Known-key Distinguisher on Full PRESENT

Céline Blondeau[1], Thomas Peyrin[2] and Lei Wang[2,3⋆]

[1] Department of Computer Science, School of Science, Aalto University, Finland
[2] Nanyang Technological University, Singapore
[3] Shanghai Jiao Tong University, China

celine.blondeau@aalto.fi, thomas.peyrin@ntu.edu.sg, wangleihb83@gmail.com

Abstract. In this article, we analyse the known-key security of the standardized PRESENT lightweight block cipher. Namely, we propose a known-key distinguisher on the full PRESENT, both 80- and 128-bit key versions. We first leverage the very latest advances in differential cryptanalysis on PRESENT, which are as strong as the best linear cryptanalysis in terms of number of attacked rounds. Differential properties are much easier to handle for a known-key distinguisher than linear properties, and we use a bias on the number of collisions on some predetermined input/output bits as distinguishing property. In order to reach the full PRESENT, we eventually introduce a new meet-in-the-middle layer to propagate the differential properties as far as possible. Our techniques have been implemented and verified on the small scale variant of PRESENT. While the known-key security model is very generous with the attacker, it makes sense in practice since PRESENT has been proposed as basic building block to design lightweight hash functions, where no secret is manipulated. Our distinguisher can for example apply to the compression function obtained by placing PRESENT in a Davies-Meyer mode. We emphasize that this is the very first attack that can reach the full number of rounds of the PRESENT block cipher.

Key words: PRESENT, known-key model, distinguisher, differential cryptanalysis, linear cryptanalysis.

1 Introduction

The pervasive deployment of tiny computational devices brings with it many interesting, and potentially difficult, security issues. Recently, lightweight cryptography has naturally attracted a lot of attention from the symmetric-key cryptography community and many lightweight block ciphers [8,10,18] and hash functions [2,17,7] have been proposed in the past few years. Among these primitives, PRESENT [8] is probably the one which has been the most scrutinized. It remains unbroken, even though many lightweight block ciphers have been successfully attacked. As such, it has become an ISO/IEC standard [19] and is now expected to be deployed in many industrial applications.

⋆ Corresponding author

It is well known that block ciphers and hash functions are very close cryptographic primitives, as the latter can be built from the former and vice versa. For example, the Davies-Meyer construction or the Miyaguchi-Preneel construction can transform a secure block cipher into a secure compression function (which can in turn be used to build a secure hash function by plugging it into some domain extension algorithm). However, while the security is usually guaranteed with a security proof that considers the internal block cipher as a black-box, it is very important that this internal primitive presents no flaw whatsoever. A classical example is the devastating effect on the compression function security of weak keys for a block cipher [33], which are usually considered as a minor flaw for a block cipher if the set of these weak-keys is small.

Therefore, the security notions to consider for a block cipher will vary depending if this block cipher will be used in a hash function setting or not. In a hash setting, block cipher security models such as the known-key [22] (the attacker can know the key) or the chosen-key model (the attacker can choose the key) make sense since in practice the attacker has full access and control over the internal computations. Moreover, an attack in these models depicts a structural flaw of the cipher, while it should be desired to work with a primitive that does not have any flaw, even in the most generous security model for the attacker. Several known-key or chosen-key attacks on reduced-round `AES-128` were published [22,16,21,14], and Gilbert [15] eventually exhibited a known-key attack on the full 10 rounds with 2^{64} computations.

`PRESENT` is a natural candidate to build lightweight compression functions and hash functions, and such constructions were proposed in [9]. It is therefore meaningful to study `PRESENT` even in security models very generous for the attacker. Thus far, the best secret-key attacks on `PRESENT` [12,5] can reach 26 rounds over the 31 total. Related-key attacks (where the key is secret, but the attacker is allowed to ask queries for some keys related to the original one) are thus far not more powerful, probably due to the impossibility of the attacker to properly control linear/differential propagation in the `PRESENT` key schedule. Regarding known or chosen-key model, the best attack could only reach 18 rounds [23] using rebound attacks and multi-differential cryptanalysis (their distinguisher worked not only for the internal block cipher, but also for the `DM-PRESENT` compression function). Only very recently Lauridsen *et al.* [25] managed to reach 26 rounds of 80-bit key version of `PRESENT` and 27 rounds of 128-bit key version of `PRESENT` by combining rebound attacks with linear cryptanalysis in the known-key model (it works for only a small portion of keys, *e.g.* $2^{71.8}$ out of 2^{128}). It is noticeable that even though the security margin of `PRESENT` is rather small, the best attacks in the classical (secret)-single-key model and in the known or chosen-key models are almost reaching the same number of rounds. This is quite surprising as one would expect many more rounds to be broken when the attacker is given so much power. As analogy, one can remark that the best attacks on `AES-128` can break 7 rounds in the single-key or related-key model, but the full 10-round cipher can be broken when the attacker knows the key [15]. It seems that, in the case of `PRESENT`, leveraging the degrees of freedom obtained

by knowing or choosing the key will not greatly help the attacker to improve linear attacks [25].

Our contribution. In this article, we exhibit the very first known-key attack on the full PRESENT cipher. More precisely, using the framework from [5], we avoid the issues when trying to improve linear attacks with more freedom degrees. We start from some of the best differential distinguishers from [5] and we managed to extend them by several rounds by adding a meet-in-the-middle layer. Overall, storing $2^{35.58}$ bytes, we can distinguish the full 31-round PRESENT in the known-key model in a time corresponding to 2^{56} encryptions. The success of our known-key distinguisher on the full PRESENT is 50.5% and is equal to 100% when considering a version reduced to 27 rounds. More details are provided in Table 3. The distinguishing attacks presented in this paper are independent of the key-size and are valid for both PRESENT-80 and PRESENT-128.

In order to validate our results, we have implemented and verified our distinguisher on a small scale variant of PRESENT proposed in [26]. Our findings indicate that one should avoid using PRESENT as building block to create compression functions and hash functions, as it was proposed in [9]. Actually, our distinguisher can also apply to DM-PRESENT (both 80 and 128-bit versions) and to H-PRESENT. We emphasize that this cryptanalysis is the very first non-random property found for the full PRESENT.

In Section 2 we first describe the PRESENT block cipher and then we introduce our attack model in Section 3. Then, we explain the method to build our distinguisher in Section 4 and finally provide experiments and summarize our results in Section 5. Section 6 concludes this paper.

2 The PRESENT block cipher

2.1 Description of PRESENT

PRESENT [8] is a 64-bit lightweight block cipher proposed at CHES 2007. It is composed of 31 rounds and the 64-bit 32 round-keys are derived from a 80-bit or a 128-bit master key (we will respectively refer to PRESENT-80 or PRESENT-128). The round function is composed of a round-key XOR, an Sbox layer and a simple linear bit permutation layer, as depicted in Figure 1.

The permutation layer operates linearly on the 64 bits as follows: the bit i of the state is moved to the bit position $P(i)$ where

$$P(i) = \begin{cases} 16 \times i \mod (63) & \text{for } 0 \leq i < 63, \\ 63 & \text{for } i = 63. \end{cases}$$

Even though the same Sbox is applied for all nibbles at each round, we numbered the Sboxes from 0 to 15 (see Figure 1) to simplify the description of our attacks. Note that, as in the original PRESENT paper, the least significant bit and the least significant Sbox are on the right. In particular the input of the

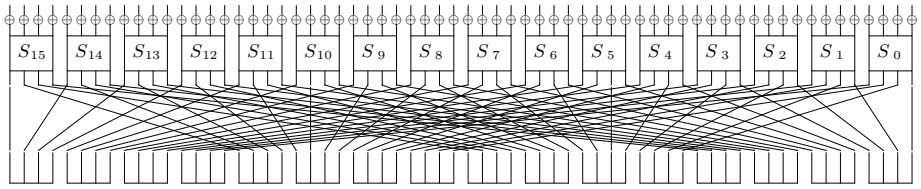

Fig. 1. One round of PRESENT.

Sbox S_i, $0 \leq i \leq 15$, corresponds to the bits $4i, 4i+1, 4i+2, 4i+3$ denoted by $[4i, 4i+3]$. The 4-bit PRESENT Sbox can be described with the following table in hexadecimal display:

$$S[] = \{\texttt{0xC}, \texttt{0x5}, \texttt{0x6}, \texttt{0xB}, \texttt{0x9}, \texttt{0x0}, \texttt{0xA}, \texttt{0xD}, \texttt{0x3}, \texttt{0xE}, \texttt{0xF}, \texttt{0x8}, \texttt{0x4}, \texttt{0x7}, \texttt{0x1}, \texttt{0x2}\}.$$

We do not describe the key schedule of PRESENT as it has no impact on our attack, yet we refer to [8] for a more complete description of the cipher.

2.2 Previous results on PRESENT

In the last couple of years, various analyses [12,31,1,13,3,27,32,4,5] on reduced versions of PRESENT in the (secret)-single-key model have been proposed. Among these analyses, the most important one remains the multidimensional linear attack from Cho [12], which takes advantage of the easy-to-trace linear trails with large correlations, and eventually threatens the security of PRESENT up to 26 rounds. Until 2014, as shown in Table 1, linear cryptanalysis-based attacks were much more powerful against PRESENT, as only 19 rounds were reachable using differential cryptanalysis-based attacks. Some of these key-recovery attacks take advantage of the key-schedule and their time complexity have often been computed for a single version.

However, based on a link between differential probability and linear correlation, it has recently been shown in [5] that one can convert a multidimensional linear distinguisher into a truncated differential one. In Section 4.2 we will provide more details on this technique, which permitted to push truncated differential attacks up to 26 rounds of PRESENT.

Regarding known-key or chosen-key settings, in [23] the authors presented an analysis of DM-PRESENT, *i.e.* the compression function built by placing PRESENT in a Davis-Meyer mode. Based on a combination of differential distinguishers and rebound techniques, they manage to obtain collision and second-preimage attacks on 12 rounds of DM-PRESENT-80, but also a distinguisher up to 18 rounds (this distinguisher can also be applied to the cipher itself). Nevertheless, this approach does not seem to be the most promising one since using classical methods, simple differential-based attacks on PRESENT have been much less powerful than linear-based attacks, as illustrated by Table 1. In the next sections, we will use

Table 1. Relevant attacks on PRESENT in the (secret)-single-key model

#rounds	Version	Attack	Data	Time	Mem.	Ref.	Year
16	80	differential	$2^{64.0}$	$2^{64.0}$	$2^{32.0}$	[31]	2008
19	128	algebraic differential	$2^{62.0}$	$2^{113.0}$	n/r	[1]	2009
19	128	multiple differential	2^{62}	2^{120}	2^{60}	[4]	2013
25	128	linear	$2^{64.0}$	$2^{96.7}$	$2^{40.0}$	[28]	2009
26	80	multidimensional linear	$2^{64.0}$	$2^{72.0}$	$2^{32.0}$	[12]	2010
26	80	truncated differential	$2^{63.16}$	$2^{76.0}$	$2^{29.0}$	[5]	2014

the model provided in [5] to take advantage of much longer truncated differential distinguishers.

Very recently, Lauridsen *et al.* [25] combined linear cryptanalysis and rebound attacks to obtain known-key distinguishers on the PRESENT cipher, for both 80-bit and 128-bit versions. Eventually, they managed to reach 27 rounds, that is one more round than the best (secret)-single-key model attack on PRESENT. Their distinguisher on 27 rounds of PRESENT-128 requires 2^{10} computations and $2^{61.67}$ steps of verification, but works only with probability $2^{-56.2}$ since the distinguisher is considered valid for $2^{71.8}$ keys among the 2^{128} possible. The issue with this method is that it seems not very well fit for known-key or chosen-key scenarios, as only one extra round is reached compared to the best attack in the (secret)-single-key model.

In the next sections, we will describe a meet-in-the-middle approach that fits very well with differential-based attacks, and that will allow us to reach 5 more rounds compared to the best attack in the (secret)-single-key model. Moreover, our distinguishers can apply to DM-PRESENT and H-PRESENT [9] as well.

3 Known-key distinguisher

The known-key model has been introduced by Knudsen and Rijmen [22] to analyse the security of AES-128 and some Feistel-based ciphers. The goal of this model was to get a better estimation of the security margin of a cipher, but also to encompass the scenario of block cipher-based hashing, where the key is known and even chosen by the attacker. The property exhibited for their distinguisher was an integral structure on the input and output of a set of plaintext/ciphertext pairs, for a given known key. Several other types of known-key distinguishers were subsequently proposed, such as the subspace distinguisher [24], the limited-birthday distinguisher [16,20], and more recently a quite complex property related to an integral structure was described by Gilbert [15] to reach the full AES-128.

When one proposes a new known-key distinguisher, it is important to prove or at least give very strong arguments that there is no generic attack that can obtain the same property with an equal or lower complexity than the distinguisher. In other words, an attacker having only blackbox access to encryption

and decryption oracles of the cipher should not be able to obtain the same property with equal or lower complexity than for the distinguisher. In our case, the property we will exhibit is quite trivial: we will observe a bias on the number of collisions on some predetermined input and output bits for a set of many plaintext/ciphertext pairs. More precisely, let the cipher block size to be n bits and let s (respectively q) denote the number of bits from the input (respectively the output) on which we will observe these collisions. We will generate N messages, such that they all have the same value on the s input bits and such that there is a bias on the number of collisions observed on the q output bits.

When having blackbox access to encryption/decryption oracles, an attacker would maximise his success rate by asking only encryption queries. Indeed it is much harder for him to ensure that he will get exactly the required value on the s input bits (which is basically the strongest bias possible) when asking decryption queries, than trying to obtain a weak bias on the q output bits when asking encryption queries. In other words, the best strategy for him is to ensure that the strongest bias is pre-verified when building its queries, and then hoping to observe the weakest bias on the outputs of the oracle. Moreover, to further maximize his success rate, all its encryption queries should have the same value on the s input bits. Indeed, since the encryption oracle is a blackbox to him, all the queries which have different value on the s input bits can be considered completely independent, and therefore will not help him (this is similar to the reasoning given in the limited-birthday problem proof [20]). To summarize, in order to validate our distinguisher, we must compare with the generic attack that consists in simply picking N random inputs (all having the same value on the s predefined bits), querying them to the encryption oracle, and counting the number of collisions obtained on the q predefined bits of the output. In Section 4.2, we will explain the details regarding the computation of the distinguisher's success probability against this type of generic attacker.

Moreover, it is important that this exhibited property can be checked efficiently, and one should count this cost in the overall complexity of the distinguisher. Our distinguishing property can be very easily checked by simply verifying that all the N plaintext/ciphertext pairs have indeed the same value on the s predefined input bits, and by maintaining counters for each possible value taken by the q predefined ciphertext bits. Then, according to these counters, the distinguisher will compute a simple scoring function and decide if he believes to be communicating with `PRESENT` or with a random permutation (see Section 4.2). We note that our known-key distinguishers will work for any key choice. Thus, one can actually have the key value used as challenge for the attacker, which further confirms the validity of our model.

4 Distinguishing full `PRESENT`

4.1 Distinguisher overview

While previous known-key distinguishers on other ciphers benefit much from a start-from-the-middle approach, it cannot effectively be applied to `PRESENT` (at

least in a straightforward way). Typically, those distinguishers are built upon a differential characteristic with desired input/output differences such that extra short differential characteristics with high probability can be pre- and post-added in order to attack as many rounds as possible. The start-from-the-middle approach is then to firstly find solutions of the intermediate differential characteristic. Although such a characteristic usually has a very low probability, thanks to the degrees of freedom obtained from knowing or choosing the key and the internal state values, the distinguisher is able to efficiently compute its solutions. After that, the distinguisher propagates these solutions backwards and forwards to probabilistically satisfy the pre- and post-added extra characteristics. As demonstrated in [23], a rather straightforward application of the start-from-the-middle approach works on very limited number of rounds of `PRESENT`. The difficulty comes from the impossibility to find an intermediate differential characteristic with a large number of rounds, while maintaining an affordable time complexity to find its solutions even leveraging the degree of freedom from knowing or choosing the key and the internal state values. We refer the interested readers to [31,23].

Instead of differential characteristic, our distinguisher on `PRESENT` is built based on the truncated differential of [5]. This is motivated by the fact that truncated differential attack reaches the maximum number of attacked rounds so far as shown in Table 1. Moreover, as far as we know, it is much easier to handle than multidimensional linear attack. Hence, the distinguishing property is a statistical bias of the number of collisions on a few predetermined output bits, where the inputs collide on a few predetermined input bits. We note that such a bias is a very small value and thus cannot be observed with a non-negligible success probability unless a very large set of input/output pairs are provided. Moreover, the necessary number of input/output pairs increases with the number of attacked rounds of truncated differential in order to have a non-negligible success probability. Therefore, there are two issues when adding extra rounds to the truncated differential to extend the number of attacked rounds of the distinguisher. The first one is that we cannot post-add extra rounds, because the predetermined colliding output bits of truncated differential will be input to different Sboxes in the next round and as a result the bias cannot be observed any more from the final outputs. The second one is that if we pre-add a differential characteristic, it sets extra constraints on the inputs of the truncated differential, *i.e.* the inputs must satisfy the extra differential characteristic, which consequently reduces the total number of available inputs to the truncated differential and, a fortiori, lowers the success probability of the distinguisher. On one hand, one surely prefers to use a longer extra characteristic in order to attack more rounds. On the other hand, a longer extra characteristic sets more constraints on the inputs of the truncated differential path. Particularly, if the total number of the available inputs of truncated differential is lower than the necessary number to observe the statistical bias, the overall distinguisher fails.

Thus instead of pre-adding extra differential characteristics, we propose a new layer called meet-in-the-middle (MitM) layer in order to pre-add extra rounds to

the truncated differential. It sets constraints only at its input bits and its output bits, but not at any of its internal state bits. More precisely, the constraints on its input bits is trivially due to defining the distinguishing property.[4] The constraints on its output bits are coming from the truncated differential, *i.e.* the output difference of the MitM layer must satisfy the input constraints of the truncated differential.

Before providing the details of our known-key distinguisher on PRESENT, we give a general overview. As illustrated in Figure 2, the main idea is to take advantage of a strong truncated differential distinguisher ($\Delta \rightarrow \Gamma$) over the $r_1 \leq 24$ last rounds. We denote by p the probability of this distinguisher. From the knowledge of the key, using a MitM approach, we are able to generate a large number of plaintexts which fulfill the following property: for all plaintexts with input difference in the set Λ, their differences after 7 rounds is in the set Δ. The truncated differential distinguisher is described in Section 4.2, the MitM layer in Section 4.3.

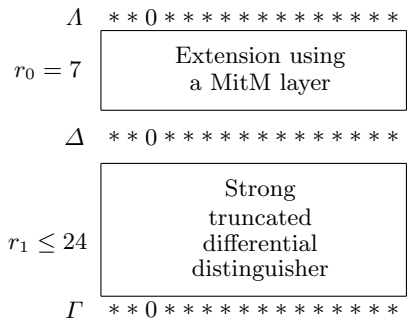

Fig. 2. Distinguisher overview. Each symbol represents 4 bits.

4.2 A statistical bias on reduced-round PRESENT

Given an n-bit permutation F, splitting the input space into $s + t$ bits and the output space into $q + r$ bits, we have the following results which link the probability of a truncated differential with the capacity of a multidimensional linear approximation.

Theorem 1 ([5]). *Let* $\mathbb{F}_2^n = \mathbb{F}_2^s \times \mathbb{F}_2^t = \mathbb{F}_2^q \times \mathbb{F}_2^r$ *and*

$$F : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n,\ x = (x_s, x_t) \mapsto (y_q, y_r).$$

[4] When attacking DM-PRESENT or H-PRESENT, the input bits with constraints of the MitM layer must be located in the same bit positions with the output bits with constraints of the truncated differential, due to the feed-forward operation.

Given a multidimensional approximation $[(a_s,0),(b_q,0)]_{a_s\in\mathbb{F}_2^s,\, b_q\in\mathbb{F}_2^q}$ *with capacity*

$$C=\sum_{(a_s,b_q)\neq(0,0)}\mathbf{cor}^2\left(a_s\cdot x_s\oplus b_q\cdot y_q\right),$$

and a truncated differential composed of 2^t *input differences* $(0,\delta_t)\in\{0\}\times\mathbb{F}_2^t$, *and* 2^r *output differences* $(0,\gamma_r)\in\{0\}\times\mathbb{F}_2^r$ *with probability*

$$p=\frac{1}{2^q}\sum_{\delta_t,\gamma_r\in\mathbb{F}_2^t\times\mathbb{F}_2^r}\mathbf{P}[(0,\delta_t)\rightarrow(0,\gamma_r)],$$

where $\mathbf{P}[(0,\delta_t)\xrightarrow{F}(0,\Delta_r)]=2^{-n}\#\{x\in\mathbb{F}_2^n\,|\,F(x)\oplus F\left(x\oplus(0,\delta_t)\right)=(0,\gamma_r)\}$. *We have*

$$p=2^{-q}(C+1). \tag{1}$$

Truncated differential with strong bias. While the previous attack on DM-PRESENT [23] is derived from a differential with high probability, in this paper, we take advantage of the strong relation between differential probability and capacity to derive a large truncated differential over up to 24 rounds with large bias.

Throughout this paper, we make a distinction on the set of output differences depending if we want to distinguish PRESENT or DM-PRESENT. For our distinguisher on the PRESENT cipher, the sets of differences Λ and Γ do not need to be similar and the last permutation can be omitted which is not the case if we want to distinguish DM-PRESENT or H-PRESENT. Depending of the context, the set of considered output differences will be denoted Γ if it is placed after the last permutation layer and Γ' if it is placed after the last Sbox layer.

We denote by I (resp. J) the number of Sboxes in the first round (resp. last round) with no difference in their input. In the context of the distinguishing attack on PRESENT, we express Theorem 1 as follows.

Corollary 1. *Given a multidimensional linear approximation involving the input bits* $\cup_{i\in\{i_1,\cdots,i_I\}}[4i,4i+3]$ *and the output bits* $\cup_{j\in\{j_1,\cdots,j_J\}}[4j,4j+3]$ *with capacity* C, *we have a truncated differential with input difference*

$$\Delta=\left\{\delta=(\delta_0,\cdots,\delta_{63})\in\mathbb{F}_2^{64}|\delta_b=0\ \textit{for}\ b\in\cup_{i\in\{i_1,\cdots,i_I\}}[4i,4i+3]\right\},$$

and output difference

$$\Gamma=\left\{\gamma=(\gamma_0,\cdots,\gamma_{63}),\in\mathbb{F}_2^{64}|\gamma_b=0\ \textit{for}\ b\in\cup_{j\in\{j_1,\cdots,j_J\}}[4j,4j+3]\right\}.$$

The probability of this differential is given as $p=2^{-4J}(C+1)=2^{-4J}+2^{-4J}C$. *Following the notation of Theorem 1, we have* $s=4I$ *and* $q=4J$. *We call* $2^{-4J}C$ *the bias of this truncated differential approximation.*

While in classical truncated differential attacks only few differentials are involved, for this distinguisher derived from a multidimensional linear distinguisher the number of involved differentials is $2^{128-4I-4J}$ (as PRESENT is a 64-bit cipher).

Part of the analysis consists at selecting a truncated differential with high relative bias. To understand the meaning of high relative bias we first study the success of a distinguishing truncated differential attack.

Success of the distinguishing attack. Given a truncated differential with probability $p = 2^{-q}(C+1)$, we use the following method with data complexity N, time complexity N encryptions and negligible memory complexity to distinguish the cipher from a random permutation.

1. Set a table T of size 2^q to 0
2. For all N messages x with same value on the bits $\cup_{i\in\{i_1,\cdots,i_I\}}[4i, 4i+3]$
   (a) Compute $y = E_K(x)$
   (b) Given y_q the truncation of y reduced to q bits, increment $T[y_q]$
3. Compute $D = \sum_{0\leq\ell\leq 2^q-1} T[\ell](T[\ell]-1)/2$
4. If $D > \tau$, consider that this is the cipher

Without comparing the pairs directly, the scoring function D gives us number of pairs which fulfill the differential [5]. From N messages with same values on the bits $\cup_{i\in\{i_1,\cdots,i_I\}}[4i, 4i+3]$ we can generate $N_S = N^2/2$ pairs of message with no difference on these bits, meaning that for a random permutation the expected number of pairs fulfilling the truncated differential should be $\mu_W = N_S \cdot 2^{-q} = N_S \cdot 2^{-4J}$. We can show [6] that the random variable $\mathcal{D}_R$ corresponding to this scoring function for the given permutation follows a normal distribution with mean $\mu_R = N_S \cdot 2^{-q}(1+C)$ and variance $\sigma_R^2 \approx N_S \cdot 2^{-q}(1+C) \approx N_S \cdot 2^{-q}$. On the other hand we have $\mu_W = N_S \cdot 2^{-q}$ and $\sigma_W^2 \approx N_S \cdot 2^{-q}$. We can show that when using N_S pairs, the success probability P_S of the distinguishing attack is given by,

$$P_S(N_S) = \Phi\left(\frac{\mu_R - \mu_W}{\sigma_R + \sigma_W}\right) \approx \Phi\left(\frac{\sqrt{2^{-q}N_S}\cdot C}{2}\right) \tag{2}$$

where Φ the cumulative distribution function of the central normal distribution. This success probability corresponds to a threshold $\tau = \mu_R - \sigma_R \cdot \Phi^{-1}(P_S) = \mu_W + \sigma_W \cdot \Phi^{-1}(P_S)$.

Strong truncated differential on PRESENT. For a fixed N_S number of pairs, we derive from (2) that the best truncated differentials are the ones which maximize $2^{-q/2}C$. As explained in the previous section the number N_S of available pairs is fixed by the MitM part and the size of Λ, Δ. For the purpose of this attack, we computed the capacity C of different set of linear approximations. From this analysis it turns out that in combination with the MitM phase, if we want to be able to transform this known distinguisher to a distinguisher on DM-PRESENT, the best choice is achieved for $I = 1$ and $J = 1$.

As explained in [12], the capacity of a multidimensional linear approximation can be obtained from the 1-bit linear trails. Given the multidimensional linear input space involving the bits $[4i, 4i+3]$, $0 \leq i \leq 15$ and an output space after the Sbox layer involving the bits $[4j, 4j+3]$, $0 \leq j \leq 15$, we denote by U the set $\{P(4i), P(4i+1), P(4i+2), P(4i+3)\}$ and V the set $\{4j, 4j+1, 4j+2, 4j+3\}$. We can show (see the explanation in [12]) that an estimate of the capacity C'_{r_1} over r_1 rounds without the last linear layer is obtained from the following formula

$$C'_{r_1} = \sum_{u \in U, v \in V} M^{r_1-2}[u, v], \tag{3}$$

where M denotes the 64×64 matrix with coefficients the square correlation of the 1-bit linear approximations over one round in rest of the paper. On the other hand, when the last linear layer is included, since the linear trails activate different Sboxes in the last round, we can estimate the capacity C_{r_1} over r_1 rounds as follows

$$C_{r_1} = \sum_{u \in U, v \in V} M^{r_1-1}[u, v]. \tag{4}$$

From our computation we found that when selecting $\Delta = \{\delta | \delta_b = 0 \text{ for } b \in [52, 55]\}$, the best truncated differentials are obtained for $\Gamma'_i = \{\gamma' | \gamma_b = 0 \text{ for } b \in [4i, 4i+3]\}$ and $i = 5, 7, 13, 15$. For instance such truncated differential distinguisher on 24 rounds has a probability of $2^{-4}(1 + 2^{-58.77}) = 2^{-4} + 2^{-62.77}$ to be fulfilled. By using 2^{56} messages (the reason of this number is due to the MitM layer explained in Section 4.3), we can distinguish 24-round of `PRESENT` from a random permutation in 50.5% of the cases.

In the next section we explain how in the known-key model we can extend this distinguisher to reach more rounds. More explicitly we explain how we can ensure that all the generated messages have a fixed value over the bits $[52, 55]$ after 7 rounds.

4.3 The meet-in-the-middle layer

This section illustrates the meet-in-the-middle (MitM) layer, which is prepended to the truncated differential in order to extend the number of attacked rounds of the known-key distinguisher. It consists of several rounds, and sets constraints on the differences of input/output bits of these rounds. Moreover, the constraints on the output bits of the MitM layer must be exactly the same with those set on the input bits of the truncated differential layer. Then next is to identify a set of plaintexts which can satisfy the constraints on both input and output of the MitM layer. Namely, if these plaintexts are input to `PRESENT`, their internal state after several rounds as the output of the MitM layer can satisfy the input constraints of the truncated differential. Thus these plaintexts can be used to launch a distinguisher on the (reduced) `PRESENT` consisting of both the MitM layer and the truncated differential layer. To efficiently identify such a set of plaintexts, we adopt a meet-in-the-middle approach, which benefits from the

small Sbox and the bit-permutation linear layer of PRESENT. More precisely, for two rounds of computations, an input bit (or four input bits of an Sbox) interacts with only few other bits, and with those bits together can determine partial output bits. Thus we carry out a forward computation to get partial internal state bits for the first two rounds of the MitM layer by guessing just few bits. Similarly we carry out an independent backward computation to get partial internal state bits for the last one and half round of the MitM layer. Finally we carry out a gradually matching process to link and meanwhile fully determine the internal states obtained from the forward and from the backward computations, which can work up to 3 rounds.

We describe in detail the concrete MitM procedure that is used in our attack on full PRESENT. It consists of 7 rounds. The constraints on the inputs are that they share the same values at bits $[52, 55]$, *i.e.* the input bits to S_{13}. The constraints on the outputs are that they share the same values at bits $[52, 55]$, *i.e.* the input bits to S_{13} for next round. In this section, we denote by X_i the internal state after i-th round of PRESENT, and by Y_i the internal state after applying Sbox layer to X_i.

Firstly, we set the bits $[52, 55]$ of plaintext to a randomly chosen 4-bit value, and compute bits 13, 29, 45 and 61 of X_1 in the forward direction. These bits are input to Sboxes S_3, S_7, S_{11} and S_{15} in the second round. Then we exhaustively guess the other 12 bits input to these Sboxes, that include bits 12, 14, 15, 28, 30, 31, 44, 46, 47, 60, 62 and 63 of X_1, and continue to compute in the forward direction to get 16 bits of X_2, *i.e.* bits $4i + 3$ for $0 \leq i \leq 15$. It is also depicted as the first two rounds in Figure 3. In total we get a set of 2^{12} such values of X_2 and each value has 16 bits determined.

Secondly, we set the bits $[52, 55]$ of X_7 to a randomly chosen 4-bit value, and compute bits 19, 23, 27 and 31 of Y_6 in the backward direction. These bits are input to compute the inversion of Sboxes S_4, S_5, S_6 and S_7 in sixth round. We guess the other 12 bits input to the inversion of these Sboxes, that include bits $[16, 18]$, $[20, 22]$, $[24, 26]$ and $[28, 30]$, and continue to compute in the backward direction to get 16 bits of Y_5, *i.e.* bits $4i + 1$ for $0 \leq i \leq 15$. It is also depicted as the last two rounds in Figure 3. In total we get a set of 2^{12} such values of Y_5 and each value has 16 bits determined.

Finally, we carry out a gradually-matching algorithm for each pair of X_2 and Y_5 obtained from the forward and the backward computations respectively. Recall that each of X_2 and Y_5 has 16 bits fixed, which will be named *fixed bits* in the following description. The algorithm is to find a set of internal state values of X_4, whose corresponding values of X_2 and Y_5 can satisfy all the fixed bits, and in turn the corresponding plaintexts can satisfy the constraints on the input and output of the MitM layer. In details, at the third round of the MitM layer, we re-group the bits of X_2 into 4 groups; the i-th group contains bits $[16i, 16i + 15]$ for $0 \leq i \leq 3$. Hence each group contains input bits to 4 consecutive Sboxes, and has 4 bits fixed, *i.e.* bits $16i + 3$, $16i + 7$, $16i + 11$ and $16i + 15$ for the i-th group. Then for each group independently, we exhaustively guess its 12 unfixed bits, and compute in the forward direction to get 16 bits of X_4, that

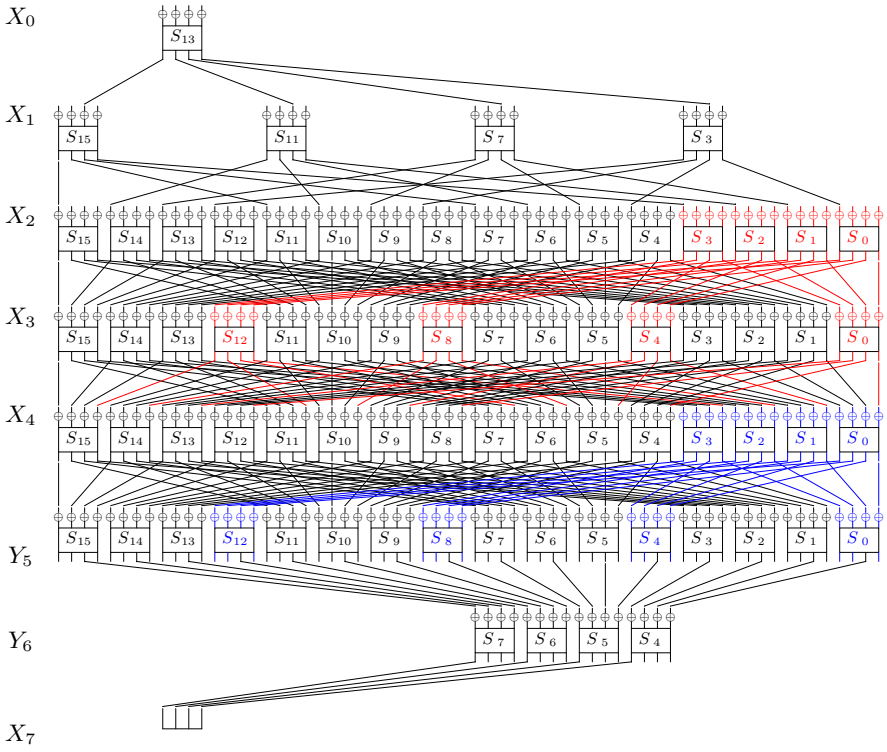

Fig. 3. MitM over the 7 first rounds of PRESENT

is bits $4j + i$, $0 \leq j \leq 15$, for the i-th group. We store the values of partially determined X_4 computed from the i-th group in a table TF_i. See Figure 3 for an example group in red color. Independently and similarly, at the sixth round of the MitM layer, we also re-group the bits of Y_5 to 4 groups; the i-th group contains bits $[4i, 4i+3] \bigcup [4i+16, 4i+19] \bigcup [4i+32, 4i+35] \bigcup [4i+48, 4i+51]$ for $0 \leq i \leq 3$. Then for each group independently, we exhaustively guess the unfixed 12 bits, and compute in the backward direction to get 16 bits of X_4, that is bits $[16i, 16i+15]$, for the i-th group. We store the values of partially determined X_4 computed from the i-th group in a table TB_i. See Figure 3 for an example group in blue color. After that, we merge those tables to find a set of fully-determined values of X_4. To begin with, we merge TF_i and TB_i, and the merged table is denoted as T_i, independently for each $0 \leq i \leq 3$. By merging these two tables, we mean to merge every two partially-determined values of X_4, each from a table and sharing the same bit values at the common determined bit positions, into a new (partially-determined) value of X_4 with all their determined bits, and then to include this new value of X_4 in table T_i. Note that each value of TF_i and each value of TB_i share 4 determined bit positions. Hence table T_i has on average 2^{20} values. Then, we merge T_0 and T_1 and merge T_2 and T_3 independently, and store the results in two tables $T_{0,1}$ and $T_{2,3}$ respectively. As T_0 (respectively T_2)

shares 8 common bits with T_1 (respectively T_3), we get that each of resulted tables has on average 2^{32} values. In the end, we merge $T_{0,1}$ and $T_{2,3}$, which gives on average 2^{32} values of fully-determined X_4 since they share 32 common bits.

Overall, there are 2^{24} pairs of partially-determined X_2 and Y_5 obtained from the forward and the backward computations respectively, and each pair results on average 2^{32} fully-determined values of X_4. Thus in total we can get on average $2^{56(=24+32)}$ plaintexts by inversely computing from the fully-determined values of X_4, and these plaintexts can satisfy the constraints on the input and output of the MitM layer.

It is important to note that by running over all pairs of X_2 and Y_5, we have filtered out all the plaintexts that can satisfy the constraints on both input and output of the MitM layer. In fact it is trivial to evaluate the expected number of such plaintexts. Since there are 4 bit-constraints at bits $[52, 55]$ of plaintext and 4 bit-constraints at bits $[52, 55]$ of X_7, the expected number of desired plaintexts should be $2^{56(=64-4-4)}$. This means that on average (at most) 2^{56} values can be input to the truncated differential, which contributes to 2^{111} pairs, to observe the bias. It has an impact to the success probability of overall distinguisher. More details are given in Section 5.

Complexity. The complexities of both the forward computation and the backward computations are 2^{12} computations of 2 PRESENT-rounds. For the gradually-matching phase, the algorithm is executed 2^{24} times since there are 2^{12} X_2 from the forward computation and 2^{12} Y_5 from the backward computations. The complexity of each execution is obviously dominated by merging $T_{0,1}$ and $T_{2,3}$, which needs 2^{32} table lookups. Hence in total the complexity of the gradually-matching phase is 2^{56} table lookups.

Once a match of the MitM layer has been found, we can encrypt this value X_4 over the $r_1 + 3$ rounds and increment the counter D given in the previous section. Therefore the memory complexity of this attack is dominated by the storage of the table $T_{0,1}$ and $T_{2,3}$ which is $2 \cdot 2^{32} \cdot 6$ bytes. Overall the total time complexity of the distinguisher is 2^{56} table lookups and 2^{56} encryptions.

5 Results

5.1 Experiments

To confirm the validity of the distinguisher presented in this paper, we implemented a similar known-key distinguisher on SMALLPRESENT-[8], a 32-bit scaled-version of PRESENT [26]. A general overview of the cipher as well as a description of the parameters for this known-key distinguishing attack are provided in Figure 4.

For this experimental attack with $I = 2$ and $J = 2$, the expected number of messages obtained from the MitM layer should be $2^{32-4I-4J} = 2^{16}$. We repeated the experiments with different keys by 100000 times, and computed that the average number of generated messages was $2^{16.0009}$. We also computed the

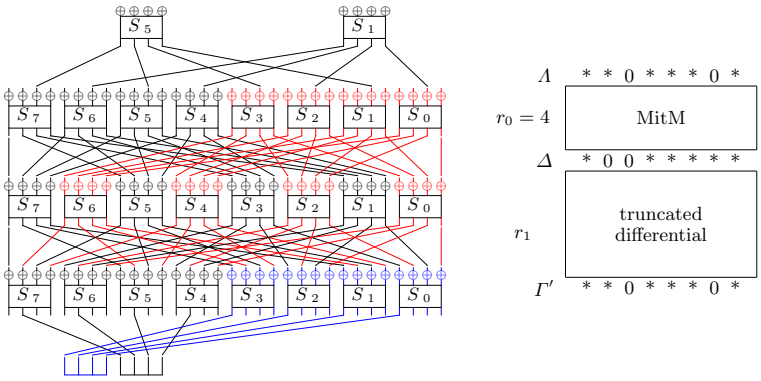

Fig. 4. Left: The MitM part of our experiments on SMALLPRESENT-[8]. Right: Description of the differential involved in our experimental attacks.

standard deviation of these experiments, which was $2^{12.73}$. Based on this deviation, we got that for more than 99.9% of the experiments, the number of messages generated by the MitM phase was greater than 2^{15}. Therefore we take the value $N = 2^{15}$ ($N_S = N^2/2 = 2^{29}$) into consideration to compute a conservative success probability of the attack. A resume of the obtained results for $5 \leq r_1 \leq 8$ and a comparison with the theoretical success probability obtained by formula (2) are given in Table 2.

Table 2. Experimental attacks on SMALLPRESENT-[8]

#Rounds	C over $r-4$ rounds	$P_S(2^{29})$	Exp. P_S
9	$2^{-5.42}$	100%	100%
10	$2^{-8.46}$	98.0%	96.6%
11	$2^{-10.30}$	75.6%	78.9%
12	$2^{-16.17}$	54.3%	54.6%

As expected, these results confirm the validity of the known-key distinguishing model presented in this paper.

5.2 Results

Distinguisher on PRESENT. The results of our known-key distinguishing attack on PRESENT are given in Table 3. The input difference set is $\Lambda = \{\lambda \mid \lambda_b = 0 \text{ for } b \in [52, 55]\}$. For this distinguishing attack we selected the output difference set after the last Sbox layer to be $\Gamma' = \{\gamma' \mid \gamma'_b = 0 \text{ for } b \in [52, 55]\}$. From this set Γ' we derive the following set of output differences after the last linear layer $\{\gamma \mid \gamma_b = 0 \text{ for } b \in \{13, 29, 45, 61\}\}$

As from the MitM phase we can extend the truncated differential distinguisher over 7 rounds, its probability has been computed over $r-7$ rounds. As explained in Section 4.3, from the MitM phase we can generate in average 2^{111} plaintext pairs with difference in the set Λ leading after 7 rounds to 2^{111} pairs with difference in Δ. As from the MitM phase the number of generated pairs is not the always same, we computed a conservative success probability assuming that from the MitM phase only 2^{109} pairs are generated. Note that we have conducted experiments on the 7 first rounds of PRESENT showing that when merging TF_i and TB_i, $0 \leq i \leq 3$, the expected number of matches was, as expected, 2^{20} and the standard deviation was dependent of the group and lower than $2^{14.5}$. These experiments support the fact that assuming that only 2^{109} pairs are generated should gives us an underestimate of the success of the attack. The memory complexity of this distinguishing attack is dominated by

Table 3. Success probability of the known-key distinguisher ($\Lambda \rightarrow \Gamma'$) on PRESENT. The probability of the truncated differential over $r-7$ rounds is obtained from the formula $p = 2^{-4}(C'_{r-7}+1)$ with C'_{r-7} computed from (3).

#Rounds	C'_{r-7}	$P_S(2^{111})$	$P_S(2^{109})$
27	$2^{-48.33}$	100%	100%
28	$2^{-50.94}$	99.8%	93.0%
29	$2^{-53.55}$	68.6%	59.5%
30	$2^{-56.16}$	53.2%	51.5%
31	$2^{-58.77}$	50.5%	50.3%

the storage of the tables in the MitM phase and corresponds to the storage of $2^{35.58}$ bytes. The time complexity of this known-key distinguisher on the full PRESENT is 2^{56} table lookups plus 2^{56} encryptions.

Using multiple truncated differentials. In contrary to the distinguishers on DM-PRESENT and H-PRESENT, for the distinguisher on PRESENT the input and output differences can be selected independently of each other, in particular it is possible to consider different sets of output differences simultaneously. For instance we can simultaneously check that the output pairs have difference in the set Γ'_5 or Γ'_7 or Γ'_{13} or Γ'_{15} (see Section 4.2 for the notation). Meaning that we can simultaneously check that the output pairs have no-difference on the bits $[20,23]$ or $[28,31]$ or $[52,55]$ or $[60,63]$. Given the four multidimensional linear approximations with input masks involving the bits $[52,55]$ and output masks involving the bits of one of the previous set with same capacity C we derive that the probability of the union of these four events is $p \approx 4 \cdot 2^{-4}(1+C)$. The success

of such multiple truncated differential distinguisher is

$$P_S \approx \Phi\left(\frac{N_S \cdot 4 \cdot 2^{-4} C}{2\sqrt{N_S \cdot 4 \cdot 2^{-4}}}\right) = \Phi\left(2^{-2}\sqrt{N_S} C\right).$$

Using this multiple truncated differential distinguisher, having 2^{109} plaintext pairs the success of the attack on 31 rounds is 50.5%. Using different distribution table T_i and different counter values D_i for each set of differentials $1 \leq i \leq 4$, the time complexity of this attack remains the same than that of the simple distinguisher.

Distinguisher on DM-PRESENT and H-PRESENT. For these two compression functions DM-PRESENT and H-PRESENT, the last linear layer has to be considered. In particular, as explained in Section 4.2 the probability of a truncated differential distinguisher with output difference equal to 0 on the bits $[52, 55]$ after the last linear layer can be computed from the capacity of the related multidimensional linear approximation using (4). From the linear properties of the Sbox of PRESENT, we derive the particularity that for all $v \in \{4j\}_{0 \leq j < 16}$ we have $M[u, v] = 0$. Meaning that (4) is equivalent to

$$C_{r_1} = \sum_{u \in U, v \in V} M^{r_1 - 1}[u, v],$$

where $V = \{4j + 1, 4j + 2, 4j + 3\}$. Reducing the output multidimensional linear space from 2^4 values to 2^3 values we can increase the success of our distinguishing attack on PRESENT. In this case we define the set $\tilde{\Gamma} = \{\gamma | \gamma_b = 0 \text{ for } b \in [53, 55]\}$. Our truncated differential distinguisher $(\Delta \rightarrow \tilde{\Gamma})$ has a probability $p = 2^{-3}(C_{r_1} + 1)$.

In this case the value of $X_0[52]$ does not have to be fixed and the MitM presented in Section 4.3 can be repeated for the two values of $X_0[52]$, meaning that in average 2^{57} messages (2^{113} pairs) with fixed $X_0[53, 55]$ and $X_7[52, 55]$ can be generated. The set Λ is now equal to $\{\lambda | \lambda_b = 0 \text{ for } b \in [53, 55]\}$. Meaning that an attack on 31-round DM-PRESENT and H-PRESENT can be performed in time 2^{57} table lookups and 2^{57} encryptions with success probability 50.3%. This distinguishing attack requires the storage of $2^{35.58}$ bytes. The success of this distinguishing attack on reduced-round DM-PRESENT and H-PRESENT is given in Table 4.

The same attack with same success probability could also be performed on PRESENT. However, the time complexity of this new known-key distinguishing attack on PRESENT is twice the one of the previous attack.

6 Conclusion

In this article, we proposed a known-key distinguisher on the full PRESENT block cipher, in both 80- and 128-bit versions. This is the very first non-random property exhibited for the full number of rounds of this standardized cipher. It seems

Table 4. Success probability of the known-key distinguisher ($\Lambda \to \Gamma'$) on DM-PRESENT and H-PRESENT. The probability of the truncated differential over $r-7$ rounds is obtained from the formula $p = 2^{-3}(C_{r-7}+1)$ with C_{r-7} computed from (4). $N_S = 2^{111}$ has been chosen in a conservative way to have a good estimate of the success probability.

#Rounds	C_{r-7}	$P_S(2^{113})$	$P_S(2^{111})$
27	$2^{-50.94}$	100%	100%
28	$2^{-53.55}$	100%	97.3%
29	$2^{-56.16}$	73.7%	62.4%
30	$2^{-58.77}$	54.1%	52.1%
31	$2^{-61.39}$	50.7%	50.3%

an interesting future work to analyse what an attacker would be able to do when not only knowing the key, but when he can actually choose it (chosen-key model). Similarly to the previous strange situation that no attack improvement could be obtained when switching from the secret-key model to the known-key model, it would be surprising that no further improvement could be obtained in the chosen-key model.

Acknowledgements. The authors would like to thank the anonymous referees for their helpful comments. The second and third authors are supported by the Singapore National Research Foundation Fellowship 2012 (NRF-NRFF2012-06). Lei Wang is also supported by Major State Basic Research Development Program (973 Plan) (2013CB338004), National Natural Science Foundation of China (61472250), and Innovation Plan of Science and Technology of Shanghai (14511100300).

References

1. Albrecht, M., Cid, C.: Algebraic Techniques in Differential Cryptanalysis. In: Dunkelman, O. (ed.) FSE. Lecture Notes in Computer Science, vol. 5665, pp. 193–208. Springer (2009)
2. Aumasson, J.P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: A Lightweight Hash. In: Mangard, S., Standaert, F.X. (eds.) CHES. Lecture Notes in Computer Science, vol. 6225, pp. 1–15. Springer (2010)
3. Blondeau, C., Gérard, B.: Multiple Differential Cryptanalysis: Theory and Practice. In: Joux, A. (ed.) FSE. Lecture Notes in Computer Science, vol. 6733, pp. 35–54. Springer (2011)
4. Blondeau, C., Nyberg, K.: New Links between Differential and Linear Cryptanalysis. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT. Lecture Notes in Computer Science, vol. 7881, pp. 388–404. Springer (2013)
5. Blondeau, C., Nyberg, K.: Links between Truncated Differential and Multidimensional Linear Properties of Block Ciphers and Underlying Attack Complexities.

In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT. Lecture Notes in Computer Science, vol. 8441, pp. 165–182. Springer (2014)
6. Blondeau, C., Nyberg, K.: Links Between Truncated Differential and Multidimensional Linear Properties of Block Ciphers and Underlying Attack Complexities. In: Oswald, E., Nguyen, P.Q. (eds.) Eurocrypt 2014. vol. 8441. Springer-Verlag (2014)
7. Bogdanov, A., Knezevic, M., Leander, G., Toz, D., Varici, K., Verbauwhede, I.: SPONGENT: A Lightweight Hash Function. In: Preneel and Takagi [29], pp. 312–325
8. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES. Lecture Notes in Computer Science, vol. 4727, pp. 450–466. Springer (2007)
9. Bogdanov, A., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y.: Hash Functions and RFID Tags: Mind the Gap. In: Oswald, E., Rohatgi, P. (eds.) CHES. Lecture Notes in Computer Science, vol. 5154, pp. 283–299. Springer (2008)
10. Cannière, C.D., Dunkelman, O., Knezevic, M.: KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In: Clavier, C., Gaj, K. (eds.) CHES. Lecture Notes in Computer Science, vol. 5747, pp. 272–288. Springer (2009)
11. Canteaut, A. (ed.): Fast Software Encryption - 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers, Lecture Notes in Computer Science, vol. 7549. Springer (2012)
12. Cho, J.Y.: Linear Cryptanalysis of Reduced-Round PRESENT. In: Pieprzyk, J. (ed.) CT-RSA. Lecture Notes in Computer Science, vol. 5985, pp. 302–317. Springer (2010)
13. Collard, B., Standaert, F.: A Statistical Saturation Attack against the Block Cipher PRESENT. In: Fischlin, M. (ed.) CT-RSA 2009. Lecture Notes in Computer Science, vol. 5473, pp. 195–210. Springer (2009)
14. Fouque, P.A., Jean, J., Peyrin, T.: Structural Evaluation of AES and Chosen-Key Distinguisher of 9-Round AES-128. In: Canetti, R., Garay, J.A. (eds.) CRYPTO (1). Lecture Notes in Computer Science, vol. 8042, pp. 183–203. Springer (2013)
15. Gilbert, H.: A Simplified Representation of AES. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part I. Lecture Notes in Computer Science, vol. 8873, pp. 200–222. Springer (2014)
16. Gilbert, H., Peyrin, T.: Super-Sbox Cryptanalysis: Improved Attacks for AES-Like Permutations. In: Hong, S., Iwata, T. (eds.) FSE 2010. Lecture Notes in Computer Science, vol. 6147, pp. 365–383. Springer (2010)
17. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON Family of Lightweight Hash Functions. In: Rogaway [30], pp. 222–239
18. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.J.B.: The LED Block Cipher. In: Preneel and Takagi [29], pp. 326–341
19. ISO/IEC: Information Technology – Security Techniques –Lightweight cryptography– Part 2: Block ciphers. ISO/IEC 29192-2:2012 (2012)
20. Iwamoto, M., Peyrin, T., Sasaki, Y.: Limited-Birthday Distinguishers for Hash Functions - Collisions beyond the Birthday Bound Can Be Meaningful. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT (2). Lecture Notes in Computer Science, vol. 8270, pp. 504–523. Springer (2013)
21. Jean, J., Naya-Plasencia, M., Peyrin, T.: Multiple Limited-Birthday Distinguishers and Applications. In: Lange, T., Lauter, K.E., Lisonek, P. (eds.) Selected Areas in

Cryptography. Lecture Notes in Computer Science, vol. 8282, pp. 533–550. Springer (2013)
22. Knudsen, L.R., Rijmen, V.: Known-Key Distinguishers for Some Block Ciphers. In: Kurosawa, K. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 4833, pp. 315–324. Springer (2007)
23. Koyama, T., Sasaki, Y., Kunihiro, N.: Multi-differential Cryptanalysis on Reduced DM-PRESENT-80: Collisions and Other Differential Properties. In: Kwon, T., Lee, M.K., Kwon, D. (eds.) ICISC. Lecture Notes in Computer Science, vol. 7839, pp. 352–367. Springer (2012)
24. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schlffer, M.: Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In: Matsui, M. (ed.) ASIACRYPT. Lecture Notes in Computer Science, vol. 5912, pp. 126–143. Springer (2009)
25. Lauridsen, M.M., Rechberger, C.: Linear Distinguishers in the Key-less Setting: Application to PRESENT. In: Leander, G. (ed.) Fast Software Encryption - FSE 2015 - to appear. Lecture Notes in Computer Science, Springer (2015)
26. Leander, G.: Small Scale Variants Of The Block Cipher PRESENT. Cryptology ePrint Archive, Report 2010/143 (2010), https://eprint.iacr.org/2010/143
27. Leander, G.: On Linear Hulls, Statistical Saturation Attacks, PRESENT and a Cryptanalysis of PUFFIN. In: Paterson, K.G. (ed.) EUROCRYPT. Lecture Notes in Computer Science, vol. 6632, pp. 303–322. Springer (2011)
28. Nakahara, J., Sepehrdad, P., Zhang, B., Wang, M.: Linear (Hull) and Algebraic Cryptanalysis of the Block Cipher PRESENT. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) Cryptology and Network Security, CANS 2009, Proceedings. Lecture Notes in Computer Science, vol. 5888, pp. 58–75. Springer (2009)
29. Preneel, B., Takagi, T. (eds.): Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings, Lecture Notes in Computer Science, vol. 6917. Springer (2011)
30. Rogaway, P. (ed.): Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings, Lecture Notes in Computer Science, vol. 6841. Springer (2011)
31. Wang, M.: Differential Cryptanalysis of Reduced-Round PRESENT. In: Vaudenay, S. (ed.) Progress in Cryptology - AFRICACRYPT 2008. Lecture Notes in Computer Science, vol. 5023, pp. 40–49. Springer (2008)
32. Wang, M., Sun, Y., Tischhauser, E., Preneel, B.: A Model for Structure Attacks, with Applications to PRESENT and Serpent. In: Canteaut [11], pp. 49–68
33. Wei, L., Peyrin, T., Sokolowski, P., Ling, S., Pieprzyk, J., Wang, H.: On the (In)Security of IDEA in Various Hashing Modes. In: Canteaut [11], pp. 163–179