Cryptanalysis of 256-Bit Key HyRAL via Equivalent Keys

Yuki Asano, Shingo Yanagihara, and Tetsu Iwata

Nagoya University, Japan
{y_asano,s_yanagi}@echo.nuee.nagoya-u.ac.jp,
iwata@cse.nagoya-u.ac.jp

Abstract. HyRAL is a blockcipher whose block size is 128 bits, and it supports the key lengths of $128, 129, \ldots, 256$ bits. The cipher was proposed for the CRYPTREC project, and previous analyses did not identify any security weaknesses. In this paper, we consider the longest key version, 256-bit key HyRAL, and present the analysis in terms of equivalent keys. First, we show that there are $2^{51.0}$ equivalent keys (or $2^{50.0}$ pairs of equivalent keys). Next, we propose an algorithm that derives an instance of equivalent keys with the expected time complexity of $2^{48.8}$ encryptions and a limited amount of memory. Finally, we implement the proposed algorithm and fully verify its correctness by showing several instances of equivalent keys.

Keywords: Cryptanalysis, blockcipher, HyRAL, equivalent key.

1 Introduction

HyRAL is a blockcipher whose block size is 128 bits, and it supports the key lengths of $128, 129, \ldots, 256$ bits [6,7,8]. The overall structure of HyRAL is the Generalized Feistel Structure with four branches, and 128-bit key HyRAL consists of 24 rounds, and $129\text{-}, 130\text{-}, \ldots, 256$-bit key HyRAL consist of 32 rounds. The CRYPTREC project, running in Japan, is maintaining the e-Government recommended ciphers list, which was first established in 2003, and the list is planned to be revised in 2013 [3]. A call for algorithms was announced in 2009, and HyRAL is one of the proposed algorithms for the call [6]. The security of HyRAL against various attacks has been evaluated. The security against differential attacks [2] and linear attacks [15,16] is analyzed in [6,9,24], impossible differential attacks [1] is analyzed in [17,18], saturation attacks [5] and higher order differential attacks [13,14] is analyzed in [19,20,21,22,23,25,26], and boomerang attacks is analyzed in [10]. [27] also presents security analyses against various attacks, and so far, no security weaknesses have been identified. Therefore, identifying a security weakness of this cipher is of interest from a cryptanalytic view point.

For a blockcipher $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ of k-bit keys and an n-bit block, two distinct keys $K, K' \in \{0,1\}^k$ that satisfy $E_K(M) = E_{K'}(M)$ for all $M \in \{0,1\}^n$ are called *equivalent keys* [12]. In this paper, out of the 129 key lengths supported in the specification of HyRAL, we consider the longest key

F. Bao, P. Samarati, and J. Zhou (Eds.): ACNS 2012, LNCS 7341, pp. 257–274, 2012.

version, 256-bit key HyRAL, and present the analysis in terms of equivalent keys. We show the following three results.

- First, we show that there are $2^{51.0}$ equivalent keys (or $2^{50.0}$ pairs of equivalent keys).
- Next, we propose an algorithm that derives an instance of equivalent keys with the expected time complexity of $2^{48.8}$ encryptions and a limited amount of memory.
- Finally, we implement the proposed algorithm and fully verify its correctness by showing several instances of equivalent keys.

The first result is obtained by an analysis of the differential characteristic of the particular component in the cipher called the Key Generation Algorithm, which we write KGA. KGA is used twice in the cipher, and their outputs are xor'ed to generate round keys. We show that, for KGA, there exist differential characteristics of probability higher than 2^{-128}, and hence the output differences collide with probability higher than 2^{-256}. Equivalent keys are obtained as the result of this internal collision. In general, the existence of equivalent keys directly implies the theoretical cryptanalysis of the cipher, as the time complexity of the brute-force attack becomes less than the time complexity implied by its key length.

The second result is the main technical contribution of this paper. We develop an algorithm to generate the input of KGA that follows the differential characteristic. The core of the algorithm is to derive the input of KGA that satisfies conditions on the 64-bit intermediate variables. More precisely, it inverts the 5 round Generalized Feistel Structure that has a feed forward at the input of the 5th round, where there are conditions that the xor of three 32-bit input variables of the 5th round is fixed to some constant, and that the xor of three 32-bit output variables of the 5th round is also fixed to some constant.

We obtain the third result by making use of a supercomputer system. It is well known that obtaining concrete instances of equivalent keys implies that we obtain collisions on the Davies-Meyer compression function based on the blockcipher. It is also easy to obtain collisions on the Merkle-Damgård hash function based on the compression function. Therefore, the existence of equivalent keys limits the use of the cipher in the widely deployed hash function mode.

With respect to the status of HyRAL in the CRYPTREC project, we note that the results of this paper were reported to the project [11], and it was announced in June 2011 that, based on the results, HyRAL did not proceed to the second round evaluation process [4].

2 Specification of HyRAL

We first define notation used throughout this paper. For an integer $n \geq 0$, $\{0,1\}^n$ is the set of n-bit strings. For two bit strings X and Y of the same length, $X \oplus Y$ is their xor. For integers $n, m \geq 0$ and a bit string $X \in \{0,1\}^{nm}$, $(X[1], \ldots, X[m]) \stackrel{n}{\leftarrow} X$ is the partitioning operation into n-bit strings, i.e., $X[1], \ldots, X[m]$ are unique n-bit strings such that $(X[1], \ldots, X[m]) = X$.

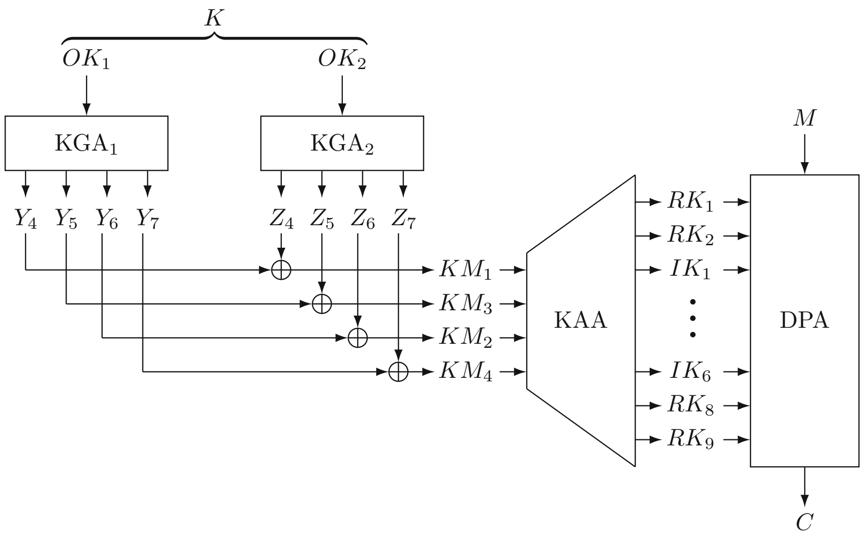

Fig. 1. The overall structure of 256-bit key HyRAL

Outline of 256-Bit Key HyRAL. HyRAL is a blockcipher whose block size is 128 bits, and it supports the key lengths of $128, 129, \ldots, 256$ bits. This paper deals with the 256-bit key version only, and the specifications of other key lengths are in [6,7,8].

The overall structure of 256-bit key HyRAL is shown in Fig. 1. The inputs are a key $K \in \{0,1\}^{256}$ and a plaintext $M \in \{0,1\}^{128}$, and the output is a ciphertext $C \in \{0,1\}^{128}$. 256-bit key HyRAL consists of the Key Generation Algorithm (KGA), the Key Assignment Algorithm (KAA), and the Data Processing Algorithm (DPA). KGA is used twice by changing the internal constants, and they are respectively denoted KGA_1 and KGA_2. For given $K \in \{0,1\}^{256}$ and $M \in \{0,1\}^{128}$, the encryption proceeds as follows.

1. Let $(OK_1, OK_2) \stackrel{128}{\leftarrow} K$. That is, let OK_1 be the most significant 128 bits of K, and OK_2 be the least significant 128 bits.
2. We then run KGA_1 and KGA_2 with OK_1 and OK_2, respectively, to generate $(Y_4, Y_5, Y_6, Y_7) \leftarrow \text{KGA}_1(OK_1)$ and $(Z_4, Z_5, Z_6, Z_7) \leftarrow \text{KGA}_2(OK_2)$, where $Y_i, Z_i \in \{0,1\}^{128}$.
3. Let $(KM_1, KM_3, KM_2, KM_4) \leftarrow (Y_4 \oplus Z_4, Y_5 \oplus Z_5, Y_6 \oplus Z_6, Y_7 \oplus Z_7)$, where $KM_i \in \{0,1\}^{128}$. We write $(KM_1, KM_3, KM_2, KM_4) = KM$.
4. Next, we run KAA with KM to generate $(RK_1, \ldots, RK_9, IK_1, \ldots, IK_6) \leftarrow \text{KAA}(KM)$, where $RK_i, IK_i \in \{0,1\}^{128}$.
5. Finally, we run DPA with $(RK_1, \ldots, RK_9, IK_1, \ldots, IK_6)$ and the plaintext M to generate the ciphertext $C \leftarrow \text{DPA}(RK_1, \ldots, RK_9, IK_1, \ldots, IK_6, M)$, and then C is returned.

In KAA, (KM_1, KM_3, KM_2, KM_4) are first parsed into 32-bit strings, and then $(RK_1, \ldots, RK_9, IK_1, \ldots, IK_6)$ are generated by taking their linear

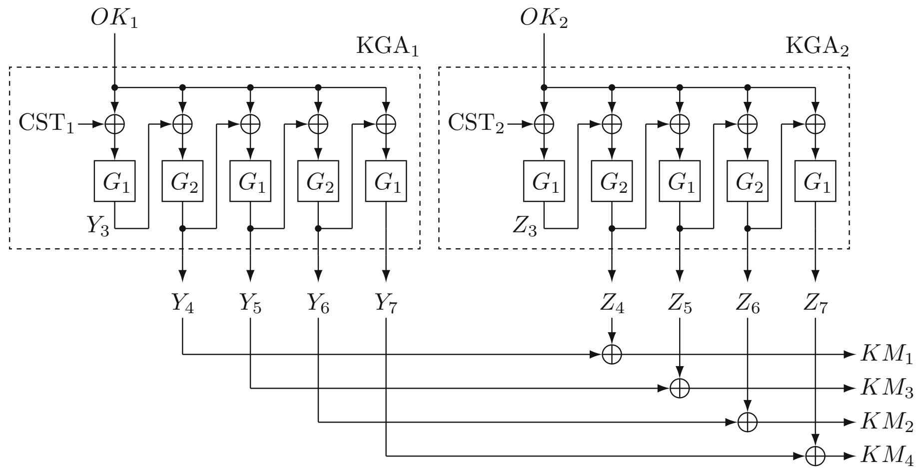

Fig. 2. KGA_1 and KGA_2

combinations. The overall structure of DPA is the 32 round Generalized Feistel Structure with four branches. Further details of KAA and DPA are not necessary in order to present the results of this paper, and their specifications can be found in [6,7,8].

The Key Generation Algorithms KGA_1 *and* KGA_2*.* KGA_1 and KGA_2 are shown in Fig. 2. For the input $OK_1 \in \{0,1\}^{128}$, KGA_1 outputs $(Y_4, Y_5, Y_6, Y_7) \in \{0,1\}^{512}$. Similarly, KGA_2 takes $OK_2 \in \{0,1\}^{128}$ and outputs $(Z_4, Z_5, Z_6, Z_7) \in \{0,1\}^{512}$. KGA_1 and KGA_2 internally use G_1 and G_2 functions, which are keyless permutations over $\{0,1\}^{128}$. KGA_1 and KGA_2 differ only in the internally used constants. The following 128-bit constants, CST_1 and CST_2, are used in KGA_1 and KGA_2, respectively, where the prefix `0x` indicates that the value is in the hexadecimal form.

$$\begin{cases} \mathrm{CST}_1 = \texttt{0xf9251a2365cd3c2e8066cbbbfe316b7b} \\ \mathrm{CST}_2 = \texttt{0x5de28625656b71ff9ffb1e12eef127f5} \end{cases}$$

G_1 *and* G_2 *Functions.* G_1 and G_2 functions are shown in Fig. 3.

G_1 and G_2 functions take $(X_1^{(1)}, X_2^{(1)}, X_3^{(1)}, X_4^{(1)}) \in \{0,1\}^{128}$ as the input, and output $(X_1^{(5)}, X_2^{(5)}, X_3^{(5)}, X_4^{(5)}) \in \{0,1\}^{128}$. They consist of four rounds of the Generalized Feistel Structure with four branches. G_1 function internally uses f_1, f_2, f_3, and f_4 functions, and G_2 function internally uses f_5, f_6, f_7, and f_8 functions.

$f_1, \ldots, f_8$ *Functions.* $f_1, \ldots, f_8$ functions are keyless permutations over $\{0,1\}^{32}$. They take a 32-bit string as the input and generate a 32-bit string as the output. The structure of f_i function is the SP-network, and a detailed specification is presented in Appendix A.

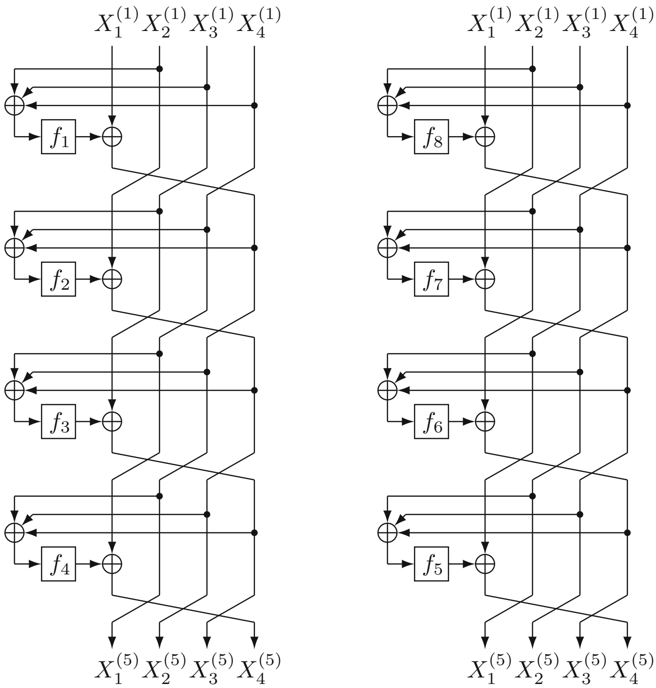

Fig. 3. G_1 function (left) and G_2 function (right)

3 Existence of Equivalent Keys

Overall Strategy. We make use of the differential cryptanalysis of Biham and Shamir [2] to show the existence of equivalent keys.

Let $(OK_1, OK_2) \in \{0,1\}^{256}$ be the key. Let ΔOK_1 be the input difference for KGA_1 and $(\Delta Y_4, \Delta Y_5, \Delta Y_6, \Delta Y_7)$ be the corresponding output difference. Similarly, let ΔOK_2 and $(\Delta Z_4, \Delta Z_5, \Delta Z_6, \Delta Z_7)$ be the input and output differences of KGA_2, respectively. We have

$$(\Delta Y_4, \Delta Y_5, \Delta Y_6, \Delta Y_7) = \text{KGA}_1(OK_1) \oplus \text{KGA}_1(OK_1 \oplus \Delta OK_1) \qquad (1)$$

and

$$(\Delta Z_4, \Delta Z_5, \Delta Z_6, \Delta Z_7) = \text{KGA}_2(OK_2) \oplus \text{KGA}_2(OK_2 \oplus \Delta OK_2). \qquad (2)$$

If the two output differences collide and

$$(\Delta Y_4, \Delta Y_5, \Delta Y_6, \Delta Y_7) = (\Delta Z_4, \Delta Z_5, \Delta Z_6, \Delta Z_7) \qquad (3)$$

holds, we see that the differences are canceled by the xor operation and the input difference $(\Delta KM_1, \Delta KM_3, \Delta KM_2, \Delta KM_4)$ of KAA becomes null. Therefore, if

(3) holds, we have the following equivalent keys.

$$(K, K') = \begin{cases} ((OK_1, OK_2), (OK_1 \oplus \Delta OK_1, OK_2 \oplus \Delta OK_2)) \\ ((OK_1 \oplus \Delta OK_1, OK_2 \oplus \Delta OK_2), (OK_1, OK_2)) \\ ((OK_1 \oplus \Delta OK_1, OK_2), (OK_1, OK_2 \oplus \Delta OK_2)) \\ ((OK_1, OK_2 \oplus \Delta OK_2), (OK_1 \oplus \Delta OK_1, OK_2)) \end{cases} \quad (4)$$

In this paper, these are counted as four equivalent keys (or two pairs of equivalent keys).

Since KGA_1 and KGA_2 are the same algorithms except for the internally used constants, we may regard them identically as long as we consider their differential characteristics. In what follows, let $\text{KGA} \in \{\text{KGA}_1, \text{KGA}_2\}$ be the Key Generation Algorithm. We next analyze the differential characteristic of KGA.

Differential Characteristic of KGA*.* We regard one round of G_1 and G_2 functions as one round of KGA. Then KGA is a function that consists of 20 rounds in total. For $r = 1, 2, \ldots, 20$, we write $f_i^{(r)}$ for f_i function used in the r-th round.

Let $\Delta OK \in \{0,1\}^{128}$ be the input difference of KGA and $(\Delta Y_4, \Delta Y_5, \Delta Y_6, \Delta Y_7) \in \{0,1\}^{512}$ be the corresponding output difference.

For $r = 1, 2, \ldots, 20$, let $\Delta X^{(r)} = (\Delta X_1^{(r)}, \Delta X_2^{(r)}, \Delta X_3^{(r)}, \Delta X_4^{(r)}) \in \{0,1\}^{128}$ be the input difference of the r-th round, and $\Delta Z^{(r)} = (\Delta Z_1^{(r)}, \Delta Z_2^{(r)}, \Delta Z_3^{(r)}, \Delta Z_4^{(r)}) \in \{0,1\}^{128}$ be its output difference. A *differential characteristic* is a tuple

$$((\Delta X^{(1)}, \Delta Z^{(1)}), \ldots, (\Delta X^{(20)}, \Delta Z^{(20)}))$$

of the input and output differences of each round, that satisfies the following conditions: First, it corresponds with the input and output differences of KGA, and hence we have $\Delta X^{(1)} = \Delta OK$, $\Delta Z^{(8)} = \Delta Y_4$, $\Delta Z^{(12)} = \Delta Y_5$, $\Delta Z^{(16)} = \Delta Y_6$, and $\Delta Z^{(20)} = \Delta Y_7$. Second, for $r = 1, 2, \ldots, 20$, we have $(\Delta X_2^{(r)}, \Delta X_3^{(r)}, \Delta X_4^{(r)}) = (\Delta Z_1^{(r)}, \Delta Z_2^{(r)}, \Delta Z_3^{(r)})$. Third, for $r \in \{4, 8, 12, 16\}$, we have $\Delta X^{(r+1)} = \Delta Z^{(r)} \oplus \Delta OK$, and for $r \in \{1, 2, \ldots, 19\} \setminus \{4, 8, 12, 16\}$, we have $\Delta X^{(r+1)} = \Delta Z^{(r)}$.

For a differential characteristic $((\Delta X^{(1)}, \Delta Z^{(1)}), \ldots, (\Delta X^{(20)}, \Delta Z^{(20)}))$, its probability is defined as

$$\text{DCP}^{\text{KGA}}((\Delta X^{(1)}, \Delta Z^{(1)}), \ldots, (\Delta X^{(20)} \Delta Z^{(20)})) = \prod_{1 \le r \le 20} \text{DP}^{f_i^{(r)}}(\Delta I_i^{(r)}, \Delta O_i^{(r)}),$$

where $\Delta I_i^{(r)} = \Delta X_2^{(r)} \oplus \Delta X_3^{(r)} \oplus \Delta X_4^{(r)}$ is the input difference of $f_i^{(r)}$, $\Delta O_i^{(r)} = \Delta X_1^{(r)} \oplus \Delta Z_4^{(r)}$ is the corresponding output difference, and the differential probability $\text{DP}^{f_i}(\Delta I_i, \Delta O_i)$ of f_i function for the input difference ΔI_i and the output difference ΔO_i is defined as

$$\text{DP}^{f_i}(\Delta I_i, \Delta O_i) = \frac{\#\{I \mid f_i(I) \oplus f_i(I \oplus \Delta I_i) = \Delta O_i\}}{2^{32}}.$$

For a given differential characteristic, we say that f_i function is *active* if its input difference is non-zero. In KGA, there are 20 f_i functions and hence the maximum number of active f_i functions is 20. In the following lemma, we show that there exists a differential characteristic with only four active f_i functions.

Lemma 1. *For* KGA*, there exists a differential characteristic with four active f_i functions.*

Proof. Let $\delta \in \{0,1\}^{32}$ be any non-zero bit string. Let $\Delta OK = (\delta, \delta, \delta, \delta)$ be the input difference of KGA, and $(\Delta Y_4, \Delta Y_5, \Delta Y_6, \Delta Y_7)$ be the output difference, where $\Delta Y_4 = (\delta, \delta, 0, 0)$, $\Delta Y_5 = (0, 0, 0, \delta)$, $\Delta Y_6 = (\delta, \delta, \delta, \delta)$, and $\Delta Y_7 = (0, 0, 0, 0)$. Consider the differential characteristic given in Table 1, which is also shown in Fig. 4. Then one can verify that there are four active f_i functions, which are $f_1^{(1)}$, $f_7^{(6)}$, $f_3^{(11)}$, and $f_5^{(16)}$. □

We see that the input and output differences of active f_i functions in the differential characteristic of Lemma 1 are both δ. Under the condition that both the input and output differences are the same, we have counted the number of active f_i functions for the 15 non-zero input differences, which are $(0, 0, 0, \delta)$, $(0, 0, \delta, 0)$, $(0, 0, \delta, \delta), \ldots, (\delta, \delta, \delta, \delta)$. The results are summarized in Table 2.

From the table, we see that the number of active f_i functions of Lemma 1 is the minimum among the 15 differential characteristics.

Differential Probability of f_i Function. For f_i function, let $\mathrm{DP}^{f_i}(\delta)$ be the probability that both the input and output differences of f_i function are δ, i.e.,

$$\mathrm{DP}^{f_i}(\delta) = \mathrm{DP}^{f_i}(\delta, \delta) = \frac{\#\{I \mid f_i(I) \oplus f_i(I \oplus \delta) = \delta\}}{2^{32}}.$$

The probability of the differential characteristic in Lemma 1 depends only on δ, and we write the probability as $\mathrm{DCP}^{\mathrm{KGA}}(\delta)$, which is given as

$$\mathrm{DCP}^{\mathrm{KGA}}(\delta) = \mathrm{DP}^{f_1}(\delta) \times \mathrm{DP}^{f_3}(\delta) \times \mathrm{DP}^{f_5}(\delta) \times \mathrm{DP}^{f_7}(\delta).$$

We present the following lemma with respect to $\mathrm{DCP}^{\mathrm{KGA}}(\delta)$.

Lemma 2. *There exists non-zero $\delta \in \{0,1\}^{32}$ such that* $\mathrm{DCP}^{\mathrm{KGA}}(\delta) > 2^{-128}$.

Proof. For all the $(2^{32} - 1)$ possible values `0x00000001`, …, `0xffffffff` of non-zero $\delta \in \{0,1\}^{32}$, we computed the value of $\mathrm{DCP}^{\mathrm{KGA}}(\delta)$. The results are summarized in Table 3. From the table, we see that there exist 89938 values of δ such that $\mathrm{DCP}^{\mathrm{KGA}}(\delta) > 2^{-128}$. □

We note that, for $\delta =$ `0xd7d7d0d7`, we have

$$\mathrm{DP}^{f_1}(\delta) = 2^{-25}, \mathrm{DP}^{f_3}(\delta) = 2^{-26}, \mathrm{DP}^{f_5}(\delta) = 2^{-26}, \text{ and } \mathrm{DP}^{f_7}(\delta) = 2^{-26}. \quad (5)$$

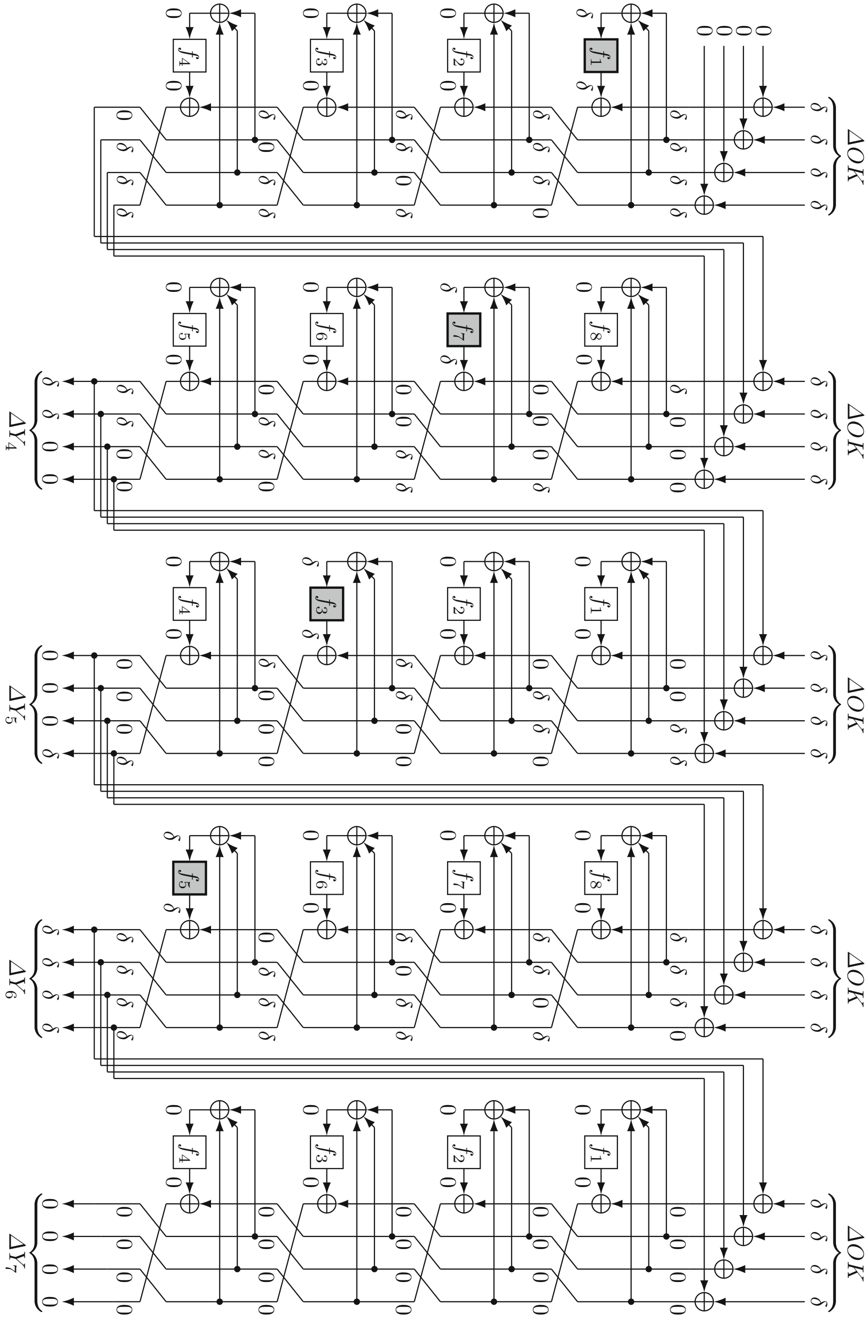

Fig. 4. The differential characteristic and active f_i functions (shown in gray) of Lemma 1

Table 1. The differential characteristic and active f_i functions of Lemma 1

r	Input diff. $\Delta X^{(r)}$	Output diff. $\Delta Z^{(r)}$	Active f_i function
1	$(\delta,\delta,\delta,\delta)$	$(\delta,\delta,\delta,0)$	f_1
2	$(\delta,\delta,\delta,0)$	$(\delta,\delta,0,\delta)$	
3	$(\delta,\delta,0,\delta)$	$(\delta,0,\delta,\delta)$	
4	$(\delta,0,\delta,\delta)$	$(0,\delta,\delta,\delta)$	
5	$(\delta,0,0,0)$	$(0,0,0,\delta)$	
6	$(0,0,0,\delta)$	$(0,0,\delta,\delta)$	f_7
7	$(0,0,\delta,\delta)$	$(0,\delta,\delta,0)$	
8	$(0,\delta,\delta,0)$	$(\delta,\delta,0,0)$	
9	$(0,0,\delta,\delta)$	$(0,\delta,\delta,0)$	
10	$(0,\delta,\delta,0)$	$(\delta,\delta,0,0)$	
11	$(\delta,\delta,0,0)$	$(\delta,0,0,0)$	f_3
12	$(\delta,0,0,0)$	$(0,0,0,\delta)$	
13	$(\delta,\delta,\delta,0)$	$(\delta,\delta,0,\delta)$	
14	$(\delta,\delta,0,\delta)$	$(\delta,0,\delta,\delta)$	
15	$(\delta,0,\delta,\delta)$	$(0,\delta,\delta,\delta)$	
16	$(0,\delta,\delta,\delta)$	$(\delta,\delta,\delta,\delta)$	f_5
17	$(0,0,0,0)$	$(0,0,0,0)$	
18	$(0,0,0,0)$	$(0,0,0,0)$	
19	$(0,0,0,0)$	$(0,0,0,0)$	
20	$(0,0,0,0)$	$(0,0,0,0)$	

Table 2. The number of active f_i functions for a given input difference

Input diff. ΔOK	Number	Input diff. ΔOK	Number
$(0,0,0,\delta)$	9	$(\delta,0,0,\delta)$	10
$(0,0,\delta,0)$	9	$(\delta,0,\delta,0)$	10
$(0,0,\delta,\delta)$	10	$(\delta,0,\delta,\delta)$	7
$(0,\delta,0,0)$	9	$(\delta,\delta,0,0)$	10
$(0,\delta,0,\delta)$	10	$(\delta,\delta,0,\delta)$	7
$(0,\delta,\delta,0)$	10	$(\delta,\delta,\delta,0)$	7
$(0,\delta,\delta,\delta)$	7	$(\delta,\delta,\delta,\delta)$	4
$(\delta,0,0,0)$	9		

Table 3. Examples of δ that satisfies $\mathrm{DCP}^{\mathrm{KGA}}(\delta) > 2^{-128}$ and the number of such δ

$\mathrm{DCP}^{\mathrm{KGA}}(\delta)$	Example of δ	Number
2^{-103}	`0xd7d7d0d7`	1
2^{-104}	`0xc5c5d254`	1
2^{-105}	`0x4e4ec554`	1
2^{-106}	`0x3c3cf4ff`	8
2^{-107}	`0x6161f9d9`	1
2^{-108}	`0x054d9797`	34
2^{-109}	`0x0101019a`	157
2^{-110}	`0x0159591a`	1579
2^{-111}	`0x0101e818`	7685
2^{-112}	`0x01010520`	80471

Existence of Equivalent Keys. We are now ready to present our main result of this section. Fix any δ such that $\mathrm{DCP}^{\mathrm{KGA}}(\delta) > 2^{-128}$. For randomly chosen $OK_1 \in \{0,1\}^{128}$, (1) is satisfied for

$$\begin{cases} \Delta OK_1 = (\delta,\delta,\delta,\delta), \\ \Delta Y_4 = (\delta,\delta,0,0), \Delta Y_5 = (0,0,0,\delta), \Delta Y_6 = (\delta,\delta,\delta,\delta), \Delta Y_7 = (0,0,0,0) \end{cases} \quad (6)$$

with at least a probability of $\mathrm{DCP}^{\mathrm{KGA}}(\delta)$. This implies that at least $2^{128} \times \mathrm{DCP}^{\mathrm{KGA}}(\delta)$ values of $OK_1 \in \{0,1\}^{128}$ satisfy (1). Similarly, at least $2^{128} \times \mathrm{DCP}^{\mathrm{KGA}}(\delta)$ values of $OK_2 \in \{0,1\}^{128}$ satisfy (2) for

$$\begin{cases} \Delta OK_2 = (\delta,\delta,\delta,\delta), \\ \Delta Z_4 = (\delta,\delta,0,0), \Delta Z_5 = (0,0,0,\delta), \Delta Z_6 = (\delta,\delta,\delta,\delta), \Delta Z_7 = (0,0,0,0). \end{cases} \quad (7)$$

If we fix a value of $(OK_1, OK_2) \in \{0,1\}^{256}$ that satisfies (1), (6), (2), and (7), we see that (3) is also satisfied, and hence we obtain four equivalent keys (or two pairs of equivalent keys) of (4). From Table 3 and by eliminating the duplications, the number of equivalent keys can be derived as

$$\frac{4 \times (2^{50} \times 1 + 2^{48} \times 1 + 2^{46} \times 1 + 2^{44} \times 8 + \cdots + 2^{32} \times 80471)}{4} \geq 2^{51.0},$$

and the number of pairs is the half of $2^{51.0}$, which is $2^{50.0}$.

From the discussions above, we obtain the following theorem.

Theorem 1. *In 256-bit key HyRAL, there exist* $2^{51.0}$ *equivalent keys (or* $2^{50.0}$ *pairs of equivalent keys).*

4 Derivation of Equivalent Keys

From the result of the previous section, we know that there are $2^{51.0}$ equivalent keys in 256-bit key HyRAL. In this section, we consider the problem of deriving a concrete instance of equivalent keys.

4.1 Equivalent Key Derivation Algorithm

As in the previous section, let $\mathrm{KGA} \in \{\mathrm{KGA}_1, \mathrm{KGA}_2\}$. Recall that one round of G_1 and G_2 functions are regarded as one round of KGA, and hence KGA is a function that consists of 20 rounds in total. Let $OK \in \{OK_1, OK_2\}$ be the input of KGA, and let $(K_1, K_2, K_3, K_4) \overset{32}{\leftarrow} OK \in \{0,1\}^{128}$ be its partition into 32-bit strings. Similarly, let $\mathrm{CST} \in \{\mathrm{CST}_1, \mathrm{CST}_2\}$ be the constant used in KGA, and let $(C_1, C_2, C_3, C_4) \overset{32}{\leftarrow} \mathrm{CST} \in \{0,1\}^{128}$ be its partition into 32-bit strings. KGA is the function that consists of 20 rounds in total, and we write the input and output strings of $f_i^{(r)}$ as $I_i^{(r)} \in \{0,1\}^{32}$ and $O_i^{(r)} \in \{0,1\}^{32}$, respectively, where $r = 1, 2, \ldots, 20$ and $f_i^{(r)}$ is f_i function used in the r-th round. Figure 5 shows the first 8 rounds of KGA.

We consider the case of $\delta = \texttt{0xd7d7d0d7}$. For $i \in \{1,3,5,7\}$, let $\mathcal{I}_i$ be a list of $I_i \in \{0,1\}^{32}$ that satisfies $f_i(I_i) \oplus f_i(I_i \oplus \delta) = \delta$. From (5), $\mathcal{I}_1$ consists of 128 elements, and each of $\mathcal{I}_3$, $\mathcal{I}_5$, and $\mathcal{I}_7$ consists of 64 elements, and we may thus write down the lists as $\mathcal{I}_1 = \{I_1[0], \ldots, I_1[127]\}$, $\mathcal{I}_3 = \{I_3[0], \ldots, I_3[63]\}$, $\mathcal{I}_5 = \{I_5[0], \ldots, I_5[63]\}$, and $\mathcal{I}_7 = \{I_7[0], \ldots, I_7[63]\}$.

Now if we can derive (K_1, K_2, K_3, K_4) that satisfies

$$I_1^{(1)} \in \mathcal{I}_1, I_7^{(6)} \in \mathcal{I}_7, I_3^{(11)} \in \mathcal{I}_3, \text{ and } I_5^{(16)} \in \mathcal{I}_5,$$

then this implies that we have derived OK that satisfies (1) and (6), or (2) and (7).

It is easy to obtain (K_1, K_2, K_3, K_4) that satisfies one of the four conditions, $I_1^{(1)} \in \mathcal{I}_1$, since this is simply (K_1, K_2, K_3, K_4) such that $K_2 \oplus C_2 \oplus K_3 \oplus C_3 \oplus K_4 \oplus C_4 \in \mathcal{I}_1$. In the following lemma, we show that one can derive (K_1, K_2, K_3, K_4) that satisfies two of the four conditions, namely, one can derive (K_1, K_2, K_3, K_4) such that both $I_1^{(1)} \in \mathcal{I}_1$ and $I_7^{(6)} \in \mathcal{I}_7$ are satisfied.

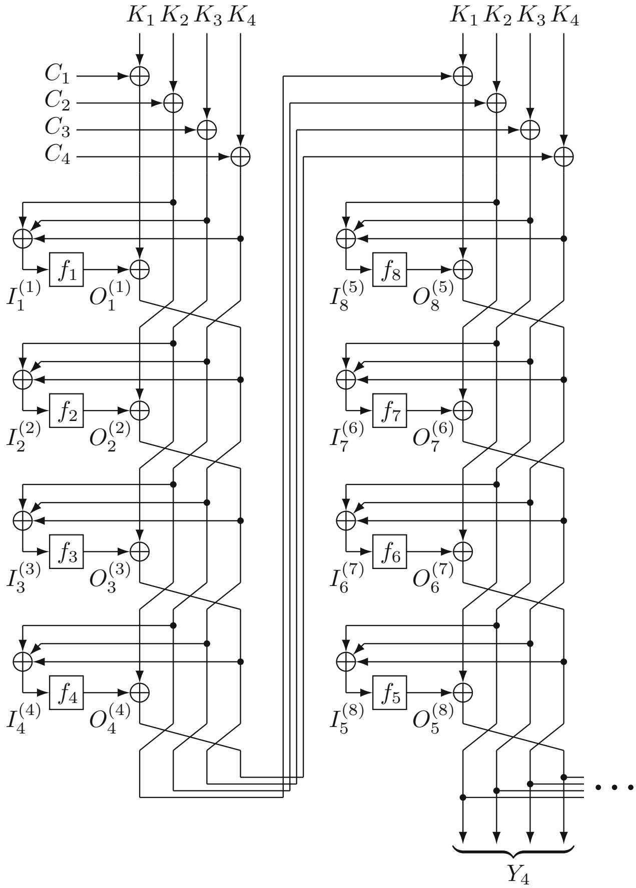

Fig. 5. The first 8 rounds of KGA

Lemma 3. *For arbitrarily fixed* $\tilde{K}_1$, $I_1^{(1)}$, $I_8^{(5)}$, *and* $I_7^{(6)}$, *where* $\tilde{K}_1 = K_1 \oplus K_3$, *the corresponding value of* (K_1, K_2, K_3, K_4) *can be derived.*

Proof. Since $I_1^{(1)}$ and $I_8^{(5)}$ are fixed, $O_1^{(1)} = f_1(I_1^{(1)})$ and $O_8^{(5)} = f_8(I_8^{(5)})$ are also fixed. To simplify the notation, let $\tilde{C}_1, \ldots, \tilde{C}_5$ be the fixed constants defined as $\tilde{C}_1 = C_1 \oplus C_3 \oplus C_4 \oplus O_1^{(1)}$, $\tilde{C}_2 = C_1 \oplus C_3 \oplus I_1^{(1)} \oplus O_1^{(1)}$, $\tilde{C}_3 = C_1 \oplus C_4 \oplus I_1^{(1)} \oplus O_1^{(1)}$, $\tilde{C}_4 = C_2 \oplus C_3 \oplus C_4$, and $\tilde{C}_5 = C_1 \oplus C_2 \oplus O_1^{(1)} \oplus I_7^{(6)}$. We also let $\tilde{K}_2 = K_1 \oplus K_3 \oplus K_4$ and $\tilde{K}_3 = K_1 \oplus K_4$.

First, $I_1^{(1)}$ has to satisfy $I_1^{(1)} = K_2 \oplus C_2 \oplus K_3 \oplus C_3 \oplus K_4 \oplus C_4$, which is equivalent to

$$K_2 = I_1^{(1)} \oplus C_2 \oplus K_3 \oplus C_3 \oplus K_4 \oplus C_4. \tag{8}$$

Next, since $I_2^{(2)} = \tilde{K}_2 \oplus \tilde{C}_1$, we have

$$O_2^{(2)} = f_2(\tilde{K}_2 \oplus \tilde{C}_1). \tag{9}$$

Similarly, since $I_3^{(3)}$ can be written as $I_3^{(3)} = \tilde{K}_1 \oplus \tilde{C}_2 \oplus O_2^{(2)}$ by using (8), we obtain

$$O_3^{(3)} = f_3(\tilde{K}_1 \oplus \tilde{C}_2 \oplus O_2^{(2)}). \tag{10}$$

Besides, since $I_4^{(4)}$ can be written as $I_4^{(4)} = \tilde{K}_3 \oplus \tilde{C}_3 \oplus O_2^{(2)} \oplus O_3^{(3)}$ by using (8), we obtain

$$O_4^{(4)} = f_4(\tilde{K}_3 \oplus \tilde{C}_3 \oplus O_2^{(2)} \oplus O_3^{(3)}). \tag{11}$$

Now since the input of the 5th round is $(C_1 \oplus O_1^{(1)}, C_2 \oplus O_2^{(2)}, C_3 \oplus O_3^{(3)}, C_4 \oplus O_4^{(4)})$ and $I_8^{(5)}$ is fixed,

$$I_8^{(5)} = \tilde{C}_4 \oplus O_2^{(2)} \oplus O_3^{(3)} \oplus O_4^{(4)} \tag{12}$$

has to be satisfied. Furthermore, since $I_7^{(6)}$ is fixed, $I_7^{(6)} = C_1 \oplus C_3 \oplus C_4 \oplus O_1^{(1)} \oplus O_3^{(3)} \oplus O_4^{(4)} \oplus O_8^{(5)}$ needs to be satisfied, which is equivalent to

$$\tilde{C}_5 \oplus O_2^{(2)} \oplus I_8^{(5)} = O_8^{(5)} \tag{13}$$

by using (12).

At this point, since $\tilde{C}_5$, $I_8^{(5)}$, and $O_8^{(5)}$ are all fixed, $O_2^{(2)}$ that satisfies (13) is uniquely determined. As we have now fixed $O_2^{(2)}$, $\tilde{K}_2$ that satisfies (9) is also uniquely determined, which is $\tilde{K}_2 = f_2^{-1}(O_2^{(2)}) \oplus \tilde{C}_1$. We also see that since $O_2^{(2)}$ is now fixed and $\tilde{K}_1$ is a fixed constant, $O_3^{(3)}$ that satisfies (10) is now uniquely fixed. Upon fixing both $O_2^{(2)}$ and $O_3^{(3)}$, we obtain unique $O_4^{(4)}$ that satisfies (12), and for these fixed $O_2^{(2)}$, $O_3^{(3)}$, and $O_4^{(4)}$, we obtain the corresponding $\tilde{K}_3$, which is $\tilde{K}_3 = f_4^{-1}(O_4^{(4)}) \oplus \tilde{C}_3 \oplus O_2^{(2)} \oplus O_3^{(3)}$.

Finally, we obtain (K_1, K_2, K_3, K_4) as $(K_1, K_2, K_3, K_4) \leftarrow (\tilde{K}_1 \oplus \tilde{K}_2 \oplus \tilde{K}_3, \tilde{K}_1 \oplus \tilde{K}_3 \oplus I_1^{(1)} \oplus \tilde{C}_4, \tilde{K}_2 \oplus \tilde{K}_3, \tilde{K}_1 \oplus \tilde{K}_2)$. □

We are now ready to present the basic version of our equivalent key derivation algorithm based on Lemma 3.

1. Fix arbitrarily $I_1^{(1)}$ and $I_7^{(6)}$ that satisfy $I_1^{(1)} \in \mathcal{I}_1$ and $I_7^{(6)} \in \mathcal{I}_7$.
2. Fix arbitrarily $I_8^{(5)}$ and $\tilde{K}_1$.
3. Derive (K_1, K_2, K_3, K_4) by using Lemma 3.
4. Compute $I_3^{(11)}$ from (K_1, K_2, K_3, K_4) by following the specification of 256-bit key HyRAL, and proceed to Step 5 if $I_3^{(11)} \in \mathcal{I}_3$ is satisfied. Otherwise return to Step 2.

5. Compute $I_5^{(16)}$ from (K_1, K_2, K_3, K_4) by following the specification of 256-bit key HyRAL, and output (K_1, K_2, K_3, K_4) and halt if $I_5^{(16)} \in \mathcal{I}_5$ is satisfied. Otherwise return to Step 2.

If we assume that $I_3^{(11)}$ and $I_5^{(16)}$ are independently and uniformly distributed random strings over $\{0,1\}^{32}$, then the probability that both $I_3^{(11)} \in \mathcal{I}_3$ and $I_5^{(16)} \in \mathcal{I}_5$ are satisfied is $(64/2^{32})^2 = 2^{-52}$, since there are 64 elements in each of $\mathcal{I}_3$ and $\mathcal{I}_5$. Therefore, we may expect that the algorithm returns (K_1, K_2, K_3, K_4) after trying 2^{52} values of $(I_8^{(5)}, \tilde{K}_1)$.

4.2 Time Complexity of the Algorithm

In the basic algorithm presented in Sect. 4.1, the test $I_3^{(11)} \in \mathcal{I}_3$ is executed for 2^{52} different values of $(I_8^{(5)}, \tilde{K}_1)$. This test of $I_3^{(11)} \in \mathcal{I}_3$ is the main cost in the time complexity of the algorithm, and the following lemma can be used in the actual implementation.

Lemma 4. *For arbitrarily fixed $\tilde{K}_1$, $I_1^{(1)}$, $O_1^{(1)}$, $I_8^{(5)}$, $I_7^{(6)}$, and $O_7^{(6)}$, the corresponding value of $I_3^{(11)}$ can be derived by seven computations of f_i functions.*

Proof. $I_3^{(11)}$ can be derived by the following steps.

1. $O_8^{(5)} \leftarrow f_8(I_8^{(5)})$
2. $O_2^{(2)} \leftarrow \tilde{C}_5 \oplus I_8^{(5)} \oplus O_8^{(5)}$
3. $\tilde{K}_2 \leftarrow f_2^{-1}(O_2^{(2)}) \oplus \tilde{C}_1$
4. $O_3^{(3)} \leftarrow f_3(\tilde{K}_1 \oplus \tilde{C}_2 \oplus O_2^{(2)})$
5. $O_4^{(4)} \leftarrow \tilde{C}_4 \oplus I_8^{(5)} \oplus O_2^{(2)} \oplus O_3^{(3)}$
6. $O_6^{(7)} \leftarrow f_6(C_1 \oplus C_2 \oplus C_4 \oplus O_1^{(1)} \oplus O_2^{(2)} \oplus O_4^{(4)} \oplus O_8^{(5)} \oplus O_7^{(6)})$
7. $O_5^{(8)} \leftarrow f_5(C_1 \oplus C_2 \oplus C_3 \oplus O_1^{(1)} \oplus O_2^{(2)} \oplus O_3^{(3)} \oplus O_8^{(5)} \oplus O_7^{(6)} \oplus O_6^{(7)})$
8. $O_1^{(9)} \leftarrow f_1(I_1^{(1)} \oplus O_2^{(2)} \oplus O_3^{(3)} \oplus O_4^{(4)} \oplus O_7^{(6)} \oplus O_6^{(7)} \oplus O_5^{(8)})$
9. $O_2^{(10)} \leftarrow f_2(\tilde{K}_2 \oplus \tilde{C}_1 \oplus O_3^{(3)} \oplus O_4^{(4)} \oplus O_8^{(5)} \oplus O_6^{(7)} \oplus O_5^{(8)} \oplus O_1^{(9)})$
10. $I_3^{(11)} \leftarrow \tilde{K}_1 \oplus C_1 \oplus C_3 \oplus I_1^{(1)} \oplus O_1^{(1)} \oplus O_2^{(2)} \oplus O_4^{(4)} \oplus O_8^{(5)} \oplus O_7^{(6)} \oplus O_5^{(8)} \oplus O_1^{(9)} \oplus O_2^{(10)}$

We see that the above steps run with seven computations of f_i functions. □

In the proof of Lemma 4, one can run Steps 1, 2, and 3 without using $\tilde{K}_1$. Therefore, one possible implementation is to search 2^{52} values of $(I_8^{(5)}, \tilde{K}_1)$ by searching 2^{20} values of $I_8^{(5)}$, and for each value of $I_8^{(5)}$, we first run Steps 1, 2, and 3 and then search all the 2^{32} possible values of $\tilde{K}_1$. Then the main cost of running the algorithm becomes 5×2^{52} computations of f_i functions assuming that 2^{26} computations of f_i functions can be ignored.

In order to derive both OK_1 and OK_2, we need to run the algorithm twice by changing the constant (C_1, C_2, C_3, C_4), and hence the time complexity of the algorithm is 10×2^{52} computations of f_i functions, which amount to running $2^{48.8}$ encryption functions as there are 96 f_i functions in the encryption function of 256-bit key HyRAL. We note that the memory requirement of the algorithm is small.

Table 4. Summary of the implementation. The "Cores" column indicates the number of cores used in running the program

	System	Queue name	Cores	Search range of $I_8^{(5)}$	Number of $(I_8^{(5)}, \tilde{K}_1)$	Running time
OK_1	HX600	h1024	1024	0x00000000, ..., 0x0000ffff	2^{48}	8h 48min 56s
		h1024	1024	0x00010000, ..., 0x0001ffff	2^{48}	8h 28min 4s
OK_2	FX1	f1024	1024	0x00000000, ..., 0x0003ffff	2^{50}	50h 36min 2s
		f512	512	0x00040000, ..., 0x0007ffff	2^{50}	92h 24min 15s
	HX600	h256	256	0x00080000, ..., 0x0009ffff	2^{49}	67h 42min 47s
		h256	256	0x000a0000, ..., 0x000bffff	2^{49}	67h 29min 1s
		h256	256	0x000c0000, ..., 0x000dffff	2^{49}	67h 34min 55s
		h256	256	0x000e0000, ..., 0x000fffff	2^{49}	67h 29min 57s

5 Deriving Equivalent Keys

We have implemented our algorithm in Sect. 4.2 on a supercomputer system. The systems we used are the server systems called HX600 and FX1. HX600 has 96 nodes, which are equivalent to 384 CPUs or 1536 cores, it has a total of 6TB of memory, and the CPU is AMD Opteron 8380 (4 cores, 2.5GHz). FX1 has 768 nodes, which are equivalent to 768 CPUs or 3072 cores, it has 24TB of memory, and CPU is SPARC64 VII (4 cores, 2.52GHz). We used C language for the implementation of the algorithm, and MPI library for the message passing library for the parallel process execution.

The values of δ, $I_1^{(1)}$, and $I_7^{(6)}$ that were used in the implementation are δ = 0xd7d7d0d7, $I_1^{(1)}$ = 0x17170c17, and $I_7^{(6)}$ = 0x1717292b. For deriving OK_1, we searched 2^{17} values of $I_8^{(5)}$, and for each value of $I_8^{(5)}$, we searched all the 2^{32} possible values of $\tilde{K}_1$. The program was divided into two programs by halving the search range of $I_8^{(5)}$, and a total of 2^{49} values of $(I_8^{(5)}, \tilde{K}_1)$ were tested. For deriving OK_2, we searched 2^{20} values of $I_8^{(5)}$, and for each value of $I_8^{(5)}$, we searched all the 2^{32} possible values of $\tilde{K}_1$. The program was divided into six programs depending on the range of $I_8^{(5)}$, and a total of 2^{52} values of $(I_8^{(5)}, \tilde{K}_1)$ were tested. The summary of the implementation is in Table 4.

As a result, we have successfully derived one value of OK_1 and three values of OK_2. The values, together with the corresponding values of $I_8^{(5)}$ and $\tilde{K}_1$, are in Table 5.

Table 5. Results of running the algorithm in Sect. 4.2

OK_1	0x2fd918837136d461f4bc99938907dd0b ($I_8^{(5)}$ = 0x00014b73, $\tilde{K}_1$ = 0xdb658110)
OK_2	0xa20ed0f467141b2a3b038abb5f61d59e ($I_8^{(5)}$ = 0x0005b394, $\tilde{K}_1$ = 0x990d5a4f)
	0xe3a1902aa60b6c3582a9131527d43b2f ($I_8^{(5)}$ = 0x000f8a7f, $\tilde{K}_1$ = 0x6108833f)
	0x3218a5b25828a0b7d2122283894cc63b ($I_8^{(5)}$ = 0x000f9953, $\tilde{K}_1$ = 0xe00a8731)

For $\delta = \texttt{0xd7d7d0d7}$, $\Delta OK_1 = \Delta OK_2 = (\delta, \delta, \delta, \delta)$, and OK_1 and OK_2 in Table 5, (K, K') in (4) are all equivalent keys, which can be verified by the reference code available in [6].

6 Discussions

The existence of equivalent keys generally implies that the cipher is theoretically cryptanalyzed, as the time complexity of the brute-force attack becomes less than the time complexity implied by its key length. As there are $2^{50.0}$ pairs of equivalent keys in 256-bit key HyRAL, the search space of the brute-force attack is reduced from 2^{256} to $2^{256} - 2^{50.0}$. Although the fraction of equivalent keys, $2^{51.0}$, compared to 2^{256} is small, as a practical implication of identifying equivalent keys, in the rest of this section, we discuss well known observations that one can obtain collisions on the Davies-Meyer compression function based on 256-bit key HyRAL, and on the Merkle-Damgård hash function based on the compression function.

The Davies-Meyer Compression Function. Let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a blockcipher with k-bit keys and an n-bit block. The Davies-Meyer compression function $h : \{0,1\}^{n+k} \to \{0,1\}^n$, one of the standard constructions of a compression function, is defined as $h(H, M) = E_M(H) \oplus H$.

Let E be 256-bit key HyRAL. If we let (M, M') be one of the equivalent keys (K, K') in (4), then for any $H \in \{0,1\}^{128}$, we have $h(H, M) = h(H, M')$. Therefore, for each equivalent keys (K, K') in (4), one can generate 2^{128} different collisions $((H, M), (H, M'))$ on h.

The Merkle-Damgård Hash Function. Let $h : \{0,1\}^{n+k} \to \{0,1\}^n$ be a compression function. The Merkle-Damgård hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^n$ is the construction of a hash function from h, and is defined as follows. Let $H_0 \in \{0,1\}^n$ be a fixed initial value. For an input string $M \in \{0,1\}^*$, let $\tilde{M} \in \{0,1\}^{mn}$ be the padded string in a standard and appropriate way, and let $(M_1, M_2, \ldots, M_m) \xleftarrow{n} \tilde{M}$ be its partition into n-bit strings. The hash value $\mathcal{H}(M)$ is $H_m \in \{0,1\}^n$, where $H_i \leftarrow h(H_{i-1}, M_i)$ for $i = 1, 2, \ldots, m$.

Let E be 256-bit key HyRAL, h be the Davies-Meyer compression function based on E, and $\mathcal{H}$ be the Merkle-Damgård hash function based on h. Let $M, M' \in \{K, K'\}^m$ be bit strings such that $M \neq M'$, where (K, K') is any equivalent keys in (4). Assume that the standard padding is used, e.g., appending a bit "1" and then bits "0" followed by the encoding of the length of the input, then we have $\mathcal{H}(M) = \mathcal{H}(M')$ and hence we obtain a collision on $\mathcal{H}$.

For example, for $m = 3$, $(M, M') = ((K, K, K), (K', K', K'))$, $((K, K, K'), (K', K', K))$, $((K, K', K), (K', K, K'))$, and $((K, K', K'), (K', K, K))$ all satisfy $\mathcal{H}(M) = \mathcal{H}(M')$. Similarly, when M and M' are m blocks in length, we obtain 2^{m-1} different collisions.

7 Summary

We presented the analysis of 256-bit key HyRAL in terms of equivalent keys. We showed that there are $2^{50.0}$ pairs of equivalent keys, leading to the theoretical cryptanalysis of the cipher as a blockcipher with 256-bit keys. We also developed the algorithm to derive an instance of equivalent keys, and demonstrated that we were able to derive concrete instances with the current computing environment.

Acknowledgments. The experiment in Sect. 5 was conducted using a supercomputer system at Information Technology Center, Nagoya University. The authors would like to thank Hideki Ando for advice on the experiment. This work was supported in part by CRYPTREC and in part by MEXT KAKENHI, Grant-in-Aid for Young Scientists (A), 22680001.

References

1. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999)
2. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. J. Cryptology 4(1), 3–72 (1991)
3. Cryptography Research and Evaluation Committees (CRYPTREC), `http://www.cryptrec.go.jp/english/index.html`
4. Cryptography Research and Evaluation Committees (CRYPTREC): CRYPTREC Report, Report of the Scheme Committee (2010) (in Japanese), `http://www.cryptrec.go.jp/english/report.html`
5. Daemen, J., Knudsen, L.R., Rijmen, V.: The Block Cipher SQUARE. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 149–165. Springer, Heidelberg (1997)
6. Hirata, K.: Submission Documents of HyRAL to the CRYPTREC Project (2010), `http://www.cryptrec.go.jp/english/topics/cryptrec_20101001_callforattack.html`
7. Hirata, K.: The 128bit Block Cipher HyRAL (Hybrid Randomization Algorithm): Common Key Block Cipher. In: Proceedings of the 2010 International Symposium on Intelligence Information Processing and Trusted Computing, IPTC 2010, pp. 9–14. IEEE Computer Society, Washington, DC (2010), `http://dx.doi.org/10.1109/IPTC.2010.179`
8. Hirata, K.: The 128bit Blockcipher HyRAL. In: The 2010 Symposium on Cryptography and Information Security, 1D1-1, SCIS 2010 (2010) (in Japanese)
9. Igarashi, Y., Takagi, Y., Kaneko, T.: Security Evaluation of HyRAL against Linear Cryptanalysis. In: The 2010 Symposium on Cryptography and Information Security, 1D1-3, SCIS 2010 (2010) (in Japanese)
10. Inoue, T., Kaneko, T.: Security Evaluation of HyRAL against Boomerang Attack. IEICE Tech. Rep. 111(142), 1–6 (2011) (in Japanese); IT 2011-07-14
11. Iwata, T.: Security Evaluation Report of HyRAL. In: Technical Report of CRYPTREC, Investigation Reports Related to Cryptographic Techniques in FY 2010 (2011) (in Japanese)
12. Knudsen, L.R.: Cryptanalysis of LOKI. In: Matsumoto, T., Imai, H., Rivest, R.L. (eds.) ASIACRYPT 1991. LNCS, vol. 739, pp. 22–35. Springer, Heidelberg (1993)

13. Knudsen, L.R.: Truncated and Higher Order Differentials. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 196–211. Springer, Heidelberg (1995)
14. Lai, X.: Higher Order Derivatives and Differential Cryptanalysis. In: Blahut, R.E., Massey, J.L. (eds.) Communications and Cryptography: Two Sides of One Tapestry, pp. 227–233. Kluwer Academic Publishers, Norwell (1994)
15. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
16. Matsui, M.: The First Experimental Cryptanalysis of the Data Encryption Standard. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 1–11. Springer, Heidelberg (1994)
17. Shibayama, N., Igarashi, Y., Kaneko, T., Hangai, S.: Impossible Differential Attack on HyRAL. In: Forum on Information Technology, L-022 (2010) (in Japanese)
18. Shibayama, N., Igarashi, Y., Kaneko, T., Hangai, S.: On Impossible Differential of HyRAL Using MDS Characteristic. In: The 2010 IEICE Engineering Sciences Society Conference, A-7-8 (2010) (in Japanese)
19. Shibayama, N., Igarashi, Y., Kaneko, T., Hangai, S.: Security Evaluation of HyRAL against Saturation Cryptanalysis. In: The 33rd Symposium on Information Theory and its Applications, SITA 2010, 10.1 (2010) (in Japanese)
20. Shibayama, N., Igarashi, Y., Kaneko, T., Hangai, S.: Higher Order Differential Attack on HyRAL. IEICE Tech. Rep. 110(443), 341–347 (2011) (in Japanese); ISEC 2010-123
21. Shibayama, N., Igarashi, Y., Kaneko, T., Hangai, S.: Security Evaluation of HyRAL against Saturation Cryptanalysis (II). IEICE Tech. Rep. 111(123), 103–109 (2011) (in Japanese); ISEC 2011-19
22. Shibayama, N., Kaneko, T., Hangai, S.: New Saturation Characteristics of HyRAL. IEICE Tech. Rep. 111(455), 53–60 (2012) (in Japanese); ISEC 2011-81
23. Taga, B., Tanaka, H.: Higher Order Differential Characteristics of HyRAL. In: The 2011 Symposium on Cryptography and Information Security, 2B2-2, SCIS 2011 (2011) (in Japanese)
24. Takagi, Y., Igarashi, Y., Kaneko, T.: Security Evaluation of HyRAL against Differential Attack. In: The 2010 Symposium on Cryptography and Information Security, 1D1-2, SCIS 2010 (2010) (in Japanese)
25. Yamaguchi, Y., Shibayama, N., Kaneko, T.: Higher Order Differential Property of HyRAL (II). In: The 2012 Symposium on Cryptography and Information Security, 1C3-4, SCIS 2012 (2012) (in Japanese)
26. Yamaguchi, Y., Igarashi, Y., Kaneko, T.: Higher Order Differential Property of HyRAL. In: The 63rd Joint Conference of Electrical and Electronics Engineers in Kyushu, 02-1A-06 (2010) (in Japanese)
27. Youm, H.Y., Song, J.H., Lee, S.Y.: Security Analysis of HyRAL. In: Technical Report of CRYPTREC, Investigation Reports Related to Cryptographic Techniques in FY 2010 (2011)

A Details of $f_1, \ldots, f_8$ Functions

We present the details of the specification of $f_1, \ldots, f_8$ functions that were omitted from Sect. 2.

$f_1, \ldots, f_8$ functions are permutations over $\{0,1\}^{32}$. For a given input $I = (x_1, x_2, x_3, x_4) \in \{0,1\}^{32}$, f_i function generates the output as follows.

1. Let $(x_1, x_2, x_3, x_4) \leftarrow T_i(x_1, x_2, x_3, x_4)$.
2. Let $(x_1, x_2, x_3, x_4) \leftarrow (S(x_1), S(x_2), S(x_3), S(x_4))$.
3. Compute (o_1, o_2, o_3, o_4) by

$$\begin{pmatrix} o_1 \\ o_2 \\ o_3 \\ o_4 \end{pmatrix} \leftarrow \begin{pmatrix} \texttt{0x03} & \texttt{0x03} & \texttt{0x02} & \texttt{0x01} \\ \texttt{0x01} & \texttt{0x02} & \texttt{0x02} & \texttt{0x02} \\ \texttt{0x07} & \texttt{0x03} & \texttt{0x01} & \texttt{0x02} \\ \texttt{0x07} & \texttt{0x04} & \texttt{0x05} & \texttt{0x03} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \oplus \begin{pmatrix} \texttt{0x11} \\ \texttt{0x22} \\ \texttt{0x44} \\ \texttt{0x88} \end{pmatrix},$$

   where the arithmetic is over GF(2^8) defined by the irreducible polynomial $p(x) = x^8 + x^4 + x^3 + x + 1$.
4. The output is $O = (o_1, o_2, o_3, o_4)$.

T_i function is defined in Table 6. The S-box S is the composition of an affine mapping over GF(2) and the inversion over GF(2^8). Table 7 shows the input and output of the S-box. The values are in hexadecimal form. The input x is regarded as two hexadecimal digits, and if the first digit is i and the last is j, then the output is a value written in the i-th row and j-th column. For example, $S(\texttt{0x12}) = \texttt{0x06}$.

Table 6. T_i function

i	$T_i(x_1, x_2, x_3, x_4)$
1	(x_1, x_2, x_3, x_4)
2	(x_2, x_3, x_4, x_1)
3	(x_3, x_4, x_1, x_2)
4	(x_4, x_1, x_2, x_3)
5	(x_4, x_3, x_2, x_1)
6	(x_3, x_2, x_1, x_4)
7	(x_2, x_1, x_4, x_3)
8	(x_1, x_4, x_3, x_2)

Table 7. S-box S

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	5e	d3	af	36	43	a6	49	33	93	3b	21	91	df	47	f4
1	b6	70	06	d0	81	82	fa	a1	10	b5	3c	ba	97	85	b7	79
2	ed	5c	ca	05	87	bf	24	4c	51	ec	17	61	22	f0	3e	18
3	a7	64	13	ab	e9	09	25	54	2d	31	69	f5	37	67	fe	1d
4	0b	28	a3	2f	e4	0f	d4	da	1b	fc	e6	ac	53	04	27	a9
5	94	8b	d5	c4	90	6b	f8	9d	c5	db	ea	e2	ae	63	07	7a
6	5b	23	34	38	03	8c	46	68	cd	1a	1c	41	7d	a0	9c	dd
7	08	4e	e3	d7	1e	b3	50	5d	c6	0e	ad	cf	d6	eb	0d	b1
8	fb	7c	c3	2e	65	48	b8	8f	ce	e7	62	d2	12	4a	c8	26
9	a5	8e	3d	76	86	57	bc	bd	11	75	71	78	1f	ef	e0	0c
a	de	6a	6d	32	84	72	8a	d8	f9	dc	9a	89	9f	88	14	2a
b	9b	9e	d9	95	b9	a4	02	f7	96	73	56	be	7f	80	7e	83
c	00	01	f6	8d	7b	d1	52	cb	b0	e1	c7	e5	29	c0	4f	e8
d	58	3f	cc	fd	ee	b2	40	ff	99	2b	5f	60	aa	4b	b4	74
e	2c	45	6c	92	66	42	39	f3	77	bb	19	59	20	6f	35	f2
f	c1	0a	15	98	a2	c2	44	30	55	4d	c9	a8	5a	f1	6e	3a