Real Time Related Key Attack on Hummingbird-2

Kai Zhang, Lin Ding, Junzhi Li and Jie Guan
Zhengzhou Information Science and Technology Institute
Zhengzhou 450000, China
[e-mail: zhkai2010@139.com, dinglin_cipher@163.com, ljunhi@163.com, guanjie007@163.com]
*Corresponding author: Jie Guan

Received May 2, 2012; revised June 26, 2012; accepted July 24, 2012; published August 30, 2012

Abstract

Hummingbird is a lightweight encryption and message authentication primitive published in RISC'09 and WLC'10. In FSE'11, Markku-Juhani O.Saarinen presented a differential divide-and-conquer method which has complexity upper bounded by 2^{64} operations and requires processing of few megabytes of chosen messages under two related nonces (IVs). The improved version, Hummingbird-2, was presented in RFIDSec 2011. Based on the idea of differential collision, this paper discovers some weaknesses of the round function WD16. Combining with the simple key loading algorithm, a related-key chosen-IV attack which can recover the full secret key is proposed. Under 15 pairs of related keys, the 128 bit initial key can be recovered, requiring 2^{27} chosen IV and the computational complexity is $O(2^{27})$. In average, the attack needs several minutes to recover the full 128-bit secret key on a PC. The experimental result corroborates our attack. The result shows that the Hummingbird-2 cipher can't resist related key attack.

Key Words: Cryptanalysis; Hummingbird-2; related key attack; lightweight cipher; hybrid symmetric cipher

[*] A preliminary version of this paper appeared in eprint archive which is available at http://eprint.iacr.org/2012/207. The authors would like to thank the anonymous reviewers and Dr. Markku-Juhani O. Saarinen for helpful comments. This work was supported by the National Natural Science Foundation of China under Grant No.61202491, No.61272041 and No.61272488.

http://dx.doi.org/10.3837/tiis.2012.08.004

1. Introduction

Symmetric encryption algorithms are traditionally categorized into two types of schemes: block ciphers and stream ciphers. Stream ciphers distinguish themselves from block ciphers by the fact that they process plaintext symbols (typically bits) as soon as they arrive by applying a very simple but ever changing invertible transformation, it's based on the idea of "*One Time Pad Assumption*". As for block ciphers, their security are from the complexity of the encryption transformation, it's based on the theory of "*Confusion and Diffusion*". Nowadays, people try to combine the stream ciphers and the block ciphers together to make safer ciphers, such as *CSA*[12], Hummingbird family ciphers[1][2] etc.

Hummingbird-1 is a recent cryptographic algorithm proposal for RFID tags and other constrained devices. It is covered by several pending patents and is being commercially marketed by the Revere Security. Revere has invested into Hummingbird's cryptographic security assurance before its publication by contracting ISSI, a private consultancy employing some ex-NSA staff and members of U.Waterloo CACR. In FSE 2011, Markku-Juhani O. Saarinen proposed a differential divide-and-conquer method which has complexity upper bounded by 2^{64} operations and requires processing of few megabytes of chosen messages under two related nonces(*IV*s). In RFIDSec 2011, the improved version, Hummingbird-2, was presented. It is also an encryption and message authentication primitive that has been designed particularly for resource-constrained devices such as RFID tags, wireless sensors, smart meters and industrial controllers. In 2011, Xinxin Fan and Guang Gong proposed a side channel cube attack[14] against Hummingbird-2 which can recover the first 48 bit initial key for the data complexity of $O(2^{18})$. Recently, using two pairs of related keys, Qi Chai and Guang Gong proposed a probabilistic attack[11]encompassing two phases, the preparation phase and the key recovery phase. In the preparation phase, the attack requires 2^{80} effort in time, aims to reach an internal state, with 0.5 success probability which satisfies particular conditions. In the key recovery phase, using the proposed differential sequence analysis it is able to recover 36 bits of the 128-bit key. The rest 48 bits of the key can be exhaustively searched and the overall time complexity of the key recovery phase is $2^{49.63}$.

Related key cryptanalysis is first introduced by Biham and independently by Knudsen in 1993[3][9]it is a type of chosen-key attack, in which the relationship between the keys used is known. People try to get the information of the initial key by analyzing the ciphertexts under certain related keys. Combined with differential attack, Kelsey proposed Related Key differential cryptanalysis in [8], and it is also combined with other attacks such as impossible differential attack and high order differential attack.

In the specification of Hummingbird-2, the authors referred to a related key differential characteristic, but didn't make an attack. In the present report we show that the published version of Hummingbird-2 is susceptible to a related-key chosen-*IV* attack that under 15 pairs of related keys, the 128 bit initial key can be recovered with the computational complexity of $O(2^{27})$ and 2^{27} chosen *IV*s. When compared with the attack[11] proposed by Qi Chai, to succeed with probability 1, the preparation phase of their attack requires more effort in time than the exhaustive search, while our attack works on Hummingbird-2 cipher with manageably low data complexity and time complexity, and our attack can succeed with probability of almost 1. But the related keys used in their attack model are less than ours.

This paper is organized as follows. In Section 2 we give a description of Hummingbird-2. In Section 3 we present a key observation about the initialization and encryption procedure of algorithm, then we propose an attack which recovers the full secret key, moreover we use a new strategy to improve our attack, followed by conclusion in Section 4.

2. Description of Hummingbird-2

The Hummingbird-2 cipher has a 128-bit secret key K and a 128-bit internal state R which is initialized using a 64-bit Initialization Vector (i.e. *IV*). The key, registers and *IV* are denoted as follows:

$$K = (K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_8)$$
$$R = (R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8)$$
$$IV = (IV_1, IV_2, IV_3, IV_4)$$

Let $S(x)$ denotes the computation of four S-Boxes and $L(x)$ the linear transformation which is expressed using the left circular shift (rotation) operator ("<<<"). We may write the nonlinear function $f(x)$ and $WD16(x,a,b,c,d)$ as

$$x = (x_0, x_1, x_2, x_3)$$
$$S(x) = S_1(x_0) \mid S_2(x_1) \mid S_3(x_2) \mid S_4(x_3)$$
$$L(x) = x \oplus (x <<< 6) \oplus (x <<< 10)$$
$$f(x) = L(S(x))$$
$$WD16(x,a,b,c,d) = f(f(f(f(x \oplus a) \oplus b) \oplus c) \oplus d)$$

The *S*-Boxes S_1,S_2,S_3 and S_4 are given in **Table 1** below.

Table 1. The *S*-Boxes of Hummingbird-2

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S_1(x)$	7	c	e	9	2	1	5	f	b	6	d	0	4	8	a	3
$S_2(x)$	4	a	1	6	8	f	7	c	3	0	e	d	5	9	b	2
$S_3(x)$	2	f	c	1	5	6	a	d	e	8	3	4	0	b	9	7
$S_4(x)$	f	4	5	8	9	7	2	1	a	3	0	e	6	c	d	b

(1) The Initialization Process

First of all, the initial state of the registers denoted as $R^{(0)}$ is filled with *IV* as follows:

$$R^{(0)}=(R_1^{(0)},R_2^{(0)},R_3^{(0)},R_4^{(0)},R_5^{(0)},R_6^{(0)},R_7^{(0)},R_8^{(0)})=(IV_1, IV_2, IV_3, IV_4, IV_1, IV_2, IV_3, IV_4)$$

Then iterate for i=0,1,2,3 as follows ("(" represents "addition module 2^{16}"):

$$t_1 = WD16(R_1^{(i)} \boxplus <i>, K_1, K_2, K_3, K_4)$$
$$t_2 = WD16(R_2^{(i)} \boxplus t_1, K_5, K_6, K_7, K_8)$$
$$t_3 = WD16(R_3^{(i)} \boxplus t_2, K_1, K_2, K_3, K_4)$$
$$t_4 = WD16(R_4^{(i)} \boxplus t_3, K_5, K_6, K_7, K_8)$$
$$R_1^{(i+1)} = (R_1^{(i)} \boxplus t_4) \lll$$
$$R_2^{(i+1)} = (R_2^{(i)} \boxplus t_1) \ggg$$
$$R_3^{(i+1)} = (R_3^{(i)} \boxplus t_2) \lll$$
$$R_4^{(i+1)} = (R_4^{(i)} \boxplus t_3) \lll$$
$$R_5^{(i+1)} = R_5^{(i)} \oplus R_1^{(i+1)}$$
$$R_6^{(i+1)} = R_6^{(i)} \oplus R_2^{(i+1)}$$
$$R_7^{(i+1)} = R_7^{(i)} \oplus R_3^{(i+1)}$$
$$R_8^{(i+1)} = R_8^{(i)} \oplus R_4^{(i+1)}$$

The initial state of registers for encrypting the first plaintext word is $R^{(4)}$.

(2) The Encryption Process

The encryption of the ith plaintext P_i to C_i needs four iteration of $WD16$ as follows:

$$t_1 = WD16(R_1^{(i)} \boxplus P_i, K_1, K_2, K_3, K_4)$$
$$t_2 = WD16(R_2^{(i)} \boxplus t_1, K_5 \oplus R_5^{(i)}, K_6 \oplus R_6^{(i)}, K_7 \oplus R_7^{(i)}, K_8 \oplus R_8^{(i)})$$
$$t_3 = WD16(R_3^{(i)} \boxplus t_2, K_1 \oplus R_5^{(i)}, K_2 \oplus R_6^{(i)}, K_3 \oplus R_7^{(i)}, K_4 \oplus R_8^{(i)})$$
$$C_i = WD16(R_4^{(i)} \boxplus t_3, K_5, K_6, K_7, K_8) \boxplus R_1^{(i)}$$

The registers R_1 to R_8 are refreshed as follows:

$$R_1^{(i+1)} = R_1^{(i)} \boxplus t_3$$
$$R_2^{(i+1)} = R_2^{(i)} \boxplus t_1$$

$$R_3^{(i+1)} = R_3^{(i)} \boxplus t_2$$
$$R_4^{(i+1)} = R_4^{(i)} \boxplus R_1^{(i)} \boxplus t_3 \boxplus t_1$$
$$R_5^{(i+1)} = R_5^{(i)} \oplus (R_1^{(i)} \boxplus t_3)$$
$$R_6^{(i+1)} = R_6^{(i)} \oplus (R_2^{(i)} \boxplus t_1)$$
$$R_7^{(i+1)} = R_7^{(i)} \oplus (R_3^{(i)} \boxplus t_2)$$
$$R_8^{(i+1)} = R_8^{(i)} \oplus (R_4^{(i)} \boxplus R_1^{(i)} \boxplus t_3 \boxplus t_1)$$

3. Cryptanalysis of Hummingbird-2

In this section, we introduce a key recovery attack on Hummingbird-2. Here is the clue: Firstly, we introduce some differential characteristics of the four *S* boxes for Hummingbird-2, then some differential collision characteristics of the nonlinear function *WD*16 are presented. Secondly, we obtain a series of differential characteristics based on the thought of related key attack and some differential collisions occur with high probability through the initialization and the encryption process of the algorithm. Finally, we proposed a differential-related key attack on Hummingbird-2 which can recover the full key in real time.

The attack process is as follows: First of all, we use proper related keys which can induce partial differential and can be counteracted with high probability, the differentials can be limited to the inside of the nonlinear function *WD*16. Then we detect whether the differential pairs we built has occurred by examining the differential of the ciphertexts. If it occurs, we can use the differential cryptanalysis techniques to recover the key.

3.1 Differential Properties of S-Boxes on Hummingbird-2

First of all, we introduce some concepts of differential cryptanalysis.

Definition 1[4] A differential of a function $f: F_2^n \to F_2^n$ is a pair $(\alpha, \beta) \in F_2^n \times F_2^n$ such that $f(x+\alpha)=f(x)+\beta$ for some $x \in F_2^n$. We call α the input differential and β the output differential. The differential probability $p_f(\alpha \to \beta)$ of a differential (α, β) with respect to $f(x)$ is defined as

$$p_f(\alpha \to \beta) = p\{(x_1, x_2) \in F_2^n \times F_2^n : f(x_1) - f(x_2) = \beta \mid x_1 - x_2 = \alpha\}$$

Through analyzing the four *S*-Boxes of Hummingbird-2, we study the distribution of the probability of differentials, and get various differential pairs with different differential probability. As for our attack, we only use the highest differential probability which is 1/4 actually for all of the four *S*-Boxes, so we only illustrate these differential pairs in **Table**

2.(In Table 2, $\alpha \to \beta$ represents the input differential and output differential respectively.)

Table 2. Highest probability differential pairs of the four S-Boxes for Hummingbird-2

S box	Highest probability differential pairs
S_1	$1\to d, 2\to 6, 2\to e, 3\to 2, 3\to b, 5\to e, 6\to 8, 7\to 8, 8\to 9, 8\to c, 9\to 5, b\to 1, b\to b, c\to 4, e\to 1, e\to f, f\to 4, f\to 7$
S_2	$1\to 3, 1\to 7, 2\to d, 3\to 2, 3\to e, 4\to 5, 4\to 6, 6\to 9, 7\to 8, 7\to e, a\to 2, b\to 4, b\to 9, c\to 1, d\to d, e\to 4, e\to f, f\to 1$
S_3	$1\to 7, 1\to d, 2\to c, 2\to e, 3\to 3, 4\to 3, 5\to 4, 6\to 7, 6\to f, 7\to 4, 8\to 5, a\to 1, b\to f, c\to 9, d\to 8, d\to e, f\to 1, f\to 5$
S_4	$1\to e, 2\to a, 2\to b, 3\to 1, 7\to 1, 7\to e, 8\to 5, 8\to f, 9\to c, a\to 4, a\to f, b\to 2, c\to 3, c\to 8, e\to 2, e\to 9, f\to 7, f\to 9$

Then, let us introduce a property of these S-Boxes which will be used later when we choose the related keys to recover the initial key.

Property 1 For any one of the S-Boxes of Hummingbird-2 denoted as S_i (i=1,2,3,4), there must exist at least one element x, which as the input of S_i satisfies two different differential characteristics $\alpha_1 \to \beta_1$, $\alpha_2 \to \beta_2$ at the same time, and x is the only input satisfies these two differential characteristics.

Table 3. Input sets for different differential characteristics of the four S-Boxes

S-Box	Differential Characteristic	Input Set	S-Box	Differential Characteristic	Input Set
S_1	$(1,d)(2,6)(3,b)$	$8,9,a,b$	S_3	$(1,7)(c,9)(d,e)$	$6,7,a,b$
	$(2,e)(9,5)(b,b)$	$5,7,c,e$		$(1,d)(2,e)(3,3)$	$0,1,2,3$
	$(3,2)(c,4)(f,6)$	$1,2,d,e$		$(2,c)(4,3)(6,f)$	$9,b,d,f$
	$(5,e)(b,1)(e,f)$	$1,4,a,f$		$(5,4)(a,1)(f,5)$	$0,5,a,f$
	$(6,8)(8,9)(e,1)$	$3,5,b,d$		$(6,7)(b,f)(d,8)$	$3,5,8,e$
	$(7,8)(8,c)(f,4)$	$0,7,8,f$		$(7,4)(8,5)(f,1)$	$3,4,b,c$
S_2	$(1,3)(2,d)(3,e)$	$8,9,a,b$	S_4	$(1,e)(e,9)(f,7)$	$4,5,a,b$
	$(1,7)(6,9)(7,e)$	$2,3,4,5$		$(2,a)(8,5)(a,f)$	$0,2,8,a$
	$(3,2)(d,d)(e,f)$	$0,3,d,e$		$(2,b)(8,f)(a,4)$	$4,6,c,e$
	$(4,5)(b,4)(f,1)$	$1,5,a,e$		$(3,1)(c,8)(f,9)$	$1,2,d,e$
	$(4,6)(a,2)(e,4)$	$2,6,8,c$		$(7,1)(b,2)(c,3)$	$3,4,8,f$
	$(7,8)(b,9)(c,1)$	$0,7,b,c$		$(7,e)(9,c)(e,2)$	$0,7,9,e$

As we only concern those differential characteristics which the differential probability is 1/4. First of all, we list all the possible input for different differential characteristics of the four S-Boxes which the differential probability is 1/4. (α,β) in Table 3 represents the differential characteristic $\alpha \to \beta$.

As for S_1, x=8 is the only element which satisfies differential characteristics (1,d) and (7,8), and x=8 is the only element for these two differential characteristics. Similarly, for S_1, x=a is the only input satisfies differential characteristics (1,d) and (5,e). x=c is the only input satisfies differential characteristics (4,6) and (7,8) for S_2 , x=0 is the only input satisfies differential characteristics (1,d) and (5,4) for S_3 and x=4 is the only input satisfies differential characteristics (1,e) and (2,b) for S_4 . It's easy to find that, for all the S-Boxes, there are plenty of x which satisfy property 1 . This property will be used later when we choose related keys.

3.2 Differential Collision Properties for Nonlinear Function *f* and *WD*16

Nonlinear function f and WD16 are the basic elements of Hummingbird-2. In this section, we will discuss the differential collision properties of nonlinear function f and WD16.

First of all, we illustrate the structure of the two functions as follow.

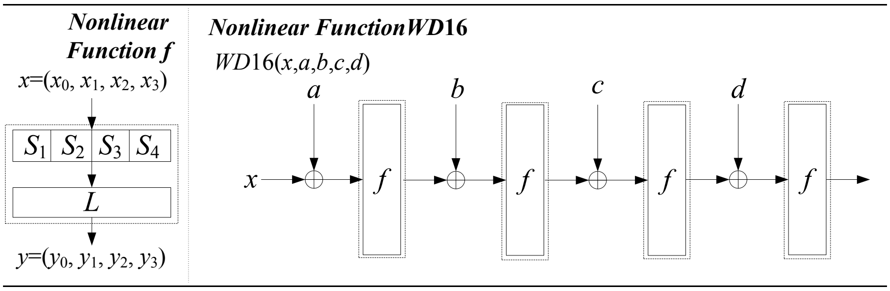

Fig. 1. Nonlinear Function f and WD16

The nonlinear function WD16 is a function with 80 bit input and 16 bit output. There must exist collisions during the encryption and decryption process, but whether these collisions satisfy certain structure character is still a problem to be further studied. Next we will use an example to show the existence of this character.

The round function WD16 can be viewed as a small "block cipher" and the structure of WD16 is the simplest SP structure. Denoted that the input differential of WD16 is $(\Delta a, \Delta b, 0, 0)$, when the equation (1) below satisfies, the output differential for WD16 is zero.

$$f(x \oplus a) \oplus b = f(x \oplus a \oplus \Delta a) \oplus b \oplus \Delta b \qquad (1)$$

The equation (1) is equal to equation (2) below:

$$S(x \oplus a) \oplus S(x \oplus a \oplus \Delta a) = L^{-1}(\Delta b) \qquad (2)$$

We can split the equation (2) as the following equations:

$$\begin{cases} S_1(x_0 \oplus a_0) \oplus S_1(x_0 \oplus a_0 \oplus \Delta a_0) = c_0 \\ S_2(x_1 \oplus a_1) \oplus S_2(x_1 \oplus a_1 \oplus \Delta a_1) = c_1 \\ S_3(x_2 \oplus a_2) \oplus S_3(x_2 \oplus a_2 \oplus \Delta a_2) = c_2 \\ S_4(x_3 \oplus a_3) \oplus S_4(x_3 \oplus a_3 \oplus \Delta a_3) = c_3 \end{cases} \tag{3}$$

Among which $a = (a_0 \| a_1 \| a_2 \| a_3), b = (b_0 \| b_1 \| b_2 \| b_3), L^{-1}(\Delta b) = c = (c_0 \| c_1 \| c_2 \| c_3)$.

According to the analysis of the *S*-Boxes in section 3.1 we can get that the highest differential probability for all of the *S*-Boxes is 1/4, if we choose proper Δa and Δb, the collision probability we listed above can reach 1/4. Take the input differential $\Delta a = (\Delta a_0 \| \Delta a_1 \| \Delta a_2 \| \Delta a_3)$ as an example, if $\Delta a_0 \neq 0$, and S_1 is the only active *S*-Box, in addition $\Delta a_0 \rightarrow c_0$ is the highest probability differential characteristic for S_1, the output differential for *WD*16 is zero with probability of 1/4. This result will be used later.

3.3 Differential Properties of Hummingbird-2

In section 3.2, we have analyzed the differential collision character for *WD*16, in this section, we will show how to use the collision character into analyzing *Hummingbird*-2.

The main structure of *Hummingbird*-2 is four round iteration of *WD*16, during the initialization and the encryption process, the input for *WD*16 is (*x*,*a*,*b*,*c*,*d*), *x* is either *IV*s or the intermediate variable, (*a*,*b*,*c*,*d*) are the independent keys injected directly. According to the analysis in section 3.2 we can get to know that if we choose proper related keys we can make the output differential for *WD*16 to be zero with high probability, furthermore the differentials for intermediate variables and 8 registers are zero with high probability at the same time, if these characters can be kept to the output of the ciphertext, we can judge whether the differential character we constructed using the related keys occurred by examining the differential of the ciphertext. Then we can recover the corresponding subkey blocks through differential techniques.

To avoid misunderstanding, we denote “one round of the algorithm” represents 4 iterations of *WD*16, either in the initialization process or encryption process, so the initialization process consists of four rounds of the algorithm and each round of the algorithm generates 16 bits ciphertext during the encryption process.

To minimize the probability of differential over round function *WD*16, the number of active *S*-Boxes must be minimized. As the algorithm consists of four round functions, and for each block of subkey it is used twice, on the same location of first round and the third round or the second round and the fourth round. So if we introduce a differential on the subkey of the first round or the second round which causes an active *S*-Box, at the same position on the third round or the fourth round must emerge an active *S*-Box. That is to say, the number of the active S-boxes is even, at least 2.

We denote subkey K_i as (K_i[0],K_i[1],K_i[2],K_i[3]), supposing $\Delta K_1=K_1\oplus K_1$'=(0,0,0, ΔK_1[3])$_{16}$, ($\Delta K_1[3]\neq 0$), if *IV*=*IV* ', during the first *WD*16 function of the initialization process, only S_4 is active, if $\Delta K_1[3]\rightarrow \Delta Z[3]$ is any one of the differential characteristic with probability of *p*, according to the differential collision properties in section 3.2, if we choose related keys $\Delta K_2=L(0,0,0,\Delta Z[3])$, $\Delta K_3,\cdots,\Delta K_8$ are all zero, it is obvious that for each round function *WD*16, the probability for zero output differential is *p*. Furthermore, each encryption process (or initialization process) consists of four round function *WD*16, according to the algorithm, in the same position of the third round of *WD*16 function the differential pair $\Delta K_1[3]\rightarrow \Delta Z[3]$ also exists, so if the differential of the *IV* is zero, under the related keys above, the differential after the first round of the algorithm is zero with probability of p^2.

We take ΔK_1 =(0003)$_{16}$ as an example, the initialization and the encryption process of the algorithm have the properties below:

Property 2 Differential characteristic of the initialization for each round: Under the key differential $\Delta K=(\Delta K_1,\Delta K_2,\Delta K_3,\Delta K_4,\Delta K_5,\Delta K_6,\Delta K_7,\Delta K_8)=$ (0003,0441,0000,0000,0000,0000, 0000,0000)$_{16}$, the differential characteristic below passes each round of initialization for the probability of $1/2^4$:

$$\begin{gathered}\Delta(IV_1,IV_2,IV_3,IV_4)=(0000,0000,0000,0000)\\ \downarrow\\ \Delta(R1_{-3},R2_{-3},R3_{-3},R4_{-3},R5_{-3},R6_{-3},R7_{-3},R8_{-3})=(0000,0000,0000,0000,0000,0000,0000,0000)\end{gathered}$$

If we find some *IV*s which make the differential characteristic above occurs, we can use the differential pair 0*x*3→0*x*1 of S_4 to recover the input, i.e. $IV_1[3]\oplus K_1[3]$, as IV_1 is known, we can recover the subkey block K_1[3] accordingly.

Property 3 Differential characteristic of the whole initialization process: Under the key differential $\Delta K=(\Delta K_1,\Delta K_2,\Delta K_3,\Delta K_4,\Delta K_5,\Delta K_6,\Delta K_7,\Delta K_8)=(0003,0441,0000,0000,0000,0000,$ $0000,0000)_{16}$, the differential characteristic below passes the whole initialization process for the probability of $1/2^{16}$:

$$\begin{tikzcd} \Delta(IV_1, IV_2, IV_3, IV_4) = (0000, 0000, 0000, 0000) \arrow[d] \\ \Delta(R1_0, R2_0, R3_0, R4_0, R5_0, R6_0, R7_0, R8_0) = (0000, 0000, 0000, 0000, 0000, 0000, 0000, 0000) \end{tikzcd}$$

As the initialization process are four rounds in all, the characteristic in property 2 can hold through the whole initialization process with probability of $1/2^{16}$.

Property 4 An iterated differential characteristic during the encryption process: Under the key differential in the property 2, the differential characteristic below passes each encryption process for the probability of $1/2^4$:

$$\begin{tikzcd} \Delta(IV_1, IV_2, IV_3, IV_4) = (0000, 0000, 0000, 0000) \arrow[d] \\ \Delta(R1_0, R2_0, R3_0, R4_0, R5_0, R6_0, R7_0, R8_0) = (0000, 0000, 0000, 0000, 0000, 0000, 0000, 0000) \end{tikzcd}$$

The property 4 denotes that if the differential of the plaintext is (0000), based on the condition of property 3, the differential of the ciphertext is (0000) for the probability of $1/2^4$.

As for the several properties above, under the condition of related keys, if the *IV* differential and the plaintext(*P*) differential are both zero, when we change the value of *IV*, we can always find such values which can satisfy these three properties.

3.4 Key Recovery Attack on Hummingbird-2

In this section, we introduce a key recovery attack algorithm on Hummingbird-2. Here is the clue of the attack: Firstly, we construct differentials through related keys, then we use different *IV*s to run the initialization process and the first round of the encryption process until we find a proper *IV* which satisfy the three properties in the section 3.3, whether a *IV* satisfies these properties can be shown through the ciphertext differential. If the *IV* can make the ciphertext differential to be zero, we will use the second filter to guarantee that it's the related key we constructed makes the ciphertext differential to be zero rather than the random case. If the *IV* can get through the second filter, we can get the input of the active *S*-Box for the first *WD*16 function of the initialization process, then the subkey candidates can be calculated. Here we use two different related keys that satisfy property 1 to make the subkey unique. Subkeys K_1, ···, K_3, K_5, ···, K_7 can be recovered using this method but K_5, ···, K_7 can be recovered on condition of K_4 is known, then we introduce a novel technique to recover K_4. In addition, K_8 can be recovered by exhaustive search.

Next, we take the recovery process of the four most significant bits of subkey K_1, ie. $K_1[3]$ as an example to introduce the procedure of the key recover, among which we select $\Delta K^{(1)}$ =(0003,0441,0000,0000,0000,0000,0000,0000)$_{16}$ and $\Delta K^{(2)}$ =(000a,1104, 0000,0000,0000,0000,0000,0000)$_{16}$.

Algorithm 1 The key recovery algorithm 1

Phase0. Define sets σ_1 and σ_2 , $\Phi \to \sigma_1, \Phi \to \sigma_2$;

Phase1. Encrypt using related keys K and $K \oplus \Delta K^{(1)}$, changing IV until we find a IV which make $C_0 = C_0'$;

(Remark: P_0 can be any value but the difference ΔP_0 must be zero)

Phase2. Encrypt with the IV we got from Phase1 and different P_0 (ΔP_0 is zero), if we use N different P_0, calculate the number of P_0 which makes $C_0 = C_0'$ (denoted as t), if $t >> N \times \frac{1}{2^{16}}$ (The random case), goto Phase 3, or discard the IV and goto Phase1;

Phase3. As the IV we got from Phase2 must satisfy the three properties in section 3.3. So the first differential characteristic for S_4 must be $0x3 \to 0x1$, searching S-Box distribution of the probability of differentials we can get to know that the input (Actually $K_1[3] \oplus IV_1[3]$) must be one element of the set {0x1,0x2,0xd,0xe}, compare $IV_1[3]$ with the elements in σ_1 , if $IV_1[3] \in \sigma_1$, goto Phase1, or add $IV_1[3]$ into the set σ_1 and make the intersection of the sets { $IV_1[3]$ } and σ_2 , if the intersection set is empty, goto Phase4, or goto Phase5;

Phase4. We call the process of Phase1 and Phase2 to be "Ciphertext Filter". Filter the IVs with related keys K and $K \oplus \Delta K^{(2)}$ until a IV' can get through the "Ciphertext Filter". Compare $IV_1'[3]$ with the elements in σ_2 , if $IV_1'[3] \in \sigma_2$, goto Phase4, or add $IV_1'[3]$ into the set σ_2 and make the intersection of the sets { $IV_1'[3]$ } and σ_1 , if the intersection set is empty, goto Phase1, or goto Phase5;

Phase5. Denote σ_1 I σ_2 @$IV_1^*[3]$, so $K_1[3] = IV_1^*[3] \oplus 0xe$, and finish the algorithm.

Using the algorithm above we can always get the right value of $K_1[3]$, the rest 12 bits $K_1[0]$, $K_1[1]$ and $K_1[2]$ can be recovered in the same way.

Similarly, through using different related keys and known K_1, we can use the same technique to recover K_2, under the conditions of known K_1 and K_2 we can recover K_3. In **Table 4**, we list the related keys needed to recover the subkeys K_1, K_2 and K_3.

Table 4. Related Keys Needed to Recover the Subkeys K_1, K_2 and K_3

Subkey Blocks	The First Related Key $\Delta K^{(1)}$	The Second Related Key $\Delta K^{(2)}$
K1[0]	(3000,2088,0000,0000,0000,0000,0000,0000)16	(b000,1044,0000,0000,0000,0000,0000,0000)16
K1[1]	(0f00,4104,0000,0000,0000,0000,0000,0000)16	(0300,8208,0000,0000,0000,0000,0000,0000)16
K1[2]	(0050,1041,0000,0000,0000,0000,0000,0000)16	(00d0,2082,0000,0000,0000,0000,0000,0000)16
K1[3]	(0003,0441,0000,0000,0000,0000,0000,0000)16	(000a,1104,0000,0000,0000,0000,0000,0000)16
K2[0]	(0000,3000,2088,0000,0000,0000,0000,0000)16	(0000,b000,1044,0000,0000,0000,0000,0000)16
K2[1]	(0000,0f00,4104,0000,0000,0000,0000,0000)16	(0000,0300,8208,0000,0000,0000,0000,0000)16
K2[2]	(0000,0050,1041,0000,0000,0000,0000,0000)16	(0000,00d0,2082,0000,0000,0000,0000,0000)16
K2[3]	(0000,0003,0441,0000,0000,0000,0000,0000)16	(0000,000a,1104,0000,0000,0000,0000,0000)16
K3[0]	(0000,0000,3000,2088,0000,0000,0000,0000)16	(0000,0000,b000,1044,0000,0000,0000,0000)16
K3[1]	(0000,0000,0f00,4104,0000,0000,0000,0000)16	(0000,0000,0300,8208,0000,0000,0000,0000)16
K3[2]	(0000,0000,0050,1041,0000,0000,0000,0000)16	(0000,0000,00d0,2082,0000,0000,0000,0000)16
K3[3]	(0000,0000,0003,0441,0000,0000,0000,0000)16	(0000,0000,000a,1104,0000,0000,0000,0000)16

As it will induce the differential into the registers if we use the method above to recover K_4, we must think of other ways to recover it. Here we introduce a novel technique to filter the K_4 candidates. As the *IV*s which pass the “Ciphertext Filter” must satisfy three properties similar to those in 3.3, so if we introduce differential in K_5 and K_6, we can get the input of the first *S*-Boxes for the second *WD*16 function of the initialization process. However, though K_5 is unknown, we can eliminate this effect by differential technique, we can get to know the differential of t_1, further we can filter the right subkey K_4 accordingly. Here is the filtering algorithm for K_4.

Algorithm 2 Filtering Algorithm for K_4

Phase1. Encrypt using related keys K and $K \oplus \Delta K$, ΔK=(0000,0000,0000,0000,3000,2088,0000, $0000)_{16}$, change different *IV*s until we find a *IV* and IV' which can both pass the “Ciphertext Filter”;

Phase2. As the differential characteristic for S_1 is $0x3 \rightarrow 0x2$, searching *S*-Box distribution of the probability of differentials we can get to know that the input of the S_1, i.e. ($t_1[0] \boxplus R_2^{(0)}[0]$) $\oplus K_5[0]$ under *IV* and IV'. Encrypt the first *WD*16 function of the initialization process with *IV* or IV', K_1, K_2, K_3 and K_4 candidates to calculate $(t_1[0] \boxplus R_2^{(0)}[0]) \oplus (t_1'[0] \boxplus R_2^{(0)}{}'[0])$, if it isn't among one of the {0*x*0, 0*x*3, 0*xc*, 0*xf*}(The differential set for S_1 input), the K_4 candidate is incorrect, discard the K_4 candidate from the candidate set and goto Phase3, or goto Phase 1;

Phase3. Check the number of the candidate set, if the number is not equal to 1, goto Phase1 and keep filtering , or finish the algorithm.

If IV and IV' can pass the “Ciphertext Filter”, the differential characteristic constructed by related keys must have appeared and according to the differential characteristic $0x3 \to 0x2$ for S_1, we can conjecture that the input must be one of the $0x1, 0x2, 0xd, 0xe$, XOR the four elements with each other we can get a new set $\{0x0, 0x3, 0xc, 0xf\}$ and this is the differential set for S_1 input. As $(t_1[0] \boxplus R_2^{(0)}[0]) \oplus (t_1'[0] \boxplus R_2^{(0)}{}'[0]) = (t_1[0] \boxplus R_2^{(0)}[0]) \oplus K_5[0] \oplus (t_1'[0] \boxplus R_2^{(0)}{}'[0]) \oplus K_5[0]$, if we use the incorrect K_4, $(t_1[0] \boxplus R_2^{(0)}[0]) \oplus K_5[0]$ and $(t_1'[0] \boxplus R_2^{(0)}{}'[0]) \oplus K_5[0]$ are both random and the $(t_1[0] \boxplus R_2^{(0)}[0]) \oplus (t_1'[0] \boxplus R_2^{(0)}{}'[0])$ falls into the set $\{0x0, 0x3, 0xc, 0xf\}$ with probability of 4/16, as $(2^{16}-1)\times(1/4)^8<1$. In average, we can eliminate all the incorrect K_4 candidates through eight different IV and IV'. As $R_2^{(0)}$ and $R_2^{(0)}{}'$ are filled with IV and IV', so we can calculate $(t_1[0] \boxplus R_2^{(0)}[0]) \oplus (t_1'[0] \boxplus R_2^{(0)}{}'[0])$ easily.

After the recovery of K_4, we can recover subkeys K_5, K_6, K_7 according to Algorithm1 and the related keys needed are as follows:

Table 5. Related Keys Needed to Recover Subkeys K_5, K_6 and K_7

	The First Related Key $\Delta K^{(1)}$	The Second Related Key $\Delta K^{(2)}$
$K_5[0]$	$(0000.0000.0000.0000.3000.2088.0000.0000)_{16}$	$(0000.0000.0000.0000.b000.1044.0000.0000)_{16}$
$K_5[1]$	$(0000,0000,0000,0000,0f00,4104,0000,0000)_{16}$	$(0000,0000,0000,0000,0300,8208,0000,0000)_{16}$
$K_5[2]$	$(0000.0000.0000.0000.0050.1041.0000.0000)_{16}$	$(0000.0000.0000.0000.00d0.2082.0000.0000)_{16}$
$K_5[3]$	$(0000,0000,0000,0000,0003,0441,0000,0000)_{16}$	$(0000,0000,0000,0000,000a,1104,0000,0000)_{16}$
$K_6[0]$	$(0000.0000.0000.0000.0000.3000.2088.0000)_{16}$	$(0000.0000.0000.0000.0000.b000.1044.0000)_{16}$
$K_6[1]$	$(0000,0000,0000,0000,0000,0f00,4104,0000)_{16}$	$(0000,0000,0000,0000,0000,0300,8208,0000)_{16}$
$K_6[2]$	$(0000,0000,0000,0000,0000,0050,1041,0000)_{16}$	$(0000,0000,0000,0000,0000,00d0,2082,0000)_{16}$
$K_6[3]$	$(0000,0000,0000,0000,0000,0003,0441,0000)_{16}$	$(0000,0000,0000,0000,0000,000a,1104,0000)_{16}$
$K_7[0]$	$(0000.0000.0000.0000.0000.0000.3000.2088)_{16}$	$(0000.0000.0000.0000.0000.0000.b000.1044)_{16}$
$K_7[1]$	$(0000.0000.0000.0000.0000.0000.0f00.4104)_{16}$	$(0000.0000.0000.0000.0000.0000.0300.8208)_{16}$
$K_7[2]$	$(0000,0000,0000,0000,0000,0000,0050,1041)_{16}$	$(0000,0000,0000,0000,0000,0000,00d0,2082)_{16}$
$K_7[3]$	$(0000,0000,0000,0000,0000,0000,0003,0441)_{16}$	$(0000,0000,0000,0000,0000,0000,000a,1104)_{16}$

We have recovered 112 bits of the key $(K_1,\cdots,K_7)$ so far, and the remaining 16 bits K_8 can be recovered by exhaustive search.

3.5 Complexities of the Attack

First of all, let us calculate the amount of IVs needed to recover 4 bits of the key for algorithm 1. When $N=2^{20}$, in average, there are $2^{20}\times1/2^{-20}+2^{20}\times1/2^{-16}=17$ IVs which can pass the Phase1 test of Algorithm 1, but only one of them can pass the Phase 2 test. In worst case, the sets σ_1, σ_2 generate the final $IV_i[j]$ when $|\sigma_1| = 4$ and $|\sigma_2| = 4$. So the number of *IV*s needed is about $2\times4\times2^{20}=2^{23}$. As for the filter for Phase 2 of Algorithm 1, the maximum amount for (P_0, C_0) is far less than $17\times2^{16}\approx2^{20.1}$ which can be ignored when compared with the complexity of *IV* filtering process for Phase 1. To recover K_1,K_2,K_3, the *IV*s needed are $12\times2^{23}\approx2^{26.6}$ and accordingly the computational complexity is $O(2^{26.6})$.

To recover K_4, we need eight *IV* pairs in average, which makes us need about $8\times2\times2^{20}\approx2^{24}$ chosen *IV*s. For computational complexity, during the process of choosing *IV*, the computational complexity is $O(2^{24})$, and during the filtering of the correct K_4, the computational complexity is $8\times2\times3+2^{16}\times(1+(1-1/4)+(1-1/4)^2+(1-1/4)^3\cdots+(1-1/4)^8)\approx2^{17.9}$ one iteration of *WD*16 function, which can be ignored when compared with the computational complexity of choosing *IV*, so the computational complexity to recover K_4 is $O(2^{24})$.

The *IV*s needed and computational complexity to recover K_5,K_6,K_7 are the same as those to recover K_1,K_2,K_3.

To sum up, we need $2^{27.6}$ chosen *IV*s and the computational complexity is $O(2^{27.6})$ to recover the first 112 bits key. The data complexity to recover K_8 is $O(1)$ and computational complexity of $O(2^{16})$ which can be ignored when compared with the complexities above. So the computational complexity to recover the full key is $O(2^{27.6})$, which needs $2^{27.6}$ chosen *IV*s, the computational complexity is $O(2^{27.6})$. The related keys needed are shown in **Table 4** and **Table 5**, we need 48 related keys in all.

3.6 An Improvement of the Attack

We can use Algorithm 2 to recover K_1, K_2, K_5, K_6 which can further improve our result. The related keys and algorithm used are as follows:

Note that the related key pair used to recover K_2 (or K_6) is included in the related keys to recover K_3 (or K_7). So in this strategy, we totally need 19 related keys. The *IV*s needed to recover K_3 and K_7 is 2^{26} and the computational complexity is $O(2^{26})$ accordingly. Otherwise, the *IV*s needed to recover K_1,K_2,K_4,K_5,K_6 is $5\times2^{24}\approx2^{26.3}$, and the computational complexity is $O(2^{26.3})$

Table 6. Another Strategy to Recover the Key

Related Keys Needed	Keys Recovered	The Number of Related Keys	Algorithm Used
K_2-K_3	K_1	1	Algorithm 2
K_3-K_4	K_2	1	Algorithm 2
K_3-K_4	K_3	8	Algorithm 1
K_5-K_6	K_4	1	Algorithm 2
K_6-K_7	K_5	1	Algorithm 2
K_7-K_8	K_6	1	Algorithm 2
K_7-K_8	K_7	8	Algorithm 1

What's more, we could recover $K_7[2]$ and $K_7[3]$ by exhaustive search, this will add the exhaustive search complexity to $O(2^{24})$, but this will decrease 4 pairs of related key used, in this scenario, the *IV*s needed is 2^{27}, and the computational complexity is $O(2^{27})$.

So, we totally just need 15 related keys, 2^{27} chosen *IV*s and the computational complexity is $O(2^{27})$ in our improved attack.

3.7 Experimental Verification

To guarantee the correctness and demonstrate the efficiency of our attack we implemented the above attack algorithms. The experiment environment, experiment process and result are as follow.

Experiment Environment: Microsoft Visual C++(SP6), Windows XP professional SP3, Pentium(R)-4, CPU 2.5GHz, 1.0 Gb RAM.

Experiment Process: Choose 50 keys randomly, then try to recover the key according to our attack algorithm and the improved attack algorithm, the output is the key recovered and the time cost.

Result: All of the keys are recovered correctly, time spent to recover one key ranges from 10.2 minutes to 16.1 minutes. In average, we need about 13.2 minutes to recover a key in our experiment environment.

The result shows that our attack can break Hummingbird-2 in real time under the related key model.

4. Conclusion

The designers of Hummingbird-2 claimed that Hummingbird-2 is resistant to all previously known cryptanalytic attacks, including related key attack. However, in this paper, we present a related-key chosen *IV* attack combining with differential techniques on Hummingbird-2 which can break the algorithm in real time. First of all, we use related keys to construct partial differential with high probability and we ensure the collision with sufficient chosen *IV*s. Then we judge the inner collision through the differential of the ciphertext. Finally, we use differential techniques to recover the initial key. As the key loading algorithm is too simple, though adding the influence of the registers, these effects can be eliminated by differential techniques, which makes the attack possible. Under 15 pairs of related keys, we can recover the 128 bit initial key with computational complexity of $O(2^{27})$ and 2^{27} chosen *IV*s. Experiment shows that the keys can be recovered in real time. Compared with the attack proposed by *Markku-Juhani O. Saarinen*, our attack uses the inner differential characteristic of round function *WD*16 rather than the outer differential characteristic and our attack works on Hummingbird-2 cipher with manageably low space and time complexity. Furthermore, we have proved that the Hummingbird-1 can also be analyzed in the same way. The result in this paper shows that Hummingbird-2 cipher can't resist the related-key attack. As the related key attack is a strong model for cryptanalysis, the ability of Hummingbird family ciphers to resist other cryptanalysis in weaker model is further to be studied.

References

[1] D. Engels, M.J.O. Saarinen, and E.M. Smith, "The Hummingbird-2 Lightweight Authenticated Encryption Algorithm," in *Proc. of the 7th Workshop on RFID Security and Privacy-RFIDSec 2011*, 2011. Article (CrossRef Link).

[2] D. Engels, X. Fan, G. Gong, H. Hu, and E.M. Smith, "Hummingbird: Ultra-Lightweight Cryptography for Resource-Constrained Devices," in *Proc. of FC 2010 Workshops*, *RLCPS*, *WECSR*, *and WLC 2010*, LNCS 6054, Springer-Verlag, pp.3-18, 2010. Article (CrossRef Link).

[3] E. Biham, "New types of cryptanalytic attacks using related keys," *in Proc. of EUROCRYT 1993*, LNCS 765. Springer-Verlag, pp. 398-309, 1994. Article (CrossRef Link).

[4] E. Biham, A. Shamir, "Differential Cryptanalysis of DES-like Cryptosystems," in *Proc. of CRYPTO 1990*. LNCS 537. Springer-Verlag, pp. 2-21, 1990. Article (CrossRef Link).

[5] E. Biham, E. Dunkelman, O. Keller, "Related-key boomerang and rectangle attacks," in *Proc. of EUROCRYPT 2005*, LNCS 3494. Springer-Verlag, pp. 507-525, 2005. Article (CrossRef Link).

[6] E. Biham, O. Dunkelman, N. Keller, "Related-key impossible differential attacks on 8-round AES-192," *in Proc. of CT-RSA 2006*, LNCS 3860. Springer-Verlag, pp. 21-33, 2006. Article (CrossRef Link).

[7] G. Jakimoski, Y. Desmedt, "Related-Key differential cryptanalysis of 192-bit key AES Variants," *in Proc. of SAC 2003*, LNCS 3006. Springer-Verlag, pp.209 – 221, 2004. Article (CrossRef Link).

[8] J. Keysey, B. Schneier, and D. Wanger, "Key-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES," in *Proc. of CRYPTO 1996*, LNCS 1109. Springer-Verlag, pp. 237-251, 1996. Article (CrossRef Link).

[9] L. Knudsen, "Cryptanalysis of LOKI," in *Proc. of ASIACRYPT 1992*, LNCS 739, Springer-Verlag, pp.22-35, 1993. Article (CrossRef Link).

[10] M.J.O. Saarinen, "Cryptanalysis of Hummingbird-1," in *Proc. of FSE 2011*. LNCS 6733, Springer-Verlag, pp.328-341, 2011. Article (CrossRef Link).

[11] Q. Chai and G. Gong, "A Cryptanalysis of HummingBird-2: The Differential Sequence Analysis," *Cryptology ePrint Archive*. Report 2012/233 (2012).
http://eprint.iacr.org/2012/233.pdf

[12] R.P. Weinmann and K. Wirt, "Analysis of the DVB Common Scrambling Algorithm," *in Proc. of the Eighth IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, CMS 2004*, Kluwer Academic Publishers, vol. 175, pp.195-207, 2005. Article (CrossRef Link).

[13] W. Zhang, W. Wu, L. Zhang, and D. Feng, "Improved related-key impossible differential attack on reduced-round AES-192," *in Proc. of SAC 2006*, LNCS 4356. Springer-Verlag, , pp.15-27, 2007. Article (CrossRef Link).

[14] X. Fan and G. Gong, "On the Security of Hummingbird-2 against Side Channel Cube Attack," in *Proc. of the 2011 West European Workshop on Research in Cryptography-WEWoRC 2011*, Springer-Verlag, pp.100-104, 2011.
http://www.uni-weimar.de/cms/fileadmin/medien/medsicherheit/WEWoRC2011/files/conference_record3.pdf#page=106

Kai Zhang received B.S. degree from Zhengzhou Information Science and Technology Institute in 2010. He is studying for M.S. degree in cryptography in the same university. His current research interests include design and analysis of symmetric ciphers.

Lin Ding received B.S. degree from Zhengzhou Information Science and Technology Institute in 2009. He is studying for M.S. degree in cryptography in the same university. His research interests include design and analysis of stream ciphers.

Junzhi Li is studying for B.S. in information security in Zhengzhou Information Science and Technology Institute. His research interests include cryptanalysis of symmetric ciphers.

Jie Guan is an associate professor of Zhengzhou Information Science and Technology Institute. Her main subject interest is cryptography and her main teaching lies in the areas of information systems, the theory of cryptography and quantum computation. She received Ph.D. degree in cryptography from Zhengzhou Information Science and Technology Institute in 2004.