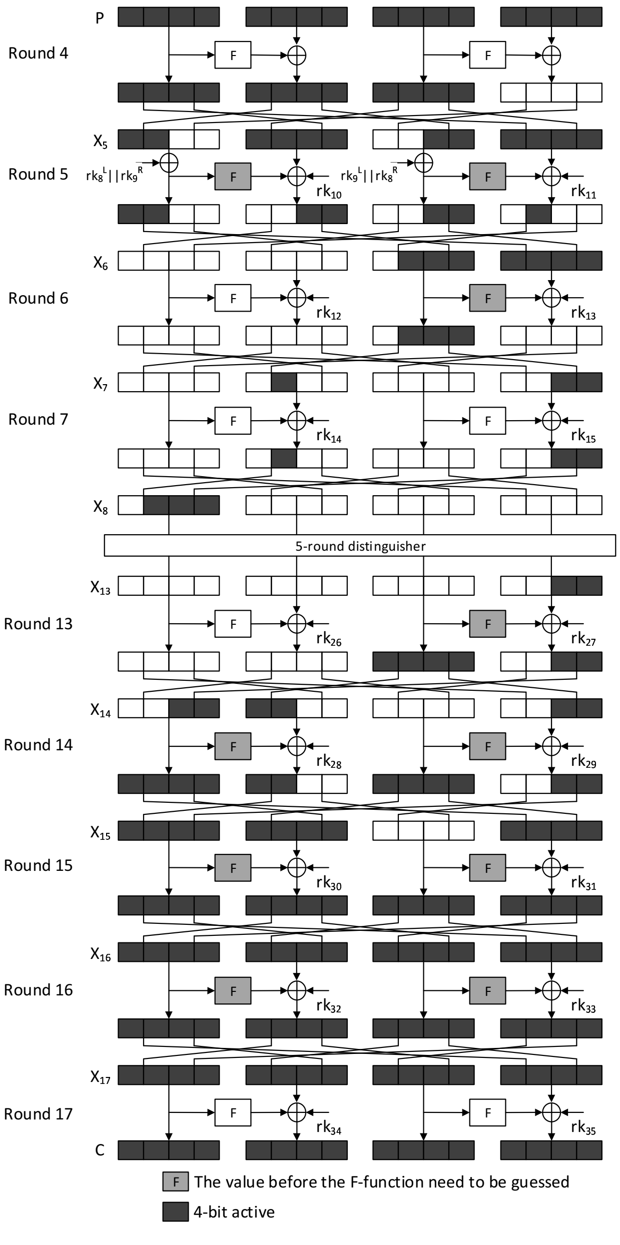

Fig. 3. 14-Round Attack on Piccolo-80

$X^0_{16}[0:3]$, $X^0_{16}[8:11]$ that are required to decrypt the ciphertext differences $[C^0 \oplus C^1, C^0 \oplus C^2, \cdots, C^0 \oplus C^{15}]$.

5. The guessed state variables are filtered by checking if the computed ordered sequence exists in H or not.

The evaluation of the δ-set and the corresponding ordered sequence as demonstrated in steps 2 and 4 require the guessing of 43 internal state nibbles. Guessing these 43 internal state nibbles makes the attack complexity exceeds the exhaustive search. Therefore, we analyze the key schedule searching for relations between the round keys to reduce the number of guessed parameters. As a result, we find that starting the attack from the 5^{th} round, i.e., round 4 is the best choice to reduce the number of the guessed parameters. Indeed, by only guessing k_0, k_1, k_2, k_3 and with the knowledge of P^0, we are able to compute $X^0_6[8:11]$, $X^0_6[6:7, 12:13]$, $X^0_6[4:5, 14:15]$, and $X^0_8[1:3]$. The knowledge of $[C^0$, C^1, $\cdots$, $C^{15}]$, $[C^0 \oplus C^1, C^0 \oplus C^2, \cdots, C^0 \oplus C^{15}]$ and the same keys guessed above enables us to evaluate the state variables $X^0_{13}[8:11]$, $X^0_{14}[0:3]$, $X^0_{14}[8:11]$, $X^0_{15}[0:3]$, $X^0_{15}[8:11]$, $X^0_{16}[0:3]$, $X^0_{16}[8:11]$. Consequently, we have to guess 4 round keys (16 nibbles) instead of guessing 43 internal state nibbles. Moreover and in order to reduce the memory complexity of the attack even further, we choose to compute the ordered sequence at only 6-bit instead of 8-bit, where any arbitrary 6-bit from the 8-bit can be chosen. Therefore, the probability of a wrong key to be a key candidate is $2^{80-(15\times 6)} = 2^{-10}$. As we have 2^{64} keys to be guessed, we expect that only $2^{64-10} = 2^{54}$ keys to remain after step 5. Hence, to recover the master key we guess k_4 and test the 2^{54} key candidates along with k_4 with just two plaintext/ciphertext pairs.

Attack Complexity. The memory complexity is determined by the size of the hash table H created in the offline phase. This table contains 2^{80} ordered sequences, where each ordered sequence has 15 6-bit differences. Therefore, the memory complexity is $2^{80} \times 90/64 = 2^{80.49}$ 64-bit blocks. To reduce the memory complexity below 2^{80}, we use a simple tradeoff and store a fraction $1/\alpha$ of H and repeat the attack α times as now we have decreased the chance to hit one element in H. We choose $\alpha = 2^7$ to reduce the memory complexity while still having a non-marginal time complexity. Hence, the memory complexity of the attack is $2^{73.49}$. As depicted in Figure 3, we shift the round keys rk_8, rk_9 from the 5^{th} round to the 6^{th} round. This round keys shift enable us to append 4 rounds, and not just 3 rounds, on top of our 5-round distinguisher with the same data complexity and without requiring the full codebook. To illustrate how this is possible, we choose our plaintexts such that after the 5^{th} round the words $X_5[2:3, 8:9]$ take a fixed value while the remaining words of X_5 take all the possible values. Hence, the data required can be formed using one structure that contains 2^{48} states of X_5. In order to obtain its corresponding plaintexts, we simply decrypt this structure as no keys are involved in this round any more. Accordingly, the data complexity is upper bounded by 2^{48} chosen plaintexts. Repeating the attack 2^7 times does not increase the data complexity as we just choose a different reference plaintext P^0. The time complexity of the offline phase is determined by the time needed to build the hash table H that now contains 2^{73}, instead of 2^{80}, ordered sequences. Therefore, the time complexity of the offline phase is $2^{73} \times 16 \times 5/(2 \times 14) = 2^{74.51}$. The time complexity of the online phase consists of two parts: the time required to filter the key space which is estimated to be $2^7 \times 2^{64} \times 16 \times (6+9)/(2 \times 14) = 2^{74.1}$ and the time to recover the master key which is estimated to be $2 \times 2^{(64-10)} \times 2^{16} = 2^{71}$. Hence, the total time complexity of the attack is $2^{74.51} + 2^{74.1} + 2^{71} \approx 2^{75.39}$ 14-round Piccolo-80 encryptions.

3.2 A MitM Attack on 16-Round Piccolo-128

Reusing the ideas of the attack on Piccolo-80 does not lead to the best attack on Piccolo-128 because the key schedule of the latter is different. Therefore, we use the key dependent sieving technique in order to build a longer distinguisher with the least number of parameters. As depicted in Figure 4, we construct a 7-round distinguisher, that we employ to attack 16-round Piccolo-128 from the 2^{nd} round (round 1) to the 17^{th} round (round 16) with the post-whitening keys. The δ-set of our 7-round distinguisher is chosen to be active at $P[7]$ and our distinguisher is built using Proposition 2.

Proposition 2. *Consider the encryption of a δ-set $\{P^0, P^1, \cdots, P^{15}\}$ through 7 rounds of Piccolo. The ordered sequence $[X_7^0[13:15] \oplus X_7^1[13:15], X_7^0[13:15] \oplus X_7^2[13:15], \cdots, X_7^0[13:15] \oplus X_7^{15}[13:15]]$ is fully determined by the following 8 16-bit parameters $X_1^0[8:11]$, $X_2^0[0:3]$, $X_2^0[8:11]$, $X_3^0[0:3]$, $X_3^0[8:11]$, $X_4^0[0:3]$, $X_4^0[8:11]$ and $X_5^0[0:3]$.*

The previous 5-round distinguisher of our attack on 14-round Piccolo-80 is independent of the round keys while our 7-round distinguisher that we utilize to attack 16-round Piccolo-128 uses the round keys to reduce the number of parameters. Assuming that we know the internal state X_3^0, i.e., 4 parameters, and the round keys rk_6, rk_7, we can evaluate X_4^0. Therefore, the 6 F-functions from the third round to the fifth round of the distinguisher can be bypassed. To bypass the other two F-functions, we need to know rk_4^L, rk_5^R, rk_8^L, and rk_9^R only. If rk_4, rk_8 depend on the same k_i and rk_5, rk_9 rely on the same k_j then we can bypass the other two F-functions by guessing only k_i^L, and k_j^R. In such case, we can bypass the 8 F-functions of our 7-round distinguisher by guessing 7 parameters only. By placing our distinguisher to cover from the 5^{th} round (round 4) to the 11^{th} round (round 10), the number of parameters in proposition 2 is reduced to 7 parameters only. In that case, k_i is k_4 and k_j is k_5 and the 7 16-bit parameters of our distinguisher are the state X_7^0, k_1, k_6, k_4^L, and k_5^R. Our 16-round attack is then built by appending 3 and 6 rounds at the top and the bottom of our 7-round distinguisher, respectively. As shown in Figure 5, the attack follows the same steps as the previous attack on Piccolo-80 while considering the new position of the δ-set at $X_4[7]$ and the different position of the corresponding ordered sequence at $X_{11}[13:15]$. In the online phase, the knowledge of P^0 and

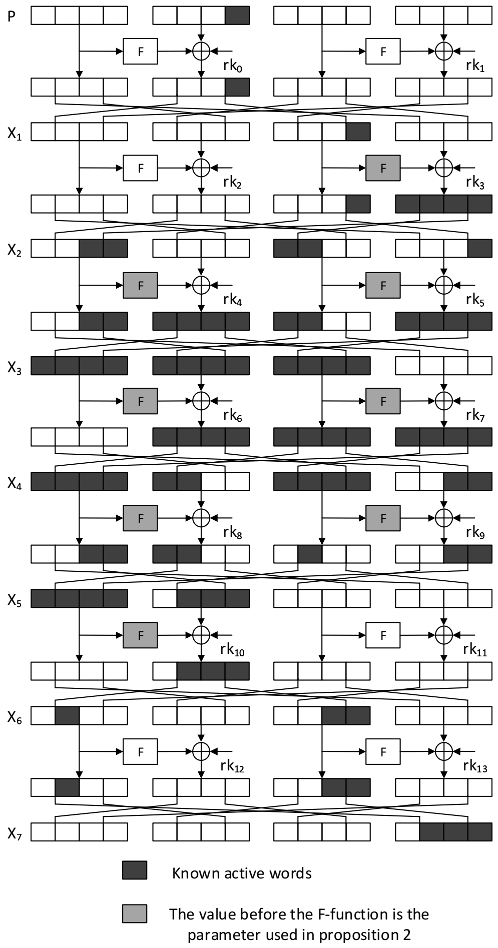

Fig. 4. 7-Round Distinguisher to attack 16-round Piccolo-128

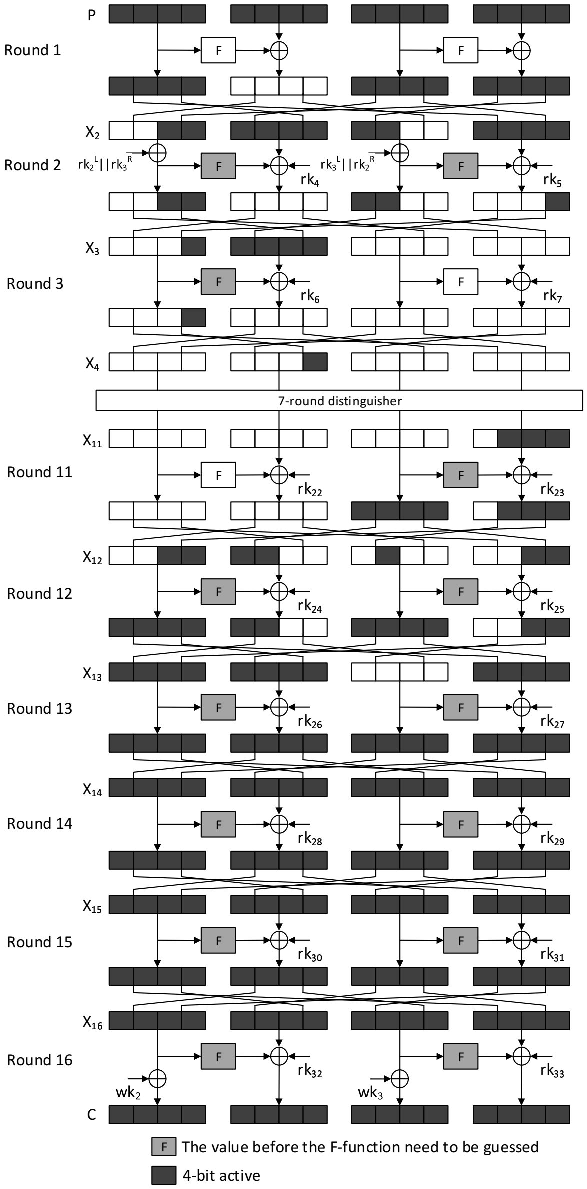

Fig. 5. 16-Round Attack on Piccolo-128

the guessing of k_4, k_5, k_6^L, and k_7^R enable us to partially decrypt $X_4[7]$ and determine the δ-set. From the other direction, by the knowledge of $[C^0, C^1, \cdots, C^{15}]$, $[C^0 \oplus C^1, C^0 \oplus C^2, \cdots, C^0 \oplus C^{15}]$ and the guessing of k_0, k_1, k_2, k_3^R, k_5, k_6, and k_7, we can compute the ordered sequence at $X_{11}[13:15]$. Hence, in total we need to guess seven and half keys, i.e., k_0, k_1, k_2, k_3^R, k_4, k_5, k_6, and k_7, in order to mount our attack on 16-round Piccolo-128.

Attack Complexity. The memory complexity is estimated to be $2^{7*16} \times (15 \times 12)/64 \approx 2^{113.49}$ 64-bit blocks and the data complexity is 2^{48} chosen plaintexts. The time complexity is $2^{112} \times 16 \times 8/(2 \times 16) + 2^{120} \times 16 \times (5+11)/(2 \times 16) + 2 \times 2^{(120-68)} \times 2^8 = 2^{114} + 2^{123} + 2^{61} \approx 2^{123}$ 16-round Piccolo-128 encryptions.

3.3 A MitM Attack on 17-Round Piccolo-128

To extend the attack on Piccolo-128 by one more round, we have to build another distinguisher, as illustrated in Figure 6 because using the previous 7-round distinguisher requires the guessing of the whole key space. Using this new 6-round distinguisher, which needs 8 parameters, we attack 17-round Piccolo-128 from the 5^{th} round (round 4) to the 21^{st} round (round 20) with the post-whitening keys. We append 4 and 7 rounds at the top and the bottom of our 6-round distinguisher, respectively. To launch the attack on 17-round Piccolo-128, we need to guess seven and half keys, as shown in Figure 7. These keys are $k_0^R, k_1, k_2, k_3, k_4, k_5, k_6, k_7$. The attack procedure follows the same steps of the previous attacks.
Attack Complexity. The memory complexity is estimated to be $2^{8\times16} \times (15 \times 12)/64 \approx 2^{129.49}$ 64-bit blocks. Since the memory complexity exceeds 2^{128}, we store a fraction $1/\alpha$ of the hash table H. $\alpha = 2^{3.5}$ is chosen so that the memory complexity does not exceed 2^{128} while having a non-marginal time complexity. Therefore, the memory complexity is $2^{125.99}$ 64-bit blocks. The data complexity is 2^{48} chosen plaintexts. Regarding the time complexity, since we do not store a fraction of the hash table, we have to repeat the online attack $2^{3.5}$ times. The time complexity of the offline phase is estimated to be $2^{128-3.5} \times 16 \times 8/(2 \times 17) \approx 2^{126.41}$. We use the partial computation technique in order to reduce the time complexity of the online phase. First, guessing the keys k_0^R, k_3^L, k_6, k_7 enables us to identify the δ-set and the time of this step is evaluated to be $2^{48} \times 16 \times 5/(2\times17) \approx 2^{49.23}$. By guessing k_4, k_5 we can partially decrypt through round 20 and this step is estimated to be $2^{80} \times 16 \times 2/(2 \times 17) \approx 2^{79.91}$. Then, guessing k_2 enables us to compute the output of the first F-function in round 19 and is estimated to be $2^{96} \times 16 \times 1/(2 \times 17) \approx 2^{94.91}$. Afterwards, guessing k_1 enables us to partially decrypt through round 19 and 18 as well as the first F-function of round 17 and needs $2^{112} \times 16 \times 4/(2 \times 17) \approx 2^{112.91}$ encryptions. Finally, guessing k_3^R enables us to compute the ordered sequence and this step needs $2^{120} \times 16 \times 6/(2\times17) \approx 2^{121.5}$ encryptions. Accordingly, the time complexity of the online phase is $2^{49.23} + 2^{79.91} + 2^{94.91} + 2^{112.91} + 2^{121.5} \approx 2^{121.5}$ and it will be repeated $2^{3.5}$ times so, all in all, it is estimated to be 2^{125}. Recovering the master key using two plaintext/ciphertext pairs requires $2 \times 2^{120} \times 2^{128-180} \times 2^8 =$

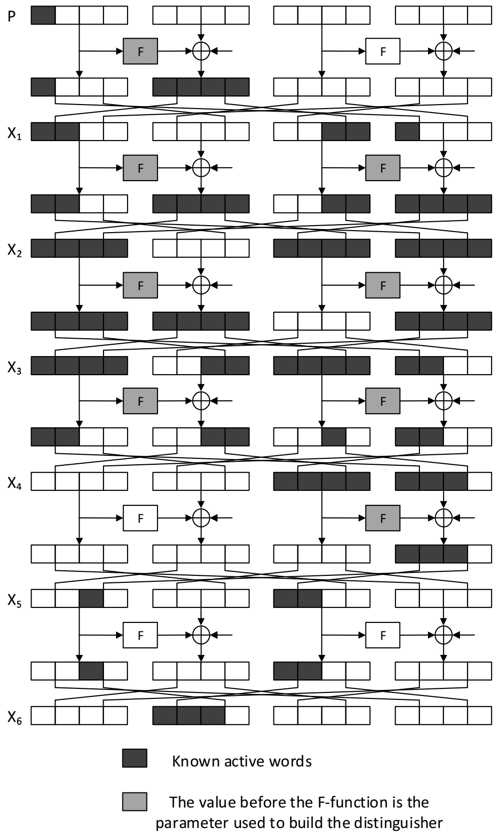
P
F
F
X_1
F
F
X_2
F
F
X_3
F
F
X_4
F
F
X_5
F
F
X_6
Known active words
The value before the F-function is the parameter used to build the distinguisher

Fig. 6. 6-Round Distinguisher to attack 17-round Piccolo-128

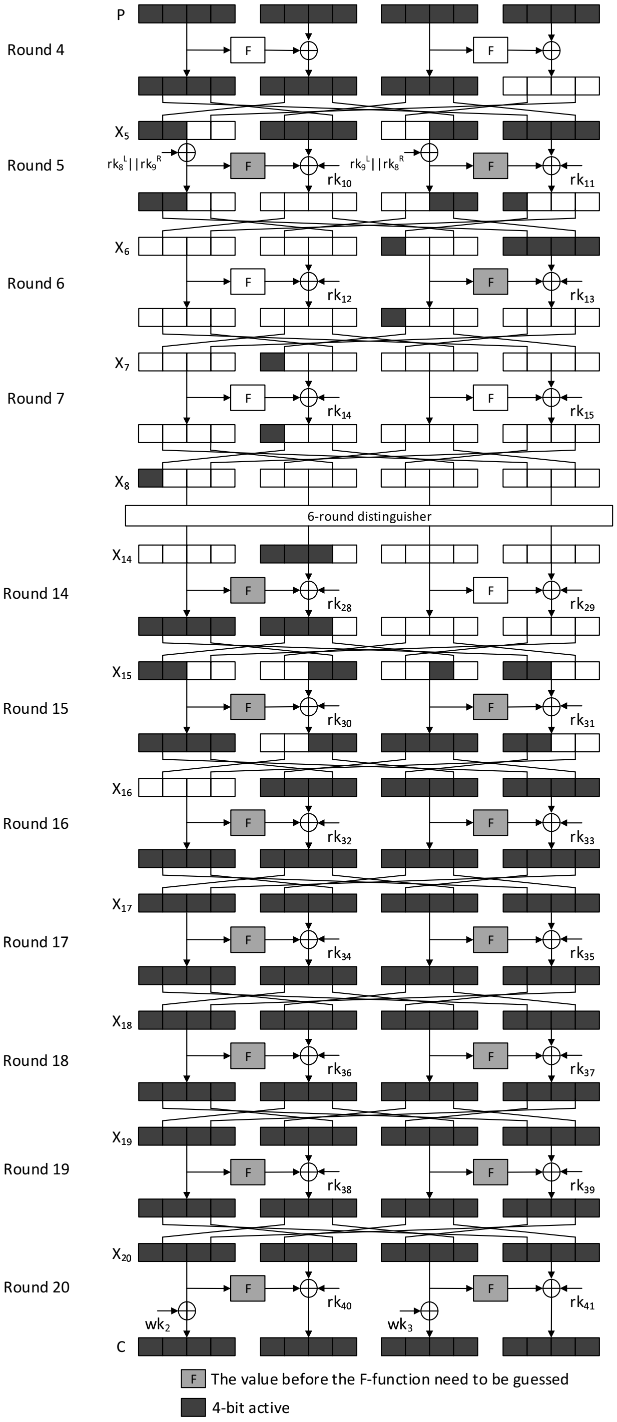

Fig. 7. 17-Round Attack on Piccolo-128

2^{77}. The total time complexity of the attack is $2^{126.41} + 2^{125} + 2^{77} \approx 2^{126.87}$ encryptions.

4 Conclusion

In this work, we presented MitM attacks on 14-round reduced Piccolo-80 and 16, 17-round reduced piccolo-128. All these attacks on Piccolo-80 and Piccolo-128 require the same data complexity of 2^{48} chosen plaintexts. The time complexities of the MitM attacks on 14-round Piccolo-80 and 16, 17-round Piccolo-128 are $2^{75.39}$, 2^{123}, and $2^{126.87}$, respectively. Their memory complexities are $2^{73.49}$, $2^{113.49}$, and $2^{125.99}$ for the 14-round Piccolo-80 and 16, 17-round Piccolo-128, respectively.

References

1. AlTawy, R., and Youssef, A. Preimage Attacks on Reduced-Round Stribog. In *Progress in Cryptology AFRICACRYPT 2014*, D. Pointcheval and D. Vergnaud, Eds., vol. 8469 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 109–125.
2. Azimi, S., Ahmadian, Z., Mohajeri, J., and Aref, M. Impossible differential cryptanalysis of Piccolo lightweight block cipher. In *Information Security and Cryptology (ISCISC), 11th International ISC Conference on* (Sept 2014), pp. 89–94.
3. Biryukov, A., Derbez, P., and Perrin, L. P. Differential Analysis and Meet-in-the-Middle Attack against Round-Reduced TWINE. Fast Software Encryption 2015. *to appear*.
4. Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J., Seurin, Y., and Vikkelsoe, C. PRESENT: An Ultra-Lightweight Block Cipher. In *Cryptographic Hardware and Embedded Systems - CHES 2007* (2007), P. Paillier and I. Verbauwhede, Eds., vol. 4727 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 450–466.
5. Bogdanov, A., and Rechberger, C. A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In *Selected Areas in Cryptography* (2010), A. Biryukov, G. Gong, and D. R. Stinson, Eds., vol. 6544 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 229–240.
6. Cannière, C., Dunkelman, O., and Knežević, M. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In *Cryptographic Hardware and Embedded Systems - CHES 2009* (Berlin, Heidelberg, 2009), C. Clavier and K. Gaj, Eds., vol. 5747 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 272–288.
7. Daemen, J., Knudsen, L., and Rijmen, V. The block cipher SQUARE. In *Fast Software Encryption*, E. Biham, Ed., vol. 1267 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1997, pp. 149–165.
8. Demirci, H., and Selçuk, A. A. A Meet-in-the-Middle Attack on 8-Round AES. In *Fast Software Encryption*, K. Nyberg, Ed., vol. 5086 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, pp. 116–126.

9. Derbez, P., Fouque, P.-A., and Jean, J. Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In *Advances in Cryptology EUROCRYPT 2013*, T. Johansson and P. Q. Nguyen, Eds., vol. 7881 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 371–387.
10. Derbez, P., and Perrin, L. Meet-in-the-Middle Attacks and Structural Analysis of Round-Reduced PRINCE. Fast Software Encryption 2015. *to appear*.
11. Diffie, W., and Hellman, M. E. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Computer 10*, 6 (June 1977), 74–84.
12. Dunkelman, O., Keller, N., and Shamir, A. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. In *Advances in Cryptology - ASIACRYPT 2010*, M. Abe, Ed., vol. 6477 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2010, pp. 158–176.
13. Guo, J., Jean, J., Nikolić, I., and Sasaki, Y. Meet-in-the-Middle Attacks on Generic Feistel Constructions. In *Advances in Cryptology ASIACRYPT 2014*, P. Sarkar and T. Iwata, Eds., vol. 8873 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2014, pp. 458–477.
14. Hong, D., Sung, J., Hong, S., Lim, J., Lee, S., Koo, B.-S., Lee, C., Chang, D., Lee, J., Jeong, K., Kim, H., Kim, J., and Chee, S. HIGHT: A New Block Cipher Suitable for Low-resource Device. In *Cryptographic Hardware and Embedded Systems - CHES 2006* (Berlin, Heidelberg, 2006), L. Goubin and M. Matsui, Eds., vol. 4249 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 46–59.
15. Isobe, T., and Shibutani, K. Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo. In *Information Security and Privacy*, W. Susilo, Y. Mu, and J. Seberry, Eds., vol. 7372 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 71–86.
16. Jeong, K. Cryptanalysis of block cipher Piccolo suitable for cloud computing. *The Journal of Supercomputing 66*, 2 (2013), 829–840.
17. Jeong, K., Kang, H., Lee, C., Sung, J., and Hong, S. Biclique Cryptanalysis of Lightweight Block Ciphers PRESENT, Piccolo and LED. IACR Cryptology ePrint Archive, 2012/621, 2012. https://eprint.iacr.org/2012/621.pdf.
18. Knudsen, L., Leander, G., Poschmann, A., and Robshaw, M. PRINTcipher: A Block Cipher for IC-Printing. In *Cryptographic Hardware and Embedded Systems, CHES 2010*, S. Mangard and F.-X. Standaert, Eds., vol. 6225 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2010, pp. 16–32.
19. Leander, G., Paar, C., Poschmann, A., and Schramm, K. New Lightweight DES Variants. In *Fast Software Encryption*, A. Biryukov, Ed., vol. 4593 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, pp. 196–210.
20. Li, L., Jia, K., and Wang, X. Improved Meet-in-the-Middle Attacks on AES-192 and PRINCE. IACR Cryptology ePrint Archive, 2013/573, 2013. https://eprint.iacr.org/2013/573.pdf.
21. Lim, C., and Korkishko, T. mCrypton A Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors. In *Information Security Applications*, J.-S. Song, T. Kwon, and M. Yung, Eds., vol. 3786 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2006, pp. 243–258.
22. Lin, L., and Wu, W. Improved Meet-in-the-Middle Distinguisher on Feistel Schemes. IACR Cryptology ePrint Archive, 2015/051, 2015. https://eprint.iacr.org/2015/051.pdf.
23. Minier, M. On the Security of Piccolo Lightweight Block Cipher against Related-Key Impossible Differentials. In *Progress in Cryptology INDOCRYPT 2013*,

G. Paul and S. Vaudenay, Eds., vol. 8250 of *Lecture Notes in Computer Science*. Springer International Publishing, 2013, pp. 308–318.

24. Sasaki, Y., Wang, L., Wu, S., and Wu, W. Investigating Fundamental Security Requirements on Whirlpool: Improved Preimage and Collision Attacks. In *Advances in Cryptology ASIACRYPT 2012*, X. Wang and K. Sako, Eds., vol. 7658 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 562–579.
25. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., and Shirai, T. Piccolo: An Ultra-Lightweight Blockcipher. In *Cryptographic Hardware and Embedded Systems CHES 2011*, B. Preneel and T. Takagi, Eds., vol. 6917 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 342–357.
26. Song, J., Lee, K., and Lee, H. Biclique Cryptanalysis on Lightweight Block Cipher: HIGHT and Piccolo. *Int. J. Comput. Math. 90*, 12 (2013), 2564–2580.
27. Wang, Y., Wu, W., and Yu, X. Biclique Cryptanalysis of Reduced-Round Piccolo Block Cipher. In *Information Security Practice and Experience*, M. Ryan, B. Smyth, and G. Wang, Eds., vol. 7232 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 337–352.