

A 2^{70} Attack on the Full MISTY1

Achiya Bar-On^{1*} and Nathan Keller¹

¹ Department of Mathematics
Bar Ilan University
Ramat Gan, 52900, Israel

Abstract. MISTY1 is a block cipher designed by Matsui in 1997. It is widely deployed in Japan, and is recognized internationally as a European NESSIE-recommended cipher and an ISO standard. After almost 20 years of unsuccessful cryptanalytic attempts, a first attack on the full MISTY1 was presented at CRYPTO 2015 by Yosuke Todo. The attack, using a new technique called *division property*, requires almost the full codebook and has time complexity of $2^{107.3}$ encryptions.

In this paper we present a new attack on the full MISTY1. It is based on Todo's division property, along with a variety of refined key-recovery techniques. Our attack requires almost the full codebook (like Todo's attack), but allows to retrieve 49 bits of the secret key in time complexity of only 2^{64} encryptions, and the full key in time complexity of $2^{69.5}$ encryptions.

While our attack is clearly impractical due to its large data complexity, it shows that MISTY1 provides security of only 2^{70} — significantly less than what was considered before.

1 Introduction

MISTY1 [10] is a 64-bit block cipher with 128-bit keys designed in 1997 by Matsui. In 2002, MISTY1 was selected by the Japanese government to be one of the e-government candidate recommended cipher, and since then, it became widely deployed in Japan. MISTY1 also gained recognition outside Japan, when it was selected to the portfolio of European NESSIE-recommended ciphers, and approved as an ISO standard in 2005. Furthermore, the block cipher KASUMI [1] designed as a slight modification of MISTY1 is used in the 3G cellular networks, which makes it one of the most widely used block ciphers today.

MISTY1 has an 8-round recursive Feistel structure, where the round function FO is in itself a 3-round Feistel construction, whose F-function FI is in turn a 3-round Feistel construction using 7-bit and 9-bit invertible S-boxes. The specific choice of S-boxes and the recursive structure ensure provable security against differential and linear cryptanalysis. In order to thwart other types of attacks, after every two rounds an FL function is applied to each of the two halves

* This research was partially supported by the Israeli Ministry of Science, Technology and Space, and by the Check Point Institute for Information Security.

independently. The FL functions are key-dependent linear functions which play the role of whitening layers.

In the 18 years since its design, MISTY1 withstood numerous cryptanalytic attempts. More than a dozen of papers analyzed its reduced-round variants (see, e.g., [2, 7, 13, 14]), yet the full 8-round variant seemed completely out of reach. The situation changed when Yosuke Todo presented at CRYPTO'2015 [11] the first attack on the full MISTY1. The attack is based on a new variant of integral cryptanalysis, called *division property*, presented by Todo at EUROCRYPT'2015. Using the new technique, Todo showed that there exist seven independent integral structures of 2^{63} plaintexts, whose propagation can be traced through 6 rounds of MISTY1. (This should be compared with 4 rounds achieved by the best previously known integral characteristics.) These characteristics allow to break the full MISTY1 with data complexity of $2^{63.994}$ chosen plaintexts and time complexity of $2^{107.3}$ encryptions.

In this paper, we present an improved attack on the full MISTY1, which allows to significantly reduce the time complexity. Our attack uses Todo's division property both in the encryption direction (like Todo used it) and in the decryption direction (which turns out to be more useful than the encryption direction due to key schedule considerations). In addition, we use refined key recovery techniques, including the partial sums technique [6], and a two-dimensional meet-in-the-middle attack (like the one is used in [4]). Our attack has two phases. The first phase requires $2^{64} - 2^{50}$ chosen ciphertexts and allows to recover the equivalent of 49 key bits in time of 2^{64} encryptions (i.e., dominated by the time required for encrypting the plaintexts!). The second phase requires almost all the rest of the codebook and allows to recover all remaining key bits in time of $2^{69.5}$ encryptions. Alternatively, the rest of the key can be found in time of 2^{79} encryptions without additional data. A comparison of our attack with the best previously known attacks on full-round and reduced-round MISTY1 is presented in Table 1.

The paper is organized as follows. In Section 2 we describe the structure of MISTY1 and introduce some notations that will be used throughout the paper. The division property is described in Section 3, as well as the 6-round integral characteristic based on its propagation. Our improved attack on full MISTY1 is presented in Section 4. Finally, in Section 5 we summarize the paper.

2 Brief Description of MISTY1

MISTY1 is an 8-round Feistel construction, where the round function, FO , is in itself a variant of a 3-round Feistel construction, defined as follows. The input to FO is divided into two halves. The left one is XORed with a subkey, enters a keyed permutation FI , and the output is XORed with the right half. After the XOR, the two halves are swapped, and the same process (including the swap) is repeated two more times. After that, an additional swap and an XOR of the left half with a subkey is performed (see Fig. 1).

Table 1: Summary of the best known single-key attacks on MISTY1

FO rounds	FL layers	Data complexity	Time complexity	Type
7	3	2^{58} KP	$2^{124.4}$	ID attack [7]
7	4	$2^{62.9}$ KP	2^{118}	MZC attack [14]
7	4	$2^{49.7}$ CP	$2^{116.4}$	HOD attack [13]
7	4	$2^{50.1}$ CP	$2^{100.4}$	HOD attack [2]
7	5	$2^{51.45}$ CP	2^{121}	HOD attack [2]
8	5	$2^{63.58}$ CP	2^{121}	IDP attack [11]
8	5	$2^{63.994}$ CP	$2^{107.3}$	IDP attack [11]
8	5	$2^{63.9999}$ CC	2^{64}	IDP attack [†] (Section 4)
8	5	$2^{63.9999}$ CC	2^{79}	IDP attack (Section 4)
8	5	$2^{64} - 2^{36}$ CPC	$2^{69.5}$	IDP attack (Section 4)

ID attack: Impossible Differential attack

HOD attack: Higher Order Differential attack

MZC attack: Multi-Dimensional Zero Correlation attack

IDP attack: Integral attack using division property

[†] Attack recovers 49 key bits

KP: Known Plaintexts; CP: Chosen Plaintexts; CC: Chosen Ciphertexts; CPC: Chosen Plaintexts and Ciphertexts

The FI function in itself also has a Feistel-like structure. Its 16-bit input is divided into two unequal parts – one of 9 bits, and the second of 7 bits. The left part (which contains 9 bits) enters an S-box, S_9 , and the output is XORed with the right 7-bit part (after padding the 7-bit value with two zeroes as the most significant bits). The two parts are swapped, the 7-bit part enters a different S-box, S_7 , and the output is XORed with 7 bits out of the 9 of the right part. The two parts are then XORed with a subkey, and swapped again. The 9-bit value again enters S_9 , and the output is XORed with the 7-bit part (after padding). The two parts are then swapped for the last time.

Every two rounds, starting before the first one, each of the two 32-bit halves enters an FL layer. The FL layer is a simple linear transformation. Its input is divided into two halves of 16 bits each, the AND of the left half with a subkey is XORed to the right half, and the OR of the updated right half with another subkey is XORed to the left half. We outline the structure of MISTY1 and its parts in Fig. 1.

The key schedule of MISTY1 takes the 128-bit key, and treats it as eight 16-bit words K_1, K_2, \dots, K_8 . From this sequence of words, another sequence of eight 16-bit words is generated, according to the rule $K'_i = FI_{K_{i+1}}(K_i)$.

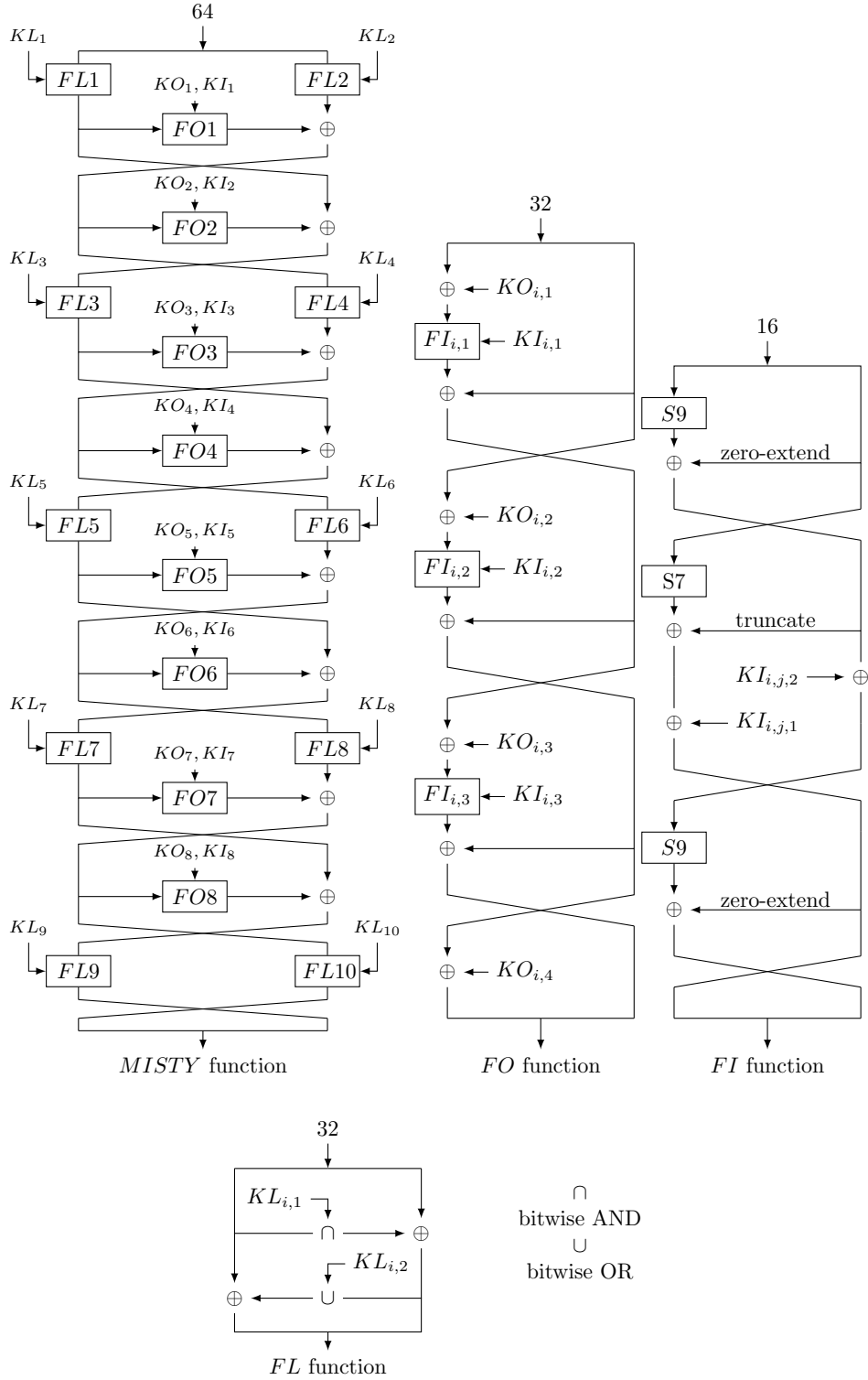


Fig. 1: Outline of MISTY1

In each round, seven words are used as the round subkey, and each of the FL functions accepts two subkey words. We give the exact key schedule of MISTY1 in Table 2.

Table 2: The Key Schedule of MISTY1

$KO_{i,1}$	$KO_{i,2}$	$KO_{i,3}$	$KO_{i,4}$	$KI_{i,1}$	$KI_{i,2}$	$KI_{i,3}$	$KL_{i,1}$	$KL_{i,2}$
K_i	K_{i+2}	K_{i+7}	K_{i+4}	K'_{i+5}	K'_{i+1}	K'_{i+3}	$K_{\frac{i+1}{2}}$ (odd i) $K'_{\frac{i}{2}+2}$ (even i)	$K'_{\frac{i+1}{2}+6}$ (odd i) $K_{\frac{i}{2}+4}$ (even i)

2.1 Notations Used in the Paper

Throughout the paper, we use the following notations for intermediate values during the MISTY1 encryption process.

- The plaintext and the ciphertext are denoted, as usual, by P and $C = E(P)$.
- The input of the i 'th round ($1 \leq i \leq 8$) is denoted by X_i . If we want to emphasize that the intermediate value corresponds to the plaintext P , we denote it by $X_i(P)$.
- For odd rounds, we denote by X'_i the intermediate value after application of the FL functions.
- The output of the FO function of round i is denoted Out_i .
- For any intermediate value Z , $Z[k-l]$ denotes bits from k to l (inclusive) of Z . The special case $Z[i]$ denotes the i 'th bit of Z .
- For any intermediate value Z , the right and left halves of Z are denoted by Z_R and Z_L , respectively.
- For any 16-bit key K , the 9 rightmost bits and 7 left most bits of K are denoted by K^R and K^L , respectively.
- The first $S9$ function of $FI_{i,j}$ is denoted by $S9_{i,j,1}$ and its input is denoted by $ES9_{i,j,1}$. Similarly, $S7_{i,j}$, $ES7_{i,j}$ and $S9_{i,j,2}$, $ES9_{i,j,2}$ denote the other S-boxes, $S7$ and the second $S9$, and their inputs in $FI_{i,j}$.

3 Integral Cryptanalysis using Division Property and its Application to MISTY1

3.1 Integral Cryptanalysis

Integral cryptanalysis is a powerful cryptanalytic technique presented in [3, 8]. The basic idea behind integral cryptanalysis is to trace the encryption process of a structured set of plaintexts, called *integral structure*, during part of the cipher's rounds. Usually, the information on the intermediate values gradually reduces as the encryption proceeds, so that eventually, one can predict only the *sum* of the set of intermediate values in part of the state.

Definition 1. A set of values V is called balanced if $\bigoplus_{x \in V} x = 0$.

An *integral characteristic* predicts that for a structured set V of plaintexts (usually, an affine subspace of the plaintext space), the set of corresponding intermediate values after i rounds is balanced in some bit j . That is,

$$\bigoplus_{x \in V} X_{i+1}[j](x) = 0. \quad (1)$$

Note that unlike differential and linear cryptanalysis, an integral characteristic is combinatorial and not statistical, meaning that (1) holds with probability 1. A characteristic that satisfies (1) is called an i -round characteristic of order $\log_2 |V|$.

An example of an integral characteristic is an 8-order 3-round characteristic of AES presented in [3], which predicts that if V consists of 256 plaintexts that are constant in all bytes but one, and assume all possible values in the remaining byte, then the corresponding set of intermediate values after 3 rounds is balanced in each of the 16 bytes. This property is used in [3] to attack up to 6 rounds of AES with a practical complexity.

Once an integral characteristic is found, it can be used to mount a key-recovery attack. Suppose that for some block cipher $E: \{0, 1\}^n \rightarrow \{0, 1\}^n$, there exists an i -round integral characteristic that satisfies (1). Denote by E_1^{-1} the Boolean function that represents the mapping from the ciphertext of E to the intermediate state bit $X_i[j]$ (see Fig. 2). Then Equation (1) can be rewritten as

$$\bigoplus_{x \in V} E_1^{-1}(E(x)) = 0. \quad (2)$$

(Equation (2) is called *the attack equation*). The adversary asks for the encryption of several structured sets of plaintexts of the form V , partially decrypts the corresponding ciphertexts (by guessing the key material used in E_1^{-1}), and checks whether Equation (2) holds.

3.2 Division Property

The idea behind the division property is to increase the precision of the information on intermediate values traced by the integral attack. As the idea is rather complex in its general form, we present in this section all definitions and notations required for it (mostly taken from [11]), and in the next section its application to MISTY1. For the general *division property* technique, we refer the reader to [12].

For a linear subspace $U \subset \mathbb{F}_2^n$ and a vector $v \in \mathbb{F}_2^n$, we call $V = \{u + v | u \in U\}$ an *affine subspace* of \mathbb{F}_2^n . The dimension of V is, as usual, $\log_2 |V|$.

For $u \in \mathbb{F}_2^n$ we denote by w_u the Hamming weight of u (i.e., $w_u = \sum_{i=1}^n u[i]$) and by $\mathbb{S}_k^n = \{u \in \mathbb{F}_2^n : w_u \geq k\}$ the set of all values with Hamming weight larger than (or equal to) k . For $x, u \in \mathbb{F}_2^n$ we define $x^u = \prod_{i=1}^n x[i]^{u[i]}$. In general, for

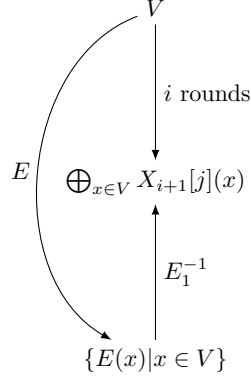


Fig. 2: Outline of the Integral Attack

$x = (x_1, x_2, \dots, x_m)$, $u = (u_1, u_2, \dots, u_m) \in \mathbb{F}_2^{n_1} \times \mathbb{F}_2^{n_2} \times \dots \times \mathbb{F}_2^{n_m}$, we write x^u for $\prod_{i=1}^m x_i^{u_i}$ and $\mathbb{S}_{[k_1, k_2, \dots, k_m]}^{n_1, n_2, \dots, n_m}$ for the set $\{u \in \mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_m} : w_{u_i} \geq k_i \text{ for all } i\}$.

Let $\mathbb{X} \subset \mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_m}$ be a multiset. We say that \mathbb{X} has the division property $\mathcal{D}_{[k_1, k_2, \dots, k_m]}^{n_1, n_2, \dots, n_m}$ if

$$\bigoplus_{x \in \mathbb{X}} x^u = 0, \quad \text{for all } u \in (\mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_m} \setminus \mathbb{S}_{[k_1, \dots, k_m]}^{n_1, \dots, n_m})$$

(with no restriction on $\bigoplus_{x \in \mathbb{X}} x^u$ for $u \in \mathbb{S}_{[k_1, \dots, k_m]}^{n_1, \dots, n_m}$). We also define

$$\mathbb{S}_{\mathbf{k}_1, \dots, \mathbf{k}_t}^{n_1, n_2, \dots, n_m} = \bigcup_{i=1}^t \mathbb{S}_{\mathbf{k}_i}^{n_1, n_2, \dots, n_m}$$

($\mathbf{k}_i \in \mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_m}$) and define the division property $\mathcal{D}_{\mathbf{k}_1, \dots, \mathbf{k}_t}^{n_1, n_2, \dots, n_m}$ similarly to the above definition of $\mathcal{D}_{[k_1, k_2, \dots, k_m]}^{n_1, n_2, \dots, n_m}$.

Example 2. Any 4-dimensional affine subspace $V \subset \mathbb{F}_2^7$ has the property

$$\bigoplus_{x \in V} x_{i_1} x_{i_2} x_{i_3} = 0$$

for all $1 \leq i_1 < i_2 < i_3 \leq 7$ (where $x_j = x[j]$ denote the j th bit of x). Hence, V has the division property \mathcal{D}_4^7 .

Example 3. For $1 \leq i \leq m$, let $V_i \subset \mathbb{F}_2^{n_i}$ be an affine subspace of dimension k_i . Then $\mathbb{X} = \{(x_1, \dots, x_m) \in \mathbb{F}_2^{n_1} \times \dots \times \mathbb{F}_2^{n_m} : x_i \in V_i\}$ has the division property $\mathcal{D}_{[k_1, k_2, \dots, k_m]}^{n_1, n_2, \dots, n_m}$.

Notation 4. Let $\mathbf{k}_i, \mathbf{k}_j \in \mathbb{Z}^m$. We write $\mathbf{k}_i \leq \mathbf{k}_j$ if $\mathbf{k}_i[\ell] \leq \mathbf{k}_j[\ell]$ for all $1 \leq \ell \leq m$.

Example 5. Let $\mathbb{X} \subseteq \mathbb{F}_2^7 \times \mathbb{F}_2^7$ be a multiset that has the division property $\mathcal{D}_{[5,0],[1,4],[2,3]}^{7,7}$. Then

$$\bigoplus_{(x,y) \in \mathbb{X}} x_{i_1} \cdots x_{i_t} y_{j_1} \cdots y_{j_s}$$

is unknown if $(5, 0) \leq (t, s)$ or $(1, 4) \leq (t, s)$ or $(2, 3) \leq (t, s)$ and equals 0 otherwise.

Observation 6. If $\mathbf{k}_i \leq \mathbf{k}_j$ ($\iff \mathbb{S}_{\mathbf{k}_i} \subseteq \mathbb{S}_{\mathbf{k}_j}$) then we omit \mathbf{k}_j from $\mathbb{S}_{\mathbf{k}_1, \dots, \mathbf{k}_t}^{n_1, n_2, \dots, n_m}$ because $\mathbb{S}_{\mathbf{k}_1, \dots, \mathbf{k}_t}^{n_1, n_2, \dots, n_m} = \mathbb{S}_{\mathbf{k}_1, \dots, \mathbf{k}_{j-1}, \mathbf{k}_{j+1}, \dots, \mathbf{k}_t}^{n_1, n_2, \dots, n_m}$ in this case. For example, we replace $\mathcal{D}_{[5,0],[1,4],[2,4],[2,3]}^{7,7}$ with $\mathcal{D}_{[5,0],[1,4],[2,3]}^{7,7}$.

Remark 7. Let \mathbb{X} be a multiset that has the division property $\mathcal{D}_{\mathbf{k}_1, \dots, \mathbf{k}_t}^{n_1, n_2, \dots, n_m}$. If $(2, 0, 0, \dots, 0) \leq \mathbf{k}_j$ for some j then

$$\bigoplus_{x \in \mathbb{X}} x_i = 0$$

for $1 \leq i \leq n_1$. In other words, \mathbb{X} is balanced in the n_1 first bits.

For the full details of the propagation of the division property, we refer the interested reader to the original paper in [12].

3.3 An Integral characteristic of 6-round MISTY1

In [11], Yosuke Todo presented a new integral characteristic for 6-round MISTY1, constructed by tracing the propagation of a division property. Todo showed that if a set V of values after the first FL layer (i.e., a set of X'_1 values) has the division property $\mathcal{D}_{[6,2,7,7,2,7,2,7,2,7,2,7]}^{7,2,7,7,2,7,2,7,2,7,2,7}$ then the corresponding set of X_7 values has the division property $\mathcal{D}_{\mathbf{k}_1, \dots, \mathbf{k}_{132}}^{7,2,7,7,2,7,2,7,2,7,2,7}$, where $\mathbf{k}_1, \dots, \mathbf{k}_{132}$ is a list of vectors presented in [11]. For our purposes, it is sufficient to know that one of the \mathbf{k}_i 's is $[2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$. In particular, if we take V_{63} to be a 63-dimensional affine subspace of the values X'_1 of a specific form, then the following equation holds:

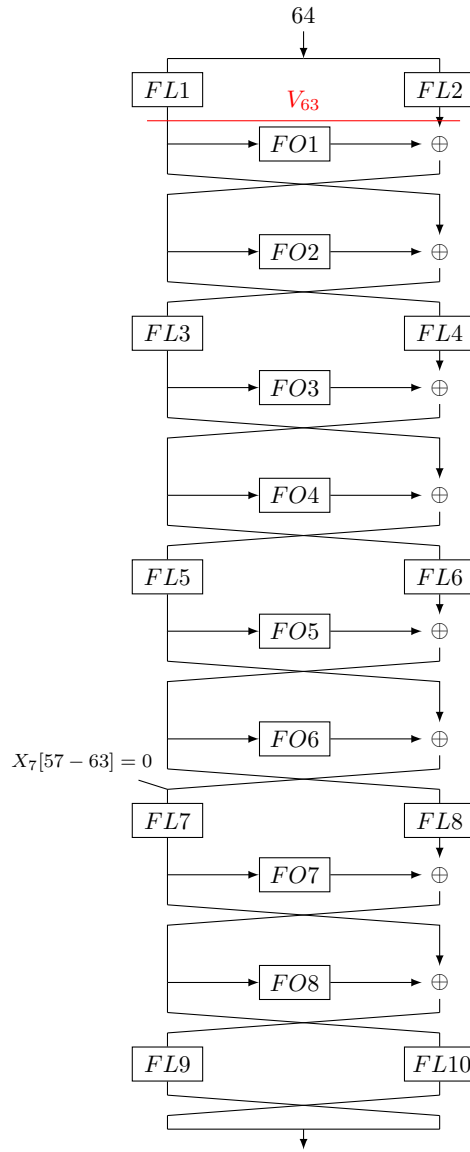
$$\bigoplus_{x \in V_{63}} X_7[57-63](x) = 0. \quad (3)$$

The “specific form” of V_{63} is defined as follows: For every 6-dimensional affine subspace $V_6 \subset \mathbb{F}_2^7$, the set $V_{63} = \{(x_6, x_{57}) : x_6 \in V_6, x_{57} \in \mathbb{F}_2^{57}\}$ has the “correct” form. Note that while there are many options for “correct” V_{63} (since there are many options for V_6), we can construct only 7 independent Eq. 3 equations. For example, define 7 particular V_{63} 's by fixing one of the 7 bits $X'_1[57-63]$ and take all the values in the other bits. Knowing that Eq. 3 holds for these seven V_{63} 's implies that Eq. 3 for all possible V_{63} 's. Therefore, attacks using Eq. 3 exploit up to 7 V_{63} 's simultaneously. Fig. 3 illustrates the 6-round integral characteristic.

3.4 Todo's Integral Attack on Full MISTY1

Todo's integral attack on full MISTY1 [11] uses the 6-round integral characteristic described above and has the following steps.

1. Choose V to be one of the “possible” V_{63} 's and ask for the encryption of $2^{63 \cdot 58}$ chosen plaintexts such that V is included in the set of intermediate X'_1 values.



V_{63} is a 63-dimensional affine subspace of the form
 $V_{63} = \{(x_6, x_{57}) : x_6 \in V_6, x_{57} \in \mathbb{F}_2^{57}\}$.

Fig. 3: A 6-Round Integral Characteristic

2. As the values of plaintexts that correspond to the values in V depend on the value of 2 key bits used in FL_1 , guess those key bits, and for each guess, perform the following steps assuming that Eq. (3) holds.
 - (a) Guess the key bits needed to partially decrypt the ciphertexts of V and get the corresponding values $X_7[57-63]$. (This step is performed efficiently using the *partial sums* technique.)
 - (b) Check whether Eq. (3) holds. A wrong key-guess will pass this seven-bit condition with probability 2^{-7} , and thus, $2^{128-7} = 2^{121}$ wrong keys are expected to remain.
3. Check the 2^{121} remaining options by exhaustive search.

The time complexity of the attack is dominated by the last step. Hence, the attack requires $2^{63.58}$ chosen plaintexts and has time complexity of 2^{121} encryptions.

To reduce the time complexity, more V_{63} 's can be used. As we can use only independent subspaces, we can use up to 7 of them. Increasing the number of the V_{63} 's has two effects. On the one hand, there is an increase in the time complexity of the filtering step and in the data complexity. On the other hand, as the filtering of wrong key-guesses is stronger, the time complexity of the exhaustive search step is reduced. The tradeoff between those two effects was considered in [11]. The optimal time complexity is $2^{107.3}$, achieved by using 4 subspaces of the form V_{63} that require $2^{64} - 2^{56} = 2^{63.994}$ chosen plaintexts.

4 Improved Attack on Full MISTY1

In this section we present our improved attack on the full MISTY1. Our attack is based on using Todo's characteristic both in the encryption and the decryption directions and on a mixture of improved key-recovery techniques. First, we discuss application of Todo's characteristic in the decryption direction and present several observations used in the attack. Then we present the first phase of the attack that requires $2^{64} - 2^{50}$ chosen ciphertexts and recovers 49 key bits in time complexity of 2^{64} encryptions. Finally, we present the second phase of the attack that recovers the rest of the key in $2^{69.5}$ time, given almost the entire codebook.

4.1 Using Todo's Characteristic in the Decryption Direction

We observe that while Todo's characteristic (described in Section 3.3 and illustrated in Fig. 3) holds in the encryption direction of MISTY1, it can be used in the decryption direction as well. Indeed, since the characteristic exploits only the general structure of MISTY1 and not the exact subkeys used in the encryption process, and as MISTY1 is a Feistel construction, the characteristic holds also for the inverse cipher MISTY1^{-1} , which possesses the same general structure. (It should be noted that MISTY1^{-1} is not exactly equivalent to MISTY1, since the FL functions are not involutions. However, as we verified experimentally, this difference does not affect the applicability of Todo's characteristic.)

Hence, we have the following "dual" claim:

Claim 8. *The equation:*

$$\bigoplus_{x \in V_{63}} X'_3[25-31](x) = 0 \quad (4)$$

holds for every 63-dimensional affine subspace V_{63} of the values X_9 of the form $V_{63} = \{(x_{57}, x_6) : x_6 \in V_6, x_{57} \in \mathbb{F}_2^{57}\}$ (where $V_6 \subset \mathbb{F}_2^7$ is a 6-dimensional affine subspace).

Our attack (see Sections 4.3, 4.4) takes advantage of Todo's characteristic in both the encryption and the decryption directions, and by that increases the filtering of wrong key-guesses. The reverse 6-round integral characteristic is illustrated in Fig. 4.

4.2 Preliminaries

In this section we give several useful observations exploited in our attack.

Computing the Attack Equation by Independent Calculations Consider the attack equation $\bigoplus_{x \in V_{63}} X'_3[25-31](x) = 0$ (see also Fig. 5a). The last functions applied during the evaluation of the attack equation are the bit-wise AND and OR of FL_4 . Since both functions are linear/affine functions, an equivalent function EFL_4 can be used. In EFL_4 the OR operation is replaced by AND operation and those for every value x it holds that $FL_4(x, K'_4, K_6) = EFL_4(x, K'_4, K_6 \oplus 1^{16}) \oplus (K_6, 0^{16})$. The constant $(K_6, 0^{16})$ can be omitted because we sum over even number of values¹. So, it is sufficient to compute the contributions of Out_1 and $FL_2(P_R)$ independently and then XOR them together. Moreover, the bits $X'_3[25-31]$ depend only on bits 9–15, 25–31 of the input of FL_4 . Thus, it is sufficient to compute the contributions of $Out_1[9-15, 25-31]$ and $FL_2(P_R)[9-15, 25-31] = FL_2(P_R[9-15, 25-31])$ independently.

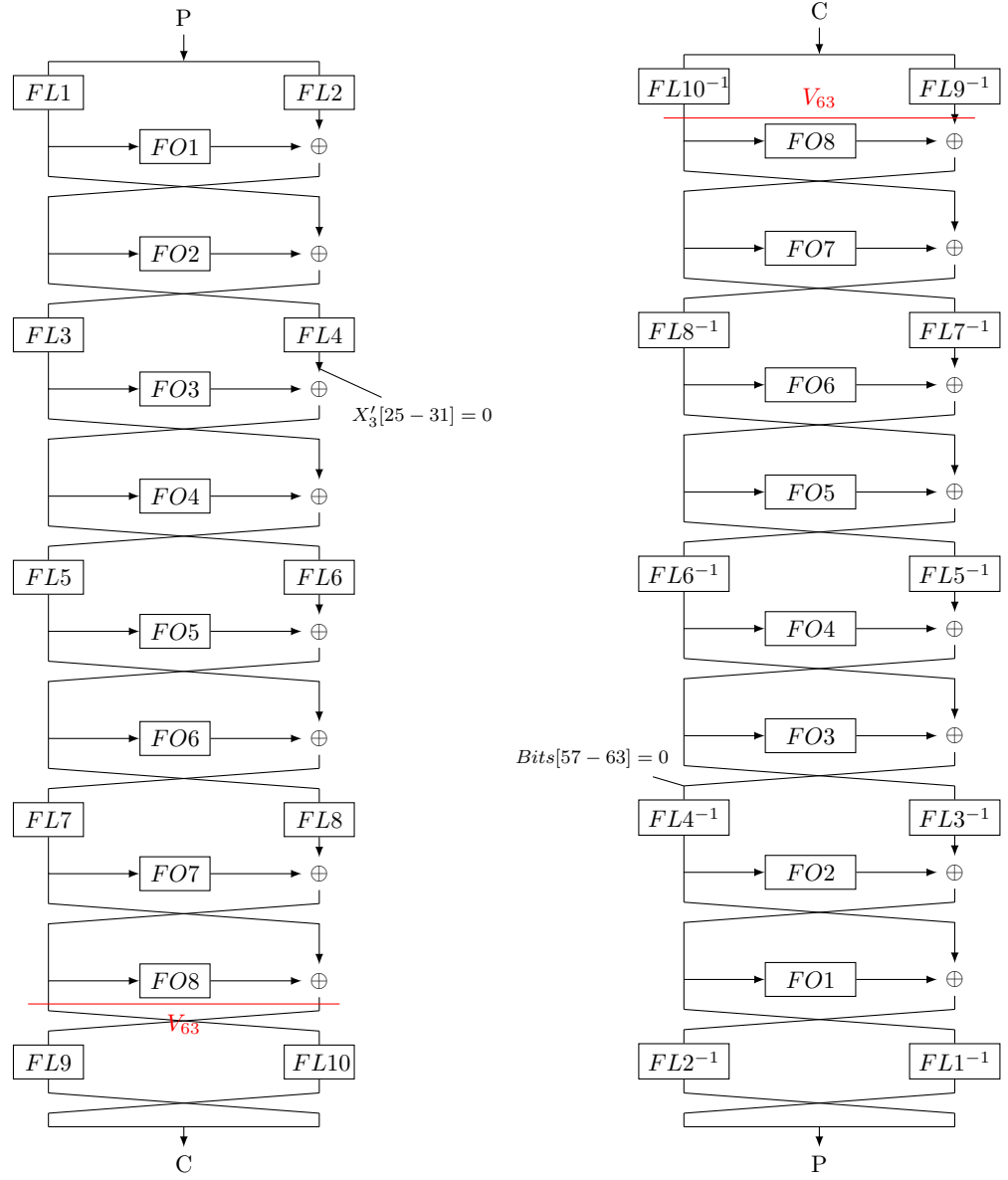
Another independence appears in computing Out_1 given the inputs to $FI_{1,2}$ and to $FI_{1,3}$. (As the computation is up to XORs with key bits, we assume that these bits have already been guessed.) Denoting the inputs to $FI_{1,2}$ and to $FI_{1,3}$ by I_2 and I_3 , respectively, the explicit formula for Out_1 is $Out_1 = (FI_{1,2}(I_2) \oplus I_3, FI_{1,2}(I_2) \oplus I_3 \oplus FI_{1,3}(I_3))$. Furthermore, we need only the values in $Out_1[9-15, 25-31]$ and these values can be found by independent computations of the four S-boxes $S_{9,1,2,1}, S_{7,1,2}, S_{9,1,3,1}, S_{7,1,3}$.

As a result, we get the following Observation:

Observation 9. Suppose we want to compute $\bigoplus_{x \in S} X'_3[25-31](x)$ for some set S of plaintexts and we already partially encrypted S using known $KL_1, KO_{1,1}, KI_{1,1}$. The rest of the computation can be done as follows. For every $x \in S$ and for every guess of the 14 bits of K_4^L, K_6^L :

1. Guess the 14 relevant bits of KL_2 (the bits K_3^L, K_5^L). For each guess, calculate the contribution of $FL_2(P_R[9-15, 25-31])$ to $\bigoplus_{x \in S} X'_3[25-31](x)$.

¹ We will write FL even when the meaning is of the equivalent function EFL .



V_{63} is a 63-dimensional affine subspace of the form
 $V_{63} = \{(x_6, x_{57}) : x_6 \in V_6, x_{57} \in \mathbb{F}_2^{57}\}.$

Fig. 4: A 6-Round Integral Characteristic of MISTY1 in the Decryption Direction

2. Guess the 9 leftmost bits of $KO_{1,2}$. For each guess, calculate the value $ES9_{1,2,1}$ (the bits that enter $S9_{1,2,1}$), encrypt it through $S9_{1,2,1}$, and store in a table its contribution to $\bigoplus_{x \in S} X'_3[25-31](x)$.
3. Guess the 7 rightmost bits of $KO_{1,2}$. For each guess, calculate the value $ES7_{1,2}$, encrypt it through $S7_{1,2}$, and store in a table its contribution to $\bigoplus_{x \in S} X'_3[25-31](x)$.
4. Guess the 9 leftmost bits of $KO_{1,3}$. For each guess, calculate the value $ES9_{1,3,1}$, encrypt it through $S9_{1,3,1}$, and store in a table its contribution to $\bigoplus_{x \in S} X'_3[25-31](x)$.
5. Guess the 7 rightmost bits of $KO_{1,3}$. For each guess, calculate the value $ES7_{1,3}$, encrypt it through $S7_{1,3}$, and store in a table its contribution to $\bigoplus_{x \in S} X'_3[25-31](x)$.
6. Finally, $\bigoplus_{x \in S} X'_3[25-31](x)$ is equal to the XOR of the five contributions listed above.

The addition of $KI_{1,2}$ and $KI_{1,3}$ was omitted from the calculation in Observation 9. We give the explanation for this in the next section.

Removing Unnecessary Key-Bits and Equivalent Keys As observed by Kühn ([9], see also [5]), the structure of FO and FI allows to use equivalent keys. In MISTY1, each $FI_{i,j}$ computation involves 32 key bits – 16 bits of $KO_{i,j}$ and 16 bits of $KI_{i,j}$. These 32 bits can be replaced by an equivalent 25-bit subkey, by “pushing” all key additions forward until they meet a non-linear operation, using the fact that linear operations can be interchanged easily. (The 7 leftmost bits of $KI_{i,j}$ do not meet any S-box of $FI_{i,j}$ and therefore are “pushed” outside $FI_{i,j}$ and added at a later place where they are merged into another subkey.)

Another way to decrease the key material involved in the computations is removing unnecessary key bits. When we compute $\bigoplus_{x \in S} X'_3[25-31](x)$ for a set S of even size, there is no effect of the key bits $KI_{1,2}$, $KI_{1,3}$ and $KO_{1,4}$ since they all affect $X'_3[25-31]$ in the same (constant) way for every plaintext and their contributions cancel each other. In our attack we consider equivalent keys and remove unnecessary key bits, as described in Fig. 5a.

Partial Sums and Piles Construction Key guessing becomes more efficient if we combine it with the partial sums technique, presented by Ferguson et al [6]. Here is an example that illustrates the technique.

Suppose there are 2^n values $\{D_i\}$ after the partial encryption of one FI function in FO_1 (the first round) and we want to calculate the XOR of the outputs from $S9_{1,2,1}$. If two values are equal in the 9 input bits of $S9_{1,2,1}$, then their contributions to the XOR of the output from $S9_{1,2,1}$ cancel each other. Hence, we sort the 2^n values according to those 9 bits, check which of the 2^9 values appears an odd number of times and encrypt through $S9_{1,2,1}$ only them (and each of them only once). Hence, it is sufficient to encrypt 2^9 values instead of 2^n values. We use the term *pile* for a set of the values that appears an odd number of times in the 9 examined bits (so that there are at most 2^9 values in a pile in total), and *constructing pile* for the process of reducing the 2^n inputs to 2^9 values. In general, we define:

Definition 10. Let $S = \{D_i\}$ be a set of n -bit values that are required for some calculation. Assume that the calculation can be done only with the knowledge of $\{D_i \cap \Omega\}$, where Ω is a mask with Hamming weight d and \cap means bitwise AND (i.e., knowing only d of the n bits is sufficient for the calculation). Denote by B the position of those d bits. Constructing pile of size 2^d in B means reducing the number of values from $|S|$ to at most 2^d by considering only the values $\{D_i \cap \Omega\}$ that appear an odd number of times.

4.3 A New Integral Attack on Full MISTY1 – The First Phase

In this section we present the first phase of our attack on the full MISTY1. This phase uses only the “reverse” 6-round characteristic presented in Section 4.1. It requires $2^{64} - 2^{50}$ chosen ciphertexts and recovers the equivalent of 49 key bits in time complexity dominated by the decryption of the ciphertexts.

Remark 11. We chose to begin with the “reverse” characteristic due to key scheduling arguments (namely, this allows to exploit the fact that the subkeys KL_1 and KO_1 share 16 key bits).

First, we choose seven independent structures V_{63} of X_9 states, to be used in the integral characteristics exploited by the attack. Then, we choose $2^{64} - 2^{50}$ ciphertexts, structured such that for any value of the subkey KL_{10} , the corresponding intermediate X_9 values contain all seven V_{63} structures. (The way to choose ciphertexts such that this property holds is presented in [11].)

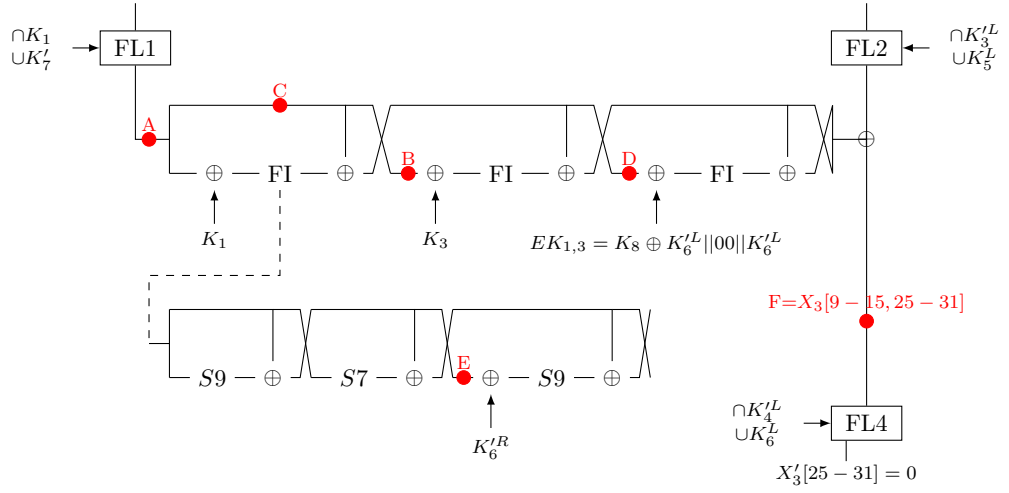
After choosing the ciphertexts, we guess the subkeys K_1 and K'_7 . This allows us to identify the right 2^{63} ciphertexts that yield each of the seven chosen V_{63} structures. An efficient procedure for this identification is presented below.

The goal of this phase is to discard wrong key guesses using the attack equation $\bigoplus_{x \in V_{63}} X'_3[25-31](x) = 0$ (derived from the “reverse” 6-round characteristic). We use a meet-in-the-middle (MITM) approach: We split the 7-bit attack equation into two equations, a 4-bit equation and a 3-bit equation. We treat each equation separately and then combine the results, getting an extra filtering by comparing the key bits involved in both equations.

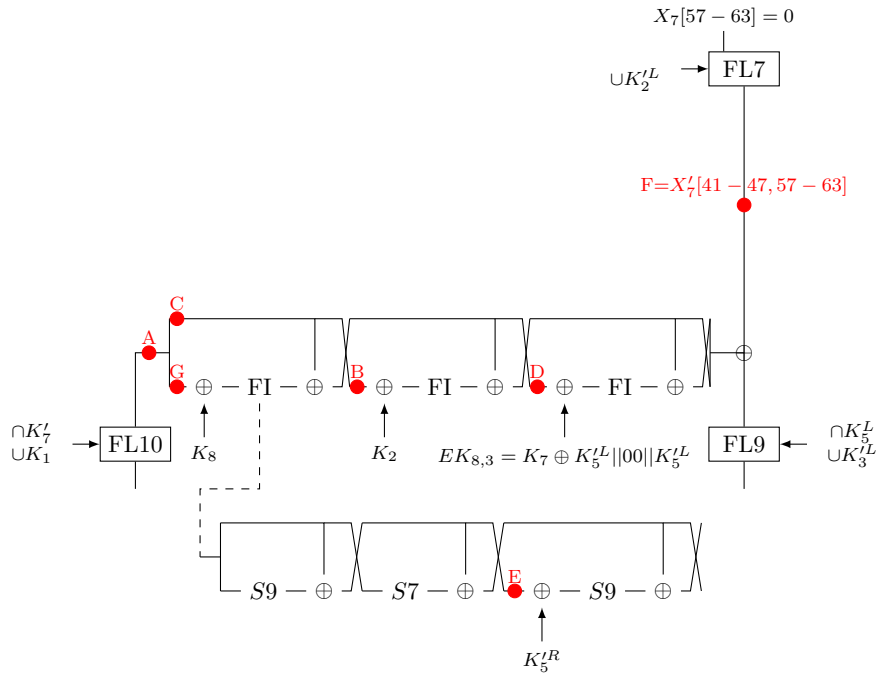
To check whether the attack equation holds, we construct the following piles (see Def. 10 and Fig. 5a):

- (i) The pile $\alpha_R = \bigoplus_{x \in V_{63}} P_R$ (which is of size 1).
- (ii) A pile of size 2^9 in the 9 leftmost bits of B.
- (iii) A pile of size 2^7 in the 7 rightmost bits of B.
- (iv) A pile of size 2^9 in the 9 leftmost bits of D.
- (v) A pile of size 2^7 in the 7 rightmost bits of D.

The total sum $\bigoplus_{x \in V_{63}} X'_3[25-31](x)$ is the sum of the contributions of the five piles. We use the MITM approach once again (thus, getting a two-dimensional MITM attack) by dividing the piles into two sets – piles (i),(ii), and (iii) on the one hand and piles (iv),(v) on the other hand. We then compute the contributions of each set of piles separately and check whether they are equal. For the right key, the contributions must be equal (since the contributions of the five piles



(a) Phase 1



(b) Phase 2

Fig. 5: Reference Figures for the Attack

sum to zero), while for a wrong key the contributions are equal with probability 2^{-7} (per V_{63} structure). Since we use seven V_{63} structures, we obtain $7 \cdot 7 = 49$ bits of filtering overall.

As mentioned above, the sets of ciphertexts that correspond to the seven V_{63} 's can be computed once K_1 and K_7' are guessed. Those key bits are indeed guessed at the beginning of this phase but if we identify the V_{63} 's *after* guessing K_1, K_7' then the time complexity of this identification would be at least $2^{32} \cdot 2^{63} \cdot 7$. We can reduce this time complexity by a precomputation, as follows.

Recall that for each structure V_{63} , the exact 2^{63} ciphertexts that are decrypted to V_{63} are determined by the value of only two key bits (one bit of K_1 and another of K_7'). For each of the 4 options, we identify the 2^{63} ciphertexts $\{C^i\}$, get the corresponding plaintexts $\{P^i\}$ and save $\alpha_R = \bigoplus_i P_R^i$. Additionally, construct a pile of size 2^{32} in P_L and save it. All information we need for the attack is now contained in the computed piles (so that once K_1, K_7' are guessed, we can continue the attack with the right piles). In this way, the time complexity of the identification step becomes $7 \cdot 4 \cdot 2^{63} = 2^{67.8}$ operations.

The procedure of Phase 1 consists of several steps for each one of the seven V_{63} 's. First, guess K_1 and K_7' , get α_R and the pile of size 2^{32} . Partially encrypt the 2^{32} values through FL_1 to the point A (the time complexity of this step is $2^{32} \cdot 2^{32} = 2^{64}$) and continue as follows:

1. Construct piles of size 2^9 and 2^7 in B (piles (ii),(iii)).
 - 1.1. Guess the 9 leftmost bits of K_3 and partially encrypt the 2^9 values of pile number (ii) to the point F.
 - 1.2. Guess the 7 rightmost bits of K_3 , encrypt the 2^7 values of pile number (iii) to the point F and calculate their XOR. Note that the XOR of the values is sufficient (instead of the values themselves) because FL_4 is linear. Explicitly, for a set of values S , the equation

$$\bigoplus_{x \in S} FL_4(x) = EFL_4\left(\bigoplus_{x \in S} x\right) \oplus \bigoplus_{x \in S} const$$

holds for the right key (where $const = (K_6, 0^{16})$).

- 1.3. Guess $K_3'[12-15], K_5[12-15]$ (8 bits of KL_2) and encrypt $\alpha_R[12-15, 28-31]$ to the point F.
- 1.4. XOR the results from the previous steps in F and call the joint XOR J_1 .
- 1.5. Guess $K_4'[12-15], K_6[12-15]$ (8 bits of KL_4), encrypt J_1 and save in a table T_1 the contribution to $X_3'[28-31]$ (4 bits of the attack equation) with the relevant key (a table of size $2^{9+7+8+8} = 2^{32}$).
2. Partially encrypt the 2^{32} values through $S9_{1,1,1}$ and $S7_{1,1}$ (note that we already know K_1), get partial outputs from FI_1 and add them to the values in C. To construct pile number (iv), we first construct a pile of size 2^{18} corresponding to the 9 leftmost bits in C and the 9 bits in E. Similarly, to construct pile number (v), we first construct a pile of size 2^{16} corresponding to the 7 rightmost bits in C and the 9 bits in E.
 - 2.1. Guess the 9 bits $K_6'^R$, encrypt the 2^{18} and 2^{16} values through FI_1 to construct piles of size 2^9 and 2^7 in D (piles number (iv) and (v)).

- 2.2. Guess the 9 leftmost bits of $EK_{1,3}$, encrypt the 2^9 values of pile number (iv) to the point F and calculate their XOR.
- 2.3. Guess the 7 rightmost bits of $EK_{1,3}$, encrypt the 2^7 values of pile number (v) to the point F and calculate their XOR.
- 2.4. XOR the results from the previous steps in the point F and call the joint XOR J_2 .
- 2.5. Guess $K'_4[12-15], K_6[12-15]$ (8 bits of KL_4), encrypt J_2 , calculate the contribution to $X'_3[28-31]$ (4 bits of the attack equation) and search for a collision in T_1 (i.e., a collision both in the contributions and in the common key bits). Save the collision in a table T_2 .
- 2.6. The size of T_2 is $2^{9+7+8+8} \cdot 2^{9+9+7+8} \cdot 2^{-8} \cdot 2^{-4 \cdot 7} = 2^{29}$, since a match must occur in $K'_4[12-15], K_6[12-15]$ (8 bits of KL_4) and in the contribution to $X'_3[28-31]$ (a 4-bit attack equation, for the seven V_{63} 's).
3. Produce T'_2 using the 3-bit attack equation $\bigoplus_{x \in V_{63}} X'_3[25-27] = 0$ by a similar process. The difference is that we guess $K'_4[9-11], K_6[9-11]$ instead of $K'_4[12-15], K_6[12-15]$ and $K'_3[9-11], K_5[9-11]$ instead of $K'_3[12-15], K_5[12-15]$. The size of T'_2 is $2^{9+7+6+6} \cdot 2^{9+9+7+6} \cdot 2^{-6} \cdot 2^{-3 \cdot 7} = 2^{32}$.
4. There are $9 + 16 + 16 = 41$ shared bit-guesses in T_2 and T'_2 (the bits $K_6'^L, K_3, EK_{1,3}$). Search for a collision in those bits and store them in a table T_3 . The size of T_3 is $2^{29} \cdot 2^{32} \cdot 2^{-41} = 2^{20}$.

We save in T_3 the corresponding guess of K_1 and K_7' , and thus, the size of T_3 is $2^{20} \cdot 2^{32} = 2^{52}$. For each suggestion in T_3 , we guess the subkey $K_6'^L$ (seven bits), retrieve the entire subkey K_6' , and then:

- Retrieve K_8 from $K_8 \oplus K_6'^L || 00 || K_6'^L$ and K_6' ,
- Retrieve K_7 from K_7' and K_8 ,
- Retrieve K_6 from K_6' and K_7 .
- Compare with the known K_6^L and discard wrong guesses (we guess seven bits of $K_6'^L$ and have a 7-bit condition, so we remain with a table of the same size).
- Retrieve K_4^L from $K_3'^L$ and K_3 .

The output of Phase 1 is two tables. The first is the table T_{phase1} , that contains 2^{52} suggestions for five full subkeys K_1, K_3, K_6, K_7', K_8 and four 7-bit subkeys $K_3'^L, K_5^L, K_4'^L, K_4^L$, sorted according to K_1, K_7', K_8 . The second is the table T'_{phase1} with the same 2^{52} suggestions, but sorted according to $K_1, K_7', K_3'^L, K_5^L$.

The time complexity of Phase 1 for a single V_{63} and a fixed guess of K_1, K_7' is less than $2^{33.7}$ operations. The calculation is given in Table 3 and composed of the sum of time complexities of the steps involving key guessing and (partial) encryption. Since we use seven V_{63} 's, the total time complexity of Phase 1 (with the precomputation) is bounded by

$$\mathbf{T1} = 2^{67.8} + 7 \cdot 2^{32} \cdot 2^{33.7} = 2^{69.2}$$

operations, which is less than 2^{64} encryptions.

Table 3: Time complexity of Phase 1 for single V_{63} and K_1, K_7' guess

Step	Time complexity	Description
1.1.	$2^{18} = 2^9 \cdot 2^9$	Guess 9 bits of K_3 and partially encrypt 2^9 values.
1.2.	$2^{14} = 2^7 \cdot 2^7$	Guess 7 bits of K_3 and partially encrypt 2^7 values.
1.3.	$2^8 = 2^8 \cdot 1$	Guess 8 bits of KL_2 and partially encrypt $\alpha_R[12-15,28-31]$.
1.4.	$2^{24} = 2^9 \cdot 2^7 \cdot 2^8$	Calculate J_1 .
1.5.	$2^{32} = 2^{24} \cdot 2^8$	Guess 8 bits of KL_4 and partially encrypt J_1 .
2.1.	$2^{27.3} = 2^9 \cdot (2^{18} + 2^{16})$	Guess 9 bits of K_6^R and partially encrypt 2^{18} and 2^{16} values.
2.2.	$2^{27} = 2^{9+9} \cdot 2^9$	Guess 9 bits of $EK_{1,3}$ and partially encrypt 2^9 values.
2.3.	$2^{23} = 2^{9+7} \cdot 2^7$	Guess 7 bits of $EK_{1,3}$ and partially encrypt 2^7 values.
2.4.	$2^{25} = 2^9 \cdot 2^9 \cdot 2^7$	Calculate J_2 .
2.5.	$2^{33} = 2^{25} \cdot 2^8$	Guess 8 bits of KL_4 and partially encrypt J_2 .
3	< step 1+ step 2	Similar to 1 and 2, using $\oplus X'_3[25-27] = 0$.
4	$2^{32.2} = 2^{29} + 2^{32}$	Compare two tables of sizes 2^{29} and 2^{32} .
Total	< $2^{33.7}$	

After the first phase is completed, the rest of the key can be found immediately in time complexity of 2^{79} encryptions, by guessing the rest of the key (for each of 2^{52} entries of T_{stage1} , guess 2^{27} key bits $K_4^R, K_5^R, K_3'^R$ and derive a master-key candidate) and checking it by a trial encryption. Of course, this approach does not require additional data. In the following section we show that if more data is available, the time complexity of the second phase can be reduced to $2^{69.5}$ encryptions.

4.4 A New Integral Attack on Full MISTY1 – The Second Phase

The second phase of our attack uses the 6-round characteristic presented in Section 3.3 to apply a second filtering to the key suggestions remaining from Phase 1.

The goal of this phase is to discard wrong key guesses that pass Phase 1, using the attack equation $\bigoplus_{x \in V_{63}} X_7[57-63](x) = 0$ (derived from the 6-round characteristic).

To check whether the attack equations holds, we construct the following piles (see Fig. 5b):

- (i) The pile $\beta_R = \bigoplus_{x \in V_{63}} C_R$ (which is of size 1).
- (ii) A pile of size 2^9 in the 9 leftmost bits of B.
- (iii) A pile of size 2^7 in the 7 rightmost bits of B.
- (iv) A pile of size 2^{16+9} in the 9 leftmost bits of C + 16 bits of D.

(v) A pile of size 2^7 in the 7 rightmost bits of C + 16 bits of D.

The sum $\bigoplus_{x \in V_{63}} X_7[57-63](x)$ is the sum of the contributions of each pile separately. As in Phase 1 above, we calculate the contribution of piles number (i),(ii),(iii) and of piles number (iv),(v) separately and check whether the two contributions are equal. For the right key they must be equal, and for a wrong key guess they are equal with probability 2^{-7} (per V_{63} structure). There are 7 optional V_{63} 's and we use only two of them to get $2 \cdot 7 = 14$ bits filtering.

The procedure of Phase 2 consists of several steps (similarly to Phase 1) for each one of the seven V_{63} 's.

First, we ask for the encryption of $2^{64} - 2^{50}$ chosen plaintexts, such that each of the seven chosen V_{63} structures (of X'_1 values) is covered by the texts. (This data requirement, combined with the requirement of Phase 1, makes the data complexity equal to $2^{64} - 2^{36}$ chosen plaintexts and ciphertexts, which is rather close to the entire codebook). Second, we construct pile number (i) and a pile of size 2^{32} in C_L for constructing the other piles.

Then, we guess K_1 and K'_7 and partially decrypt the 2^{32} values through FL_{10} to the point A (the time complexity of this step is negligible compared to the total time complexity). The procedure continues as follows:

1. Construct piles of size 2^9 and 2^7 in B (piles (ii),(iii)).
 - 1.1. Guess the 9 leftmost bits of K_2 and decrypt the 2^9 values of pile number (ii) to the point F.
 - 1.2. Guess the 7 rightmost bits of K_2 , decrypt the 2^7 values of pile number (iii) to the point F and calculate their XOR.
 - 1.3. Guess $K_3^{L'}, K_5^L$ (14 bits of KL_9 and decrypt the $\beta_R[9-15,25-31]$ to the point F.
 - 1.4. XOR the results from the previous steps in the point F and call the joint XOR J_1 .
 - 1.5. Get the $K_1, K'_7, K_3^{L'}, K_5^L$ entry of T'_{stage1} . The entry consists of $2^{52-46} = 2^6$ values for $K_3, K_6, K_8, K_4^{L'}, K_4^L$. Compute K'_2 from K_2, K_3 , decrypt J_1 and save in a table T_1 the contribution to $X'_7[57-63]$, along with the relevant key (a table of size $2^{9+7+14+6} = 2^{36}$).
2. To construct pile number (iv), we first construct a pile of size 2^{16+9} that corresponds to the 9 leftmost bits of C + 16 bits of G. Similarly, to construct pile number (v), we first construct a pile of size 2^{16+7} that corresponds to the 7 rightmost bits in C + 16 bits of G.
 - 2.1. Guess K_8 and construct a pile of size 2^{9+9} that corresponds to the 9 leftmost bits of C + 9 bits of E. Construct a pile of size 2^{9+7} that corresponds to the 7 rightmost bits of C + 9 bits of E.
 - 2.2. Guess the 9 bits K_5^{LR} , decrypt the 2^{18} and 2^{16} values of the piles from the previous step through FI_1 to construct piles of size 2^9 and 2^7 in D (piles number (iv) and (v)). For the piles of size 2^9 , this step can be performed more efficiently by guessing key bits one by one, as described in [2].

- 2.3. Get the K_1, K_7', K_8 entry of T_{stage1} . The entry consists of $2^{52-48} = 2^4$ values for $K_3, K_6, K_3^L, K_5^L, K_4^L, K_4^L$. For each value, guess K_5^R , derive K_5 , compute K_5' (from K_5, K_6) and compare with $K_5'^R$ that was guessed. We remain with 2^4 values for K_5' and hence 2^4 values for $EK_{8,3}$.
 - 2.4. With the known $EK_{8,3}$, decrypt the 2^9 values of pile number (iv) to the point F and calculate their XOR. In addition, decrypt the 2^7 values of pile number (v) to the point F and calculate their XOR.
 - 2.5. XOR the results from the previous steps at the point F and call the joint XOR J_2 .
 - 2.6. Guess $K_2'[9-15]$, decrypt J_2 , calculate the contribution to $X_7[57-63]$ and search for a collision in the table T_1 (i.e., collision in the contributions + in the common key bits).
 - 2.7. The expected number of collisions is $2^{9+7+14+6} \cdot 2^{16+9+4+7} \cdot 2^{-7} \cdot 2^{-2 \cdot 7} \cdot 2^{-20} = 2^{31}$, since a match must occur in $K_2'[9-15]$, in the contributions in $X_7[57-63]$ (for two V_{63} 's) and in the entry of T_{stage1} .
3. For each of the 2^{31} suggestions, guess K_4^R , and use the knowledge of K_4, K_5, K_4^L to get a 7-bit filtering. This yields 2^{65} suggestions for the entire key. Test them with a single plaintext/ciphertext pair. Only $2^{65} \cdot 2^{-64} = 2$ suggestions are expected to remain. Test them with another plaintext/ciphertext pair and find the key.

The time complexity of stage 2 for each structure V_{63} and each guess of K_1, K_7' is less than $2^{42.5}$. The calculation is given in Table 4 and is composed of the sum of the time complexities of the steps involving key guessing and (partial) decryption. Since we use two V_{63} 's, the total time complexity of Phase 2 is bounded by

$$\mathbf{T2} = 2 \cdot 2^{32} \cdot 2^{42.5} = 2^{75.5}$$

operations.

Table 4: Time complexity of Phase 2 for single V_{63} and K_1, K_7^l guess

Step	Time complexity	Description
1.1.	$2^{18} = 2^9 \cdot 2^9$	Guess 9 bits of K_2 and partially decrypt 2^9 values.
1.2.	$2^{14} = 2^7 \cdot 2^7$	Guess 7 bits of K_2 and partially decrypt 2^7 values.
1.3.	$2^{14} = 2^{14} \cdot 1$	Guess 14 bits of KL_9 and partially decrypt $\beta_R[9-15,25-31]$.
1.4.	$2^{30} = 2^9 \cdot 2^7 \cdot 2^{14}$	Calculate J_1 .
1.5.	$2^{36} = 2^{30+6} \cdot 1$	Partially decrypt J_1 .
2.1.	$2^{41.3} = 2^{16} \cdot (2^{25} + 2^{23})$	Guess K_8 and partially decrypt 2^{25} and 2^{23} values.
2.2.	$2^{41.3} = 2^{16+9} \cdot (2^{14} + 2^{16})$	Guess 9 bits of K_5^{lR} and partially encrypt 2^{18} and 2^{16} values.
2.3.	$2^{38} = 2^{16+9+4+9}$	Get 2^4 values from T_{stage2} and for each value guess 9 bits of K_5^R .
2.4.+2.5.	$2^{38.3} = 2^{16+9+4} \cdot (2^9 + 2^7)$	Partially decrypt 2^9 and 2^7 values. Calculate J_2 .
2.6.	$2^{36} = 2^{16+9+4+7} \cdot 1$	Partially decrypt J_2 .
3	$2^{33} = 2^{31} \cdot 2^2$	Obtain another 7-bit filtering (upon the attack equation filtering of the two V_{63} 's), and then test remaining key suggestions by trial encryptions.
Total	$< 2^{42.5}$ simple operations + 2^{33} full MISTY1 encryptions	

All the operations of Phase 2 are simple operations besides Step 3 that consists of 2^{33} full MISTY1 encryptions (for each K_1, K_7^l guess). Thus, the time complexity **T2** is $2^{75.5}$ simple operations + 2^{65} full MISTY1 encryptions. Assuming that each simple operation is comparable to an S-box evaluation, the time complexity of stage 2 (in terms of full encryptions) is $\frac{2^{75.5}}{8.9} + 2^{65} = 2^{69.5}$, since MISTY1 has 9 S-boxes in each of its 8 rounds.

The output of Phase 1 required tables of size 2^{52} . In a naive approach, this is the memory complexity of the attack but maybe it can be reduced.

5 Summary and Conclusions

In this paper we presented a new attack on the full MISTY1. The attack uses Todo's 6-round integral characteristic[11] both in the encryption direction (as it was used by Todo) and in the decryption direction. The attack equations derived from the characteristics provide a filtering for wrong key guesses. Exploiting the filtering efficiently by using partial sums, two-dimensional meet-in-the-middle and other techniques, our attack has time complexity of $2^{69.5}$ encryptions. This is a reduction by a factor of 2^{38} over Todo's attack that has time complexity of $2^{107.3}$ encryptions.

While our attack is clearly impractical due to its high data complexity, it shows that MISTY1 has a rather low security margin, providing only 70 bits of security.

As a problem for further research, it will be interesting to find out whether the data complexity can be reduced. A possible direction for achieving this is finding additional 6-round integral characteristics of a lower order.

References

1. 3rd Generation Partnership Project. Specification of the 3GPP Confidentiality and Integrity Algorithms - Document 2: KASUMI Specification (Release 6). Technical Report 3GPP TS 35.202 V6.1.0 (2005-09), September 2005.
2. Achiya Bar-On. Improved Higher-Order Differential Attacks on MISTY1. In *Fast Software Encryption, 22nd International Workshop, FSE '15, Istanbul, TURKEY, March 8-11, 2015, to appear*, 2015.
3. Joan Daemen, Lars Knudsen, and Vincent Rijmen. The block cipher Square. In Eli Biham, editor, *Fast Software Encryption*, volume 1267 of *Lecture Notes in Computer Science*, pages 149–165. Springer Berlin Heidelberg, 1997.
4. Itai Dinur, Orr Dunkelman, and Adi Shamir. Improved Attacks on Full GOST. In Anne Canteaut, editor, *Fast Software Encryption*, volume 7549 of *Lecture Notes in Computer Science*, pages 9–28. Springer Berlin Heidelberg, 2012.
5. Orr Dunkelman and Nathan Keller. Practical-Time Attacks Against Reduced Variants of MISTY1. *Design, Codes and Cryptography, to appear*, 2013.
6. Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Michael Stay, David Wagner, and Doug Whiting. Improved Cryptanalysis of Rijndael. In *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, pages 213–230, 2000.
7. Keting Jia and Leibo Li. Impossible Differential Attacks on Reduced-Round MISTY1. In Dong Hoon Lee and Moti Yung, editors, *WISA*, volume 7690 of *Lecture Notes in Computer Science*, pages 15–27. Springer, 2012.
8. Lars Knudsen and David Wagner. Integral Cryptanalysis. In Joan Daemen and Vincent Rijmen, editors, *Fast Software Encryption*, volume 2365 of *Lecture Notes in Computer Science*, pages 112–127. Springer Berlin Heidelberg, 2002.
9. Ulrich Kühn. Improved Cryptanalysis of MISTY1. In Joan Daemen and Vincent Rijmen, editors, *Fast Software Encryption*, volume 2365 of *Lecture Notes in Computer Science*, pages 61–75. Springer Berlin Heidelberg, 2002.
10. Mitsuru Matsui. New block encryption algorithm misty. In Eli Biham, editor, *Fast Software Encryption*, volume 1267 of *Lecture Notes in Computer Science*, pages 54–68. Springer Berlin Heidelberg, 1997.
11. Yosuke Todo. Integral Cryptanalysis on Full MISTY1. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, to appear*, 2015.
12. Yosuke Todo. Structural Evaluation by Generalized Integral Property. In *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, pages 287–314, 2015.
13. Yukiyasu Tsunoo, Teruo Saito, Takeshi Kawabata, and Hirokatsu Nakagawa. Finding Higher Order Differentials of MISTY1. *IEICE Transactions*, 95-A(6):1049–1055, 2012.

14. Wentan Yi and Shaozhen Chen. Multidimensional Zero-Correlation Linear Attacks on Reduced-Round MISTY1. *CoRR*, abs/1410.4312, 2014.