Improved Key Recovery Attack on Round-reduced Hierocrypt-L1 in the Single-Key Setting

Ahmed Abdelkhalek, Mohamed Tolba, and Amr M. Youssef

Concordia Institute for Information Systems Engineering,
Concordia University, Montréal, Quebéc, Canada

Abstract. Hierocrypt-L1 is a 64-bit block cipher with a 128-bit key. It was selected among the Japanese e-Government 2003 recommended ciphers list and has been reselected in the 2013 candidate recommended ciphers list. In this work, we cryptanalyze Hierocrypt-L1 in the single-key setting. In particular, we construct a 5 S-box layers distinguisher that we utilize to launch a meet-in-the-middle attack on 8 S-box layers round-reduced Hierocrypt-L1 using the differential enumeration technique. Our attack allows us to recover the master key with data complexity of 2^{49} chosen plaintexts, time complexity of $2^{114.8}$ 8-Sbox layers Hierocrypt-L1 encryptions and memory complexity of 2^{106} 64-bit blocks. Up to the authors' knowledge, this is the first cryptanalysis result that reaches 8 S-box layers of Hierocrypt-L1 in the single-key setting.
Keywords: Cryptanalysis, Hierocrypt-L1, Meet-in-the-Middle attack, Differential Enumeration.

1 Introduction

Hierocrypt-L1 (HC-L1) [24,10,29], designed by Toshiba Corporation in 2000, is a 64-bit block cipher with a 128-bit key. The cipher employs a nested Substitution Permutation Network (SPN) structure [24], where each S-box in a higher SPN level encompasses the lower-level SPN structure. HC-L1 was submitted to the New European Schemes for Signatures, Integrity, and Encryption (NESSIE) project [23]. In 2003, HC-L1 was selected as one of the Japanese e-Government recommended ciphers [9], and its security was reaffirmed by CRYPTREC in 2013 where it was included in the candidate recommended ciphers list [8].

The best known attack on HC-L1 in the single-key setting is the square attack on 7 S-box layers which was proposed by the designers [19] and independently by Barreto *et al.* [5]. Later, Cheon *et al.* proposed a 4 S-box layers impossible differential [29] and utilized it to attack HC-L1 reduced to 6 S-box layers. In the related-key setting, Taga *et al.* utilized a differential characteristic in the key scheduling of HC-L1 to attack 8 S-box layers [28].

The meet-in-the-middle (MitM) attack, first proposed by Diffie and Hellman in 1977, is considered as one of the major attacks on cryptographic primitives. Following its introduction, a number of variants has been developed to study the security of many block ciphers such as DES [15], XTEA, [27], KTANTAN [7], LBlock [2], and Twine [6]. MitM attacks were also employed in the cryptanalysis of a number of hash functions [17,26,3] and public key cryptosystems [18].

A new line of research was opened when Demirci and Selçuk presented the first MitM attack on AES at FSE 2008 [12]. They presented small data complexity attacks on 8-round AES-256 and 7-round AES-192 using time/memory trade-off. The major downside of their attack is the high memory required to store a precomputation table. The issue of the high memory requirement remained severe untill Dunkelman, Keller, and Shamir [14] introduced the notions of multisets and differential enumeration that have reduced the memory requirement drastically but with higher data complexity. Furthermore, Derbez, Fouque and Jean revised Dunkleman *et al.*'s attack, by borrowing ideas from the rebound attack [22], rendering the attack feasible on AES-128 [13]. Then, Li, Jia and Wang [20] presented attacks on 9-round AES-192 and 8-round PRINCE using a key-dependent sieving. MitM attacks were also applied on other block ciphers such as Hierocrypt-3 [1] and mCrypton [16].

In this work, we first construct a 5 S-box layers truncated differential characteristic for HC-L1. Then, we utilize this characteristic as a distinguisher to launch a MitM attack based on the differential enumeration technique against HC-L1 reduced to 8 S-box layers. Unlike the majority of existing MitM attack results, the matching step in our attack is performed around the linear transformation. Particularly, in the offline phase, we compute two specific bytes of the input of the linear transformation and store their xor in a precomputation table. Then, in the online phase, we compute two particular bytes of the output of that linear transformation, compute their xor which is equivalent to the xor of the two input bytes, and look for a match in the precomputation table. If no match is found, the key is discarded. Our attack recovers the master key with data complexity of 2^{49} chosen plaintexts, time complexity of $2^{114.8}$ 8-Sbox layers HC-L1 encryptions and memory complexity of 2^{106} 64-bit blocks.

The rest of the paper is organized as follows. Section 2 provides a description of HC-L1 and the notations adopted in the paper. Section 3 describes our 5 S-box layers distinguisher and how it is used to launch our MitM attack to recover the master key. Then, our conclusion is provided in Section 4.

2 Specification of Hierocrypt-L1

HC-L1 is an iterated block cipher with 64-bit blocks and 128-bit key. It adopts a nested SPN construction which embeds a lower level SPN structure within a higher SPN one. It has 6 rounds where the last round is slightly different than

the others. As shown in Figure 1, the higher SPN level of HC-L1 consists of the following three operations:

- AK: Mixes 64-bit layer key with the 64-bit internal state.
- XS: Two 32×32-bit keyed substitution boxes that are applied simultaneously to the internal state.
- MS: A diffusion layer consisting of a byte-wise linear transform defined by the matrix

$$MDS_H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

The lower SPN level, i.e., the two 32×32-bit XS-boxes, as shown in Figure 1, comprises of:

- SB: A nonlinear byte bijective mapping layer which applies the same 8×8-bit S-box 8 times in parallel.
- MC: A diffusion layer consisting of a byte-wise linear transform defined by a 4×4 matrix called mds_l which is a Maximum Distance Separable (MDS) matrix [21,25].
- AK: The 64-bit layer key is divided into halves and each half is mixed with the 32-bit internal state of one XS box.
- SB: Another nonlinear byte bijective mapping layer which applies the same 8×8-bit S-box 8 times in parallel.

Each round of HC-L1 includes two S-box layers. The last round of HC-L1 is an output transformation where the MS linear transformation is substituted by an xor layer with a layer key. The full encryption function of HC-L1 where the ciphertext C is computed from the plaintext P is given by:

$$\begin{aligned} C = {} & AK[K_1^{(7)}] \circ ((SB \circ AK[K_2^{(6)}] \circ MC \circ SB) \circ AK[K_1^{(6)}]) \\ & \circ \cdots \circ (MS \circ (SB \circ AK[K_2^{(1)}] \circ MC \circ SB) \circ AK[K_1^{(1)}])(P) \end{aligned}$$

To facilitate the understanding of our attacks, we represent the internal state of HC-L1 as a 4×2 matrix, as depicted in Figure 2, where each 8-bit word in the i^{th} row and the j^{th} column of this matrix represent a state byte. Consequently, MC, similar to the MixColumns operation in AES, operates column-wise and MS affects the entire matrix. Moreover, we exploit the fact that both the linear transformations (MC, MS), and the key addition AK are linear and swap their order. In such case, the input data is first xored with an equivalent layer key, denoted by EK_i, and then the linear transformation is applied. The equivalent layer key at any given S-box layer i is evaluated by $EK_i = MC^{-1}(K_i)$ when i is odd and $EK_i = MS^{-1}(K_i)$ when i is even. In addition, we use the following property of the S-box:

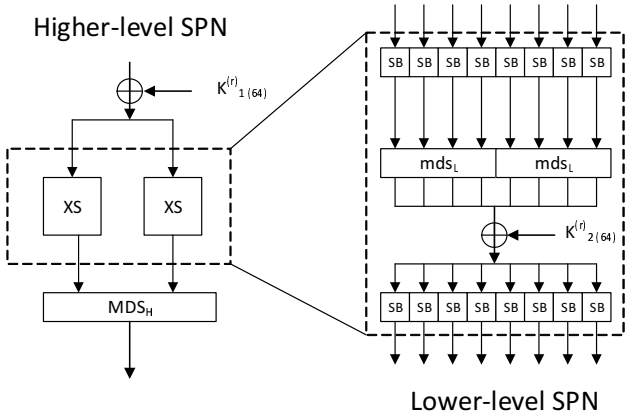

Fig. 1. One round of Hierocrypt-L1

Proposition 1 (Differential Property of S) *Given Δ_i and Δ_o two non-zero differences in $\mathbb{F}_{256}$, the equation: $S(x) + S(x + \Delta_i) = \Delta_o$ has one solution on average. This property also applies to S^{-1}.*

Key schedule. The input to the key schedule is the 128-bit master key and the output is 13 64-bit layer keys (1 key per S-box layer in addition to the final key). The master key initializes the first key state denoted by $V^{(-1)}_{1(32)} \| V^{(-1)}_{2(32)} \| V^{(-1)}_{3(32)} \| V^{(-1)}_{4(32)}$ which then undergoes 8 rounds relying on a Feistel construction and linear transformations to generate the layer keys where the first round is a bit special as it omits a linear function and does not produce any layer keys. Then, depending on the employed function, the other rounds form two groups, which we mark as 'type A' and 'type B'. The two 32-bit key state words $V_{3(32)}$ and $V_{4(32)}$ are updated linearly in each round, while the other two 32-bit key state words $V_{1(32)}$ and $V_{2(32)}$ are updated using a Feistel construction with additional input from $V_{3(32)}$ and $V_{4(32)}$. Specifically, as shown in Figure 3, one round of 'type A' of the key schedule can be described by:

$$
\begin{aligned}
(V^{(r)}_{3(32)}, V^{(r)}_{4(32)}) &\leftarrow L(V^{(r-1)}_{3(32)}, V^{(r-1)}_{4(32)});\\
V^{(r)}_{1(32)} &\leftarrow V_{2(32)})^{(r-1)};\\
V^{(r)}_{2(32)} &\leftarrow V^{(r-1)}_{1(32)}; \oplus F_\sigma(V^{(r-1)}_{2(32)} \oplus V^{(r)}_{3(32)}), \quad r = 0, 1, \cdots, 7
\end{aligned}
$$

where L is a linear function and the function F_σ is a level of S-boxes succeeded by another linear transformation. Then, the 64-bit layer keys $K^{(r)}_{1(64)}$ $(k^{(r)}_{1(32)} \| k^{(r)}_{2(32)})$

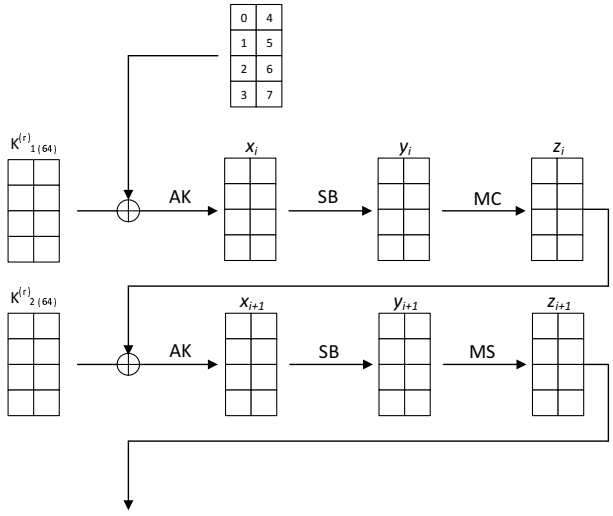

Fig. 2. Alternative representation of one round of Hierocrypt-L1

and $K^{(r)}_{2(64)}$ $(k^{(r)}_{3(32)} \| k^{(r)}_{4(32)})$ are generated in every round of 'type A' as follows:

$$
\begin{aligned}
k^{(r)}_{1(32)} &\leftarrow V^{(r-1)}_{1(32)} \oplus F_\sigma(V^{(r-1)}_{2(32)} \oplus V^{(r)}_{3(32)});\\
k^{(r)}_{2(32)} &\leftarrow V^{(r)}_{3(32)} \oplus F_\sigma(V^{(r-1)}_{2(32)} \oplus V^{(r)}_{3(32)});\\
k^{(r)}_{3(32)} &\leftarrow V^{(r)}_{4(32)} \oplus F_\sigma(V^{(r-1)}_{2(32)} \oplus V^{(r)}_{3(32)});\\
k^{(r)}_{4(32)} &\leftarrow V^{(r)}_{4(32)} \oplus V^{(r-1)}_{2(32)}, \qquad\qquad r = 0, 1, \cdots, 7
\end{aligned}
$$

The round function of 'type B' is almost equivalent to the inversion of the 'type A' round function but the linear function that operates on $V_{3(32)}$ and $V_{4(32)}$ is different. It is to be noted that in our attacks, we number the layer keys sequentially from K_1 up to K_{13} where $K_i = K^{\lceil i/2 \rceil}_{1(64)}$ when i is odd and $K_i = K^{\lceil i/2 \rceil}_{2(64)}$ when i is even.

For further details regarding the S-box, the linear transformations or the key schedule, the reader is referred to [29].

The following notations are used throughout the paper:

- x_i: The internal state at the input of layer i
- y_i: The internal state after the SB of layer i.
- z_i: The internal state after the MC (resp. MS) of layer i when i is odd (resp. even).
- $z_i^{'}$: The internal state after the AK of layer i with an equivalent key EK_i.
- $x_i[j]$: The j^{th} byte of the state x_i, where $j = 0, 1, \cdots, 7$, and the bytes are indexed as described in Figure 2.

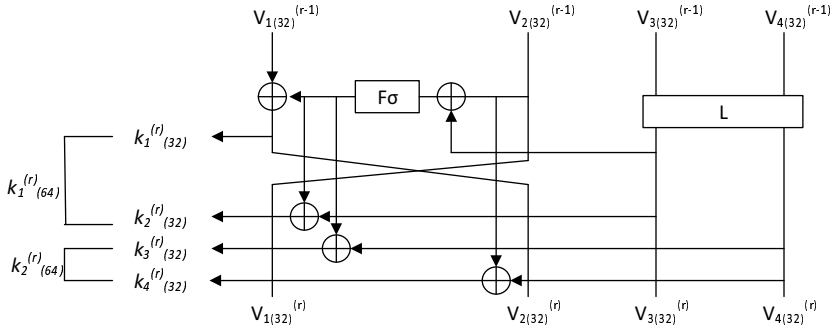

Fig. 3. 1 Round of Hierocrypt-L1 key schedule

- $x_i[j \cdots k]$: The bytes between the j^{th} position and k^{th} position of the state x_i.
- Δx_i,$\Delta x_i[j]$: The difference at state x_i and byte $x_i[j]$, respectively.
- $X_{(n)}$: An n-bit word X. Such notation is specifically used in describing the key schedule.

The memory and time complexities of our attack are measured as 64-bit HC-L1 blocks and round-reduced HC-L1 encryptions, respectively.

3 A Differential Enumeration MitM Attack on HC-L1

Generally, in a MitM attack, a round reduced block cipher E_K is split into 3 successive parts, such that $E_K = E^2_{K_2} \circ E_m \circ E^1_{K_1}$, where E_m exhibits a distinguishing property. The exploited property is used to identify the correct key by checking whether each guess of subkey (K_1, K_2) yields the property or not. In our attacks, we use a truncated differential characteristic as the distinguishing property, such that its input is a δ-set [11] captured by Definition 1. While in most of the published MitM attacks the matching is performed around a specific byte or word, adopting such approach on HC-L1 requires a time complexity that exceeds that of the exhaustive search. Therefore, as explained in details below, we opt for matching on a single equation that relates two input bytes of the linear transformation MS with two bytes at its output.

Definition 1 (δ-set of HC-L1) *Let a δ-set be a set of* 256 *HC-L1 states that are all different in one state byte (the active byte) and all equal in the other state bytes (the inactive bytes).*

In our MitM attack, we use the 5 S-box layers distinguisher embedded in the truncated differential characteristic, illustrated in Figure 4. It starts at x_2 and ends at the input of the linear transformation MS of layer 6, i.e., $z_6^{'}$. We exploit the simplicity of the MS operation by observing the below equations of two of

its output bytes:

$$z_i[0] = y_i[0] \oplus y_i[2] \oplus y_i[4] \oplus y_i[5] \oplus y_i[6] \quad (1)$$
$$z_i[7] = y_i[0] \oplus y_i[2] \oplus y_i[4] \oplus y_i[6] \oplus y_i[7] \quad (2)$$

Therefore, from (1) and (2), we have

$$z_i[0] \oplus z_i[7] = y_i[5] \oplus y_i[7] \quad (3)$$

Consequently, it follows that $x_7[0] \oplus x_7[7] = z'_6[5] \oplus z'_6[7]$ (see Figure 4) which is the single equation upon which the matching is performed as will be explained in the attack procedure. Proposition 2, below, is the core of our attack.

Proposition 2 *If a message m belongs to a pair of states conforming to the truncated differential characteristic of Figure 4, then the ordered sequence of differences $\Delta z_6^{'}[5] \oplus \Delta z_6^{'}[7]$ obtained from the δ-set constructed from m in $x_2[3]$ is fully determined by the following 14 bytes: $x_2[3], \Delta x_2[3], x_3[1,3,4,6], y_5[4 \cdots 7]$, $\Delta y_6[5], \Delta y_6[7], y_6[5]$ and $y_6[7]$.*

Proof. The proof is based on rebound-like arguments adopted from the cryptanalysis of hash functions [22] and used in [13]. Assuming that $(m, m^{'})$ is a pair that follows the truncated differential characteristic in Figure 4. In the sequel, we manifest that knowing these specific 14 bytes is sufficient to compute the ordered sequence of differences $\Delta z_6^{'}[5] \oplus \Delta z_6^{'}[7]$. To conform to the differential characteristic in Figure 4, the 14 bytes $x_2[3], \Delta x_2[3], x_3[1,3,4,6], y_5[4 \cdots 7], \Delta y_6[5], \Delta y_6[7]$, $y_6[5]$ and $y_6[7]$ can take as many as $2^{8\times 13} = 2^{104}$ possible values only. This is because $\Delta y_6[5]$ must equal $\Delta y_6[7]$ in order to result in a difference in just $x_7[0,7]$. Then for each of these 2^{104} values, we can determine all the differences shown in Figure 4.

- The value $x_2[3]$ with the difference $\Delta x_2[3]$ enable us to bypass the S-box of layer 2 and then propagate the difference linearly through MS to compute $\Delta x_3[1,3,4,6]$.
- By knowing $x_3[1,3,4,6]$, we can bypass the S-box of layer 3 to reach y_3, then linearly through MC to compute $\Delta x_4[0 \cdots 7]$.
- Similarly in the other direction, the differences $\Delta y_6[5]$ and $\Delta y_6[7]$ with the values $y_6[5]$ and $y_6[7]$ enable us to bypass the S-box of layer 6 and compute the difference $\Delta z_5[5,7]$, then linearly through MC^{-1} we compute $\Delta y_5[4 \cdots 7]$.
- By knowing $y_5[4 \cdots 7]$, we bypass the S-box of layer 5 to compute $\Delta z_4[4 \cdots 7]$ and then linearly through MS^{-1} we can compute $\Delta y_4[0 \cdots 7]$.
- Now, we have the differences $\Delta x_4[0 \cdots 7]$ and $\Delta y_4[0 \cdots 7]$. Hence, by the differential property of the HC-L1 S-box (Proposition 1), there is, on average, one solution for each of the 8 bytes of x_4.

To build the ordered sequence for each of the 2^{104} possible values of the 14 bytes from proposition 1, we consider all the $2^8 - 1$ possible values for the difference

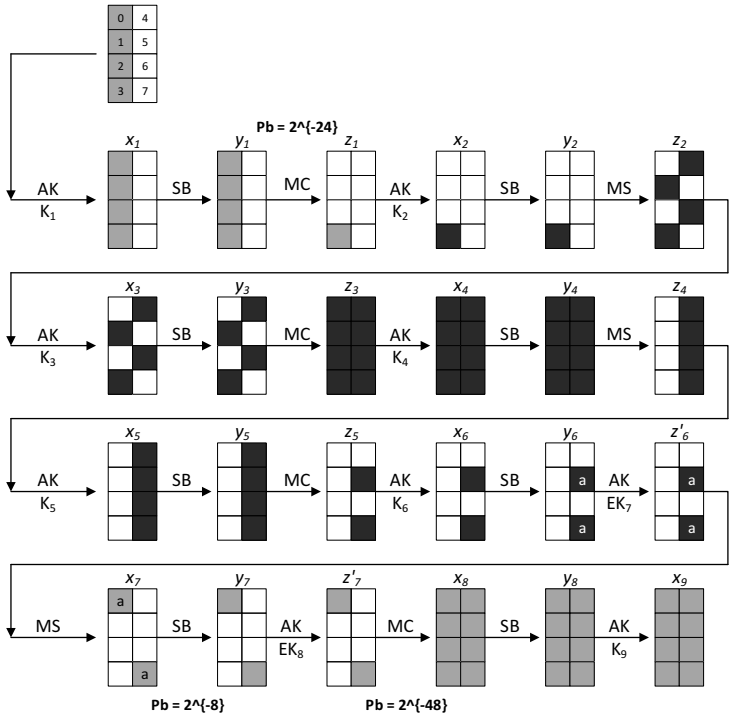

Fig. 4. The differential characteristic used in the MitM attack on HC-L1 using differential enumeration. Offline bytes are colored in black while online bytes are colored in gray.

$\Delta x_2[3]$ and propagate them until $z_6^{'}$ with the help of the internal state solutions we have. This creates an ordered sequence of $2^8 - 1$ differences in $\Delta z_6^{'}[5, 7]$.

Attack Procedure. Similar to other MitM attacks, our attack has 2 phases; an offline phase and an online phase that result in recovering the 64-bit last layer key K_9, 2 bytes of $EK_8 = MC^{-1}(K_8)$ and 4 bytes of K_1.

Offline Phase. In the offline phase, we compute all the 2^{104} values of $\Delta z_6^{'}[5] \oplus \Delta z_6^{'}[7]$ determined by the 14 bytes listed in proposition 1 and store them in a precomputation table T.

Online Phase. The online phase can be divided into two stages; data collection and key recovery. The data collection stage aims at finding pairs of messages that follow the truncated differential characteristic in Figure 4. Then, for each of the found pairs, a δ-set is created, its corresponding ordered sequence is computed and tested for a match in T to identify the correct key in the key recovery stage.

Data collection. To generate one pair of messages that conforms to the 8 S-box layers truncated differential characteristic in Figure 4, the encryption oracle is queried with structures of chosen plaintexts. In a given structure, bytes $[0 \cdots 3]$ take all the 2^{32} possible values while the other 4 bytes are fixed to some, possibly different, constants and hence, each structure generates $2^{32} \times (2^{32} - 1)/2 \approx 2^{63}$ pairs. Our 8 S-box layers truncated differential characteristic has a probability of $2^{-3\times3\times8-1\times8} = 2^{-80}$ because of the three $4 \rightarrow 1$ transitions over MC in addition to the probability that $\Delta x_7[0]$ equals $\Delta x_7[7]$, marked as a in Figure 4. Consequently, in order to find one pair that follows our chosen 2^{-80} probability truncated differential characteristic, 2^{80} pairs are needed which is equivalent to 2^{80-63}, i.e., 2^{17} structures of 2^{32} messages, each. Briefly, $2^{17+32} = 2^{49}$ messages are sent to the encryption oracle to generate the required 2^{80} pairs. It is to be noted that the distinguisher was chosen to start at $x_2[3]$ because this specific byte results into just 4 differences, i.e., $z_2[1, 3, 4, 6]$ after the application of the MS transformation (cf. 4^{th} column of MDS_H).

Key Recovery. The key bytes: $K_9[0 \cdots 7]$, $EK_8[0, 7]$ and $K_1[0 \cdots 3]$ can take $2^{(2+1+1)\times8} = 2^{32}$ possible values for each of the 2^{80} pairs. This is justified as follows:

- The difference $\Delta y_7[0, 7]$ is guessed and propagated linearly through MC to compute $\Delta x_8[0 \cdots 7]$.
- The difference $\Delta y_8[0 \cdots 7]$ is equal to the difference $\Delta x_9[0 \cdots 7]$ which, in turn, is the difference in the ciphertext pair.
- As we have the difference $\Delta x_8[0 \cdots 7]$ and $\Delta y_8[0 \cdots 7]$, the differential property of the S-box is used to deduce a solution for each byte of x_8 and y_8 which yields 2^{16} key candidates for the whole K_9 by simply xoring y_8 with the ciphertext.
- Then, $x_8[0 \cdots 7]$ is propagated linearly through MC^{-1} to deduce $z_7^{'}[0 \cdots 7]$.
- Afterwards, the difference $\Delta y_6[5, 7]$ which, as explained before, assumes 2^8 values only is guessed and propagated through MS to get the difference $\Delta x_7[0, 7]$.
- As we have $\Delta x_7[0, 7]$ and $\Delta y_7[0, 7]$ were already guessed in the first step above, the differential property of the S-box is used to deduce a solution for $y_7[0, 7]$ which with $z_7^{'}[0 \cdots 7]$, computed above, enables us to deduce 2^8 candidates for $EK_8[0, 7]$.
- Next, the difference $\Delta x_2[3]$ is guessed and propagated linearly through MC^{-1} to compute $\Delta y_1[0 \cdots 3]$.
- The difference $\Delta x_1[0 \cdots 3]$ is actually the difference in the plaintext pair. Therefore, knowing the differences $\Delta x_1[0 \cdots 3]$ and $\Delta y_1[0 \cdots 3]$ enables us to deduce 2^8 candidates for $K_1[0 \cdots 3]$ using the differential property of the S-box.

All in all, guessing 4 bytes helps deduce 14 key bytes. In other words, these 14 key bytes have just 2^{32} possible values for each of the 2^{80} pairs. Then, in order to recover the key, we enumerate each of the 2^{80} candidate pairs we obtained in

the data collection stage and deduce the corresponding 2^{32} possible key suggestions. Next, we build the plaintext δ-set and compute its corresponding ordered sequence of $\Delta x_7[0] \oplus \Delta x_7[7]$ and look for a match in T and if no match is found, this key suggestion is discarded.

A valid ordered sequence can be generated by a wrong key with a negligible probability, $2^{80+32+104-255\times 8} = 2^{-1824}$, which can be relaxed. Therefore, we use the partial sequence matching idea proposed in [4]. Instead of matching $2^8 - 1$ bytes ordered sequence, we match b bytes such that $b < 2^8$ and the probability of error, chosen to be 2^{-32}, is still small enough to be able to identity the right key. In that case, the number of required bytes b is calculated by $2^{-32} = 2^{80+32+104-8b}$ which yields $b = 31$. Therefore, it is enough to match 31 bytes of the ordered sequence to identify the right key with a negligible error probability of 2^{-32}.

So far, we have recovered 14 key bytes; the 8-byte K_9, 2 bytes of EK_8 and 4 bytes of K_1. To recover the master key, 6 bytes of EK_8 are guessed to get 2^{48} suggestions for K_8, which, using the key schedule notations, means that there are 2^{48} candidates for the keys $k^{(4)}_{3(32)}, k^{(4)}_{4(32)}$. Along with K_9 or rather $k^{(5)}_{1(32)}$ and $k^{(5)}_{2(32)}$, these keys are computed as follows:

$$k^{(4)}_{3(32)} = V^{(4)}_{4(32)} \oplus F_\sigma(V^{(3)}_{2(32)} \oplus V^{(4)}_{3(32)}) \tag{4}$$

$$k^{(4)}_{4(32)} = V^{(4)}_{4(32)} \oplus V^{(3)}_{2(32)} \tag{5}$$

$$k^{(5)}_{1(32)} = V^{(4)}_{1(32)} \oplus F_\sigma(V^{(4)}_{2(32)} \oplus V^{(5)}_{3(32)}) \tag{6}$$

$$k^{(5)}_{2(32)} = V^{(5)}_{3(32)} \oplus F_\sigma(V^{(4)}_{2(32)} \oplus V^{(5)}_{3(32)}) \tag{7}$$

Then, the 32-bit $V^{(3)}_{2(32)}$ is guessed and $V^{(4)}_{4(32)}$ is computed from equation (5) and, in turn, $V^{(4)}_{3(32)}$ is deduced from equation (4). According to the key schedule, the knowledge of $V^{(4)}_{3(32)}$ and $V^{(4)}_{4(32)}$ results in knowing $V^{(5)}_{3(32)}$ and $V^{(5)}_{4(32)}$. Next, since $V^{(4)}_{1(32)} = V^{(3)}_{2(32)}$, $V^{(4)}_{2(32)}$ is computed from equation (6) and then equation (7) is used as a 2^{-32} filter to get one solution for $V^{(3)}_{2(32)}, V^{(4)}_{1(32)}, V^{(4)}_{2(32)}, V^{(4)}_{3(32)}$, $V^{(4)}_{4(32)}, V^{(5)}_{3(32)}$ and $V^{(5)}_{4(32)}$. As we recover one full intermediate state of the key schedule and its round is bijective, we can recover the master key and get 2^{48} candidates for the master key corresponding to the 2^{48} K_8 suggestions. The correct master key is found by exhaustively searching through these 2^{48} candidates using 2 plaintext/ciphertext pairs with no significant impact on the attack overall time complexity.

Attack Complexity. The size of the precomputation table T created in the offline phase determines the memory requirement of the attack. T contains 2^{104} ordered sequences, each of $8 \times 31 = 248$ bits by using the partial sequence matching technique. Therefore, the memory complexity of the attack is $2^{104} \times$

$248/64 \approx 2^{106}$ 64-bit blocks. The data collection stage of the online phase sets the data complexity of the attack to 2^{49} chosen plaintexts. The offline phase time complexity to build T is attributed to executing 2^{104} partial encryptions on 32 messages, which is equivalent to $2^{104+5} \times 11/(8 \times 8) = 2^{106.46}$ encryptions. The online phase time complexity to recover 14 key bytes is determined by the time needed to partially decrypt the 2^5 values in a δ-set with all the 2^{32} key suggestions for all the 2^{80} generated pairs which is equivalent to $2^{80+32+5} \times (4 + 10)/(8 \times 8) = 2^{114.8}$. Finding the correct master key among the 2^{48} candidates using 2 plaintext/ciphertext pairs requires $2 \times 2^{48} = 2^{49}$ encryptions. Therefore, the time complexity of the attack is equivalent to $2^{114.8} + 2^{106.46} + 2^{49} \approx 2^{114.8}$.

4 Conclusion

In this paper, we have analyzed one of the Japanese e-Government 2013 candidate recommended block ciphers; Hierocrypt-L1 using the meet-in-the-middle (MitM) attack in the single-key setting. Our attack employs the differential enumeration technique and is launched against 8 S-box layers using a 5 S-box layers distinguisher. The attack recovers the master key with data complexity of 2^{49} chosen plaintexts, time complexity of $2^{114.8}$ 8 S-box layers Hierocrypt-L1 encryptions and memory complexity of 2^{106} 64-bit blocks. To the best of our knowledge, this is the first attack on Hierocrypt-L1 in the single-key setting that reaches 8 S-box layers.

References

1. Abdelkhalek, A., AlTawy, R., Tolba, M., and Youssef, A. M. Meet-in-the-Middle Attacks on Reduced-Round Hierocrypt-3. In *LatinCrypt* (2015), Lecture Notes in Computer Sience, Springer, *to appear*.
2. AlTawy, R., and Youssef, A. M. Differential Sieving for 2-step matching meet-in-the-middle attack with application to LBlock. In *Lightsec* (2014), Lecture Notes in Computer Sience, Springer.
3. AlTawy, R., and Youssef, A. M. Preimage Attacks on Reduced-Round Stribog. In *Progress in Cryptology AFRICACRYPT 2014*, D. Pointcheval and D. Vergnaud, Eds., vol. 8469 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 109–125.
4. AlTawy, R., and Youssef, A. M. Meet in the Middle Attacks on Reduced Round Kuznyechik. Cryptology ePrint Archive, Report 2015/096, 2015. http://eprint.iacr.org/.
5. Barreto, P., Rijmen, V., Nakahara, Jorge, J., Preneel, B., Vandewalle, J., and Kim, H. Improved Square Attacks against Reduced-Round Hierocrypt. In *Fast Software Encryption*, M. Matsui, Ed., vol. 2355 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2002, pp. 165–173.
6. Biryukov, A., Derbez, P., and Perrin, L. P. Differential Analysis and Meet-in-the-Middle Attack against Round-Reduced TWINE. Fast Software Encryption 2015. *to appear*.

7. Bogdanov, A., and Rechberger, C. A 3-Subset Meet-in-the-Middle Attack: Cryptanalysis of the Lightweight Block Cipher KTANTAN. In *Selected Areas in Cryptography*, A. Biryukov, G. Gong, and D. Stinson, Eds., vol. 6544 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 229–240.
8. CRYPTEC. e-Government Candidate Recommended Ciphers List, 2013. http://www.cryptrec.go.jp/english/method.html.
9. CRYPTEC. e-Government Recommended Ciphers List, 2003. http://www.cryptrec.go.jp/english/images/cryptrec_01en.pdf.
10. CRYPTEC. Specification on a Block Cipher: Hierocrypt-L1. http://www.cryptrec.go.jp/cryptrec_03_spec_cypherlist_files/PDF/04_02espec.pdf.
11. Daemen, J., Knudsen, L., and Rijmen, V. The block cipher SQUARE. In *Fast Software Encryption*, E. Biham, Ed., vol. 1267 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1997, pp. 149–165.
12. Demirci, H., and Seluk, A. A Meet-in-the-Middle Attack on 8-Round AES. In *Fast Software Encryption*, K. Nyberg, Ed., vol. 5086 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008, pp. 116–126.
13. Derbez, P., Fouque, P.-A., and Jean, J. Improved Key Recovery Attacks on Reduced-Round AES in the Single-Key Setting. In *Advances in Cryptology EUROCRYPT 2013*, T. Johansson and P. Nguyen, Eds., vol. 7881 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, pp. 371–387.
14. Dunkelman, O., Keller, N., and Shamir, A. Improved Single-Key Attacks on 8-Round AES-192 and AES-256. In *Advances in Cryptology - ASIACRYPT 2010*, M. Abe, Ed., vol. 6477 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2010, pp. 158–176.
15. Dunkelman, O., Sekar, G., and Preneel, B. Improved Meet-in-the-Middle Attacks on Reduced-Round DES. In *Progress in Cryptology INDOCRYPT 2007*, K. Srinathan, C. Rangan, and M. Yung, Eds., vol. 4859 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, pp. 86–100.
16. Hao, Y., Bai, D., and Li, L. A Meet-in-the-Middle Attack on Round-Reduced mCrypton Using the Differential Enumeration Technique. In *Network and System Security*, M. Au, B. Carminati, and C.-C. Kuo, Eds., vol. 8792 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 166–183.
17. Hong, D., Koo, B., and Sasaki, Y. Improved Preimage Attack for 68-Step HAS-160. In *Information, Security and Cryptology ICISC 2009*, D. Lee and S. Hong, Eds., vol. 5984 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2010, pp. 332–348.
18. Howgrave-Graham, N. A Hybrid Lattice-Reduction and Meet-in-the-Middle Attack Against NTRU. In *Advances in Cryptology - CRYPTO 2007*, A. Menezes, Ed., vol. 4622 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2007, pp. 150–169.
19. K. Ohkuma, F. Sano, H. Muratani, M. Motoyama, and S. Kawamura. On security of block ciphers Hierocrypt-3 and Hierocrypt-L1. The 2001 Symposium on Cryptography and Information Security (SCIS 2001), 11A-4, Jan. 2001.
20. Li, L., Jia, K., and Wang, X. Improved Meet-in-the-Middle Attacks on AES-192 and PRINCE. Cryptology ePrint Archive, Report 2013/573, 2013. http://eprint.iacr.org/.
21. MacWilliams, F. J., and Sloane, N. J. A. *The theory of error correcting codes*, vol. 16. Elsevier, 1977.
22. Mendel, F., Rechberger, C., Schlffer, M., and Thomsen, S. The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grϕstl. In *Fast Software Encryp-*

tion, O. Dunkelman, Ed., vol. 5665 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009, pp. 260–276.
23. New European Schemes for Signatures, Integrity, and Encryption. https://www.cosic.esat.kuleuven.be/nessie.
24. Ohkuma, K., Muratani, H., Sano, F., and Kawamura, S. The Block Cipher Hierocrypt. In *Selected Areas in Cryptography*, D. Stinson and S. Tavares, Eds., vol. 2012 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2001, pp. 72–88.
25. Rijmen, V. *Cryptanalysis and design of iterated block ciphers*. PhD thesis, Doctoral Dissertation, October 1997, KU Leuven, 1997.
26. Sasaki, Y., Wang, L., Wu, S., and Wu, W. Investigating Fundamental Security Requirements on Whirlpool: Improved Preimage and Collision Attacks. In *Advances in Cryptology ASIACRYPT 2012*, X. Wang and K. Sako, Eds., vol. 7658 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012, pp. 562–579.
27. Sekar, G., Mouha, N., Velichkov, V., and Preneel, B. Meet-in-the-Middle Attacks on Reduced-Round XTEA. In *Topics in Cryptology CT-RSA 2011*, A. Kiayias, Ed., vol. 6558 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2011, pp. 250–267.
28. Taga, B., Moriai, S., and Aoki, K. Differential and Impossible Differential Related-Key Attacks on Hierocrypt-L1. In *Information Security and Privacy*, W. Susilo and Y. Mu, Eds., vol. 8544 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014, pp. 17–33.
29. Toshiba Corporation. Block Cipher Family Hierocrypt . http://www.toshiba.co.jp/rdc/security/hierocrypt/index.htm.