J. Cryptol. (2015) 28: 209–239
DOI: 10.1007/s00145-013-9162-9

Journal of
CRYPTOLOGY

New Attacks on IDEA with at Least 6 Rounds*

Eli Biham
Computer Science Department, Technion, Haifa 32000, Israel
biham@cs.technion.ac.il

Orr Dunkelman†
Computer Science Department, University of Haifa, Haifa 31905, Israel
orrd@cs.haifa.ac.il
and
Faculty of Mathematics and Computer Science, Weizmann Institute of Science, P.O. Box 26, Rehovot 76100, Israel

Nathan Keller‡
Department of Mathematics, Bar Ilan University, Ramat Gan 52900, Israel
nkeller@math.biu.ac.il
and
Faculty of Mathematics and Computer Science, Weizmann Institute of Science, P.O. Box 26, Rehovot 76100, Israel

Adi Shamir
Faculty of Mathematics and Computer Science, Weizmann Institute of Science, P.O. Box 26, Rehovot 76100, Israel
adi.shamir@weizmann.ac.il

Communicated by Willi Meier

Received 22 August 2011
Online publication 16 November 2013

Abstract. IDEA is a 64-bit block cipher with 128-bit keys which was introduced by Lai and Massey in 1991. The best previously published attack on IDEA could only handle 6 of its 8.5-rounds. In this paper, we combine a highly optimized meet-in-the-middle attack with a keyless version of the Biryukov–Demirci relation to obtain a greatly improved attack on 6-round IDEA which requires only two known plaintexts, and the first key recovery attacks on versions of IDEA with 6.5 to 8.5 rounds.

Key words. IDEA, Cryptanalysis, Biryukov–Demirci relation, Zero-in-the-Middle attack

* This paper is partially based on [4,5], presented at ASIACRYPT 2006 and FSE 2007, respectively.

† O. Dunkelman was supported in part by the German–Israeli Foundation for Scientific Research and Development through grant No. 2282-2222.6/2011.

‡ N. Keller was supported by the Alon Fellowship.

1. Introduction

IDEA (which is an acronym for International Data Encryption Algorithm) was introduced by Lai and Massey in 1991, and quickly became one of the best known and most well studied block ciphers. Even though it has only 8.5 relatively simple rounds which consist of just XOR's, additions, and multiplications of 16-bit values, it withstood more than 20 years of cryptanalysis surprisingly well (e.g., [2–6,8,9,11–14,17–21,23,25–28]).

The best attack on IDEA published until 2006 was the improved Demirci–Selçuk–Türe [2] attack on 5 rounds, whose 2^{124} time complexity is only slightly better than exhaustive search. At ASIACRYPT 2006 [4], we introduced the keyless Biryukov–Demirci relation and used it to reduce the time complexity of the attack on 5-round IDEA to 2^{103}. At FSE 2007 [5], we used an improved version of the same technique to devise the first attack on a 6-round variant of IDEA, but its complexity was extremely high: it was only twice as fast as exhaustive search, and required essentially the whole codebook of 2^{64} plaintext/ciphertext pairs. This 6-round attack was considerably improved by Sun and Lai at ASIACRYPT 2009 [28], who showed at how to reduce the data complexity from 2^{64} to 2^{49} chosen plaintexts, while at the same time reducing the time complexity from $2^{126.8}$ to $2^{112.1}$.

In this paper we combine the keyless Biryukov–Demirci relation with a highly optimized meet-in-the-middle attack and obtain a new attack on 6-round IDEA which reduces the data complexity from 2^{49} chosen plaintexts to 16 known plaintexts while remaining faster than the Sun and Lai attack (alternatively, we can reduce the data complexity all the way to its information-theoretic lower bound of 2 known plaintexts, but then its time complexity increases to $2^{123.4}$).

By using higher data complexities, we can attack larger variants of IDEA, which could not be successfully attacked by any previously published technique. By combining the keyless Biryukov–Demirci relation with the splice-and-cut variant of the meet-in-the-middle attack [1,24,29], we can break 6.5 rounds using about one thousand plaintexts in 2^{122} time. With further optimizations, we can attack 7 rounds in 2^{112} time using 2^{48} data, 7.5 rounds in 2^{114} time using 2^{63} data, and the full 8.5-round IDEA in $2^{126.8}$ time using only 16 plaintexts.

After the submission of our paper, Khovratovich, Leurent, and Rechberger [21] independently developed a different type of attacks on 5, 6, 7.5 and 8.5 round IDEA, using their new biclique approach in order to slightly optimize the complexity of exhaustive search. Compared to our techniques, their approach requires much larger data complexities to achieve tiny time savings. For example, their attack on full IDEA requires 2^{52} chosen plaintexts in order to reduce the time complexity to 2^{126} time, whereas our attack needs only 16 chosen plaintexts to reduce the time complexity to $2^{126.8}$.

Table 1 summarizes the major previously published attacks on reduced-round IDEA variants, and compares them to the new attacks presented in this paper.

The paper is organized as follows: In Sect. 2 we describe the structure of IDEA and introduce our notations. In Sect. 3 we overview the techniques used in this paper, including a keyless version of the Biryukov–Demirci relation (which gets rid of all the subkeys in the equation). In Sect. 4 we present our attacks on 6-round IDEA. The attacks on up to 7.5 rounds, which incorporate the splice-and-cut technique, are presented in Sect. 5. In Sect. 6 we show how to use our techniques to speed up exhaustive search on

Table 1. Comparing other attacks on IDEA with our new results.

Rounds	Attack type	Complexity		Source & Year
		Data	Time	
2	Differential	2^{10} CP	2^{40}	[23], 1993
2.5	Differential	2^{10} CP	$2^{104.7}$	[23], 1993
3	Differential-linear	2^{29} CP	2^{44}	[9], 1997
3.5	Differential	2^{56} CP	2^{67}	[9], 1997
4	Impossible differential	$2^{36.6}$ CP	$2^{66.6}$	[3], 1999
4.5	Impossible differential	2^{64} KP	$2^{110.4}$	[3], 1999
5	Demirci–Selçuk–Türe	$2^{24.6}$ CP	2^{124}	[2], 2006
5	ZitM BD-relation	2^{19} KP	2^{103}	[4], 2006
5.5	ZitM BD-relation	2^{32} CP	$2^{126.85}$	[5], 2007
6	ZitM BD-relation	2^{64} KP	$2^{126.8}$	[5], 2007
5.5	Key-dependent linear	2^{21} CP	$2^{112.1}$	[28], 2009
6	Key-dependent linear	2^{49} CP	$2^{112.1}$	[28], 2009
Our new results				
6	MitM BD-relation	2 KP	$2^{123.4}$	Sect. 4, 2011
6	MitM BD-relation	16 KP	$2^{111.9}$	Sect. 4, 2011
6.5	SaC MitM BD-relation	2^{10} CP	2^{122}	Sect. 5, 2011
6.5	SaC MitM BD-relation	2^{23} CP	2^{113}	Sect. 5, 2011
6.5	SaC MitM BD-relation	2^{32} CP	$2^{111.9}$	Sect. 5, 2011
7	SaC MitM BD-relation	2^{38} CP	2^{123}	Sect. 5, 2011
7	SaC MitM BD-relation	2^{48} CP	2^{112}	Sect. 5, 2011
7.5	SaC MitM BD-relation	16 CP	$2^{125.9}$	Sect. 6, 2011
7.5	SaC MitM BD-relation	2^{63} CP	2^{114}	Sect. 5, 2011
7.5	RK ZitM BD-relation	2^{25} CP	2^{103}	Sect. 7, 2011
8.5	SaC MitM BD-relation	16 CP	$2^{126.8}$	Sect. 6, 2011
The independently discovered results in [21]				
7.5	Biclique BD-relation	2^{18} CP	$2^{126.5}$	[21], 2012
7.5	Biclique BD-relation	2^{52} CP	$2^{123.9}$	[21], 2012
8.5	Biclique BD-relation	2^{52} CP	$2^{126.0}$	[21], 2012

ZitM—Zero-in-the-Middle, MitM—Meet-in-the-Middle, SaC—Splice-and-Cut, RK—Related Key, KP/CP—Known/Chosen Plaintext. Time complexity is measured in encryptions. †—This attack is a distinguishing attack.

the full IDEA. In Sect. 7 we introduce a different technique called Zero-in-the-Middle, and show how to use it to devise a related-key attack on 7.5 rounds of IDEA with a practical data complexity. In Appendix A, we present a surprising attack on 4.5-round IDEA which uses merely the meet-in-the-middle technique. We conclude with a short summary and discussion in Sect. 8.

2. Description of IDEA and Notations

IDEA [22] is a 64-bit, 8.5-round block cipher with 128-bit keys. It uses a composition of XOR operations, additions modulo 2^{16}, and multiplications over $GF(2^{16}+1)$.

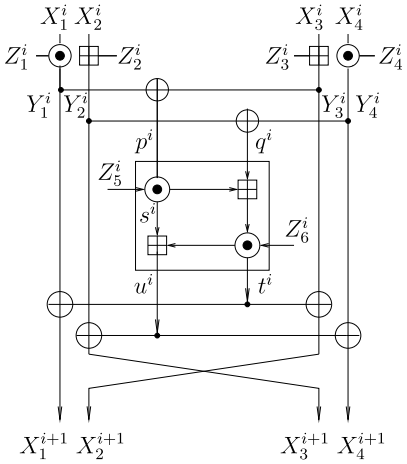

Fig. 1. One round of IDEA.

The structure of a single round of IDEA is shown in Fig. 1. As can be seen in the figure, every round of IDEA is the concatenation of two layers. The input of round i, denoted by X^i, consists of four 16-bit words, denoted by $(X_1^i, X_2^i, X_3^i, X_4^i)$. In the first layer (denoted by *KA* for Key Addition), the first and the fourth words are multiplied by subkey words (mod $2^{16}+1$) where a 0 operand is replaced by 2^{16}, and an outcome of 2^{16} is replaced by 0, and the second and the third words are added to subkey words (mod 2^{16}). The intermediate value after this half-round is denoted by $Y^i = (Y_1^i, Y_2^i, Y_3^i, Y_4^i)$. Formally, let Z_1^i, Z_2^i, Z_3^i, and Z_4^i be the four subkey words, let $\boxplus$ denote addition modulo 2^{16} and let $\odot$ be IDEA's special multiplication, then

$$Y_1^i = Z_1^i \odot X_1^i; \qquad Y_2^i = Z_2^i \boxplus X_2^i; \qquad Y_3^i = Z_3^i \boxplus X_3^i; \qquad Y_4^i = Z_4^i \odot X_4^i.$$

The pair $(p^i, q^i) = (Y_1^i \oplus Y_3^i, Y_2^i \oplus Y_4^i)$ enters the second layer, a structure composed of multiplications and additions denoted by *MA*. Denoting the subkey words that enter the *MA* function by Z_5^i and Z_6^i, the computation is performed as follows:

$$\begin{aligned} s^i &= p^i \odot Z_5^i; \\ t^i &= \left(q^i \boxplus s^i\right) \odot Z_6^i; \\ u^i &= t^i \boxplus s^i. \end{aligned}$$

The output of the *MA* function is (u^i, t^i), where u^i and t^i are related through $u^i = t^i \boxplus s^i$, a fact which is later used.

The output of the ith round is $X^{i+1} = (Y_1^i \oplus t^i, Y_3^i \oplus t^i, Y_2^i \oplus u^i, Y_4^i \oplus u^i)$. In the last round the *MA* layer is removed (i.e., the ciphertext is $Y^9 = (Y_1^9 || Y_2^9 || Y_3^9 || Y_4^9)$), and thus we refer to the full IDEA as an 8.5-round rather than as a 9-round scheme.

IDEA's key schedule is extremely simple, and turns out to be the source of many attacks. It is completely linear, and each subkey is a subset of 16 consecutive bits selected from the key. Since the exact structure of the key schedule is crucial for our attacks, the entire key schedule is described in Table 2. In this table and the remainder of this paper,

Table 2. The key schedule algorithm of IDEA. Each cell describes the bits of the secret key used in the corresponding subkey.

Round	Z_1^i	Z_2^i	Z_3^i	Z_4^i	Z_5^i	Z_6^i
$i=1$	0–15	16–31	32–47	48–63	64–79	80–95
$i=2$	96–111	112–127	25–40	41–56	57–72	73–88
$i=3$	89–104	105–120	121–8	9–24	50–65	66–81
$i=4$	82–97	98–113	114–1	2–17	18–33	34–49
$i=5$	75–90	91–106	107–122	123–10	11–26	27–42
$i=6$	43–58	59–74	100–115	116–3	4–19	20–35
$i=7$	36–51	52–67	68–83	84–99	125–12	13–28
$i=8$	29–44	45–60	61–76	77–92	93–108	109–124
$i=9$	22–37	38–53	54–69	70–85		

we denote the first bit of the key by 0 and the last bit of the key by 127, and use a cyclic interval notation such as 121–8 to denote the 16 bits $121, 122, \ldots, 127, 0, 1, \ldots, 7, 8$.

3. Overview of the Used Techniques

In this section we present the techniques we use in this paper. First we present the generic techniques—the standard Meet-in-the-Middle attack [15], along with its variant called the Splice-and-Cut attack [1,24]. Then we present the keyless Biryukov–Demirci relation, which is specific to IDEA and allows to exploit the simplicity of IDEA's operations and key schedule in an efficient way.

3.1. *The Meet-in-the-Middle Attack*

The Meet-in-the-Middle (MitM) attack, introduced by Diffie and Hellman [15] in 1977, is one of the most classic cryptanalytic techniques. The MitM attack on a block cipher uses the observation that given a (plaintext, ciphertext) pair, some (possibly partial) intermediate value V during the encryption process can be computed in two different ways:

- Using only the plaintext and part of the secret key material, denoted by K_t (where t stands for "top"), and
- Using only the ciphertext and a (possibly different) part of the key material, denoted by K_b (where b stands for "bottom").

In the attack, the adversary considers several known (plaintext, ciphertext) pairs, and for each guess of K_t, she computes from the plaintexts the corresponding V values and stores them in a hash table. Then, for each guess of K_b, she computes the V values from the ciphertexts, and searches for a match in the hash table. (If $|K_t| > |K_b|$, it is more efficient to swap the roles of K_t and K_b.)

Since the right value of the key material in K_t and K_b must lead to the same value of V in the two different computations (for each of the (plaintext, ciphertext) pairs), this right value can be found by checking all the (K_t, K_b) values which lead to a match in the hash table. If d (plaintext, ciphertext) pairs are examined, the expected number of such suggestions is $2^{|K_t|+|K_b|-d\cdot|V|}$, where $|X|$ denotes the length of X in bits. If

more than one suggestion passes the filtering, the remaining suggestions are checked by exhaustive search over the remaining bits of the secret key.

The *time complexity* of the attack is $2^{\max(|K_t|,|K_b|)}$ encryptions.

The *memory complexity* of the attack is $2^{\min(|K_t|,|K_b|)}$ blocks of size $d|V| + \min(|K_t|, |K_b|)$ bits. If the key materials guessed in the top and the bottom parts (i.e., K_t and K_b) share a common part K_c, the memory complexity can be reduced without affecting the time complexity by guessing K_c in advance, and repeating the attack for each value of K_c. The resulting memory complexity is $2^{\min(|K_t|,|K_b|)-|K_c|}$ blocks of size $d|V| + \min(|K_t|, |K_b|) - |K_c|$ bits.

The *data complexity* in a naive application of the attack is $(|K_t| + |K_b|)/|V|$ (plaintext, ciphertext) pairs, required for discarding all wrong values of (K_t, K_b) in the first filtering step. This complexity can be reduced with only a small effect on the time complexity by letting $2^{\max(|K_t|,|K_b|)}$ key candidates remain after the first filtering step, and then checking them exhaustively. (Note that this makes the time complexity of the second filtering step roughly equal to that of the first step, and hence is optimal.) The resulting data complexity is

$$\frac{|K_t| + |K_b| - |K_c| - \max(|K_t|, |K_b|)}{|V|} = \frac{\min(|K_t|, |K_b|) - |K_c|}{|V|}$$

(plaintext, ciphertext) pairs.

3.2. *The Splice-and-Cut Technique*

A promising enhancement of the meet-in-the-middle attack is the recently rediscovered (and renamed) method of Splice-and-cut. Originally presented in the attack on double-key triple-DES in [24], the attack was reintroduced in 2009 by Aoki and Sasaki [1] for cryptanalysis of hash functions, and was recently adapted to block ciphers by Wei et al. [29].

The idea behind the technique is rather simple. Instead of treating the encryption process as directed from the plaintext to the ciphertext, we consider it as a *cyclic* process, where the plaintext is connected to its corresponding ciphertext by an encryption oracle. In this treatment, the last rounds of encryption and the first ones are considered consecutive (with the oracle in between), and the plaintext and ciphertext are no longer treated as "special" points of the process. This treatment allows to apply a "cyclic" variant of the MitM attack. We fix an intermediate value I of the encryption process, and apply the MitM attack to the cyclic construction treating I as the plaintext/ciphertext point.

Specifically, like in the basic MitM attack, we compute the same intermediate value V in two ways, where in the first way we guess the key material K_t used in the rounds between I and V, and in the second way we guess the key material K_b used in the rounds below V and in the rounds above I. The new subdivision of the cipher is demonstrated in Fig. 2.

In some cases, including the case of IDEA, the ability to choose a "good" starting point I instead of the plaintext/ciphertext allows to exploit weaknesses of the key schedule to improve the attack significantly. A drawback of the attack is the need to use many chosen plaintext or chosen ciphertext queries in order to imitate the oracle used in the attack. However, if the new starting point I is relatively close to the plaintext or to the ciphertext, one can use structures to make the data complexity reasonable.

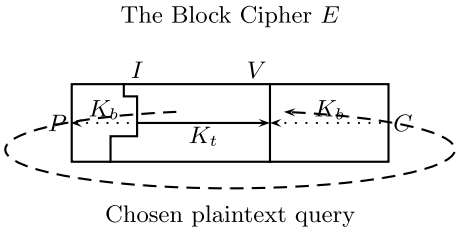

Fig. 2. The general idea of splice-and-cut.

For a full presentation of the splice-and-cut technique, we refer the reader to [1,24,29].

3.3. *The Keyless Biryukov–Demirci Relation*

In this section we present a keyless variant of the Biryukov–Demirci (BD) relation—a linear equation involving the plaintext, the ciphertext, and several intermediate values computed during the IDEA encryption process.[1]

We start with the basic observation made by Biryukov. Let us examine the second and the third words in all the intermediate stages of the encryption. There is a relation between the values of these words and the outputs of the *MA* layers in the intermediate rounds, which uses only XOR and modular addition, but not multiplication. Let $P = (P_1, P_2, P_3, P_4)$ be a plaintext and let $C = (C_1, C_2, C_3, C_4)$ be its corresponding ciphertext. Then

$$\begin{aligned}&\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(P_2 \boxplus Z_2^1\Big) \oplus u^1\Big) \boxplus Z_3^2\Big) \oplus t^2\Big) \boxplus Z_2^3\Big) \oplus u^3\Big) \boxplus Z_3^4\Big) \oplus t^4\Big) \boxplus Z_2^5\Big) \oplus u^5\Big)\\ &\quad \boxplus Z_3^6\Big) \oplus t^6\Big) \boxplus Z_2^7\Big) \oplus u^7\Big) \boxplus Z_3^8\Big) \oplus t^8\Big) \boxplus Z_2^9\Big) = C_2. \qquad (1)\end{aligned}$$

Similarly,

$$\begin{aligned}&\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(\Big(P_3 \boxplus Z_3^1\Big) \oplus t^1\Big) \boxplus Z_2^2\Big) \oplus u^2\Big) \boxplus Z_3^3\Big) \oplus t^3\Big) \boxplus Z_2^4\Big) \oplus u^4\Big) \boxplus Z_3^5\Big) \oplus t^5\Big)\\ &\quad \boxplus Z_2^6\Big) \oplus u^6\Big) \boxplus Z_3^7\Big) \oplus t^7\Big) \boxplus Z_2^8\Big) \oplus u^8\Big) \boxplus Z_3^9\Big) = C_3. \qquad (2)\end{aligned}$$

If we restrict our interest to the values of the least significant bits (LSB) of the words, modular addition is equivalent to XOR and we can simplify the above equations into

$$\begin{aligned}&LSB\big(P_2 \oplus Z_2^1 \oplus u^1 \oplus Z_3^2 \oplus t^2 \oplus Z_2^3 \oplus u^3 \oplus Z_3^4 \oplus t^4 \oplus Z_2^5 \oplus u^5 \oplus Z_3^6 \oplus t^6 \oplus Z_2^7\\ &\quad \oplus u^7 \oplus Z_3^8 \oplus t^8 \oplus Z_2^9\big) = LSB(C_2), \qquad (3)\end{aligned}$$

and

$$\begin{aligned}&LSB\big(P_3 \oplus Z_3^1 \oplus t^1 \oplus Z_2^2 \oplus u^2 \oplus Z_3^3 \oplus t^3 \oplus Z_2^4 \oplus u^4 \oplus Z_3^5 \oplus t^5 \oplus Z_2^6 \oplus u^6 \oplus Z_3^7\\ &\quad \oplus t^7 \oplus Z_2^8 \oplus u^8 \oplus Z_3^9\big) = LSB(C_3). \qquad (4)\end{aligned}$$

[1] The Biryukov–Demirci relation is based on observations made independently by Biryukov and Demirci, and was previously used by Junod in [19]. The keyless variant, which allows to enhance the attacks based on the relation significantly, was first presented in the conference version of this paper [4].

As observed by Demirci [13], $u^i = t^i \boxplus s^i$, thus, $LSB(u^i) = LSB(t^i \boxplus s^i)$, which is equivalent to $LSB(u^i \oplus t^i) = LSB(s^i)$. Taking this into consideration and XORing the two above equations, we obtain

$$\begin{aligned} &LSB\big(P_2 \oplus P_3 \oplus Z_2^1 \oplus Z_3^1 \oplus s^1 \oplus Z_2^2 \oplus Z_3^2 \oplus s^2 \oplus Z_2^3 \oplus Z_3^3 \oplus s^3 \oplus Z_2^4 \oplus Z_3^4 \oplus s^4 \\ &\quad \oplus Z_2^5 \oplus Z_3^5 \oplus s^5 \oplus Z_2^6 \oplus Z_3^6 \oplus s^6 \oplus Z_2^7 \oplus Z_3^7 \oplus s^7 \oplus Z_2^8 \oplus Z_3^8 \oplus s^8 \oplus Z_2^9 \oplus Z_3^9\big) \\ &= LSB(C_2 \oplus C_3). \end{aligned} \tag{5}$$

This equation is called in [19] "the Biryukov–Demirci relation", which we shall refer to as the BD-relation.

In this paper we use a keyless variant of the BD-relation in which all the Z_j^i subkeys are canceled. Consider any pair of known plaintexts P^1 and P^2. Denote the XOR difference between the encryptions of P^1 and P^2 (under the same secret key) in an intermediate value X by ΔX. Then, XORing the equations given by P^1 and P^2 yields

$$\begin{aligned} &LSB\big(P_2^1 \oplus P_3^1 \oplus P_2^2 \oplus P_3^2 \oplus \Delta s^1 \oplus \Delta s^2 \oplus \Delta s^3 \oplus \Delta s^4 \oplus \Delta s^5 \oplus \Delta s^6 \oplus \Delta s^7 \oplus \Delta s^8\big) \\ &= LSB\big(C_2^1 \oplus C_3^1 \oplus C_2^2 \oplus C_3^2\big). \end{aligned} \tag{6}$$

In the sequel, we refer to this equation as the *keyless BD-relation*.

4. Meet-in-the-Middle Biryukov–Demirci Attack on 6-Round IDEA

In this section, we combine the standard MitM technique with the keyless BD-relation to obtain a new attack on a 6-round variant of IDEA, which starts after the KA layer of round 2. The data complexity of our best attack is just 16 known plaintexts, its memory complexity is 2^{25} 64-bit blocks, and its time complexity is less than 2^{112} encryptions. This is a significant improvement over the best previously known attack on 6-round IDEA [28], which required 2^{49} chosen plaintexts and 2^{112} encryptions.

First we present the basic attack, and then we present a tradeoff that allows us to slightly reduce the high time complexity, at the expense of slightly increasing the low memory complexity. A reader who is mainly interested in the idea of the attack and is less concerned with the details may concentrate on Sect. 4.1 and skip the other parts of this section.

4.1. *The Basic Attack*

The idea behind the attack is that the keyless BD-relation can be incorporated into the standard meet-in-the-middle framework. Instead of computing the same intermediate value V in two different ways, we divide the terms of Eq. (6) into two sets, such that the terms in the first set can be computed using only the plaintexts and the set K_t of key bits (as defined in Sect. 3.1), and the terms in the second set can be computed using only the ciphertexts and the set K_b of key bits.

In the attack, for each guess of K_t, the adversary computes the XOR of all terms of the equation that belong to the first set, and stores it in a hash table. Then, for each guess of the subkey K_b, she computes the XOR of all terms that belong to the second set, and searches for a match in the hash table. If the equation is satisfied (which is always the

case for the correct guess of (K_t, K_b)), the XOR of all the terms in the equation is zero, which corresponds to a match in the hash table. In the sequel, we call such an attack *MitM BD attack*.

In the specific case of a 6-round variant of IDEA which starts after the *KA* layer of round 2 and ends after the *KA* layer of round 8, Eq. (6) can be written in the form

$$\begin{aligned} &LSB\big(P_2^1 \oplus P_3^1 \oplus P_2^2 \oplus P_3^2 \oplus \Delta s^2 \oplus \Delta s^3 \oplus \Delta s^4\big) \\ &\quad = LSB\big(C_2^1 \oplus C_3^1 \oplus C_2^2 \oplus C_3^2 \oplus \Delta s^5 \oplus \Delta s^6 \oplus \Delta s^7\big). \end{aligned} \tag{7}$$

We choose the sets as follows:

- The first set consists of the terms: $P_2^1, P_3^1, P_2^2, P_3^2, \Delta s^2, \Delta s^3, \Delta s^4$.
- The second set consists of the terms: $C_2^1, C_3^1, C_2^2, C_3^2, \Delta s^5, \Delta s^6, \Delta s^7$.

This division emphasizes the advantage of the MitM BD attack over the standard MitM attack. In the standard MitM attack, the adversary has to compute values from both the plaintext and ciphertext sides until she reaches a common intermediate value V. The use of the BD-relation allows us to "jump" over one round in the middle: the adversary computes only up to Δs^4 in the encryption direction and only up to Δs^5 in the decryption direction, and the meet-in-the-middle effect is achieved using Eq. (7) to bridge between these values. The attack algorithm is given by a pseudo-code in Fig. 3.

In order to compute the complexity of the attack, we have to find the values $|K_t|, |K_b|, |K_c|, |V|$ (as defined in Sect. 3.1). The terms of the first set can be computed from the plaintexts given key bits 50–33 (i.e., the entire subkeys of the *MA* layer of round 2 and the entire round 3, and the subkeys Z_1^4, Z_3^4, Z_5^4). The terms of the second set can be computed from the ciphertexts given key bits 125–99 (i.e., the entire

---

Input: 17 "plaintext"/"ciphertext" pairs $(P_1, C_1), (P_2, C_2), \ldots, (P_{17}, C_{17})$.
Divide the 17 plaintexts into 16 pairs: $(P_1, P_2), (P_1, P_3), \ldots, (P_1, P_{17})$ (with their corresponding ciphertexts).
for any key guess of key bits 0–33, 50–99, 125–127 **do**
  Initialize an empty hash table H.
  for any guess of key bits 34–49 **do**
    For each of the 16 pairs (C_1, C_i) compute $b_i \triangleq LSB(C_2^1 \oplus C_3^1 \oplus C_2^i \oplus C_3^i \oplus \Delta s^5 \oplus \Delta s^6 \oplus \Delta s^7)$, and store in H the value $(b_2, b_3, \ldots, b_{17}, K[34\text{–}49])$.
  end for
  for any guess of key bits 100–124 **do**
    For each of the 16 pairs (P_1, P_j) compute $b'_j \triangleq LSB(P_2^1 \oplus P_3^1 \oplus P_2^j \oplus P_3^j \oplus \Delta s^2 \oplus \Delta s^3 \oplus \Delta s^4)$, and check whether $(b'_2, b'_3, \ldots, b'_{17})$ is in H.
    If so, perform trial encryption under the key bits 0–33, 50–99, 125–127, the current guess of key bits 100–124, and the guess of bits 34–49 suggested in the corresponding entry of H.
  end for
end for

Fig. 3. The algorithm of our attack on 6-round IDEA (starting after the KA layer of round 2).

subkeys of the *KA* layer of round 8 and the entire round 7, in addition to the subkeys Z_1^6, Z_2^6, Z_5^5). Hence, we have $|K_t| = 112$, $|K_b| = 103$, $|K_c| = 87$ (since K_t and K_b share key bits 125–33 and 50–99), and $|V| = 1$ (since Eq. (7) considers only the LSB of the word).

Thus, using the formulas given in Sect. 3.1, the data complexity of the attack is $(\min(|K_t|, |K_b|) - |K_c|)/|V| = 16$ plaintext pairs (which can be obtained from 17 plaintexts), the memory complexity is $2^{\min(|K_t|,|K_b|)-|K_c|} = 2^{16}$ 32-bit blocks (which are equivalent to 2^{15} 64-bit blocks), and the time complexity is $2^{\max(|K_t|,|K_b|)} = 2^{112}$ partial encryptions of 17 plaintexts, which are roughly equivalent to 2^{115} encryptions.

4.2. *A Time-Memory-Data Tradeoff*

In this section we show that the time and data complexities of the attack can be slightly reduced to less than 2^{112} encryptions and 16 known plaintexts, at the expense of increasing the memory complexity to 2^{25} 64-bit blocks. The tradeoff may seem to be unattractive, but in fact, it reduces the largest complexity (time) while keeping a smaller complexity (memory) completely practical.

The most time-consuming part of the basic attack is computing the terms Δs^3 and Δs^4 for 16 plaintexts,[2] which requires the knowledge of the 112 key bits 50–33. We observe that bits 25–33 are required only for the subkey Z_5^4, which is used only in the last multiplication operation in the computation of Δs^4. Hence, at a first glance it seems that the adversary can guess the 103 key bits 50–24 and perform all operations except for the last multiplication, and then guess the remaining 9 key bits and perform a single multiplication operation for the 16 plaintexts. However, this is impossible since key bits 25–33 are also part of K_b, and hence, their value should be guessed and fixed in advance, before the beginning of the MitM phase.

This technical problem can be solved at the expense of increasing the memory complexity. The adversary simply ignores the fact that bits 25–33 are shared by K_t and K_b, and treats them as independent parts of K_t and K_b. As a result, the number of shared key bits is reduced to 78, and thus the memory complexity is increased to 2^{25} 40-bit blocks.[3] On the other hand, this allows the adversary to reduce the time complexity of the computation of Δs^3 and Δs^4, since it is now possible to postpone the guess of bits 25–33 until the last multiplication operation, as described above. As a result, this phase of the attack requires $2^{112} \cdot 16 = 2^{116}$ modular multiplications. Since each encryption with 6-round IDEA contains 24 modular multiplications (in addition to other operations), the time required is less than $2^{111.4}$ 6-round encryptions.[4]

[2] Note that the number of plaintexts is reduced to 16, which means that only 15 pairs are used in the MitM phase of the attack. As a result, in the second phase of the attack we have to check 2^{113} key guesses (instead of 2^{112} in the basic attack). We show below how this step can be performed efficiently, so that its complexity will be lower than that of the MitM phase.

[3] Note that the size of an entry in the table is 40 bits: 15 bits for the value of the evaluated keyless BD-relation in the 15 pairs, and 25 bits for the value of key bits 25–49.

[4] In order to evaluate more precisely the time complexity of the attack (and of the other attacks presented in this paper), one has to determine the ratio between the complexities of the three types of operations used in IDEA (i.e., XORs, modular additions, and modular multiplications). As this relation varies very much for different platforms, and the precise complexity is of little significance in this case, we compute the complexity according to the simplest measure that assumes that additions and XORs are negligible compared to modular multiplications.

After reducing the time complexity of the MitM phase, the second phase of the attack (i.e., discarding the 2^{113} remaining subkey candidates), becomes the most time-consuming phase of the attack. However, this part can also be performed more efficiently, as follows: at the phase of generating the hash table, the adversary also computes the entire value $p^5 = X_1^6 \oplus X_2^6$ for one of the plaintext/ciphertext pairs and stores it in the hash table. Then, for a remaining subkey guess, the adversary only computes the value p^5 for that plaintext/ciphertext pair from the plaintext side, and checks whether it matches the value in the corresponding entry of the hash table. As this is a 16-bit filtering, only 2^{97} key candidates remain after this stage, and they can be easily checked by trial encryption. Since during the computation of Δs^3 and Δs^4, the adversary already performs full encryption through round 3 and partial encryption through round 4, obtaining the value of p^5 requires only three modular multiplications, which are roughly equivalent to $1/8$ encryption. Thus, the time complexity of this phase is $(1/8) \cdot 2^{113} = 2^{110}$ encryptions.

Therefore, the total time complexity of the attack is $2^{111.4} + 2^{110} = 2^{111.9}$ encryptions. The memory complexity is increased by a small factor (due to the need to store the p^5 values) to 2^{25} 56-bit blocks, which are less than 2^{25} 64-bit blocks.

4.3. *Other Attacks on 6-Round IDEA*

For the sake of completeness, we consider in this section several other attacks on 6-round IDEA, which represent different time/memory/data tradeoffs, or target different consecutive sets of rounds.

4.3.1. *An Attack with Only Two Known Plaintexts*

A variant of the attack described above can be used to attack the same 6-round variant of IDEA with only two known plaintexts and time complexity of $2^{123.4}$ encryptions.

First, the adversary constructs the tables and performs the MitM phase of the basic attack described above. Since the adversary has only two plaintexts in his disposition, she can check the validity of the keyless Biryukov–Demirci relation only once, and thus, 2^{127} key suggestions remain after this stage.

As described in the previous section, most of these suggestions can be discarded efficiently by storing in the table also the p^5 value in one of the encryptions and computing it from the plaintext side for each subkey suggestion. In order to make this step even more efficient, the adversary can make a small change in the MitM phase of the attack: In addition to computing Δs^3 and Δs^4, she computes the intermediate values until the multiplication with the subkey Z_6^4 in the *MA* layer of round 4. Given these intermediate values, p^5 can be computed with only 2 modular multiplications, 2 modular additions, and 2 XORs, which are less than $1/12$ of a 6-round encryption.

The time complexity of the attack is dominated by the second phase (i.e., discarding the subkey suggestions), whose complexity is $2^{127} \cdot (1/12) = 2^{123.4}$ encryptions.

We note that a similar attack can be applied to any number $2 \leq k \leq 16$ of plaintexts, with time complexity of $2^{107.4} \cdot k + 2^{125.4-k}$ encryptions.

4.3.2. *Attacks on Other Reduced-Round Variants of IDEA*

Our analysis indicates that no other 6-round variant of IDEA (with a shifted starting position) can be attacked using our technique.

Two specific cases of interest are reduced-round variants in which the targeted rounds are either the first or the last rounds of IDEA. In these cases, we obtained the following results.

If the reduced-round variant must end at the *KA* layer of round 9, then 5.5 rounds can be attacked, with data complexity of 10 known plaintexts, memory complexity of 2^{24} 64-bit blocks, and time complexity of 2^{119} encryptions.

If the reduced-round variant must start at the beginning of round 1 (as considered in [21,28]), then 5 rounds can be attacked, with data complexity of 10 known plaintexts, memory complexity of 2^{24} 64-bit blocks, and time complexity of 2^{119} encryptions.

For the sake of comparison, the best previous attack on the same variant which was presented in [28] requires either 2^{17} chosen plaintexts and $2^{125.5}$ encryptions, or 2^{64} known plaintexts and $2^{115.5}$ encryptions. The attack of Khovratovich et al. [21], which was obtained independently after the first version of this paper was submitted, requires 2^{25} chosen plaintexts, and either 2^{16} memory and 2^{110} encryptions or 2^{110} memory and 2^{112} memory accesses.

In addition, [21] presents an attack on 6-round IDEA which targets the first six rounds, with data complexity of 2^{41} chosen plaintexts, time complexity of $2^{118.9}$ encryptions and memory complexity of 2^{12}. While these complexities are higher than those of our attack on 6-round IDEA presented above, this attack targets a set of consecutive rounds which cannot be attacked using our technique.

5. Splice-and-Cut Biryukov–Demirci Attacks on up to 7.5-Round IDEA

In this section, we show that by using the splice-and-cut [1] variant of the meet-in-the-middle technique, we can increase the number of rounds the MitM BD attack can target from 6 to 7.5, without affecting the time complexity of the attack.

In Sect. 5.1 we present the basic attack procedure, which can break the first 7.5 rounds of IDEA with time complexity of 2^{112} encryptions, but requires the entire codebook. In the following sections, we show how the data complexity can be reduced significantly by allowing the fixed value of the intermediate state I to vary as function of the bits of K_c which are guessed at the beginning of the attack. These sections consider separately variants of 6.5, 7, and 7.5 rounds of IDEA, where the 6.5-round variant starts at the beginning of round 2, the 7-round variant starts after the KA layer of round 1, and the 7.5-round variant consists of the first 7.5 rounds of IDEA. The complexities of the attacks are presented in Table 1.

A reader who is mainly interested in the attack's idea and is less concerned with the details, can concentrate on Sect. 5.1 and the beginning of Sect. 5.2, and skip the remaining parts of this section (which are similar in nature but more technical).

5.1. *The Basic Attack on* 7.5*-Round IDEA*

Consider a reduced-round variant which consists of the first 7.5 rounds of IDEA. We want to show that the basic attack on 6-round IDEA presented in Sect. 4.1 can be extended to this variant, without increasing the time complexity.

Recall that in our 6-round attack, K_t consists of all the key except for bits 34–49, and K_b consists of all the key except for bits 100–124.

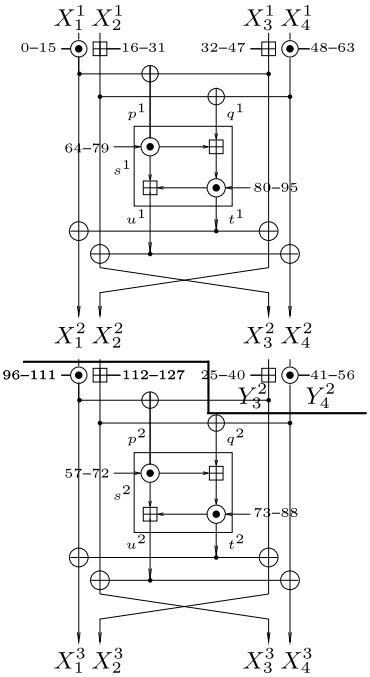

Fig. 4. The 7.5-round splice-and-cut location.

The basic observation behind the 7.5-round attack is that all the subkeys used in round 1 are included in K_b, and the four subkeys used in the *KA* layer of round 2 consist of Z_1^2, Z_2^2 which are included in K_t and Z_3^2, Z_4^2 which are included in K_b. This allows us to perform a Splice-and-Cut BD attack, where the intermediate value I is located at the *KA* layer of round 2 in a ladder-type fashion: In the first two words, I represents the value before the subkey addition/multiplication, while in the last two words I represents the value after the subkey addition/multiplication. That is, $I = (X_1^2, X_2^2, Y_3^2, Y_4^2)$. The location of I is presented in Fig. 4. We note that a similar ladder-type technique was used in another context by Biryukov and Khovratovich [7], under the name *ladder switch*.

In our case, Eq. (6) can be rewritten as

$$\begin{aligned} &LSB\big(\Delta s^2 \oplus \Delta s^3 \oplus \Delta s^4\big) \\ &\quad = LSB\big(P_2^1 \oplus P_3^1 \oplus P_2^2 \oplus P_3^2 \oplus C_2^1 \oplus C_3^1 \oplus C_2^2 \oplus C_3^2 \oplus \Delta s^5 \oplus \Delta s^6 \oplus \Delta s^7 \oplus \Delta s^1\big). \end{aligned} \tag{8}$$

Given the intermediate value in the state I, the left-hand-side of Eq. (8) can be computed using only the subkeys included in K_t. On the other hand, given the value in the state I and the subkey K_b, the adversary can partially decrypt the intermediate value at I through the first 1.5 rounds to obtain the plaintext. Assuming that the entire codebook is available, she can obtain the corresponding ciphertext, and then partially decrypt

Input: The entire code book.
Determine 17 values $I_i = (X_1^{2i}, X_2^{2i}, Y_3^{2i}, Y_4^{2i})$ as "plaintext" values.
Divide the 17 "plaintexts" into 16 pairs: $(I_1, I_2), (I_1, I_3), \ldots, (I_1, I_{17})$
for any key guess of key bits 0–33, 50–99, 125–127 **do**
  Initialize an empty hash table H.
  for any guess of key bits 34–49 **do**
    Partially decrypt all I_i to obtain the corresponding plaintext P_i. Deduce from the given data the corresponding ciphertext C_i.
    For each pair (C_1, C_i) compute $b_i = LSB(P_2^1 \oplus P_3^1 \oplus P_2^j \oplus P_3^j \oplus C_2^1 \oplus C_3^1 \oplus C_2^j \oplus C_3^j \oplus \Delta s^5 \oplus \Delta s^6 \oplus \Delta s^7 \oplus \Delta s^1)$ and store in H the value $(b_2, b_3, \ldots, b_{17}, K[34\text{–}49])$.
  end for
  for any guess of key bits 100–124 **do**
    For each of the 16 pairs (I_1, I_j) compute $b'_j \triangleq LSB(\Delta s^2 \oplus \Delta s^3 \oplus \Delta s^4)$, and check whether $(b'_2, b'_3, \ldots, b'_{17})$ is in H.
    If so, perform trial encryption under the key bits 0–33, 50–99, 125–127, the current guess of key bits 100–124, and the guess of bits 34–49 suggested in the corresponding entry of H.
  end for
end for

Fig. 5. The algorithm of the basic splice-and-cut attack on 7.5-round IDEA.

through rounds 8, 7, 6, 5 to compute the right-hand-side of Eq. (8). The attack procedure is presented as a pseudo-code in Fig. 5.

In order to compute the complexity of the attack, we have to find the values $|K_t|, |K_b|, |K_c|, |V|$. As in the basic 6-round attack in Sect. 4.1, we have $|K_t| = 112$, $|K_b| = 103$, $|K_c| = 87$, and $|V| = 1$. Thus, the attack requires $(\min(|K_t|, |K_b|) - |K_c|)/|V| = 16$ plaintext pairs (which can be obtained from 17 fixed intermediate values), the memory complexity is $2^{\min(|K_t|,|K_b|)-|K_c|} = 2^{16}$ 32-bit blocks (which are equivalent to 2^{15} 64-bit blocks), and the time complexity is $2^{\max(|K_t|,|K_b|)} = 2^{112}$ partial encryptions of 17 plaintexts, which are roughly equivalent to 2^{115} encryptions. The time-memory tradeoff presented in Sect. 4.2 works without change as well, allowing us to reduce the time complexity to less than 2^{112} encryptions.

The crucial difference between our attack and the 6-round attack is in the data complexity. In the basic form described above, the 7.5-round attack requires the entire codebook, which is used to imitate the oracle that maps plaintexts to the corresponding ciphertexts (see Sect. 3.2). In the following sections, we show that the data complexity can be reduced significantly with only a small effect on the time and memory complexities, using the ability to vary the value at the intermediate state I, depending on the key bits in K_c guessed at the beginning of the attack.

5.2. *Reducing the Data Complexity for* 6.5*-Round IDEA*

Consider a 6.5-round variant of IDEA, which starts at the beginning of round 2 and ends after the *KA* layer of round 8. The splice-and-cut attack presented above applies,

of course, to this variant as well. We would like to show that the value of the intermediate state I can be chosen such that only 2^{23} specific plaintext values are encountered during the computation of the right-hand-side of Eq. (8). In such a case, the knowledge of the 2^{23} ciphertexts corresponding to these specific plaintexts is sufficient to imitate the encryption oracle, and hence, the data complexity of the attack is reduced to 2^{23} chosen plaintexts.

First, note that in our variant, the first two words of I (i.e., X_1^2 and X_2^2) are simply the first two words of the plaintext. Hence, by fixing the value of these two words to zero in all values of I considered in the attack, we assure that only plaintexts whose first two words are equal to zero are encountered during the attack.

Second, note that the 9 most significant bits (MSBs) of the addition subkey Z_3^2 (i.e., bits 25–33) are included in K_c in the basic 6-round attack. This allows us to choose the value of the third word of I (i.e., Y_3^2) in a more sophisticated way. Recall that the inner loop of the basic 6-round attack is repeated for each possible value of the 87 bits of K_c (see Fig. 5).

We suggest to choose a different value of Y_3^2 for each value of bits 25–33, as follows. Denote the value of bits 25–33 by $v \in \{0,1\}^9$. In the application of the inner loop which corresponds to the guess v, we choose the intermediate value Y_3^2 to be $v||1111111$, where $||$ denotes concatenation of bit strings. For this choice, the 2^7 corresponding values of P_3 (obtained for the 2^7 possible values of the 7 LSBs of Z_3^2) are the 2^7 values of the form $000000000||w$, where w takes all possible values in $\{0,1\}^7$. This assures that all the plaintexts encountered during the attack have zeros as the 9 MSBs of P_3.

Therefore, by choosing the values of X_1^2, X_2^2, and Y_3^2 as a function of the bits in K_c as described, we assure that all plaintexts encountered in the attack have zeros at their 41 MSBs. This reduces the data complexity to 2^{23} chosen plaintexts.

The price of the significant data reduction is a slightly increased time complexity (from 2^{112} to 2^{113}), as the time-memory tradeoff described in Sect. 4.2 is not compatible with the sophisticated choice of Y_3^2 suggested above. Indeed, in the attack presented in Sect. 4.2, key bits 25–33 are no longer part of the external loop, and thus, Y_3^2 cannot be chosen according to their value.

To minimize the computation overhead, we note that the adversary can still perform part of the computation of Δs^3 and Δs^4 before guessing all the 25 key bits 100–124. Specifically, she can compute Δs^3 and perform the multiplication with Z_4^3 before guessing subkey bits 105–120, and only then guess these key bits and perform the rest of the computation of Δs^4. As a result, this phase of the attack is roughly equivalent to $2^{112} \cdot 16 \cdot 3 = 2^{117.6}$ modular multiplications. Since each encryption with 6.5-round IDEA contains 26 modular multiplications, this is roughly equivalent to $2^{112.9}$ 6.5-round encryptions. The time complexity of the rest of the attack (which is equal to the complexity of the corresponding steps of the 6-round attack) is negligible, and hence, the overall time complexity of the attack is about 2^{113} encryptions.

The data complexity can be further reduced by another factor of 2^{13} to only 2^{10} chosen plaintexts, at the expense of increasing the time complexity by a factor of 2^9. Note that out of the 16 bits of the multiplication subkey Z_4^2, 7 bits are included in K_c. If we guess the 9 remaining bits (i.e., bits 41–49) at the beginning of the attack, we can choose the value Y_4^2 in accordance with the value of Z_4^2, such that the corresponding value of P_4 is fixed. Since the attack requires 8 intermediate values for performing

the match in the middle (such that the remaining part of the attack has a smaller time complexity), the data complexity is reduced to 2^{10} chosen plaintexts (2^7 possible values of P_3, and 8 possible values of P_4).

We note that various other tradeoffs between the data and the time complexities of the attack are possible as well.

5.3. *Reducing the Data Complexity for 7-Round IDEA*

In this section we show that for 7-round IDEA, the data complexity can be reduced to 2^{48} chosen plaintexts without affecting the time complexity of 2^{112}, and can be further reduced to 2^{38} chosen plaintexts, at the expense of increasing the time complexity to 2^{123}.

The 7-round variant we target starts after the *KA* layer of round 1 and ends after the *KA* layer of round 8. In particular, the plaintexts in this variant correspond to the state $(Y_1^1, Y_2^1, Y_3^1, Y_4^1)$. Obviously, the basic 7.5-round attack applies to this variant as well.

To obtain the first data complexity reduction, we observe that by the structure of the *MA* layer in IDEA, we have

$$X_2^1 \oplus X_2^2 = Y_1^1 \oplus Y_3^1 = P_1 \oplus P_3.$$

Hence, we can choose $X_1^2 = X_2^2$ in all values of I considered in the attack, and this assures that all encountered plaintexts satisfy $P_1 = P_3$. This reduces the data complexity to 2^{48} chosen plaintexts.

The data complexity can be further reduced from 2^{48} to 2^{39} chosen plaintexts, at the expense of increasing the time complexity by factor of about 2^{10}. As described at the end of Sect. 5.2, if we guess the value of key bits 41–49 at the beginning of the attack, we can choose the value Y_4^2 according to the value of the subkey Z_4^2, such that X_4^2 always assumes the same prescribed value. If we choose them such that $X_4^2 = 0$, then $P_2 \oplus P_4 = X_3^2$ holds throughout the attack. This allows us to choose the value Y_3^2 in such a way that the 9 MSBs of $P_2 \oplus P_4$ are equal to zero (like in the 6.5-round attack).

This reduces the data complexity to 2^{39} chosen plaintexts (since the 9 MSBs of $P_2 \oplus P_4$ and all 16 bits of $P_1 \oplus P_3$ are equal to zero in all encountered plaintexts). On the other hand, the time complexity is increased by a factor of 2^{10}, due to the guess of bits 41–49, and since this improvement is not compatible with the time-memory tradeoff presented in Sect. 4.2.

The data complexity can be reduced by another factor of 2, at the expense of increasing the time complexity by the same factor. Here we use the fact that by the BD-relation,

$$LSB\left(X_2^2 \oplus X_3^2\right) = LSB(P_2 \oplus P_3) \oplus LSB\left(s^1\right).$$

Note that since $p^1 = X_1^2 \oplus X_2^2$ and the subkey Z_5^1 is included in K_c, we can choose the values of X_1^2 and X_2^2 according to the value of Z_5^1 in such a way that $s^1 = p^1 \odot Z_5^1$ is fixed in all encountered encryptions. Furthermore, if we guess the LSB of the subkey Z_3^2 (i.e., bit 40) at the beginning of the attack, we can choose the values X_2^2 and Y_3^2 such that $LSB(X_2^2 \oplus X_3^2)$ is fixed in all encryptions encountered in the attack. By the equation above, this means that we can choose X_2^1, X_2^2, and Y_3^2 in such a way that $LSB(P_2 \oplus P_3)$ is fixed for all plaintexts encountered in the attack. This reduces the data complexity by

an additional factor of 2, at the expense of increasing the time complexity by the same factor (due to the external guess of bit 40).

The two reductions of the data complexity can be combined, resulting in data complexity of 2^{38} chosen plaintexts, and time complexity of about 2^{123} encryptions.

5.4. *Reducing the Data Complexity for 7.5-Round IDEA*

In the case of the first 7.5 rounds of IDEA, the data complexity can be reduced to 2^{63} chosen plaintexts, using the second improvement of the 7-round attack described above. We use the equation

$$LSB\left(X_2^2 \oplus X_3^2\right) = LSB(P_2 \oplus P_3) \oplus LSB\left(Z_2^1 \oplus Z_3^1\right) \oplus LSB\left(s^1\right).$$

As described above, we can choose X_2^1, X_2^2, and Y_3^2 in such a way that $LSB(X_2^2 \oplus X_3^2)$ and $LSB(s^1)$ are fixed for all encryptions encountered in the attack. If we guess $LSB(Z_2^1 \oplus Z_3^1)$ at the beginning of the attack, then we can adjust the choice of X_2^1, X_2^2, and Y_3^2 such that $LSB(X_2^2 \oplus X_3^2) \oplus LSB(Z_2^1 \oplus Z_3^1) \oplus LSB(s^1)$ will be fixed throughout the attack. By the equation, this implies that $LSB(P_2 \oplus P_3)$ will be fixed for all plaintexts encountered in the attack.

This reduces the data complexity to 2^{63} chosen plaintexts, while increasing the time complexity to slightly less than 2^{114} encryptions.

6. Reducing the Time Complexity of Exhaustive Key Search on the Full IDEA

In this section we show that the techniques presented in Sect. 5 can be used to marginally reduce the time complexity of exhaustive key search on the full 8.5-round IDEA to $2^{126.8}$ encryptions, at the expense of slightly increasing the data complexity to 16 chosen plaintexts. After the first version of this paper was submitted, Khovratovich et al. [21] obtained (independently) another attack on the full IDEA, which reduces the complexity of exhaustive search to $2^{126.0}$ encryptions, but at the expense of a much higher data complexity of 2^{52} chosen plaintexts.

We present our attack in Sect. 6.1, and in Sect. 6.2 we compare it with generic methods of optimized exhaustive search, and with the results of [21].

6.1. *Splice-and-Cut Biryukov–Demirci Attack on the Full IDEA*

The Splice-and-Cut BD attack presented in Sect. 5 cannot be extended directly to the full 8.5-round IDEA, since for any division of the terms of Eq. (6) into two sets, the computation of each set requires the knowledge of the entire secret key. What we can do is to reduce the complexity of exhaustive key search by computing *most* of the terms before guessing the entire key, such that only a few operations have to be performed for every guess of the full key.

The attack is similar to the basic Splice-and-Cut BD attack presented in Sect. 5.1. First, we treat the full IDEA in a cyclical manner, and choose the "starting point"—the intermediate value I. In order to minimize the data complexity, we choose I to be as close to the plaintext as possible. The choice, which is $I = (X_1^1, Y_2^1, X_3^1, X_4^1)$ is demonstrated in Fig. 6.

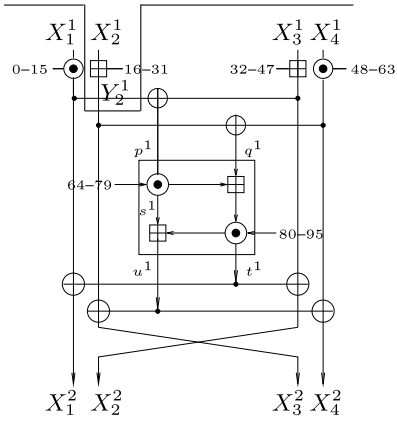

Fig. 6. The 8.5-round splice-and-cut location.

Second, we divide the terms of the BD-relation into two sets. Note that we use the original variant of the BD-relation and not the keyless one, since the keyless relation requires examining two (plaintext, ciphertext) pairs, which would double the encryption time, thus eliminating most of the time saving. The representation of the BD-relation we use is the following:

$$\begin{aligned} &LSB\big(Z_2^1 \oplus Z_3^1 \oplus Z_2^2 \oplus Z_3^2 \oplus Z_2^3 \oplus Z_3^3 \oplus Z_2^4 \oplus Z_3^4 \oplus Z_2^5 \oplus Z_3^5 \oplus Z_2^6 \oplus Z_3^6 \oplus Z_2^7 \\ &\quad \oplus Z_3^7 \oplus Z_2^8 \oplus Z_3^8 \oplus Z_2^9 \oplus Z_3^9 \oplus s^1 \oplus s^2 \oplus s^3\big) \\ &= LSB\big(P_2 \oplus P_3 \oplus C_2 \oplus C_3 \oplus s^4 \oplus s^5 \oplus s^6 \oplus s^7 \oplus s^8\big). \end{aligned} \tag{9}$$

We observe that the left-hand-side of the equation can be computed entirely using the knowledge of the value at state I and bits 0–15 and 25–127 of the secret key. On the other hand, all terms in the right-hand-side, except for s^4, s^5, s^6, can be computed using the value at state I, bits 0–12 and 16–127 of the key, and the encryption oracle (which will be later replaced by the appropriate chosen plaintext queries). Note that the two sets of bits have key bits 0–12 and 25–127 in common.

Now we are ready to present the attack algorithm.

Attack algorithm:

1. Choose an arbitrary value of $I = (X_1^1, Y_2^1, X_3^1, X_4^1)$.
2. For each value of bits 0–12 and 25–127 of the key, perform the following:
   (a) For each value of bits 13–15 of the key, perform the following:
      (i) Compute the value

$$\begin{aligned} &LSB\big(Z_2^1 \oplus Z_3^1 \oplus Z_2^2 \oplus Z_3^2 \oplus Z_2^3 \oplus Z_3^3 \oplus Z_2^4 \oplus Z_3^4 \oplus Z_2^5 \oplus Z_3^5 \oplus Z_2^6 \\ &\quad \oplus Z_3^6 \oplus Z_2^7 \oplus Z_3^7 \oplus Z_2^8 \oplus Z_3^8 \oplus Z_2^9\big). \end{aligned} \tag{10}$$

      (ii) Partially encrypt I through rounds 1,2,3, and compute the values s^1, s^2, s^3 (which allow to compute the left-hand-side of Eq. (9).

(iii) Store the value of the LHS of Eq. (9), along with the intermediate value p^3, in a table entry corresponding to the value of bits 13–15.

(b) For each value of bits 16–24 of the key, perform the following:

(i) Decrypt I through the key addition with Z_2^1 to obtain the corresponding plaintext. Consider the corresponding ciphertext,[5] and partially decrypt it[6] through rounds 9,8,7 to obtain the values s^7, s^8.

(ii) For each value of bits 13–15 of the key, continue the partial decryption to compute the values s^4, s^5, s^6 (which allow to compute the right-hand-side of Eq. (9)).

(iii) Check, using the corresponding entry in the table, whether Eq. (9) holds. If not, discard the key guess.

(iv) For the remaining keys, continue the partial decryption through rounds 5 and 4 and check whether the value of $p^3 = X_1^4 \oplus X_2^4$ matches the corresponding value in the table. As this is a 16-bit filtering, most of the key guesses are discarded at this stage.

(v) Check the remaining key guesses by a trial encryption.

Reducing the Data Complexity As we show below, this algorithm allows to speed-up exhaustive search by a factor of about 2.5. However, in a naive implementation, it increases the data complexity to 2^{16} chosen plaintexts, since for different values of key bits 16–31, the intermediate value I leads (by partial decryption) to 2^{16} different plaintexts. The data complexity can be reduced by varying the value at the state I according to the value of (part of the) bits 16–31.

Specifically, we can reduce the complexity to 2^9 chosen plaintexts by setting the 7 LSBs of Y_2^1 to be equal to bits 25–31 of the key (which are guessed in the external loop of the attack), which assures that the 7 LSBs of P_2 are zero in all encountered plaintexts.

The complexity can be reduced even further by adding part of bits 16–24 to the external loop of the attack. For example, adding bits 20–24 to the external loop increases the time complexity of the attack by less than 5 %, while reducing the data complexity to only 16 chosen plaintexts. The data complexity can be reduced even further, but at the expense of increasing the time complexity. We compute below the time complexity for the variant of the attack that requires 16 chosen plaintexts.

The Time Complexity of the Attack The most time-consuming step of the attack is Step 2(b)(ii), consisting of 11 multiplications, 10 additions, 20 XORs, and 1 table lookup, is performed for all the keys. After adding the effect of the other steps, and using the assumption that additions and XORs are negligible (compared to a modular multiplication), we conclude that the average number of modular multiplications for each key guess is 14.75. Since the full 8.5-round IDEA contains 34 multiplications, the time complexity of the attack is $\frac{14.75}{34} \cdot 2^{128} = 2^{126.8}$ encryptions.

[5] As shown below, the data complexity of the attack is only 16 chosen plaintexts. Hence, the plaintext/ciphertext pairs can be stored in a table of size 16, and the corresponding ciphertext can be retrieved by a single table lookup.

[6] Note that this operation can be performed more efficiently using the fact that key bits 125–12 are not used in the decryption direction until the multiplication with the subkey Z_5^7. This allows us to perform all the operations in this step except for the last multiplication only once for each value of bits 125–12, which makes the complexity of all these operations negligible (compared to the other parts of the attack).

6.1.1. *An attack on 7.5-Round IDEA*

We note that a similar attack can be applied to 7.5-round IDEA, starting at the plaintext. In this case, I is chosen as in the attack on the full IDEA, and the adversary uses the fact that key bits 100–124 are not used between the plaintext and I, in rounds 8,7, and in part of round 6. The data complexity of the attack is 16 chosen plaintexts, and its time complexity is 7.25 multiplications on average for each key, which are equivalent to $2^{125.9}$ 7.5-round encryptions. This attack has a higher time complexity than the 7.5-round attack presented in Sect. 5 ($2^{125.9}$ vs. 2^{114}), but we mention it due to its greatly reduced data complexity (16 vs. 2^{63}).

6.2. *Comparison with Optimized Exhaustive Search and with the Results of [21]*

When comparing our attack to optimized exhaustive search, we take into considerations well-known optimizations such as those implemented in the EFF DES cracking machine [16] or in [10]. For sake of comparison, we describe the case of an attack which divides the key into two (not necessarily disjoint) sets, and note that in many attacks these sets may contain the entire key.

The basic idea behind these optimizations is to use a meet-in-the-middle approach, deploying the following observation: Assume that there exist subsets K_t, K_b of the secret key, such that the first few operations of the encryption process require only the knowledge of K_t, and the last few operations require only the knowledge of K_b. Let $K_c = K_t \cap K_b$ denote the set of common key bits. Then, exhaustive key search can be trivially enhanced by the following algorithm:

For each value of K_c, perform the following:

1. For each value of $K_t \setminus K_c$, perform the first few operations of the encryption process (which require only the knowledge of K_t) for the given plaintext. Create[7] a table that contains the intermediate values corresponding to the values of the bits in $K_t \setminus K_c$.
2. For each value of $K_b \setminus K_c$, perform the last few operations of the encryption process (which require only the knowledge of K_b) in the decryption direction for the given ciphertext. Then, guess the remaining bits of K, compute the rest of the operations until the intermediate values, and check the match with the values stored in the pre-computed table.

In the case of IDEA, most operations in the first two rounds can be performed without the knowledge of bits 112-127 of the key. In particular, the value $p^2 = X_1^3 \oplus X_2^3$ can be computed without the knowledge of these 16 key bits. In the decryption direction, all operations of rounds 8,9 and the multiplication with Z_5^7 can be performed without the knowledge of bits 13–21 of the key.

Hence, using the above algorithm with $K_t = \{0\text{–}111\}$, $K_b = \{0\text{–}12, 22\text{–}127\}$ and matching at the value $V = p^3$, the number of modular multiplications performed for each key guess can be reduced to 18 out of 34 total multiplications in IDEA. Therefore, we estimate the time complexity of optimized exhaustive search as $\frac{18}{34} \cdot 2^{128} = 2^{127.1}$

[7] We note that when K_t or K_b compose the entire key, there is no need in a table, as the gain comes from the partial evaluation.

encryptions. The data complexity of optimized exhaustive key search is, of course, 2 known plaintexts.

For comparison, our attack presented in Sect. 6.1 allows reducing the time complexity to $2^{126.8}$ encryptions, at the expense of increasing the data complexity to 16 chosen plaintexts. The attack presented in [21] allows further reducing the time complexity to $2^{126.0}$ encryptions, at the expense of significantly increasing the data complexity to 2^{52} chosen plaintexts.

The main difference between our attack and the attack of [21] is that the attack of [21] places the "starting point" of the attack at $I = (X_1^2, X_2^2, Y_3^2, Y_4^2)$, like in our 7.5-round attack. On the one hand, this allows to slightly reduce the time complexity since more operations can be performed before the full key must be guessed. On the other hand, since the "starting point" is farther from the plaintext, the data complexity is increased significantly.

6.2.1. *The Case of 7.5-Round IDEA*

In the case of 7.5-round IDEA (where the targeted rounds are the first 7.5 rounds), the time complexity of optimized exhaustive search is slightly lower than in the case of the full IDEA, since the last 2.5 rounds of encryption can be computed without the knowledge of key bits 100–115. Applying the algorithm above with $K_t = \{0\text{–}111\}$, $K_b = \{0\text{–}99, 116\text{–}127\}$ and $V = p^2$ leads to 11 multiplications on average for each subkey guess. Since the total number of multiplications in 7.5-round IDEA is 28, the time complexity of optimized exhaustive search is $\frac{11}{28} \cdot 2^{128} = 2^{126.7}$ encryptions.

In [21], two attacks on this variant were presented. The first allows to slightly reduce the time complexity to $2^{126.5}$ encryptions, at the expense of increasing the data complexity to 2^{18} chosen plaintexts. The second allows to further reduce the time complexity to $2^{123.9}$ encryptions, but requires a larger amount of 2^{52} chosen plaintexts.

For comparison, our attack presented in Sect. 6.1 has time complexity of $2^{125.9}$ encryptions, and requires 16 chosen plaintexts. Hence, our attack is strictly better than the first attack of [21], and is incomparable with the second attack of [21]. A comparison of exhaustive search speedups on 7.5-round and 8.5-round IDEA is presented in Table 3.

Table 3. Comparison of exhaustive search speedups on 7.5-round and 8.5-round IDEA.

Rounds	Attack type	Complexity		Source
		Data	Time	
7.5	Opt. exhaustive search	2 KP	$2^{126.7}$	Sect. 6.2
7.5	Biclique BD-relation	2^{18} CP	$2^{126.5}$	[21]
7.5	Biclique BD-relation	2^{52} CP	$2^{123.9}$	[21]
7.5	SaC MitM BD-relation	16 CP	$2^{125.9}$	Sect. 6.1
8.5	Opt. exhaustive search	2 KP	$2^{127.1}$	Sect. 6.2
8.5	Biclique BD-relation	2^{52} CP	$2^{126.0}$	[21]
8.5	SaC MitM BD-relation	16 CP	$2^{126.8}$	Sect. 6.1

KP/CP—Known/Chosen Plaintext. Time complexity is measured in encryptions.

7. Zero-in-the-Middle Biryukov–Demirci Attack on Reduced-Round Variants of IDEA

The keyless Biryukov–Demirci relation was used to attack reduced-round variants of IDEA in several previous papers [4,5,28]. All these papers used a technique that can be called "Zero-in-the-Middle" (ZitM BD attack), in which the adversary uses proper choice of plaintext/ciphertext pairs, in conjunction with additional differential-type techniques, in order to ensure that some terms of the BD-relation are canceled. While all the attacks presented in [4,5,28] are inferior to the MitM BD attacks presented in the previous sections, we show in this section that there are other scenarios in which the ZitM BD technique is more efficient than the MitM BD technique.

The first such scenario is practical-time attacks. All the MitM attacks presented in the previous sections have a completely non-practical-time complexity of beyond 2^{100} encryptions. Moreover, as we argue in Sect. 8, the MitM technique is not expected to produce practical-time attacks on reduced-round variants of IDEA with at least 2 rounds. In contrast, we show in Sect. 7.2 that the ZitM BD technique can be used to distinguish 2.5-round IDEA from a random permutation using only 2^{18} data and time. This is the first attack of practical complexity of a variant of IDEA with at least 2 rounds.

The second such scenario is related-key attacks. The MitM BD attacks presented in the previous sections cannot take advantage of the ability to ask for encryptions under related (but unknown) keys. On the other hand, we show in Sect. 7.3 that the ZitM BD technique can use related-key differentials to attack a 7.5-round variant of IDEA which starts at the first round, with data complexity of 2^{25} chosen plaintexts and time complexity of $2^{103.5}$ encryptions. This is the only attack on 7.5-round IDEA (in any model) with a practical data complexity and a non-marginal time complexity.

We begin this section with briefly describing the previous ZitM BD attacks presented in [4,5,28], which are inferior to the MitM BD attacks presented in this paper. This description spans Sect. 7.1. The new ZitM BD attacks on 2.5-round IDEA and on 7.5-round IDEA in the related-key model are presented in Sects. 7.2 and 7.3, respectively.

7.1. *Previous Zero-in-the-Middle Keyless Biryukov–Demirci Attacks*

The Zero-in-the-Middle Biryukov–Demirci attack was used in several papers to attack 5-round, 5.5-round, and 6-round variants of IDEA:

1. *Differential BD attack on 5 rounds:* The first attack that exploited the keyless BD-relation is [4]. In the attack, the reduced-round variant starts after the *KA* layer of round 3 and ends after the *KA* layer of round 8, and a differential property is used to cancel the term Δs^4 in the BD-relation. The data complexity of the attack is 2^{19} known plaintexts, and the time complexity is 2^{103} encryptions. In [5] it was shown that the data complexity can be reduced to 16 known plaintexts, at the expense of increasing the time complexity to 2^{114} encryptions, and a slightly improved variant of the attack of [4] which uses only $2^{18.5}$ known plaintexts was presented.
2. *Square BD attack on 5.5 and 6 rounds:* The second attack that exploited the keyless BD-relation in larger versions of IDEA appeared in [5]. In this attack, the

reduced-round variant starts either after the *KA* layer of round 2 or at the beginning of round 3 and ends after the *KA* layer of round 8, and a Square property is used to cancel the terms Δs^3 and Δs^4 in the BD-relation. The data complexity of the attack on 6-round IDEA is almost the entire codebook, and the time complexity is $2^{126.8}$ encryptions.

3. *Key-Dependent Differential BD attack on 5.5 and 6 rounds:* The third attack that exploited the keyless BD-relation is [28]. The attack targets the same variant as [5] and uses a differential-type technique called *key-dependent attack* to cancel the terms Δs^3 and Δs^4 in the BD-relation (instead of the Square technique used in [5]). This allows to reduce the data and time complexities of the attack on 6-round IDEA to 2^{49} chosen plaintexts and $2^{112.1}$ encryptions, respectively.

All these attacks are clearly inferior to the MitM BD attack on 6-round IDEA presented in Sect. 4, whose data complexity is just 16 known plaintexts, and whose time complexity is less than 2^{112} encryptions.

7.2. *A Zero-in-the-Middle Biryukov–Demirci Distinguishing Attack on 2.5-Round IDEA*

In this section we present an extremely efficient distinguishing attack on 2.5-round IDEA, based on the Zero-in-the-Middle Biryukov–Demirci technique. The attack applies to any 2.5 consecutive rounds starting with the *KA* layer, and does not depend on any property of the IDEA key schedule. The time complexity of the attack is 2^{18}, which is significantly lower than the complexity of any previously published attack on IDEA (including attacks on 2 and 2.5 rounds).

For 2.5 rounds of IDEA, Eq. (6) is reduced to

$$LSB\big(P_2^1 \oplus P_3^1 \oplus P_2^2 \oplus P_3^2 \oplus \Delta s^1 \oplus \Delta s^2\big) = LSB\big(C_2^1 \oplus C_3^1 \oplus C_2^2 \oplus C_3^2\big). \quad (11)$$

Note that if for some round of IDEA, $\Delta p^r = 0$, then $\Delta s^r = 0$ as well. Hence, if the plaintexts and the ciphertexts are chosen such that $\Delta p^1 = \Delta p^2 = 0$, then the terms Δs^1 and Δs^2 in Eq. (11) are canceled, and the equation reduces to a simpler form:

$$LSB\big(P_2^1 \oplus P_3^1 \oplus P_2^2 \oplus P_3^2\big) = LSB\big(C_2^1 \oplus C_3^1 \oplus C_2^2 \oplus C_3^2\big), \quad (12)$$

whose validity can be checked using only the plaintexts and the ciphertexts, independently of the key.

In order to satisfy the relation $\Delta p^1 = 0$, we can consider pairs of chosen plaintexts (P^1, P^2) such that $\Delta(X_1^1, X_2^1, X_3^1, X_4^1) = (0, \beta, 0, \gamma)$ for arbitrary values of β and γ. For such pairs, $\Delta Y_1^1 = \Delta Y_3^1 = 0$ (independent of the values of Z_1^1, Z_3^1), and hence, $\Delta p^1 = 0$. We note that the same idea was used in [19].

Similarly, if we take only ciphertext pairs satisfying $\Delta(Y_1^3, Y_2^3, Y_3^3, Y_4^3) = (0, 0, \beta', \gamma')$ for arbitrary values of β' and γ', then $\Delta X_1^3 = \Delta X_2^3 = 0$, and thus, $\Delta p^2 = 0$.

Based on these observations, we can mount a simple distinguishing attack on 2.5-round IDEA, using the following algorithm:

1. Ask for the encryption of 2^{18} plaintexts of the form (A, Z, B, W), where A and B are fixed and Z and W assume arbitrary random values.
2. Insert the ciphertexts into a hash table sorted by the first two ciphertext words.

3. For every pair of ciphertexts in the same bin of the hash table, check whether Eq. (12) holds for the corresponding plaintext/ciphertext pair.
4. If there is a pair for which the equation does not hold, conclude that the cipher is not 2.5-round IDEA. Otherwise, conclude that the cipher is 2.5-round IDEA.

Due to the choice of the structure, for every pair of plaintexts in the structure we have $\Delta p^1 = 0$. Furthermore, for every pair of ciphertexts in the same bin of the hash table, we also have $\Delta p^2 = 0$. Hence, for all the checked pairs, Eq. (12) must be satisfied.

The 2^{18} plaintexts can be combined into about 2^{35} possible pairs, and a fraction of 2^{-32} of them is expected to have the required ciphertext difference of the form $(0, 0, \beta', \gamma')$. Hence, the expected number of pairs analyzed in Step 3 is 8. If there is a pair for which Eq. (12) fails, we know for sure that the cipher is not 2.5-round IDEA. On the other hand, for a random permutation, the probability that the equation holds for all the eight pairs is $1/256$. Hence, the distinguisher succeeds with a very high probability.

Since the second and the third steps of the attack are implemented using a hash table, the time complexity of the attack is dominated by the time complexity of the encryptions in the first step of the attack. Hence, the data complexity of the attack is 2^{18} chosen plaintexts and the time complexity is 2^{18} encryptions.

7.3. *Related-Key Zero-in-the-Middle Biryukov–Demirci Attack on 7.5-Round IDEA*

In this section we present a related-key attack on the first 7.5 rounds of IDEA based on the Zero-in-the-Middle Biryukov–Demirci technique. In the attack, we use the difference between the keys to construct pairs of plaintexts for which the intermediate values (when encrypted under the two different keys) are equal during 2.5 rounds. In conjunction with an appropriate choice of the plaintext/ciphertext pairs, the terms Δs^1, Δs^2, Δs^3, and Δs^4 in the keyless Biryukov–Demirci relation are canceled.

The Related-Key Differential Let K and K^* be two keys that differ only in the two bits 34 and 49. We observe that if for two plaintexts P and P^*, encrypted under K and K^*, respectively, the intermediate values of Y^2 (i.e., the values after the *KA* layer of round 2) are equal, then the intermediate encryption values remain equal until the *MA* layer of round 4. Indeed, bits 34 and 49 of the key are not used in the *MA* layer of round 2, in the entire round 3, and in the *KA* layer of round 4. Furthermore, these key bits are also not used in the subkey Z_5^4, and hence, the terms Δs^2, Δs^3, and Δs^4 in the BD-relation are equal to zero.

Therefore, for such pairs, Eq. (6) (for the first 7.5 rounds of IDEA) is reduced to

$$\begin{aligned} &LSB\big(P_2 \oplus P_3 \oplus P_2^* \oplus P_3^* \oplus \Delta s^1 \oplus \Delta s^5 \oplus \Delta s^6 \oplus \Delta s^7\big) \\ &\quad = LSB\big(C_2 \oplus C_3 \oplus C_2^* \oplus C_3^*\big). \end{aligned} \tag{13}$$

All terms of this equation can be computed given the plaintexts, the ciphertexts, and 103 key bits (specifically, bits 125–99 of the key). Hence, if the adversary can construct 25 pairs (P, P^*) for which the intermediate Y^2 values are equal, the attack can be completed within time complexity of about 2^{103} encryptions.

The Choice of the Plaintexts In order to obtain the required pairs (P, P^*) efficiently, we consider 2^8 pairs of structures (S_i, S_i^*) of 2^{16} chosen plaintexts each, to be encrypted under the keys K and K^*, respectively. In both structures S_i and S_i^*, the three first words

are fixed to constants (A_i, B_i, C_i) and (A_i^*, B_i^*, C_i^*), respectively, and the fourth word assumes all the 2^{16} possible values. The values $A_i, B_i, C_i, A_i^*, B_i^*, C_i^*$ are chosen such that

$$A_i = A_i^*; \qquad B_i \oplus B_i^* = 0040_x; \qquad C_i \oplus C_i^* = 2000_x.$$

Note that by the chosen key difference, there is no difference in the subkeys Z_1^1 and Z_2^1, and the difference in the subkey Z_3^1 is in the third-most significant bit (which is bit 34 of the secret key). Hence, the difference between the structures S_i and S_i^* in the first three words of the state Y^1 (i.e., after the *KA* layer of round 1) equals $(0, 0040_x, 0)$ with probability 2^{-2}.

In order to bypass the *MA* layer of round 1, we consider only pairs $(P_i \in S_i, P_i^* \in S_i^*)$ for which the difference in Y_4^1 is 0040_x. For each pair of structures (S_i, S_i^*) and for any value of the subkey Z_4^1, the pair of structures contains 2^{16} pairs (P_i, P_i^*) for which this condition is satisfied. Therefore, the data contains $2^8 \cdot 2^{-2} \cdot 2^{16} = 2^{22}$ pairs with difference $(0, 0040_x, 0, 0040_x)$ in the state Y^1.

Detection of the Right Pairs The right pairs, i.e., the pairs $(P_i \in S_i, P_i^* \in S_i^*)$ for which $\Delta Y^2 = 0$, are detected in a two-step procedure. First the adversary guesses the value of bits 0–63 of the key, encrypts all plaintexts through the *KA* layer of round 1 (under the corresponding keys), and chooses the 2^{22} pairs for which the difference ΔY^1 is $(0, 0040_x, 0, 0040_x)$. The time complexity of this step is less than $2^{25} \cdot 2^{64} = 2^{89}$ encryptions.

In the second step, the adversary guesses the value of bits 64–95 of the key, and for each of the 2^{22} remaining pairs, she checks whether $\Delta Y^2 = 0$.

Note that for each of the 2^{22} pairs, we have $\Delta X^2 = (0, 0, 0040_x, 0040_x)$. Since there is no difference in the subkeys Z_1^2 and Z_2^2, it is assured that $\Delta Y_1^2 = \Delta Y_2^2 = 0$, as required.[8]

In the third word, we have $\Delta X_3^2 = 0040_x$, and there is key difference in the seventh least significant bit (which is bit 34 of the secret key), and hence, $\Delta Y_3^2 = 0$ holds with probability $1/2$. In the fourth word, since the operation is modular multiplication and both the state difference and the subkey difference are non-zero, we make the randomness assumption that the values after the *KA* layer are equal with probability[9] 2^{-16}. Hence, the expected number of pairs satisfying $\Delta Y^2 = 0$ is $2^{22} \cdot 2^{-1} \cdot 2^{-16} = 32$.

The time complexity of detecting these pairs is $2^{64} \cdot 2^{32} \cdot 2^{22} = 2^{118}$ partial encryptions, which are roughly equivalent to 2^{115} full encryptions.

Checking Whether Eq. (13) Holds After the right pairs are detected, the adversary guesses 7 additional key bits (i.e., bits 96–99 and 125–127 of the key), and checks whether Eq. (13) holds. As this is a 32-bit filtering, only $2^{103} \cdot 2^{-32} = 2^{71}$ key suggestions are expected to remain, and these suggestions can be checked by guessing the remaining 25 key bits and performing a trial encryption.

[8] Note that it is important that this difference is fixed to zero independently of the subkeys Z_1^2 and Z_2^2, since these two subkeys use bits 96–127 of the secret key, and 25 of these 32 bits are not included in the 103 key bits guessed in the attack (which are bits 125–99).

[9] We have experimentally verified this claim, and we found that for all subkey pairs, this probability is at least 2^{-16}. Furthermore, our experiments revealed that for 31/32 of the subkey pairs, this probability is actually 2^{-15}. Thus, in most of the cases, the data complexity of the attack can be reduced by a factor of 2.

Checking whether Eq. (13) holds requires partial decryption of the ciphertexts through 2.5 rounds. (Note that there is no need to compute Δs^1, as for all the right pairs, $\Delta p^1 = 0$, and thus, $\Delta s^1 = 0$). Hence, a naive application of this step requires $2^{103} \cdot (32 \cdot 2) \cdot (2.5/8) = 2^{107.3}$ encryptions.

This step can be performed more efficiently by noting that half of the key guesses are discarded after considering the first right pair, half of the remaining key guesses are discarded after the second right pair, etc. Hence, instead of decrypting all the pairs at once, the adversary can decrypt the first pair and check whether the equation holds, then (if the key guess was not discarded) decrypt the second pair and check the equation for it, etc. Using this improvement, the time complexity of this step is $2^{104} + 2^{103} + 2^{102} + \cdots \approx 2^{105}$ partial decryptions, which are roughly equivalent to $2^{103.3}$ full encryptions.

However, the overall time complexity of the attack is dominated by the detection of the right pairs, whose complexity is about 2^{115} encryptions. In the next paragraph we present a more efficient algorithm that allows to detect the right pairs with time complexity of less than 2^{100} encryptions, thus reducing the overall complexity of the attack to about $2^{103.5}$ encryptions.

An Efficient Algorithm for Detecting the Right Pairs As shown above, the first step in the detection of right pairs, which consists of guessing bits 0–63 of the key and detecting 2^{22} pairs with difference $\Delta Y^1 = (0, 0040_x, 0, 0040_x)$, requires less than 2^{89} encryptions. We thus concentrate on the second step that consists of guessing bits 64–95 of the key and checking, for each of the 2^{22} pairs, whether $\Delta Y^2 = 0$.

Consider the modular multiplication with the subkey Z_4^2 in the *KA* layer of round 2. We observe that for all 2^{22} pairs, the difference before this multiplication is $\Delta X_4^2 = 0040_x$, and for the right pairs, the difference after the multiplication is $\Delta Y_4^2 = 0$. In addition, the subkey Z_4^2 consists of bits 41–55 of the key, and thus is included in bits 0–63 that are guessed during the first step of the right pairs detection.

Hence, the adversary can go over all 2^{16} pairs of 16-bit values with difference 0040_x, multiply them by the known value of Z_4^2 and find those pairs for which the difference after the multiplication is zero. For each guess of Z_4^2, one or two pairs with difference 0040_x lead after the subkey multiplication to zero difference, and thus, the adversary can compute the *actual values* (X_4^2, X_4^{*2}) which a pair must have in order to be a right pair. The time complexity of this computation is less than $2^{64} \cdot 2^{16} = 2^{80}$ encryptions.

After the adversary computes the "required" (X_4^2, X_4^{*2}) values, she guesses bits 64–79 of the key (i.e., the subkey Z_5^1), and partially encrypts the 2^{22} pairs through the *MA* layer of round 1. Then, for each pair, she assumes that indeed it is a right pair, and using the required values of (X_4^2, X_4^{*2}) on the one hand and u^1, s^1, q^1 (that can be computed from the partial encryption and the required values (X_4^2, X_4^{*2})) on the other hand, she computes the input and the output of the modular multiplication with the subkey Z_6^1.

This gives the adversary an equation of the form $a \odot Z_6^1 = b$, where a, b are known. Since the modular multiplication is performed in a field, the adversary can invert the equation and get the value of Z_6^1 with only a few operations. (For example, she can store the inverses of all elements in the field in a table of size 2^{16}, and perform a single table lookup and a single modular multiplication to compute $Z_6^1 = a^{-1} \odot b$). Hence, for

each of the 2^{22} pairs, the adversary can find the value of Z_6^1 for which that pair is a right pair.

Finally, the adversary inserts the tuples (P_1, P_2, Z_6^1) into a hash table sorted according to the value of Z_6^1, and then for each value of Z_6^1, she can get the 32 right pairs with respect to that key by a single table lookup. The time complexity of this step is $2^{64} \cdot 2^{16} \cdot 2^{22} = 2^{102}$ simple computations, which are less than 2^{100} encryptions.

Summary Using this improved algorithm, the time complexity of the attack is reduced to less than $2^{103.5}$ encryptions. The data complexity of the attack is 2^{25} chosen plaintexts, and the memory complexity is 2^{22} 32-bit blocks, or equivalently, 2^{21} 64-bit blocks.

A Known Plaintext Variant of the Attack We note that a similar attack can be performed in the known-plaintext model. In the attack, the adversary considers two structures of 2^{43} known plaintexts encrypted under the keys K and K^*, and for each guess of bits 0–63 of the key, she inserts the plaintexts into a hash table and detects the 2^{22} pairs (P, P^*) for which $\Delta Y^1 = (0, 0040_x, 0, 0040_x)$. The rest of the attack is the same as the chosen-plaintext attack described above. Since the first step can be performed efficiently, the overall time complexity of the attack is the same as that of the chosen plaintext attack. The memory complexity is increased to 2^{43} 64-bit blocks.

8. Discussion and Open Problems

In this paper, we presented the keyless Biryukov–Demirci relation and combined it with Meet-in-the-Middle type techniques to devise new attacks on up to 7.5-round IDEA, whose complexities are significantly lower than that of exhaustive search. For up to 6.5 rounds, the data complexities of the attacks are practical. All these results are major improvements over previously published attacks, which could handle at most 6 rounds using impractical amounts of chosen plaintexts. In the stronger model of related-key attacks, we could attack up to 7.5 rounds with a practical data complexity.

The two major techniques we used in this paper are Meet-in-the-Middle Biryukov–Demirci (MitM BD) and Zero-in-the-Middle Biryukov–Demirci (ZitM BD) attacks. In general, the MitM BD technique yields better attacks in terms of the number of rounds that can be attacked, but there are scenarios in which the ZitM BD technique yields better results. It seems that such scenarios are of two types:

1. *Low time complexity attacks:* The MitM BD attack inevitably requires a large time complexity, since computing even a single Δs^r value requires to guess at least 48 key bits (subkeys Z_1^r, Z_3^r, and Z_5^r in the encryption direction, or subkeys Z_1^{r+1}, Z_2^{r+1}, and Z_5^r in the decryption direction). Hence, it appears that any MitM BD attack would have time complexity of at least 2^{48}. In contrast, there is no lower bound on the complexity of a ZitM BD attack, since the adversary can choose the plaintexts such that some Δs^r terms are canceled, independently of the key. This is demonstrated in the case of 2.5-round IDEA, where the ZitM BD technique allows to mount a distinguishing attack with an extremely low time complexity of 2^{18}.

2. *Low data complexity attacks on a large number of rounds:* Due to the key schedule of IDEA, the computation of any four consecutive Δs^r values requires knowledge of the entire secret key. Hence, if the number of Δs^r terms in the BD equation is greater than 6, the equation is not vulnerable to the MitM BD attack. This obstacle can be overruled by using the splice-and-cut technique (like in our 7.5-round attack), but only at the price of a higher data complexity. In contrast, there may be special scenarios, such as the related-key model, in which a special choice of plaintexts allows to cancel more than three consecutive Δs^r values. This is demonstrated in the case of 7.5-round IDEA, where the BD-relation contains seven terms of the form Δs^r, but a special choice of plaintexts according to a related-key differential allows to cancel four consecutive Δs^r terms.

Summarizing, it seems that the MitM BD technique is better in the "usual" scenarios, where the required complexity of the attack is not "too low". However, in specific scenarios, and especially in the related-key scenario, the ZitM BD attack can perform better. It would be nice to combine these two techniques into a unified framework.

The main open problem left in this paper is to find a "real" attack on the full 8.5-round IDEA, whose running time is considerably faster than the 2^{128} complexity of exhaustive search. In our opinion, the $2^{126.8}$ attack we described in this paper, and even the $2^{126.0}$ attack presented in [21], are too marginal to justify a claim that the full IDEA is (even academically) broken, and we encourage other researchers to try to improve them.

Acknowledgements

The authors thank Willi Meier and the anonymous referees for their constructive and helpful comments.

Appendix A. A Simple Meet-in-the-Middle Attack on 4.5-Round IDEA

In this appendix we present a simple MitM attack on a 4.5-round variant of IDEA starting at the beginning of round 4, which allows to recover the full key using only 2 known plaintexts, 2^{25} memory and 2^{103} operations. Note that this is the most data-efficient attack possible, since the unicity distance of a 64-bit block cipher with a 128-bit key is 2. We were surprised by the fact that this simple attack breaks more rounds of IDEA with so little data compared with numerous previously published sophisticated attacks, including the differential [23], differential-linear [9], Square [19], and impossible differential [3][10] attacks.

Consider a 4.5-round variant of IDEA which starts at the beginning of round 4 and ends after the *KA* layer of round 8. The basic observation behind the attack is that the value $V = p^5$ can be computed from the "plaintext" (i.e., the input of round 4) given only bits 75–49 of the key, and from the "ciphertext" (i.e., the value after *KA* of round 8) given only bits 125–99 of the key. This allows us to apply the standard MitM attack described in Sect. 3.1. The attack algorithm is described in Fig. A.1.

[10] The impossible differential attack of [3] can break the same number of rounds, but with a significantly higher data, memory, and time complexities.

Input: Two "plaintext"/"ciphertext" pairs (P_1, C_1), (P_2, C_2).
for any key guess of key bits 0–49, 75–99, 125–127 **do**
  Initialize an empty hash table H.
  for any guess of key bits 100–124 **do**
    Compute p_1^5 and p_2^5 from P_1 and P_2, respectively, and store in H the value $(p_1^5, p_2^5, K[100\text{–}124])$.
  end for
  for any guess of key bits 50–74 **do**
    Compute $p_1'^5$ and $p_2'^5$ from C_1 and C_2, respectively, and check whether $(p_1'^5, p_2'^5)$ is in H.
    If so, perform trial encryption under the key bits 0–49, 75–99, 125–127, the current guess of key bits 50–74, and the guess of bits 100–124 suggested in the corresponding entry of H.
  end for
end for

Fig. A.1. The algorithm of a meet-in-the-middle attack on 4.5-round IDEA (starting at round 4).

Note that each match in the hash table H suggests a value for the entire key (which is then checked by trial encryption), and that the correct key must be suggested by one of the matches. Since the total number of matches is $2^{128} \cdot 2^{-32} = 2^{96}$, all wrong key suggestions are filtered after two trial encryptions, and only the correct key remains.

In order to evaluate the complexity of the attack, we compute $|K_t|, |K_b|, |K_c|, |V|$ (see Sect. 3.1). Since K_t consists of bits 75–49 and K_b consists of bits 125–99, we have $|K_t| = |K_b| = 103$, and $|K_c| = 78$. Also, $|V| = 16$, since $V = p^5$ is a 16-bit value. Hence, the data complexity is $\lceil \frac{\min(|K_t|,|K_b|)-|K_c|}{|V|} \rceil = 2$ known plaintexts, and the memory complexity is $2^{\min(|K_t|,|K_b|)-|K_c|} = 2^{25}$ 64-bit blocks. The time complexity of the attack is $2^{\max(|K_t|,|K_b|)} = 2^{103}$ partial encryptions of two plaintexts, which are less than 2^{103} encryptions.

For the sake of completeness, we considered all reduced-round variants of IDEA consisting of 4.5 consecutive rounds. We found two other 4.5-round variants that can be attacked using the standard MitM technique:

- A variant that starts after the *KA* layer of round 2 and ends at the end of round 6—the complexity of the attack on this variant is identical to the complexity of the attack described above.
- A variant that starts after the *KA* layer of round 1 and ends at the end of round 5—the time complexity of the attack on this variant is increased to 2^{112} encryptions, whereas the memory complexity is decreased to 2^{15} 64-bit blocks.

References

[1] K. Aoki, Y. Sasaki, Preimage attacks on one-block MD4, 63-step MD5 and more, in *Proceedings of Selected Areas in Cryptography 2008*. Lecture Notes in Computer Science, vol. 5381 (Springer, Berlin, 2009), pp. 103–119

[2] E.S. Ayaz, A.A. Selçuk, Improved DST cryptanalysis of IDEA, in *Proceedings of Selected Areas in Cryptography 2006*. Lecture Notes in Computer Science, vol. 4356 (Springer, Berlin, 2007), pp. 1–14
[3] E. Biham, A. Biryukov, A. Shamir, Miss in the middle attacks on IDEA and Khufu, in *Proceedings of Fast Software Encryption 1999*. Lecture Notes in Computer Science, vol. 1636 (Springer, Berlin, 1999), pp. 124–138
[4] E. Biham, O. Dunkelman, N. Keller, New cryptanalytic results on IDEA, in *Advances in Cryptology, Proceedings of ASIACRYPT 2006*. Lecture Notes in Computer Science, vol. 4284 (2006), pp. 412–427
[5] E. Biham, O. Dunkelman, N. Keller, A new attack on 6-round IDEA, in *Proceedings of Fast Software Encryption 2007*. Lecture Notes in Computer Science, vol. 4593 (Springer, Berlin, 2007), pp. 211–224
[6] A. Biryukov, J. Nakahara Jr., B. Preneel, J. Vandewalle, New weak-key classes of IDEA, in *Proceedings of Information and Communications Security 2002*. Lecture Notes in Computer Science, vol. 2513 (Springer, Berlin, 2002), pp. 315–326
[7] A. Biryukov, D. Khovratovich, Related-key cryptanalysis of the full AES-192 and AES-256, in *Advances in Cryptology, Proceedings of ASIACRYPT 2009*. Lecture Notes in Computer Science, vol. 5912 (Springer, Berlin, 2009), pp. 1–18
[8] N. Borisov, M. Chew, R. Johnson, D. Wagner, Multiplicative differentials, in *Proceedings of Fast Software Encryption 2002*. Lecture Notes in Computer Science, vol. 2365 (Springer, Berlin, 2002), pp. 17–33
[9] J. Borst, L.R. Knudsen, V. Rijmen, Two attacks on reduced round IDEA, in *Advances in Cryptology, Proceedings of EUROCRYPT 1997*. Lecture Notes in Computer Science, vol. 1233 (Springer, Berlin, 1997), pp. 1–13
[10] D. Chaum, J.-H. Evertse, Cryptanalysis of DES with a reduced number of rounds: sequences of linear factors in block ciphers, in *Advances in Cryptology, Proceedings of CRYPTO 1985*. Lecture Notes in Computer Science, vol. 218 (Springer, Berlin, 1986), pp. 192–211
[11] J. Daemen, R. Govaerts, J. Vandewalle, Cryptanalysis of 2.5 rounds of IDEA (Extended Abstract). Technical report 93/1, Department of Electrical Engineering, ESAT–COSIC, KU Leuven, Belgium (1993)
[12] J. Daemen, R. Govaerts, J. Vandewalle, Weak keys for IDEA, in *Advances in Cryptology, Proceedings of CRYPTO 1993*. Lecture Notes in Computer Science, vol. 773 (Springer, Berlin, 1994), pp. 224–231
[13] H. Demirci, Square-like attacks on reduced rounds of IDEA, in *Proceedings of Selected Areas in Cryptography 2002*. Lecture Notes in Computer Science, vol. 2595 (Springer, Berlin, 2003), pp. 147–159
[14] H. Demirci, A.A. Selçuk, E. Türe, A new meet-in-the-middle attack on the IDEA block cipher, in *Proceedings of Selected Areas in Cryptography 2003*. Lecture Notes in Computer Science, vol. 3006 (Springer, Berlin, 2004), pp. 117–129
[15] W. Diffie, M.E. Hellman, Exhaustive cryptanalysis of the NBS data encryption standard. *Computer* **10**(6), 74–84 (1977)
[16] Electronic Frontier Foundations, *Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design* (O'Reilly, Sebastopol, 1998)
[17] P. Hawkes, Differential-linear weak keys classes of IDEA, in *Advances in Cryptology, Proceedings if EUROCRYPT 1998*. Lecture Notes in Computer Science, vol. 1403 (Springer, Berlin, 1998), pp. 112–126
[18] P. Hawkes, L. O'Connor, On applying linear cryptanalysis to IDEA, in *Advances in Cryptology, Proceedings of ASIACRYPT 1996*. Lecture Notes in Computer Science, vol. 1163 (Springer, Berlin, 1996), pp. 105–115
[19] P. Junod, New attacks against reduced-round versions of IDEA, in *Proceedings of Fast Software Encryption 2005*. Lecture Notes in Computer Science, vol. 3557 (Springer, Berlin, 2005), pp. 384–397
[20] J. Kelsey, B. Schneier, D. Wagner, Key-schedule cryptoanalysis of IDEA, G-DES, GOST, SAFER, and triple-DES, in *Advances in Cryptology, Proceedings of CRYPTO 1996*. Lecture Notes in Computer Science, vol. 1109 (Springer, Berlin, 1996), pp. 237–251
[21] D. Khovratovich, G. Leurent, C. Rechberger, Narrow-bicliques: cryptanalysis of Full IDEA, in *Advances in Cryptology, Proceedings of EUROCRYPT 2012*. Lecture Notes in Computer Science, vol. 7237 (Springer, Berlin, 2012), pp. 392–410
[22] X. Lai, J.L. Massey, S. Murphy, Markov ciphers and differential cryptanalysis, in *Advances in Cryptology, Proceedings of EUROCRYPT 1991*. Lecture Notes in Computer Science, vol. 547 (Springer, Berlin, 1992), pp. 17–38
[23] W. Meier, On the security of the IDEA block cipher, in *Advances in Cryptology, Proceedings of EUROCRYPT 1993*. Lecture Notes in Computer Science, vol. 765 (Springer, Berlin, 1994), pp. 371–385

[24] R.C. Merkle, M.E. Hellman, On the security of multiple encryption. *Commun. ACM* **24**(7), 465–467 (1981)

[25] J. Nakahara Jr., P.S.L.M. Barreto, B. Preneel, J. Vandewalle, H.Y. Kim, SQUARE Attacks Against Reduced-Round PES and IDEA Block Ciphers, IACR Cryptology ePrint Archive, Report 2001/068 (2001)

[26] J. Nakahara Jr., B. Preneel, J. Vandewalle, The Biryukov–Demirci attack on reduced-round versions of IDEA and MESH ciphers, in *Proceedings of Australasian Conference on Information Security and Privacy 2004*. Lecture Notes in Computer Science, vol. 3108 (Springer, Berlin, 2004), pp. 98–109

[27] H. Raddum, Cryptanalysis of IDEA-X/2, in *Proceedings of Fast Software Encryption 2003*. Lecture Notes in Computer Science, vol. 2887 (Springer, Berlin, 2003), pp. 1–8

[28] X. Sun, X. Lai, The key-dependent attack on block ciphers, in *Advances in Cryptology, Proceedings of ASIACRYPT 2009*. Lecture Notes in Computer Science, vol. 5912 (2009), pp. 19–36

[29] L. Wei, C. Rechberger, J. Guo, H. Wu, H. Wang, S. Ling, Improved meet-in-the-middle cryptanalysis of KTANTAN, in *Proceedings of Australasian Conference on Information Security and Privacy 2011*. Lecture Notes in Computer Science, vol. 6812 (Springer, Berlin, 2011), pp. 433–438. Full version available at: IACR Cryptology ePrint Archive, Report 2011/201 (2011)