Differential Cryptanalysis of 18-Round PRIDE

Virginie Lallemand and Shahram Rasoolzadeh

Horst Görtz Institute for IT Security, Ruhr-Universität Bochum, Germany.
firstname.lastname@rub.de

Abstract. The rapid growth of the Internet of Things together with the increasing popularity of connected objects have created a need for secure, efficient and lightweight ciphers. Among the multitude of candidates, the block cipher PRIDE is, to this day, one of the most efficient solutions for 8-bit micro-controllers. In this paper, we provide new insights and a better understanding of differential attacks of PRIDE. First, we show that two previous attacks are incorrect, and describe (new and old) properties of the cipher that make such attacks intricate. Based on this understanding, we show how to properly mount a differential attack. Our proposal is the first single key differential attack that reaches 18 rounds out of 20. It requires 2^{61} chosen plaintexts and recovers the 128-bit key with a final time complexity of $2^{63.3}$ encryptions, while requiring a memory of about 2^{35} blocks of 64 bits.

Keywords: Block cipher, PRIDE, Differential cryptanalysis

1 Introduction

We are currently facing a growing need for secure and efficient cryptographic primitives that aim to protect the myriad of resource-constrained devices that are more and more part of our daily lives.

Most popular examples of such targeted devices of the Internet of Things include RFID tags and nodes in sensor networks. For the latter, one of the preferred platforms are 8-bit micro-controllers. Ciphers dedicated to this platform require to be lightweight and software-oriented, that is, in addition to being secure will only require a small program memory and have a small execution time. Examples of ciphers proposed to meet these needs include SEA [10], KLEIN [5], ITUbee [7], PRIDE [1] and the Feistel ciphers designed by the National Security Agency (SIMON and SPECK [2]). Among the academic proposals, the substitution permutation network (SPN) PRIDE proposed by Albrecht et al. at Crypto 2014 is the most efficient, result that sources from the designers' careful analysis of linear layers that reach interesting trade-off between security and efficiency.

Previous works on PRIDE include a side-channel attack presented at CRiSIS 2016 [9]. In the black box scenario, Dinur presented at Eurocrypt 2015 [4] a new cryptanalytic time-memory-data trade-off, while Guo et al.[6] gave observations on the impact of increasing the number of rounds of the cipher. The more

powerful attacks published to date are a related-key differential attack of the full cipher [3] by Dai and Chen, and two differential attacks on 18 [15] and 19 [14] rounds out of 20. Quoting from the specification document[1], the related key attack is out of scope: *" PRIDE does not claim any resistance against related-key attacks (and actually can be distinguished trivially in this setting)"*, so the best type of attack appears to be single key differential attack.

In this paper we provide insight on the resistance of PRIDE against this type of attack and give a twofold contribution: first, we show that the two previous attacks ([15] and [14]) are erroneous — even when taking into account the corrections proposed by [12] — due to a miscomputation of the known bits and second we show how to correctly mount a differential cryptanalysis to attack 18 rounds of PRIDE.

Our attack requires 2^{61} chosen plaintexts and the equivalent of $2^{63.3}$ encryptions. Since the security claim of the designers is that the product of data and time complexity cannot be smaller than 2^{127}, our proposal is a valid attack of the cipher reduced to 18 rounds.

The paper is organized as follows. Next section gives a short description of the block cipher PRIDE and introduces our notations. Then, we start our study with a section reporting old and new properties of PRIDE Sbox and key schedule. In Section 4, we describe our first contribution by disclosing why the two previous differential cryptanalyses of PRIDE fail to recover the key, even when the flaws spotted in previous works are corrected. We then put into practice our comprehension of PRIDE to build high probability differential characteristics (Section 5) and mount an 18-round differential attack in Section 6. The paper ends with a conclusion.

2 PRIDE Block Cipher

2.1 Description of PRIDE

PRIDE [1] is a lightweight block cipher proposed at Crypto 2014 by Albrecht, Driessen, Kavun, Leander, Paar and Yalçin. The cipher follows an SPN structure and benefits from an extensive analysis of secure and efficient linear layers, presented in the same article. It is software-oriented and reaches notable performance figures when implemented on 8-bit micro-controllers.

Round Function. PRIDE uses 64-bit blocks and 128-bit keys and makes use of the FX construction [8] in the following way: the first 64 bits of the master key k, denoted k_0, is used as pre- and post-whitening key, while the other half k_1 is used to compute the round keys. In the following, we denote the whitening key by K_0 and the round key of round i by K_i, $1 \leq i \leq 20$ (see Figure 1).

PRIDE encryption routine is made of 20 rounds. The first 19 rounds are identical and denoted by $\mathcal{R}$, while the last one does not contain the linear layer and is denoted by $\mathcal{R}'$. The cipher ends (resp. starts) with the application of a bit-permutation (resp. its inverse) for bit-sliced implementation reasons. Since

[1] Section 5.5 of [1].

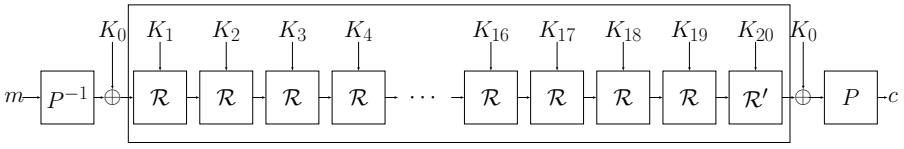

Fig. 1. Overall structure of PRIDE block cipher.

these operations can easily be inverted, what we call in the following *plaintext* and *ciphertext* are the states before (resp. after) the first (resp. last) whitening operation. The cipher is based on the following operations, combined as depicted in Figure 2:

- A key addition layer,
- An Sbox layer, which consists in applying the same 4×4 Sbox S (given in Table 1) to each nibble (group of 4 bits) of the state,
- A linear layer, combining:
  - The application of bit permutations P and P^{-1}, described in Appendix A, Table 7,
  - The application of matrices, more precisely the application of matrix L_i, $i = 0, \cdots, 3$ (given in Appendix A) to the i^{th} 16-bit word of the state.

Table 1. Definition of the Sbox of PRIDE.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	0	4	8	f	1	5	e	9	2	7	a	c	b	d	6	3

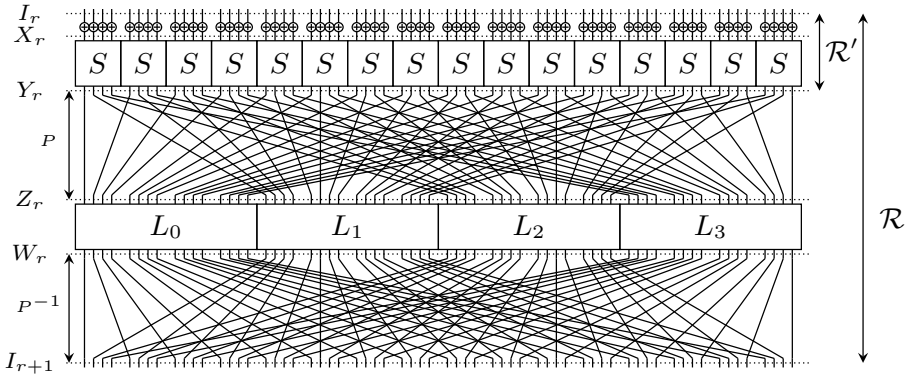

Fig. 2. $\mathcal{R}$ and $\mathcal{R}'$ round functions of PRIDE. The naming conventions used for the intermediate states are detailed in Table 2.

Key-Schedule. The round keys of PRIDE are 64-bit words given by $K_i = P^{-1}(f_i(k_1))$ $(1 \leq i \leq 20)$ where $f_i(k_1)$ is:

$$f_i(k_1) = k_{1_0}||g_i^{(0)}(k_{1_1})||k_{1_2}||g_i^{(1)}(k_{1_3})||k_{1_4}||g_i^{(2)}(k_{1_5})||k_{1_6}||g_i^{(3)}(k_{1_7})$$

k_{1_i}, $0 \leq i < 8$, is byte number i of k_1 and the g_i functions are given by:

$$g_i^{(0)}(x) = (x + 193i) \bmod 256, \quad g_i^{(1)}(x) = (x + 165i) \bmod 256,$$
$$g_i^{(2)}(x) = (x + 81i) \bmod 256, \quad g_i^{(3)}(x) = (x + 197i) \bmod 256.$$

2.2 Notations

To ease comprehension of the remainder of the paper, we use the same notation as in the two previous differential attacks on PRIDE ([15,14]). These notations are recalled in Table 2. In order to remain consistent with it, we also start counting bits from 1, and more particularly we denote by (x_1, x_2, x_3, x_4) the binary decomposition of the nibble x, where x_1 is its most significant bit.

Table 2. Notations.

Symbol	Definition
I_r	input state of r-th round
X_r	state after key addition of r-th round
Y_r	state after the Sbox layer of r-th round
Z_r	state after the application of P of r-th round
W_r	state after the matrices layer of r-th round
ΔS	*xor* difference of the state S
$S_r[i]$	i-th nibble of the state S_r
$S_r^j[i]$	j-th bit of the i-th nibble of S_r

3 Properties of PRIDE Components

In this section, we present important properties of the Sbox and of the Key-Schedule that impact a differential attack of PRIDE. These properties are crucial to understand the mistakes made in the previous differential cryptanalyses as well as to get the techniques used in our new attack.

3.1 Sbox Properties

We start by recalling the component functions of the Sbox:

Definition 1. ***(Component functions of*** PRIDE ***Sbox)*** *If we denote $x = (x_1, x_2, x_3, x_4)$ the input nibble of the Sbox, then the expressions of the corresponding output nibble $S(x) = y = (y_1, y_2, y_3, y_4)$ is given by:*

$$\begin{cases} y_1 = x_1x_2 \oplus x_3 \\ y_2 = x_2x_3 \oplus x_4 \\ y_3 = x_1x_2x_3 \oplus x_1x_2x_4 \oplus x_2x_3 \oplus x_3x_4 \oplus x_1 \\ y_4 = x_1x_2x_4 \oplus x_1x_4 \oplus x_2x_3 \oplus x_3x_4 \oplus x_2 \end{cases}$$

We can remark that y_1 and y_2 depend only on 3 bits out of 4 of the input and that only two of the input bits are involved in the degree 2 monomials. This remark turns useful in our attack since it implies that instead of requiring a complete nibble to get the value of bit number 1 or 2 we only need the value of 3 bits. Note that since PRIDE Sbox is an involution these properties also hold for its inverse.

What's more, this observation impacts the possible differential transitions of the Sbox, a property that was formalized by Tezcan in [11] and applied to PRIDE in [12].

Definition 2. *(undisturbed bit [11]) For a specific input difference of an S-box, if some bits of the output difference remain invariant, then we call such bits undisturbed.*

For instance, if the input difference of PRIDE Sbox is equal to 8 (1000), its output difference is of the form ?0?? (see [12]).

In [13], Tezcan and Özbudak introduced the notion of differential factor, that plays a role in the number of key bits one can recover and on the time complexity:

Definition 3. *(differential factor [13]) Let S be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. For all $x, y \in \mathbb{F}_2^n$ that satisfy $S(x) \oplus S(y) = \mu$, if we also have $S(x \oplus \lambda) \oplus S(y \oplus \lambda) = \mu$, then we say that the S-box has a differential factor λ for the output difference μ. (i.e. μ remains invariant for λ).*

3.2 Key-Schedule Properties

We introduce here a property of the key schedule that allows to reduce the number of key-guesses required in our attack:

Property 1. **(Difference between round keys)** The binary difference between two round keys K_i and K_j for i and j of different parity is given by the following expression, where '?' represents an unknown bit :

$$K_i \oplus K_j = (00000000||00000000||00000000||00000000||????????||????????||????????||????1111)$$

Also, the difference between K_i and K_ℓ for i and ℓ of same parity[2] is given by:

$$K_i \oplus K_\ell = (00000000||00000000||00000000||00000000||????????||????????||????????||????0000).$$

Proof. The first relation results from the definition of the round key, from which we obtain that:

$$K_i \oplus K_j = P^{-1}(00000000||???????1||00000000||???????1||00000000||???????1||00000000||???????1)$$

where the differences of '1' in bit 16, 32, 48 and 64 of $P(K_i \oplus K_{i+1})$ are easily explained by the fact that i and j have different parities and that the values added to k_1 in g functions are odd. The second relation results from the fact that i and ℓ have the same parity.

As described later, we select our characteristic so that when checking the active Sboxes we have common bits so less guesses to make.

[2]Note that simple relations can also be found between other keys; K_i and K_{i+16} for instance.

4 Previous Differential Attacks on PRIDE

Two single key differential attacks ([15],[14]) have been published prior to our work. In [12], Tezcan et al. show that the complexities of these attacks are miss-computed due to the oversight of the impact of differential factor and propose a correction. Their patch mainly results in an increase of the final time complexity.

In this section, we show that there are more problems in [15] and [14] than the ones reported in [12] and that consequently the proposed patches are insufficient. The problem we disclose and that is common to both attacks is that the attacker misses information to compute the required internal state bits.

4.1 18-Round Differential Attack of Zhao et al.

In [15], Zhao et al. proposed an attack on 18-round PRIDE. They use a 15-round characteristic[3] of probability 2^{-58} and add one round to the top and two rounds to the bottom. Their attack procedure starts by eliminating some wrong pairs by looking at the ciphertext difference. Then, they guess 10 nibbles of the whitening key K_0 – namely $K_0[1, 2, 3, 5, 6, 7, 10, 11, 14, 15]$ – in order to be able to check that the differences at the input of the corresponding Sboxes of round 18 have the right form (see Table 3).

They next introduce K'_{18}, a key that is equivalent to the last round key K_{18} and is given by: $(M \circ P)^{-1}(f_{18}(k_1))$. They make a guess on $K'_{18}[6, 10, 14]$ in order to be able to compute the difference entering Sbox number 6, 10 and 14 of penultimate round and access the corresponding sieve.

This attack suffers from several problems: first, as noted in [14] and later in [12], the authors omitted to take into account the undisturbed bits. In addition to that, [12] reveals that the 6 Sbox differences that are involved in the attack are differential factors (namely $\lambda = \mu = 8$), which implies that the attacker cannot obtain information on 6 key bits. Quoting [12], this error results in the fact that "the correct time complexity of this attack is 2^{70} 18-round Pride encryptions, not 2^{66}".

The new problem we spotted is a miscomputation of the known bits of the internal states. Namely: to compute the difference entering Sbox number 6, 10 and 14 of penultimate round we need the value of $Y_{17}[6, 10, 14]$, but it is impossible to determine the two middle bits of any of these 3 nibbles. This phenomenon appears clearly if we look at Table 3, where we have depicted the bits of rounds 17 that can be computed from the ciphertext given the key guesses on K_0. We clearly see that the 3 highlighted nibbles $Y_{17}[6], Y_{17}[10]$ and $Y_{17}[14]$ are not completely determined.

As it is, the sieve offered by these 3 Sboxes cannot be accessed, so each possible value for the 52 bits of key would be suggested 2^7 times in average, and the right value would not be distinguishable. Consequently, the attack fails.

[3] It corresponds to what we name in next section the first characteristic of type (I, a), see Table 5.

Table 3. Analysis of 18-round differential attack of PRIDE by Zhao et al. [15]. All the bit values that are computable are depicted with a '1', while other bits are shown by '0'.

$P = I_1$	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
$K_0 \oplus K_1$	0000 0000 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000
X_1	0000 0000 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000
Y_1	0000 0000 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000
$\cdots$	$\cdots$			
X_{17}	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000
K'_{18}	0000 0000 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000	0000 1111 0000 0000
Y_{17}	0000 1001 1001 0110	0000 1001 1001 0110	0000 1001 1111 0000	0000 1001 1111 0000
Z_{17}	0110 0110 0110 0110	0001 0001 0010 0010	0001 0001 0010 0010	0110 0110 0110 0110
W_{17}	1110 1110 0110 0110	1110 1110 0110 0110	1110 1110 0110 0110	1110 1110 0110 0110
X_{18}	1111 1111 1111 0000	1111 1111 1111 0000	0000 1111 1111 0000	0000 1111 1111 0000
Y_{18}	1111 1111 1111 0000	1111 1111 1111 0000	0000 1111 1111 0000	0000 1111 1111 0000
K_0	1111 1111 1111 0000	1111 1111 1111 0000	0000 1111 1111 0000	0000 1111 1111 0000
C	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111

The attack of Zhao et al. has high requirements (2^{60} messages and 2^{66} encryptions), so taking into account the correction from the differential factors already leads to an attack that does not break the security claim ($2^{60} \times 2^{70} > 2^{127}$). In addition to that, correcting the problem we spotted in a straightforward manner would require to make more guesses on K_0, that is to guess 22 bits of the nibbles $K_0[4, 8, 9, 12, 13, 16]$, so clearly fixing Zhao et al. paper does not lead to an attack that threaten the cipher.

The authors' confusion comes probably from the fact that the linear layer is not an involution. In the case it was, knowing the bits they name would have been enough to access the active Sboxes in round 17. Unfortunately, L_1 and L_2 do not define involutions so more bits are required to find the output of the 3 active Sboxes of round 17.

4.2 19-Round Differential Attack by Yang et al.

There is a similar mistake in the 19-round attack described by Yang et al. in [14].

They use a 15-round characteristic[4] and expand it two rounds to the plaintext side and two rounds to the ciphertext side. After discarding pairs that for sure do not follow the characteristic, they guess nine nibbles of key in plaintext side, namely $(K_0 \oplus K_1)[1, 2, 3, 5, 7, 9, 10, 13, 14]$, and partially encrypt the plaintext pairs through the first Sbox layer. On the ciphertext side, they guess the seven nibbles $1, 2, 5, 8, 9, 10$ and 13 of K_0 and partially decrypt the ciphertext pairs through the last Sbox layer. They next claim that they can also recover nibbles number 5 and 9 of K_2 and nibbles 5 and 9 of $K'_{19} = (M \circ P)^{-1}(f_{19}(k_1))$.

Similarly to the 18-round attack discussed in previous section and as described in [12], this attack uses difference transitions that are differential factors, meaning that it fails to recover 4 bits of the key (each most significant bit of nibbles number 5 and 9 of K_2 and nibbles 5 and 9 of K'_{19}).

[4] The fourth characteristic of type (II, a) given in Table 5.

Table 4. Analysis of 19-round differential attack of PRIDE by Yang et al. [14]. We use the same notation as before and depict known bits with '1'.

$P = I_1$	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
$K_0 \oplus K_1$	1111 1111 1111 0000	1111 0000 1111 0000	1111 1111 0000 0000	1111 1111 0000 0000
X_1	1111 1111 1111 0000	1111 0000 1111 0000	1111 1111 0000 0000	1111 1111 0000 0000
Y_1	1111 1111 1111 0000	1111 0000 1111 0000	1111 1111 0000 0000	1111 1111 0000 0000
Z_1	1110 1010 1100 1100	1110 1010 1100 1100	1110 1010 1100 1100	1110 1010 1100 1100
W_1	1000 1100 1000 1000	0100 0000 0100 0000	0000 0100 0000 0100	1000 1000 1000 1100
I_2	1001 0100 0000 0000	1001 1010 0000 0000	1001 0100 0000 0000	1001 0011 0000 0000
K_2	0000 0000 0000 0000	1111 0000 0000 0000	1111 0000 0000 0000	0000 0000 0000 0000
X_2	0000 0000 0000 0000	1001 0000 0000 0000	1001 0000 0000 0000	0000 0000 0000 0000
Y_2	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000
...		...		
X_{18}	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000	0000 0000 0000 0000
K'_{19}	0000 0000 0000 0000	1111 0000 0000 0000	1111 0000 0000 0000	0000 0000 0000 0000
Y_{18}	1001 0000 0100 0000	1001 0000 0010 0000	1001 0010 0000 0000	1001 0100 0000 0000
Z_{18}	1000 1000 1000 1000	0010 0000 0000 0100	0000 0010 0100 0000	1000 1000 1000 1000
W_{18}	1100 1001 1100 1000	1100 1001 1100 1000	1100 1001 1100 1000	1100 1001 1100 1000
X_{19}	1111 1111 0000 0000	1111 0000 0000 1111	1111 1111 0000 0000	1111 0000 0000 0000
Y_{19}	1111 1111 0000 0000	1111 0000 0000 1111	1111 1111 0000 0000	1111 0000 0000 0000
K_0	1111 1111 0000 0000	1111 0000 0000 1111	1111 1111 0000 0000	1111 0000 0000 0000
C	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111

Moreover, as in previous section, we note that there is a problem upstream to that one. Namely, the authors simply cannot compute the desired Sbox transitions given their guesses: they lack information to compute the two middle bits of $I_2[5, 9]$ and $Y_{18}[5, 9]$. Consequently, they cannot obtain information on these four nibbles, and the right key cannot be identify (even if the correction given by Tezcan et al. is applied).

The detail of which bits are computable in first and last two rounds is provided in Table 4.

The initial attack requires 2^{62} chosen plaintexts and 2^{63} 19-round encryptions. To correct the errors we spotted, it is necessary to significantly increase the number of key guesses. We need the value of 24 bits of nibbles number 3, 4, 7, 11, 12, 15, 16 of K_0 to be able to treat the 2 Sboxes of penultimate round, while we need the value of $(K_0 \oplus K_1)[12, 16]$ and 3 bits of $(K_0 \oplus K_1)[6]$ to access the 2 Sboxes of round 2. Clearly, the straightforward correction does not lead to a correct attack since the time complexity explodes.

5 Differential Characteristics for PRIDE

5.1 1 and 2-Round Iterative Differential Characteristics

As already shown in [15,14], there are 56 high-probability iterative characteristics on 1 and 2 rounds of PRIDE, each activating only 4 Sboxes on 2 rounds whose both input and output differences are equal to 8. Hence, the probability of any of these iterative characteristics is equal to $(2^{-2})^4 = 2^{-8}$. The 56 possible input/output differences are given in Table 5, where they are grouped according to the number of active Sboxes in the first round (line I, II or III) and to the index of the first active Sbox in the input difference (column a, b, c and d).

Note that all type II characteristics are iterative on 1 round while the others are iterative on 2 rounds.

Table 5. Hexadecimal value of all the 1 and 2-round iterative differential characteristics of PRIDE. The characteristic used in our attack is highlighted.

	a	b	c	d
I	8000 0000 0000 0000	0800 0000 0000 0000	0080 0000 0000 0000	0008 0000 0000 0000
	0000 8000 0000 0000	0000 0800 0000 0000	0000 0080 0000 0000	0000 0008 0000 0000
	0000 0000 8000 0000	0000 0000 0800 0000	0000 0000 0080 0000	0000 0000 0008 0000
	0000 0000 0000 8000	0000 0000 0000 0800	0000 0000 0000 0080	0000 0000 0000 0008
II	8000 8000 0000 0000	0800 0800 0000 0000	0080 0080 0000 0000	0008 0008 0000 0000
	8000 0000 8000 0000	0800 0000 0800 0000	0080 0000 0080 0000	0008 0000 0008 0000
	8000 0000 0000 8000	0800 0000 0000 0800	0080 0000 0000 0080	0008 0000 0000 0008
	0000 8000 8000 0000	0000 0800 0800 0000	0000 0080 0080 0000	0000 0008 0008 0000
	0000 8000 0000 8000	0000 0800 0000 0800	0000 0080 0000 0080	0000 0008 0000 0008
	0000 0000 8000 8000	0000 0000 0800 0800	0000 0000 0080 0080	0000 0000 0008 0008
III	0000 8000 8000 8000	0000 0800 0800 0800	0000 0080 0080 0080	0000 0008 0008 0008
	8000 0000 8000 8000	0800 0000 0800 0800	0080 0000 0080 0080	0008 0000 0008 0008
	8000 8000 0000 8000	0800 0800 0000 0800	0080 0080 0000 0080	0008 0008 0000 0008
	8000 8000 8000 0000	0800 0800 0800 0000	0080 0080 0080 0000	0008 0008 0008 0000

5.2 14-Round Differential Characteristics

Repeating any of the iterative characteristics of Table 5 gives a 14-round characteristic of probability 2^{-56}. To find out if there are other 14-round characteristics with similar or better probability, we searched for characteristics with up to 3 active Sboxes in each round. Our program returned 168 (new) 14-round characteristics of probability 2^{-56}. Unfortunately, these characteristics are less advantageous than the iterative ones since when we propagate them with probability 1 in the forward and backward direction they activate more Sboxes.

Assume then that we use a 14-round characteristic (built from one of Table 5) between round 3 and 17 of the cipher. By inverting the linear layer, we compute ΔY_2 from ΔI_3, thus capturing 56 pairs (ΔY_2 , ΔX_{17}) that hold with probability 2^{-56}. In addition to that, a 14-round characteristic defines a limited number of possible differences for ΔI_2 and ΔY_{18} that can be computed from the distribution table of the Sbox (Table 8 in Appendix). Namely, each active Sbox of ΔI_2 and ΔY_{17} can only take 4 values, so we obtain that ΔI_2 and ΔY_{17} can respectively take 4^{n_2} and $4^{n_{17}}$ values, where n_i represents the number of active Sboxes in round i.

6 Differential Cryptanalysis of 18-Round PRIDE

This section describes our 18-round differential cryptanalysis of PRIDE. We start by exposing a differential property of PRIDE Sbox and then show how to use it in an attack to easily find information on key bits. We then detail the complexities of our attack.

6.1 PRIDE Sbox Properties for our Differential Characteristics

As discussed in Section 5.2, the difference transitions made by the Sboxes of round 2 and 17 are either from $\mathsf{8}$ to $\mathsf{2}, \mathsf{3}, \mathsf{8}$ or a or from $\mathsf{2}, \mathsf{3}, \mathsf{8}$ or a to $\mathsf{8}$. For these configurations, the following property holds:

Property 2. **(Relations defined by difference transitions of the Sbox)** If two Sbox inputs differ by $\mathsf{2}$ (respectively $\mathsf{3}, \mathsf{8}$ or a) and lead to an output difference of $\mathsf{8}$ then the following relations hold:

$$\begin{array}{lll} S(x) \oplus S(x \oplus \mathsf{2}) = S(x_1x_2x_3x_4) \oplus S(x_1x_2\bar{x}_3x_4) = \mathsf{8} & \Rightarrow & x_2 = \mathsf{0}, x_4 = \mathsf{0} \\ S(x) \oplus S(x \oplus \mathsf{3}) = S(x_1x_2x_3x_4) \oplus S(x_1x_2\bar{x}_3\bar{x}_4) = \mathsf{8} & \Rightarrow & x_2 = \mathsf{1}, x_3 = x_4 \\ S(x) \oplus S(x \oplus \mathsf{8}) = S(x_1x_2x_3x_4) \oplus S(\bar{x}_1x_2x_3x_4) = \mathsf{8} & \Rightarrow & x_2 = \mathsf{1}, x_3 = \bar{x}_4 \\ S(x) \oplus S(x \oplus \mathsf{a}) = S(x_1x_2x_3x_4) \oplus S(\bar{x}_1x_2\bar{x}_3x_4) = \mathsf{8} & \Rightarrow & x_2 = \mathsf{0}, x_4 = \mathsf{1} \end{array}$$

Proof. The property results from the component functions (Definition 1).

In other words, if we are able to check that the input difference of an active Sbox is $\mathsf{2}, \mathsf{3}, \mathsf{8}$ or a and if we expect its output difference to be equal to $\mathsf{8}$ then we are able to deduce information on the value of the state entering this Sbox. Namely, we obtain the value of x_2 together with either the value of x_4 or a relation between x_4 and x_3.

This observation implies that to check if an Sbox executes the right transition (so to have access to the corresponding filter of probability 2^{-2}) we only require information on (at most) 3 bits (bits 2, 3 and 4). This can be seen as a reinterpretation of the undisturbed bits of [11].

6.2 Overview of the Attack Procedure

In our attack, we use the 14-round characteristic[5] of type II given in row 16 of Table 9, and extend it 2 rounds to the plaintext and 2 rounds to the ciphertext. This extension defines 4 rounds of key recovery that will allow us to recover 10 bits of key on the plaintext side and 10 bits of key on the ciphertext side.

The reasons why we decided to use this particular characteristic are the following. First, among the 224 (168 new and 56 previously found) characteristics we prefer the ones which imply less active Sboxes on the plaintext and ciphertext side, and consequently require to make less guesses and computations when checking that first and last round transitions are correct. This downsizes the set of candidate characteristics to 24 (8 of each type), each activating a total of 14 Sboxes on the plaintext and ciphertext side. As can be seen in Table 9 of Appendix C, type I characteristics activate 9 Sboxes on plaintext side and 5 Sboxes on ciphertext side while for type II we have 7 active Sboxes on each side, and for type III the distribution is of 6 active Sboxes on plaintext side and 8 on ciphertext side.

[5] Note that this characteristic is iterative on 1 round.

Then, among the 24 possible characteristics, we prefer the ones that lead to smaller amount of possible differences[6] when extending the characteristic with probability 1 in plaintext and ciphertext. We also take into account the number of key bits that we need to guess.

When looking for minimizing these parameters, both some of the characteristics of type I and type II seem good. Type I characteristics require 1 more key bit guess, but lead to less possible differences in plaintext and ciphertext. Eventually we prefer characteristics of type II since the memory size required to store a full structure is more reasonable.

This selection is further explained in Appendix C.

Table 6. Differential extension of the 14-round characteristic used in our attack.

$\Delta P = \Delta X_1$	0000 0000 0000 ????	???? ???? 0000 ????	???? 0000 0000 ????	0000 0000 0000 ????
ΔY_1	0000 0000 0000 ?00?	00?0 00?0 0000 ?00?	00?0 0000 0000 ?00?	0000 0000 0000 ?0??
ΔZ_1	000? 000? 000? 000?	0000 0000 0000 0000	0000 ??00 ?000 000?	000? 000? 000? 000?
ΔW_1	0000 000? 0000 000?	0000 0000 0000 0000	0000 000? 0000 000?	0000 000? 0000 000?
$\Delta I_2 = \Delta X_2$	0000 0000 0000 0000	0000 0000 0000 ?0??	0000 0000 0000 0000	0000 0000 0000 ?0??
ΔY_2	0000 0000 0000 0000	0000 0000 0000 1000	0000 0000 0000 0000	0000 0000 0000 1000
...	...			
$\Delta I_{17} = \Delta X_{17}$	0000 0000 0000 0000	0000 0000 0000 1000	0000 0000 0000 0000	0000 0000 0000 1000
ΔY_{17}	0000 0000 0000 0000	0000 0000 0000 ?0??	0000 0000 0000 0000	0000 0000 0000 ?0??
ΔZ_{17}	0000 000? 0000 000?	0000 0000 0000 0000	0000 000? 0000 000?	0000 000? 0000 000?
ΔW_{17}	000? 000? 000? 000?	0000 0000 0000 0000	?0?? 0000 ?00? 000?	000? 000? 000? 000?
$\Delta I_{18} = \Delta X_{18}$	00?0 0000 00?0 ?0??	0000 0000 0000 ?00?	00?0 0000 0000 ?0??	0000 0000 0000 ?0??
$\Delta Y_{18} = \Delta C$	???? 0000 ???? ????	0000 0000 0000 ????	???? 0000 0000 ????	0000 0000 0000 ????

For the selected characteristic, and as explained in Section 5.2, ΔX_2 can take $4^2 = 16$ possible values, implying that there are also 16 possible values for ΔY_1 (these values are given in Appendix D). As shown in Table 6, we have 7 active nibbles for the 16 possible differences of ΔY_1 (nibbles 4, 5, 6, 8, 9, 12 and 16) while the other 9 nibbles are always inactive. We use this property to reduce the necessary amount of data by building data structures: the messages we ask for encryption are organized as sets of $2^{4 \cdot 7} = 2^{28}$ plaintexts that are all equal in the corresponding 9 inactive nibbles and take all possible values in the above mentioned 7 nibbles. So in each structure there is about $2^{8 \cdot 7 - 1} = 2^{55}$ plaintext pairs that differ in the 7 nibbles of interest.

A pair from this structure has the correct difference in ΔY_2 if it reaches one of the 16 targeted values for ΔY_1 and then makes happen the correct transitions of Sbox 8 and 16 of the Sbox layer of round 2. The probability of this event can be computed as follows.

Probability that a pair takes one of the 16 targeted values in ΔY_1. Consider a complete structure, that is the set made by all the messages with the same – fixed – value on our 9 inactive nibbles and taking all possible values on the other 7 nibbles. There are exactly 2^{28} such plaintexts. Since the key addition step defines a permutation, we still have 2^{28} messages differing on 7 nibbles after the addition of the whitening key K_0 and the first round key K_1. The same reasoning

[6] Here we look at real values instead of truncated differences.

applies to the Sbox layer: the messages in our structure take all possible values at the input of 7 Sboxes, so since PRIDE Sbox is a permutation the images of these messages still correspond to 2^{28} messages differing on the same positions. Consequently, when forming pairs of these messages, every possible non null difference on the 7 nibbles appears 2^{27} times, so in one structure exactly 16×2^{27} pairs out of 2^{55} are useful for our attack (i.e. a ratio of 1 out of 2^{24}).

Probability that one of the 16 differences in ΔY_1 leads to the correct ΔY_2. If ΔY_1 is as required, the probability that the second round leads to the desired characteristic is equal to $(2^{-2})^2 = 2^{-4}$, which corresponds to the probability that the two active Sboxes of round 2 output a difference of 8 given an entering difference of $2, 3, 8$ or a.

In sum, the total probability that one of our pairs follows the characteristic is equal to:

$$2^{-24} \cdot 2^{-4} \cdot 2^{-56} = 2^{-84}$$

which corresponds to realizing the correct transitions in round 1 and 2, following the 14-round characteristic[7] and finally propagating with probability 1 in the last 2 rounds.

This indicates that we need to encrypt about $a \cdot 2^{84}$ plaintext pairs in order to obtain a pairs that follow the characteristic (also called *right pairs*). This amount can be obtained with $a \cdot 2^{84} \cdot 2^{-55} = a \cdot 2^{29}$ data structures i.e. with $a \cdot 2^{57}$ chosen plaintexts.

In the forward extension there are $4^2 = 16$ possible values for ΔY_{17}, so there are 16 possible values for ΔX_{18} (see Appendix D). As shown in Table 6, these 16 possible values for ΔX_{16} define at most 7 active nibbles (nibbles 1, 3, 4, 8, 9, 12 and 16) while other 9 nibbles are always inactive. A common technique to filter out wrong pairs consists in discarding pairs that have active Sboxes at any of these 9 positions. Given our harsh restrictions in terms of time complexity, we prefer considering a stronger filter which consists in checking that the difference observed in both plaintext and ciphertext differences are consistent with the 16 possible differences that can take Y_1 and X_{18}.

Once we generated enough messages and filtered them according to plaintext and ciphertext differences, we start making key guesses: we test together a pair with a possible value for the key by making partial encryptions and checking that the necessary conditions are fulfilled. We discard all the candidates that do not follow the characteristic.

We start by considering the ciphertext side and make a guess on the seven nibbles $K_0[1, 3, 4, 8, 9, 12, 16]$. We partially decrypt each of the pairs through the matching seven nibbles of the last Sbox layer, and look at the difference that we obtain: any candidate which difference is not one of the previously computed 16 possible values for ΔX_{18} is discarded.

We follow a similar procedure in plaintext side: we make a guess on the 28 key bits that intervene in the computation of the 7 active Sboxes ($(K_0 \oplus K_1)[4, 5, 6, 8, 9, 12, 16]$) and partially encrypt the corresponding nibbles. If the

[7] Our experiments for up to 7 rounds showed that the probability of the differential matches the one of the characteristic.

obtained difference is one of the 16 precomputed ones, we keep the candidate as possible, otherwise we discard it.

At this point, each pair is associated with $28 + 28 = 56$ bits of key corresponding to $(K_0 \oplus K_1)[4, 5, 6, 8, 9, 12, 16]$ and $K_0[1, 3, 4, 8, 9, 12, 16]$. From these possible values for parts of $(K_0 \oplus K_1)$ and of K_0, we deduce possible values for $K_1[4, 8, 9, 12, 16]$. In addition to that, Property 1 implies that we can deduce nibble 4, 8 and 16 of any round key K_i.

We now have a look at the Sbox layer of round 2. We know the value of the differences entering Sbox 8 and 16, together with the value of $K_2[8, 16]$. To check if the Sboxes execute the right transitions, we lack the value of the two middle bits of nibble $I_2[8]$ and $I_2[16]$. By inverting the linear layer, we can see that these values depend on the values of 3 unknown bits which are bit 2, 42 and 59 of state Y_1 (see Figure 3). The ANF description of the Sbox (see Section 6.1) indicates that the values of these 3 bits depend on 10 bits of the plaintext (which is known), together with 10 key bits: $(K_0 \oplus K_1)^{2,3,4}[1]$, $(K_0 \oplus K_1)^{2,3,4}[11]$ and $(K_0 \oplus K_1)[15]$ respectively. Consequently, the idea would be to make a guess on these 10 key bits, deduce the value of $X_2[8]$ and $X_2[16]$ and check whether the transitions are satisfied or not. The probability that a guess passes this test is 2^{-4}.

We follow a similar procedure to handle the last 2 rounds. Let us recall here that our 14-round characteristic ends at round 16 and that the difference spreads freely in rounds 17 and 18, which are respectively of type $\mathcal{R}$ and $\mathcal{R}'$. Our goal here is to check the transitions of Sbox 8 and 16 of round 17 by using Property 2. To limit the complexity of this step, we only check that the value of x_2 is correct instead of checking both relations, so we are only interested in $Y_{17}^2[8]$ and $Y_{17}^2[16]$ (denoted c_2 and d_2 in Figure 4). By referring to the linear layer, we obtain that their expressions in function of I_{18} are:

$$\begin{cases} Y_{17}^2[8] = I_{18}^2[6] \oplus I_{18}^2[7] \oplus I_{18}^2[14], \\ Y_{17}^2[16] = I_{18}^2[3] \oplus I_{18}^2[11] \oplus I_{18}^2[12]. \end{cases}$$

The value of I_{18} depends on ciphertext bits (state C) together with bits of K_0 and K_{18}. For instance, $Y_{17}^2[8]$ can be rewritten as:

$$\begin{aligned} Y_{17}^2[8] &= (K_{18}^2[6] \oplus X_{18}^2[6]) \oplus (K_{18}^2[7] \oplus X_{18}^2[7]) \oplus (K_{18}^2[14] \oplus X_{18}^2[14]) \\ &= (K_{18}^2[6] \oplus K_{18}^2[7] \oplus K_{18}^2[14]) \oplus (Y_{18}^4[6] \oplus Y_{18}^3[6]Y_{18}^2[6]) \oplus (Y_{18}^4[7] \oplus Y_{18}^3[7]Y_{18}^2[7]) \\ &\quad \oplus (Y_{18}^4[14] \oplus Y_{18}^3[14]Y_{18}^2[14]) \\ &= (K_{18}^2[6] \oplus K_{18}^2[7] \oplus K_{18}^2[14]) \oplus ((C^4[6] \oplus K_0^4[6]) \oplus (C^3[6] \oplus K_0^3[6])(C^2[6] \oplus K_0^2[6])) \\ &\quad \oplus ((C^4[7] \oplus K_0^4[7]) \oplus (C^3[7] \oplus K_0^3[7])(C^2[7] \oplus K_0^2[7])) \oplus ((C^4[14] \oplus K_0^4[14]) \\ &\quad \oplus (C^3[14] \oplus K_0^3[14])(C^2[14] \oplus K_0^2[14])). \end{aligned}$$

Which indicates that we need to make a guess on:

$$K_0^{2,3}[6], K_0^{2,3}[7], K_0^{2,3}[14], K_{18}^2[6] \oplus K_{18}^2[7] \oplus K_{18}^2[14] \oplus K_0^4[6] \oplus K_0^4[7] \oplus K_0^4[14].$$

We follow a similar procedure for $Y_{17}^2[16]$ and conclude that we need to guess another 3 bits, namely:

$$K_0^{2,3}[11], K_0^4[11] \oplus K_{18}^2[3] \oplus K_{18}^2[11] \oplus K_{18}^2[12].$$

To sum up the key guessing process, we started from a set of $a \cdot 2^{84}$ possible pairs, we guessed $28+28+10+10 = 76$ key bits and we had access to a filter of $2^{-36} \cdot 2^{-24} \cdot 2^{-24} \cdot 2^{-4} \cdot 2^{-2} = 2^{-54}$ (which corresponds respectively to filtering on the ciphertext difference, checking last and first round and finally second and seventeenth rounds). The number of candidates remaining in the last step is then equal to $a \cdot 2^{70}$. So, in average, each of the 76-bit key candidate will be counted $a \cdot 2^{70} \cdot 2^{-76} = a \cdot 2^{-6}$ times, while as we expect to have a right pairs, the right key candidate will be counted a times. The *signal to noise ratio* (S/N) will then be equal to 2^6, which ensure that we can distinguish the right key candidate from the wrong ones.

The last step of the attack consists in doing an exhaustive search to find the correct value for the remaining $128 - 76 = 52$ key bits.

6.3 Detailed Description of the Attack and of its Complexities

In this section, we detail the time, data and memory complexities of our attack. We show that a naive implementation of the attack procedure described in Section 6.2 would lead to a time complexity overrun, and show how to deal with this issue.

Data Complexity: As detailed previously, we need about $a \cdot 2^{57}$ chosen plaintexts in order to successfully achieve the attack. We choose $a = 2^4$, which means that the data complexity of our attack is equal to 2^{61}. We recall that the security provided by PRIDE when the attacker has access to 2^d messages is equal to 2^{127-d}. Since our attack requires 2^{61} messages, we are limited to a number of operations lower than 2^{66} encryptions.

Time Complexity: To summarize the process described in Section 6.2, the attack is made of 6 main steps:

1. Encrypt 2^{33} structures and filter wrong pairs by looking at their input and output differences.
2. Make a guess on 28 bits of K_0 and check the transitions of the active Sboxes of last round. Eliminate wrong candidates, (that are associations of a pair with a key value that do not satisfy the transitions).
3. Make a guess on 28 bits of $K_0 \oplus K_1$ and check the transitions of the active Sboxes of first round. Eliminate wrong candidates.
4. Make additional guesses on 10 bits of $K_0 \oplus K_1$ to access the value entering Sbox number 8 and 16 of round 2 and check their transitions.
5. Make additional guesses on 10 bits of K_0 and K_{18} to access the value outputting Sbox number 8 and 16 of round 17 and check their transitions.
6. The key guess that is suggested the most is the correct one. Make a guess on remaining 52 key bits and do trial encryptions to recover the 128-bit master key.

A naive implementation of this process would lead to several problems. First, the attack involves many key bit guesses and uses many pairs, which would make the time complexity exceed our upper bound of 2^{66} PRIDE encryptions as soon as step 3. Second, detecting which key candidate is the most frequent would require to keep track of 2^{76} counters, which is clearly not reasonable.

As described next, we solve those two problems by making small guesses at the time and by studying each possible key guess for all possible pairs instead of studying each pair one after the other with all the possible key candidates.

First 3 Steps of the Attack. As briefly mentioned in Section 6.2, the first step of the attack consists in filtering the 2^{88} pairs of messages by looking at their plaintext and ciphertext differences.

Starting from the known 16 possible differences in ΔY_1 and ΔX_{18}, we refer to the difference distribution table and precompute the possible differences in P and C. A search returns that ΔP can take $170164 = 2^{17.38}$ values while ΔC can take $999448 = 2^{19.93}$ values. This implies that out of the 2^{88} initial pairs of messages only $2^{88} \cdot (2^{17.38} \cdot 2^{-28}) \cdot (2^{19.93} \cdot 2^{-64}) = 2^{33.31}$ pairs will remain.

In practice, we start by filtering pairs according to the truncated difference in the ciphertext. We are left with 2^{52} pairs whose differences on the plaintext and ciphertext sides are only on (at most) 7 nibbles. We then build two tables of 2^{28} bits each: the first table indicates if a difference on 28 bits is possible in the plaintext side (so contains a '1' at the position corresponding to the $2^{17.38}$ possible ΔP), while the second indicates which 28-bit differences are valid on the ciphertext side. Each of the 2^{52} remaining pairs then requires at most two table look up to be filtered.

Then, we store all these $2^{33.31}$ pairs and evaluate them with all possible key values. This change implies that instead of needing 2^{76} counters, we have to save the $2^{33.31}$ pairs (so we require $2^{35.31}$ blocks of 64 bits).

In step 2, we start by guessing 7 nibbles of K_0 ($K_0[1, 3, 4, 8, 9, 12, 16]$). For each possible value we study the $2^{33.31}$ pairs, and cancel the ones that do not fulfill the required conditions. More precisely, the key guess is used to invert last round Sbox layer and compute the difference ΔX_{18}. We only keep pairs that lead to one of the 16 possible differences given in Table 10. Since there are 2^{28} possible values for the 7 key nibbles and that we repeat these operations for each of the $2^{33.31}$ pairs, this step is made $2^{61.31}$ times. To express this complexity in terms of PRIDE encryptions, we can see that it consists in computing two times 7 Sboxes (for a pair), while 1 full encryption with the cipher requires $18 \cdot 16 = 288 = 2^{8.17}$ Sbox computations. Step 2 is then roughly equivalent to $2^{56.95}$ PRIDE encryptions.

Step 3 consists in the same operations as step 2 but in plaintext side. We guess the 7 nibbles $(K_0 \oplus K_1)[4, 5, 6, 8, 9, 12, 16]$ and compute the corresponding 7 Sboxes of round 1 for all the remaining pairs. The pairs that are processed in this step correspond to the pairs that remain after step 2, that is on average $2^{33.31} \cdot (16 \cdot 2^{-19.93}) = 2^{17.38}$ pairs associated to each possible value for the 28 bits of key. In its naive form, the number of PRIDE encryptions made in this

step would then be equal to $2^{28} \cdot 2^{28} \cdot 2^{17.38} \cdot 7 \cdot 2 \cdot 2^{-8.17} = 2^{69.02}$, which exceeds our limit of 2^{66} PRIDE encryptions. To solve this problem, we encrypt one Sbox after the other and immediately check if the conditions are fulfilled. We start with Sbox number 5, for which according to Table 10 the targeted output difference is 2. Given one of the 2^4 possible values for $(K_0 \oplus K_1)[5]$ fixed, we compute $Y_1[5]$ (this requires a total of $2^{28} \cdot 2^4 \cdot 2^{17.38} \cdot 2 = 2^{50.38}$ Sbox operations) and check that the obtained difference is equal to 2. To compute the number of pairs that pass this test we need to take into account the proportion of pairs for which Sbox number 5 is active (equal to $\frac{152004}{170164} = 2^{-0.16}$) together with the probability that an active Sbox of our pre-filtered set leads to a difference of 2 (which is $\frac{1}{6}$). We obtain the following estimate:

$$2^{17.38} \cdot (\frac{152004}{170164} \cdot \frac{1}{6} + \frac{18160}{170164} \cdot 1) = 2^{17.38} \cdot 2^{-1.97} = 2^{15.41}.$$

In the same way, we make a guess on the 4 bits of $(K_0 \oplus K_1)[6]$, which requires $2^{28} \cdot 2^4 \cdot 2^4 \cdot 2^{15.41} \cdot 2 = 2^{52.41}$ Sbox encryptions. We then filter out wrong pairs by checking that active Sboxes give a difference of 2. The number of remaining pairs is then equal to[8] $2^{13.37}$. We then process Sbox number 9, which requires $2^{28} \cdot (2^4)^3 \cdot 2^{13.37} \cdot 2 = 2^{54.37}$ Sbox encryptions and leaves us with an average of $2^{12.33}$ pairs for each partial key guess. Next, we treat Sbox number 4 and 12, taking advantage of the fact that they must have the same output difference. The number of Sbox encryptions is equal to $2^{61.33}$ and $2^{8.73}$ pairs remain in average. We finally handle the last 2 Sboxes together, which requires $2^{65.73}$ Sbox encryptions and discard all but 2^4 pairs in average for each key candidate[9]. To sum up, total time complexity of this step is $2^{50.38} + 2^{52.41} + 2^{54.37} + 2^{61.33} + 2^{65.73} = 2^{65.80}$ which is equivalent to $2^{66.80-8.17} = 2^{57.63}$ PRIDE encryptions.

Step 4 and 5. Next operations (step 4 and 5) consist in checking the transitions of Sbox number 8 and 16 in round 2 and in round 17. As explained before, we look at the value of only 3 out of their 4 input bits in round 2 and only 1 out of 4 output bits in round 17. In the following, we name these bits a_i, b_i, $(2 \leq i \leq 4)$ and c_2, d_2, respectively (see Figure 3 and Figure 4).

We start by explaining how to check the 2 active Sboxes of round 2. We remark here that if a_2 (respectively b_2) defines a condition on its own, the condition that a_4 (resp. b_4) must fulfill sometimes depends on a_3 (resp. b_3).

As briefly mentioned in Section 6.2 and as illustrated in Figure 3, a_2 and b_2 are given by the following two expressions:

$$a_2 = K_2^2[8] \oplus Y_1^2[1] \oplus Y_1^2[8] \oplus Y_1^2[11]$$
$$b_2 = K_2^2[16] \oplus Y_1^2[8] \oplus Y_1^2[11] \oplus Y_1^2[12]$$

[8] The computation of the quantities used in this step are detailed in Appendix E.
[9] $2^{17.38} \cdot (16 \cdot 2^{-17.38}) = 2^4$

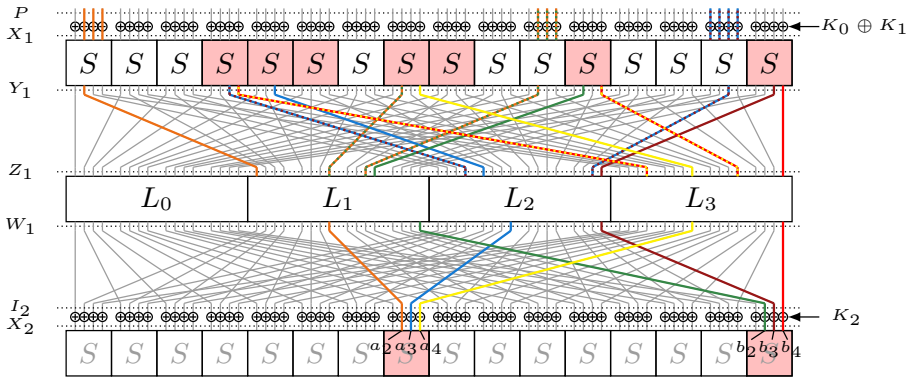

Fig. 3. Bits involved in the computation of a_2, a_3, a_4 and b_2, b_3, b_4.

that when referring to Definition 1 can be rewritten as:

$$\begin{aligned} a_2 = {} & K_2^2[8] \oplus P^4[1] \oplus (K_0 \oplus K_1)^4[1] \\ & \oplus (P^3[1] \oplus (K_0 \oplus K_1)^3[1]) \cdot (P^2[1] \oplus (K_0 \oplus K_1)^2[1]) \\ & \oplus Y_1^2[8] \oplus P^4[11] \oplus (K_0 \oplus K_1)^4[11] \\ & \oplus (P^3[11] \oplus (K_0 \oplus K_1)^3[11]) \cdot (P^2[11] \oplus (K_0 \oplus K_1)^2[11]) \qquad (1) \\ b_2 = {} & K_2^2[16] \oplus Y_1^2[8] \oplus P^4[11] \oplus (K_0 \oplus K_1)^4[11] \\ & \oplus (P^3[11] \oplus (K_0 \oplus K_1)^3[11]) \cdot (P^2[11] \oplus (K_0 \oplus K_1)^2[11]) \\ & \oplus Y_1^2[12], \qquad (2) \end{aligned}$$

for which the only unknown bits are $(K_0 \oplus K_1)^{\{2,3,4\}}[1]$ and $(K_0 \oplus K_1)^{\{2,3,4\}}[11]$. Indeed, the plaintext bits are known and $K_2^2[8]$ and $K_2^2[16]$ are deduced from key-schedule properties, while a_2 and b_2 are determined by the difference observed in X_2 together with the relations given by Property 2.

Consequently, we make a guess on these 6 key bits and check that the relations given by Equation (1) and Equation (2) hold, which happens with probability 2^{-2}.

Since we have an average of 2^4 candidates for each possibility for the 56 bits of key guessed so far, this step is repeated $2^4 \times 2^{56} \times 2^6 = 2^{66}$ times. We expect that $2^4 \times 2^{56} \times 2^6 \times 2^{-2} = 2^{64}$ candidates remain after it. Since computing a_2 and b_2 requires less operations than for an Sbox encryption, the time complexity of this step is less than $2 \cdot 2^{66-8.17} = 2^{58.83}$ full cipher encryptions.

We then look at a_3, a_4, b_3 and b_4. As can be seen in Figure 3, the bits that are necessary to compute a_4 and b_4 are $Y_1^4[4]$, $Y_1^4[8]$, $Y_1^4[12]$, $Y_1^4[16]$, $K_2^4[8]$ and $K_2^4[16]$. Since all these bits are known from previous computations, we can obtain a_4 and b_4 and deduce from the value of ΔY_1 and Property 2 the conditions that they must fulfill on their own or with respect to a_3 and b_3.

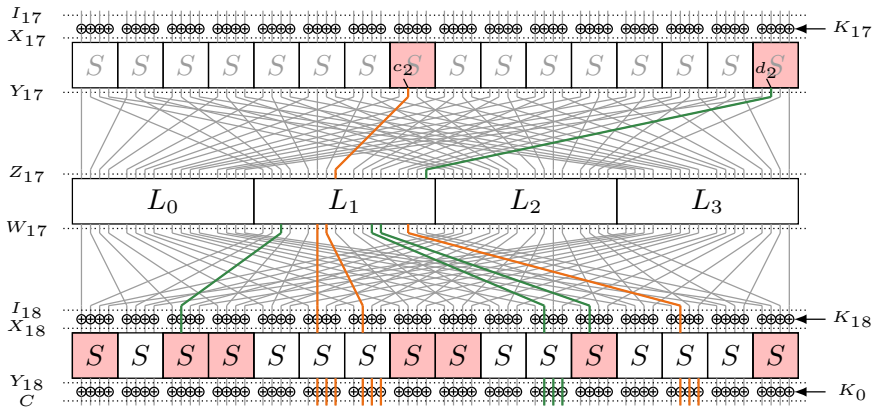

Fig. 4. Bits involved in the computation of c_2 and d_2.

To simplify the explanation, we consider that a_3 and b_3 are always necessary to check the Sboxes. Note that this simplification is at the disadvantage of the attacker and results in an over estimation of the time complexity.

Bits a_3 and b_3 are given by the following expressions (see also Figure 3):

$$a_3 = K_2^3[8] \oplus Y_1^3[4] \oplus Y_1^3[5] \oplus Y_1^3[15]$$
$$b_3 = K_2^3[16] \oplus Y_1^3[4] \oplus Y_1^3[15] \oplus Y_1^3[16]$$

in which the only unknown bit is $Y_1^3[15]$. Since this term appears linearly in both a_3 and b_3, we can obtain a relation relying only on known bits by xoring the two expressions:

$$a_3 \oplus b_3 = K_2^3[8] \oplus K_2^3[16] \oplus Y_1^3[5] \oplus Y_1^3[16].$$

Therefore without any key guessing we can filter our candidates and reduce their number by a factor of 2^{-1}: as a result, 2^{63} candidates remain at this point, while the complexity of this step is lower than $2^{64-8.17} = 2^{55.83}$ encryptions.

For the remaining candidates, we guess $(K_0 \oplus K_1)[15]$ to be able to compute $Y_1^3[15]$ and we check that a_3 takes the right value. This requires a guess of 4 bits, and leads to a reduction of the set of possible candidates by a factor of 2^{-1}. With 2^{67} simple computations (each roughly equal to one Sbox computation, so with a time complexity that is less than $2^{58.83}$ PRIDE encryptions), we reduce the number of candidates to 2^{66}.

At this point, we have 2^{66} candidates made of a pair of plaintext/ciphertext associated to a guessed value for 66 key bits. The average count for a wrong key is expected to be 1, while we built our messages so that the right key appears around $a = 2^4$ times.

The distribution of keys in the candidates follows a binomial distribution of parameters $n = 2^{66}$ and $p = 2^{-66}$ ($\mathrm{B}(2^{66}, 2^{-66})$) that can be approximated by a Poisson distribution of parameter $\lambda = np = 1$ so the probability that a wrong

key appears strictly more than t times in our set of candidates is given by:

$$P_t = 1 - \sum_{k=0}^{t} e^{-1} \cdot \frac{1^k}{k!} = 1 - e^{-1} \cdot \sum_{k=0}^{t} \frac{1}{k!}$$

The idea here is to do an additional filtering step (that is checking the 2 active Sboxes of round 17) only for candidates that are associated with a key that is suggested $t+1$ times or more. Doing so, the ratio of candidates that we have to study is equal to P_t.

We choose $t = 13$, meaning that we are now looking at $2^{66} \cdot 2^{-37.7} = 2^{28.3}$ candidates. For these, we start by computing the value of c_2 and d_2, that as can be seen in Figure 4 depend on the following unknown bits:

- For c_2: $K_0^{\{2,3\}}[6]$, $K_0^{\{2,3\}}[7]$, $K_0^{\{2,3\}}[14]$ and $K_{1s}^2[6] \oplus K_{1s}^2[7] \oplus K_{1s}^2[14] \oplus K_0^4[6] \oplus K_0^4[7] \oplus K_0^4[14]$.
- For d_2: $K_0^{\{2,3\}}[11]$ and $K_0^4[11] \oplus K_{1s}^2[3] \oplus K_{1s}^2[11] \oplus K_{1s}^2[12]$.

We start by guessing the 3 key bits required to compute d_2, and we filter our guesses by confronting the obtained value with the value given by Property 2. The filtering ratio is of 2^{-1}, so the number of candidates after this step is: $2^{28.3} \times 2^3 \times 2^{-1} = 2^{30.3}$.

Next, we repeat the same process by guessing the 7 unknown key bits that are necessary to compute c_2. The number of candidates obtained at this point is: $2^{30.3} \times 2^7 \times 2^{-1} = 2^{36.3}$, and the time complexity of these two steps is negligible in comparison to previous ones.

For all the key candidates that are (still) suggested 14 times or more, we do an exhaustive search to find the value of the $128 - 76 = 52$ unknown key bits and check them by doing a trial encryption. Since we expect $2^{11.3}$ such key candidates, this step will at most require $2^{11.3} \cdot 2^{52} = 2^{63.3}$ encryptions.

To sum up, the total data complexity of our attack is 2^{61} chosen plaintexts, its time complexity is less than $2^{63.3}$ 18-round PRIDE encryption and its memory complexity is of 2^{35} 64-bit blocks.

7 Conclusion

In this paper, we studied the resistance of PRIDE against differential cryptanalysis. We first proved that two previous differential attacks are wrong since essential bits are unknown to the attacker, making her unable to succeed. Our main contribution is a 18-round differential cryptanalysis of the cipher that results from a careful analysis of its high probability characteristics and of its diffusion layer. Our attack recovers the full 128-bit master key with 2^{61} chosen plaintexts, a time complexity equivalent to $2^{63.3}$ encryptions and requires to store around 2^{35} 64-bit blocks.

References

1. Albrecht, M.R., Driessen, B., Kavun, E.B., Leander, G., Paar, C., Yalçin, T.: Block Ciphers - Focus on the Linear Layer (feat. PRIDE). In: Garay, J.A., Gennaro, R. (eds.) Advances in Cryptology - CRYPTO 2014, Proceedings, Part I. LNCS, vol. 8616, pp. 57–76. Springer (2014)
2. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK lightweight block ciphers. In: Proceedings of the 52nd Annual Design Automation Conference, 2015. pp. 175:1–175:6. ACM (2015)
3. Dai, Y., Chen, S.: Cryptanalysis of full PRIDE block cipher. SCIENCE CHINA Information Sciences 60(5), 052108:1–052108:12 (2017)
4. Dinur, I.: Cryptanalytic Time-Memory-Data Tradeoffs for FX-Constructions with Applications to PRINCE and PRIDE. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology - EUROCRYPT 2015, Proceedings, Part I. LNCS, vol. 9056, pp. 231–253. Springer (2015)
5. Gong, Z., Nikova, S., Law, Y.W.: KLEIN: A New Family of Lightweight Block Ciphers. In: Juels, A., Paar, C. (eds.) RFID. Security and Privacy, RFIDSec 2011, Revised Selected Papers. LNCS, vol. 7055, pp. 1–18. Springer (2011)
6. Guo, J., Jean, J., Mouha, N., Nikolic, I.: More Rounds, Less Security? IACR Cryptology ePrint Archive 2015, 484 (2015)
7. Karakoç, F., Demirci, H., Harmanci, A.E.: ITUbee: A Software Oriented Lightweight Block Cipher. In: Avoine, G., Kara, O. (eds.) Lightweight Cryptography for Security and Privacy, LightSec 2013, Revised Selected Papers. LNCS, vol. 8162, pp. 16–27. Springer (2013)
8. Kilian, J., Rogaway, P.: How to Protect DES Against Exhaustive Key Search (an Analysis of DESX). J. Cryptology 14(1), 17–35 (2001)
9. Lac, B., Beunardeau, M., Canteaut, A., Fournier, J.J., Sirdey, R.: A First DFA on PRIDE: from Theory to Practice (extended version). IACR Cryptology ePrint Archive 2017, 075 (2017)
10. Standaert, F., Piret, G., Gershenfeld, N., Quisquater, J.: SEA: A Scalable Encryption Algorithm for Small Embedded Applications. In: Domingo-Ferrer, J., Posegga, J., Schreckling, D. (eds.) Smart Card Research and Advanced Applications, CARDIS 2006, Proceedings. LNCS, vol. 3928, pp. 222–236. Springer (2006)
11. Tezcan, C.: Improbable differential attacks on Present using undisturbed bits. J. Computational Applied Mathematics 259, 503–511 (2014)
12. Tezcan, C., Okan, G.O., Senol, A., Dogan, E., Yücebas, F., Baykal, N.: Differential Attacks on Lightweight Block Ciphers PRESENT, PRIDE, and RECTANGLE Revisited. In: Bogdanov, A. (ed.) Lightweight Cryptography for Security and Privacy, LightSec 2016, Revised Selected Papers. LNCS, vol. 10098, pp. 18–32. Springer (2016)
13. Tezcan, C., Özbudak, F.: Differential Factors: Improved Attacks on SERPENT. In: Eisenbarth, T., Öztürk, E. (eds.) Lightweight Cryptography for Security and Privacy, LightSec 2014, Revised Selected Papers. LNCS, vol. 8898, pp. 69–84. Springer (2014)
14. Yang, Q., Hu, L., Sun, S., Qiao, K., Song, L., Shan, J., Ma, X.: Improved Differential Analysis of Block Cipher PRIDE. In: Lopez, J., Wu, Y. (eds.) Information Security Practice and Experience - ISPEC 2015, Proceedings. LNCS, vol. 9065, pp. 209–219. Springer (2015)
15. Zhao, J., Wang, X., Wang, M., Dong, X.: Differential Analysis on Block Cipher PRIDE. IACR Cryptology ePrint Archive 2014, 525 (2014)

A Specifications of L_0, L_1, L_2, L_3 and P, P^{-1}

$$L_0 = L_0^{-1} = \begin{pmatrix}
0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1\\
1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1\\
1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0
\end{pmatrix},
\quad
L_1 = \begin{pmatrix}
1&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&1&1&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&1&1&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&1\\
0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&1&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&1&1&0&0&0&0
\end{pmatrix},$$

$$L_2 = \begin{pmatrix}
0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&0&1&1&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&0&0&0&0\\
1&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&1&1&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&1&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&1&1&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&1&1&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&1
\end{pmatrix},
\quad
L_3 = L_3^{-1} = \begin{pmatrix}
1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1\\
1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1
\end{pmatrix},$$

$$L_1^{-1} = \begin{pmatrix}
0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0\\
1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1\\
1&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&1&1&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&1&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&1&1&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&1\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&1\\
1&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&1&1&0&0&0&0
\end{pmatrix},
\quad
L_2^{-1} = \begin{pmatrix}
0&0&1&1&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0\\
1&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1\\
1&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&1&1&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&1&1&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&1\\
1&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&1&1&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&1&1&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&1&1&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&1&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&1
\end{pmatrix}.$$

Table 7. P and P^{-1} bit-permutation.

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
$P(x)$	1	17	33	49	2	18	34	50	3	19	35	51	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55	8	24	40	56
x	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
$P(x)$	9	25	41	57	10	26	42	58	11	27	43	59	12	28	44	60	13	29	45	61	14	30	46	62	15	31	47	63	16	32	48	64

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
$P^{-1}(x)$	1	5	9	13	17	21	25	29	33	37	41	45	49	53	57	61	2	6	10	14	18	22	26	30	34	38	42	46	50	54	58	62
x	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
$P^{-1}(x)$	3	7	11	15	19	23	27	31	35	39	43	47	51	55	59	63	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64

B Difference Distribution Table of PRIDE Sbox

Table 8. Difference distribution table of PRIDE Sbox.

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	4	4	4	4	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	4	0	0	4	2	2	2	2
3	0	0	0	0	0	0	0	0	4	0	0	4	2	2	2	2
4	0	4	0	0	0	0	4	0	0	2	2	0	2	0	0	2
5	0	4	0	0	0	4	0	0	0	2	2	0	2	0	0	2
6	0	4	0	0	4	0	0	0	0	2	2	0	0	2	2	0
7	0	4	0	0	0	0	0	4	0	2	2	0	0	2	2	0
8	0	0	4	4	0	0	0	0	4	0	4	0	0	0	0	0
9	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
a	0	0	0	0	2	2	2	2	4	0	4	0	0	0	0	0
b	0	0	4	4	0	0	0	0	0	0	0	0	2	2	2	2
c	0	0	2	2	2	2	0	0	0	2	0	2	2	0	2	0
d	0	0	2	2	0	0	2	2	0	2	0	2	0	2	0	2
e	0	0	2	2	0	0	2	2	0	2	0	2	2	0	2	0
f	0	0	2	2	2	2	0	0	0	2	0	2	0	2	0	2

C Best 14-Round Differential Paths

As explained in Section 5.2, there are many differential characteristics reaching the minimum probability of 2^{-56}. To find out which one is the best for our attack, we first make the following observations:

- Probability of the 14-round differential path: 2^{-p}
- Number of active Sboxes in round i: n_i
- Probability that a pair of messages of the same structure leads to the correct difference: 2^{-4n_1}
- Number of pairs needed to get around a right pairs: $a \cdot 2^{p+4n_1}$
- Number of encryptions needed to obtain a full structure: 2^{4n_1}
- Number of pairs that can be made with one structure: 2^{8n_1-1}
- Number of structures required to get enough pairs: $a \cdot 2^{p-4n_1+1}$
- Number of encryptions required to get enough pairs: $a \cdot 2^{p+1}$

From this, we can see that the number of encryptions needed to conduct the attack only depends on p, so all the 14-round characteristics with maximal probability (that is with $p = 56$) lead to the same data complexity.

Among these, we want to select the characteristic that leads to the minimal time complexity. The parameters that impact it are first the number of active Sboxes on plaintext and ciphertext side (n_P and n_C) but also the number of possible differences in plaintext and ciphertext ($N_{\Delta P}$ and $N_{\Delta C}$), and the number of key bits that we need to guess ($|K_P|$ and $|K_c|$). Table 9 gives a comparison of the best probability characteristics of minimal $n_P + n_C$.

Table 9. Comparison of 14-round characteristics with minimal $n_P + n_C$.

Type	$\Delta I_t = \Delta I_{t+14}$	n_P	n_C	$N_{\Delta P}$	$N_{\Delta C}$	$\lvert K_P \rvert$	$\lvert K_C \rvert$
I	8000 0000 0000 0000	9	5	$2^{24.86}$	$2^{10.97}$	53	27
	0800 0000 0000 0000					53	
	0080 0000 0000 0000					53	
	0008 0000 0000 0000					50	
	0000 8000 0000 0000					53	
	0000 0800 0000 0000					53	
	0000 0080 0000 0000					53	
	0000 0008 0000 0000					50	
II	8000 0000 8000 0000	7	7	$2^{17.38}$	$2^{19.93}$	40	38
	0800 0000 0800 0000					40	
	0080 0000 0080 0000					40	
	0008 0000 0008 0000					40	
	0000 8000 0000 8000					40	
	0000 0800 0000 0800					40	
	0000 0080 0000 0080					40	
	0000 0008 0000 0008					38	
III	8000 8000 0000 8000	6	8	$2^{14.10}$	$2^{24.33}$	35	45
	0800 0800 0000 0800					35	
	0080 0080 0000 0080					35	
	0008 0008 0000 0008					35	
	8000 8000 8000 0000					35	
	0800 0800 0800 0000					35	
	0080 0080 0080 0000					35	
	0008 0008 0008 0000					33	

D Possible Values for ΔY_1 and ΔX_{18}

Table 10. 16 possible hexadecimal values for ΔY_1 and ΔX_{18}.

$\Delta X_2[8,16]$	ΔY_1	$\Delta Y_{17}[8,16]$	ΔX_{18}
22	0000 2000 0000 0002	22	2022 0000 2002 0002
23	0001 2000 0001 0003	23	2023 0000 2003 0003
28	0008 2208 2008 0000	28	002a 0008 000a 0000
2a	0008 2008 0008 0002	2a	202a 0008 200a 0002
32	0001 2001 0001 0002	32	2023 0001 2003 0002
33	0000 2001 0000 0003	33	2022 0001 2002 0003
38	0009 2209 2009 0000	38	002b 0009 000b 0000
3a	0009 2009 0009 0002	3a	202b 0009 200b 0002
82	0008 0200 2008 000a	82	2008 0000 2008 000a
83	0009 0200 2009 000b	83	2009 0000 2009 000b
88	0000 0008 0000 0008	88	0000 0008 0000 0008
8a	0000 0208 2000 000a	8a	2000 0008 2000 000a
a2	0008 2000 0008 000a	a2	202a 0000 200a 000a
a3	0009 2000 0009 000b	a3	202b 0000 200b 000b
a8	0000 2208 2000 0008	a8	0022 0008 0002 0008
aa	0000 2008 0000 000a	aa	2022 0008 2002 000a

E Details of the Filtration Ratio Used in the Attack (Step 3)

We make the (sensible) assumption that the distribution observed in our set is close to the one of a complete ΔP, in particular that filtering first according to the output difference does not impact on the input difference distribution. The successive filtering ratio are then similar to the ones we would observe on a complete ΔP set.

Table 11. Successive filtering steps and expected number of remaining candidates for a complete ΔP set.

$\Delta X_2[8,16]$	$\lvert\Delta Y_1\rvert$	SB5	SB6	SB9	SB4 + SB12
(2, 2)	36	6	6	6	6
(2, 3)	576	96	96	96	6
(2, 8)	13 824	2 304	384	64	4
(2, a)	2 304	384	384	384	24
(3, 2)	2 304	384	384	384	24
(3, 3)	144	24	24	24	24
(3, 8)	110 592	18 432	3 072	512	32
(3, a)	16 128	2 688	2 688	2 688	168
(8, 2)	3 456	3 456	576	96	6
(8, 3)	13 824	13 824	2304	384	24
(8, 8)	16	16	16	16	16
(8, a)	864	864	144	24	24
(a, 2)	576	96	96	96	6
(a, 3)	1 920	320	320	320	20
(a, 8)	3 456	576	96	16	16
(a, a)	144	24	24	24	24
total	170 164 = ΔP	43 494	10 614	5 134	424

Table 11 should be read as follows: originally, ΔP is made of 170 164 differences corresponding to the 16 values that ΔX_2 can take. We make a guess on the value of $(K_0 \oplus K_1)[5]$ and compute Sbox number 5 of first round. In case the Sbox was active, the only correct output difference is 2, while a different result expresses that the key guess associated to the given pair is not correct. In case the Sbox is inactive, we cannot tell if a candidate is wrong so we keep it. This phenomenon is depicted in column "SB5" of Table 11: the number of differences that have nibble 5 active get divided by 6 (see the highlighted number) after we checked that they give the right output difference, while other sets remain unchanged. 43 494 pairs out of 170 164 remain, that is a ratio of $2^{-1.97}$. We follow a similar procedure for Sbox 6 and 9, that also have to lead to a difference of 2. We then look at Sbox number 4 and 12 together. As can be seen in Table 10, their outputs have to take the same value equal to $1, 8$ or 9. The probability of obtaining any of these 3 values from an active Sbox of our set is in the worst case for the attacker equal to $3 \cdot 2^{-2}$. Then, the probability of getting a correct pair of differences is equal to $3 \cdot 2^{-2} \cdot 2^{-2} \cdot \frac{1}{3} = 2^{-4}$, which leads to the results given in last column of Table 11.

The final step consists in checking Sbox 8 and 16 together with the consistency of all the Sbox differences as a whole. This leaves only 16 differences.