Cryptanalysis of the ISDB Scrambling Algorithm (MULTI2)

Jean-Philippe Aumasson[1⋆], Jorge Nakahara Jr.[2⋆⋆], and Pouyan Sepehrdad[2]

[1] FHNW, Windisch, Switzerland
[2] EPFL, Lausanne, Switzerland
jeanphilippe.aumasson@gmail.com,
{jorge.nakahara,pouyan.sepehrdad}@epfl.ch

Abstract. MULTI2 is the block cipher used in the ISDB standard for scrambling digital multimedia content. MULTI2 is used in Japan to secure multimedia broadcasting, including recent applications like HDTV and mobile TV. It is the only cipher specified in the 2007 Japanese ARIB standard for conditional access systems. This paper presents a theoretical break of MULTI2 (not relevant in practice), with shortcut key recovery attacks for any number of rounds. We also describe equivalent keys and linear attacks on reduced versions with up 20 rounds (out of 32), improving on the previous 12-round attack by Matsui and Yamagishi. Practical attacks are presented on up to 16 rounds.

Keywords: ISDB, ARIB, MULTI2, block cipher, linear cryptanalysis, conditional access

1 Introduction

MULTI2 is a block cipher developed by Hitachi in 1988 for general-purpose applications, but which has mainly been used for securing multimedia content. It was registered in ISO/IEC 9979[3] [8] in 1994, and is patented in the U.S. [13, 14] and in Japan [7]. MULTI2 is the only cipher specified in the 2007 Japanese standard ARIB for conditional access systems [2]. ARIB is the basic standard of the recent ISDB (for Integrated Services Digital Broadcasting), Japan's standard for digital television and digital radio (see http://www.dibeg.org/)

Since 1995, MULTI2 is the cipher used by satellite and terrestrial broadcasters in Japan [16, 18] for protecting audio and video streams, including HDTV, mobile and interactive TV. In 2006, Brazil adopted ISDB as a standard for

⋆ Supported by the Swiss National Science Foundation, project no. 113329.
⋆⋆ The work described in this paper has been supported in part by the European Commission through the ICT Programme under contract ICT-2007-216646 ECRYPT II. The information in this document reflects only the author's views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

[3] The ISO/IEC 9979, under which cryptographic algorithms were registered, was withdrawn on Feb. 2006 because of its redundancy with the ISO/IEC 18033 standard.

digital-TV, and several other countries are progressively switching to ISDB (Chile, Ecuador, Peru, Philippines and Venezuela). But for the moment only Japan uses the conditional access features of ISDB, thus MULTI2 is only used in Japan.

MULTI2 has a Feistel structure and encrypts 64-bit blocks using a 256-bit "system key" and a 64-bit "data key". The ISO register recommends at least 32 rounds. A previous work by Matsui and Yamagishi [11] reports attacks on a reduced version of MULTI2 with 12 rounds. Another work by Aoki and Kurokawa [1] reports an analysis of the round mappings of MULTI2, with results independently rediscovered in the present work.

Contribution. This paper presents new cryptanalytic results on MULTI2, including the description of large sets of equivalent keys, a guess-and-determine attack for any number of rounds, a linear attack on 20 rounds, and a related-key slide attack (see Table 1 for complexities). Despite no practical threat to conditional access systems, our results raise concerns on the intrinsic security of MULTI2.

Table 1. Summary of our attacks on MULTI2 (Data is given in known plaintexts).

#Rounds	Time	Data	Memory	Attack
4	$2^{16.4}$	$2^{16.4}$	—	linear distinguisher*
8	$2^{27.8}$	$2^{27.8}$	—	linear distinguisher*
12	$2^{39.2}$	$2^{39.2}$	—	linear distinguisher*
16	$2^{50.6}$	$2^{50.6}$	—	linear distinguisher*
20	$2^{93.4}$	$2^{39.2}$	$2^{39.2}$	linear key-recovery
r	$2^{185.4}$	3	2^{31}	guess-and-determine key-recovery
$r \equiv 0 \bmod 8$	$2^{128}/r$	2^{33}	2^{33}	related-key slide key-recovery

$\star$: time complexity is # of parity computations instead of # of encryptions.

2 Description of MULTI2

MULTI2 (Multi-Media Encryption Algorithm 2) is a Feistel block cipher that operates on 64-bit blocks, parametrized by a 64-bit data key and a 256-bit system key. Encryption depends only on a 256-bit key derived from the data and system keys. This encryption key is divided into eight subkeys. MULTI2 uses four key-dependent round functions π_1, π_2, π_3, and π_4, repeated in this order. The ISO register entry recommends at least 32 rounds, which is the number of rounds used in the ISDB standard. We denote MULTI2's keys as follows, parsing them into 32-bit words (see Fig. 1):

- $d = (d_1, d_2)$ is the 64-bit *data key*

- $s = (s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8)$ is the 256-bit *system key*
- $k = (k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8)$ is the 256-bit *encryption key*

MULTI2 uses no S-boxes, but only a combination of XOR ($\oplus$), modulo 2^{32} addition ($+$) and subtraction ($-$), left rotation ($\lll$) and logical OR ($\vee$). Below we denote L (resp. R) the left (resp. right) half of the encrypted data, and k_i a 32-bit encryption subkey:

- π_1 is the identity mapping: $\pi_1(L) = L$. It is the only surjective and key independent round transformation.
- π_2 maps 64 bits to 32 bits, and returns

$$\pi_2(R, k_i) = (x \lll 4) \oplus x \tag{1}$$

  where $x = ((R + k_i) \lll 1) + R + k_i - 1$. From the definition (1) it follows that $\pi_2(R, k_i) = \pi_2(k_i, R)$, for any $k_i, R \in \{0,1\}^{32}$. Moreover, π_2 can be expressed as a function of a single value, $R + k_i$. Due to the feed forward in (1), π_2 can not be surjective. The range of π_2 contains exactly $265\,016\,655 \approx 2^{28}$ elements (only 6.2% of $\{0,1\}^{32}$, against 63% expected for a random function [12, §2.1.6]). Moreover, the set of 32 bit values output by π_2 is always the same. This follows from the observation that for fixed R if $0 \leq k_i \leq 2^{32} - 1$, then $0 \leq R + k_i \leq 2^{32} - 1$ and the same holds if k_i is fixed and $0 \leq R \leq 2^{32} - 1$.
- π_3 maps 96 bits to 32 bits, and returns

$$\pi_3(L, k_i, k_j) = (x \lll 16) \oplus (x \vee L) \tag{2}$$

  where

$$x = \big(\big((y \lll 8) \oplus y + k_j\big) \lll 1\big) - \big((y \lll 8) \oplus y + k_j\big)$$

  where $y = ((L + k_i) \lll 2) + L + k_i + 1$. The range of π_3 spans approximately $2^{30.8}$ values, that is, 43% of $\{0,1\}^{32}$, for a *fixed encryption key*. The fraction of the range covered by π_3 is not the same for every $k_i, k_j \in \{0,1\}^{32}$, because $\pi_3(L, k_i, k_j) \neq \pi_3(L, k_j, k_i)$.
- π_4 maps 64 bits to 32 bits, and returns

$$\pi_4(R, k_i) = \big((R + k_i) \lll 2\big) + R + k_i + 1. \tag{3}$$

  From the definition of 3, it follows that $\pi_4(R, k_i) = \pi_4(k_i, R)$ for any $k_i, R \in \{0,1\}^{32}$. The range of π_4 contains exactly $1\,717\,986\,919 \approx 2^{30.7}$ elements (i.e., 40.6% of $\{0,1\}^{32}$). The reasoning is the same as for π_2.

An additional property is the fact that these π_j functions do not commute, that is, $\pi_i \circ \pi_j \neq \pi_j \circ \pi_i$, for $i \neq j$, where $\circ$ is functional composition. Thus, the π_j mapping cannot be purposefully clustered or permuted in the cipher framework to ease cryptanalysis.

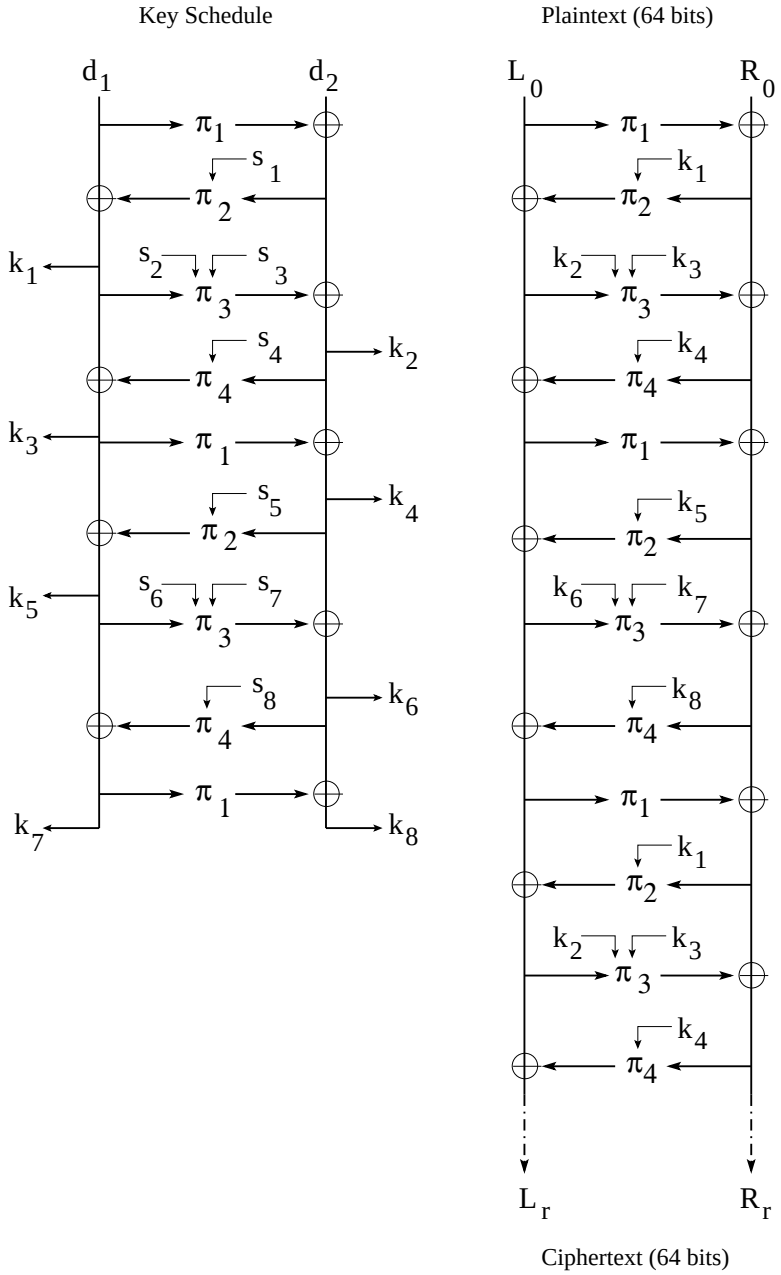

Fig. 1. Key schedule (left) and encryption (right) in MULTI2: the encryption key k is derived from the system key s and the data key d. Only k is used during the encryption.

Encryption. Given subkeys $k_1, \ldots, k_8$ and a plaintext (L_0, R_0), MULTI2 computes the first eight rounds as follows (see Fig. 1):

1. $R_1 \leftarrow R_0 \oplus \pi_1(L_0)$
2. $L_1 \leftarrow L_0$; $L_2 \leftarrow L_1 \oplus \pi_2(R_1, k_1)$
3. $R_2 \leftarrow R_1$; $R_3 \leftarrow R_2 \oplus \pi_3(L_2, k_2, k_3)$
4. $L_3 \leftarrow L_2$; $L_4 \leftarrow L_3 \oplus \pi_4(R_3, k_4)$
5. $R_4 \leftarrow R_3$; $R_5 \leftarrow R_4 \oplus \pi_1(L_4)$
6. $L_5 \leftarrow L_4$; $L_6 \leftarrow L_5 \oplus \pi_2(R_5, k_5)$
7. $R_6 \leftarrow R_5$; $R_7 \leftarrow R_6 \oplus \pi_3(L_6, k_6, k_7)$
8. $L_7 \leftarrow L_6$; $L_8 \leftarrow L_7 \oplus \pi_4(R_7, k_8)$
9. $R_8 \leftarrow R_7$

This sequence is repeated (with suitably incremented subscripts) until the desired number of rounds r, and the ciphertext (L_r, R_r) is returned. The subkeys $k_1, \ldots, k_8$ are reused for each sequence $\pi_1, \ldots, \pi_4, \pi_1, \ldots, \pi_4$.

Key Schedule. The key schedule of MULTI2 "encrypts" a data key (d_1, d_2) (as plaintext) through nine rounds, using the system key $s_1, \ldots, s_8$. The round subkeys $k_1, \ldots, k_8$ are extracted as follows (see Fig. 1):

- $k_1 \leftarrow d_1 \oplus \pi_2(d_1 \oplus d_2, s_1)$
- $k_2 \leftarrow d_1 \oplus d_2 \oplus \pi_3(k_1, s_2, s_3)$
- $k_3 \leftarrow k_1 \oplus \pi_4(k_2, s_4)$
- $k_4 \leftarrow k_2 \oplus k_3$
- $k_5 \leftarrow k_3 \oplus \pi_2(k_4, s_5)$
- $k_6 \leftarrow k_4 \oplus \pi_3(k_5, s_6, s_7)$
- $k_7 \leftarrow k_5 \oplus \pi_4(k_6, s_8)$
- $k_8 \leftarrow k_6 \oplus k_7$

MULTI2 in ISDB. In ISDB, MULTI2 is mainly used via the B-CAS card [6] for copy control to ensure that only valid subscribers are using the service. MULTI2 encrypts transport stream packets in CBC or OFB mode. The same system key is used for all conditional-access applications, and another system key is used for other applications (DTV, satellite, etc.). The 64-bit data key is refreshed every second, sent by the broadcaster and encrypted with another block cipher. Therefore only the data key is really secret, since the system key can be obtained from the receivers. Details can be found in the ARIB B25 standard [2].

3 Equivalent Keys

The key schedule of MULTI2 maps a $(256 + 64)$-bit data-and-system key to a 256-bit encryption key (see Fig. 1). This means 64 bits of redundancy (leading to 2^{64} collisions). Further, the 256-bit encryption key $k = (k_1, \ldots, k_8)$ has entropy at most 192 bits, because the key schedule sets $k_4 = k_3 \oplus k_2$ and $k_8 = k_7 \oplus k_6$. Hence, the knowledge of two subkeys in (k_2, k_3, k_4) is sufficient to compute the

third. The key schedule thus induces a loss of at least 128 bits of entropy, from the 320-bit (s, d) key. Therefore, the average size of equivalence key classes is 2^{128}.

Large sets of colliding pairs (s, d) can be found as follows: given (s, d), one just has to find s'_1 such that $\pi_2(d_1 \oplus d_2, s'_1) = \pi_2(d_1 \oplus d_2, s_1)$; or s'_2, s'_3 such that $\pi_3(k_1, s_2, s_3) = \pi_3(k_1, s'_2, s'_3)$; or s'_4 such that $\pi_4(k_2, s'_4) = \pi_4(k_2, s_4)$; or s'_5 such that $\pi_2(k_4, s_5) = \pi_2(k_4, s'_5)$; or s'_6, s'_7 such that $\pi_3(k_5, s_6, s_7) = \pi_3(k_5, s'_6, s'_7)$; or s'_8 such that $\pi_4(k_6, s_8) = \pi_4(k_6, s'_8)$. Each of these conditions are independent. The result is a (series of) equivalent keys (s', d) that lead to the same encryption key as the pair (s, d).

However, there exist no equivalent keys with the same system key and distinct data keys. This is because the key schedule uses the data key as plaintext, hence the encryption key is trivially invertible (see Fig. 1).

Note that [8] suggests to use MULTI2 as building block for constructing hash functions. If the construction is not carefully chosen, however, equivalent keys in MULTI2 could lead to simple collisions. For example, in Davies-Meyer mode the compression function would return $E_m(h) \oplus h$, with h a chaining value and m a message block; since equivalent keys are easy to find, it is easy as well to find two (or more) distinct message block that produce the same encryption key, and thus that give multicollisions.

4 Guess-and-Determine Attack

We describe a known-plaintext attack that recovers the 256-bit encryption key in about $2^{185.4}$ r-round encryptions. The attack works for any number r of rounds, and uses only three known plaintexts/ciphertext pairs.

We recall the loss of key entropy due to redundancy in the key schedule of MULTI2 described in Sect. 3.

One recovers $k_1, \ldots, k_8$ using a guess-and-determine strategy, exploiting the non-surjectivity of the round functions π_2 and π_4 (see key schedule in Fig. 1):

1. guess k_1 and k_2 (2^{64} choices)
2. guess $\pi_4(k_2, s_4)$ ($2^{30.7}$ choices), and deduce $k_3 = k_1 \oplus \pi_4(k_2, s_4)$
3. set $k_4 = k_2 \oplus k_3$
4. guess $\pi_2(k_4, s_5)$ (2^{28} choices), and deduce $k_5 = k_3 \oplus \pi_2(k_4, s_5)$
5. guess $\pi_3(k_5, s_6, s_7)$ (2^{32} choices), and deduce $k_6 = k_4 \oplus \pi_3(k_5, s_6, s_7)$
6. guess $\pi_4(k_6, s_8)$ ($2^{30.7}$ choices), and deduce $k_7 = k_5 \oplus \pi_4(k_6, s_8)$
7. set $k_8 = k_6 \oplus k_7$

A guess of $k_1, \ldots, k_8$ is verified using three known-plaintext/ciphertext pairs (each pair gives a 64-bit condition). The total cost is about $2^{185.4}$ r-round encryptions and 2^{31} 32-bit words of memory. Note that the non-surjectivity of $\pi_3(k_5, s_6, s_7)$ cannot be exploited here, because the range depends on the system subkeys, which are unknown.

Once the encryption key k is found, one can recover all equivalent 256-bit system keys s and 64-bit data keys d as follows: starting from the end of the key

schedule (Fig. 1), one iteratively searches for a valid s_8 (2^{32} π_4-computations), then a valid pair (s_6, s_7) (2^{64} π_3-computations), and so on (the complexities add up) until recovering s_4. For computing (s_2, s_3) we need the value of $d_1 \oplus d_2$. The cost for this case is $2^{32+32+32} = 2^{96}$ π_3-computations. For s_1 we need the separate values of d_1 and d_2. Since we already computed $d_1 \oplus d_2$, the cost is $2^{32+32} = 2^{64}$ π_2-computations. The final complexity is dominated by 2^{96} π_3-computations to recover all candidates pairs (s, d). The cost of computing one of the pairs (s, d) is dominated by 2^{33} π_3-computations.

5 Linear Attacks

The non-surjectivity of the round functions π_2, π_3, π_4 motivates the study of linear relations [10] for particular bitmasks. Usually, one looks for nonzero input and output bitmasks for individual cipher components, with high bias. But for MULTI2, we look for linear relations of the form $0 \xrightarrow{\pi_i} \Gamma$, $2 \leq i \leq 4$, $\Gamma \neq 0$. Because of 4-round repetition of π_i mappings in MULTI2, and to optimize the search, we looked only for *iterative* linear relations that cover all four consecutive π_i round mappings.

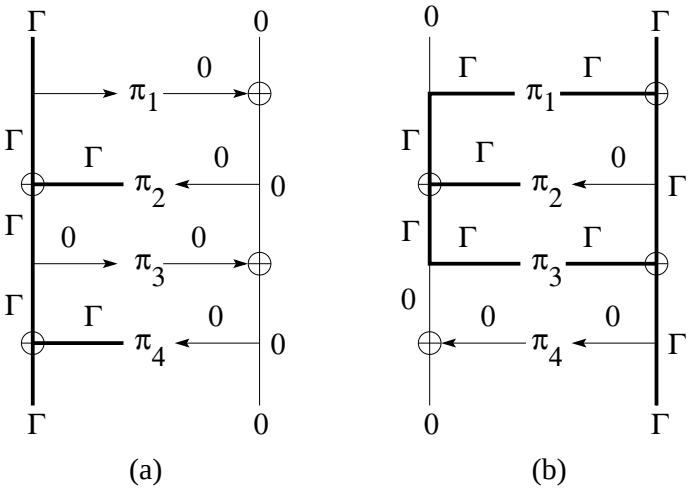

Fig. 2. Four-round iterative linear trails.

In Fig. 2(a), only π_2 and π_4 are active, that is, effectively participate in the linear relation (linear trails are depicted by thick lines). In Fig. 2(b), π_2 and π_3 are active, but for π_3 the linear approximation has the form $\Gamma \xrightarrow{\pi_3} \Gamma$, where $\Gamma \neq 0$. Alternative iterative linear relations are depicted in Fig. 3. In Fig. 3(a), there is one instance of π_2 and π_4, and two instances of π_3 active along the linear relation. Fig. 3(b) is just Fig. 3(a) slided by four rounds.

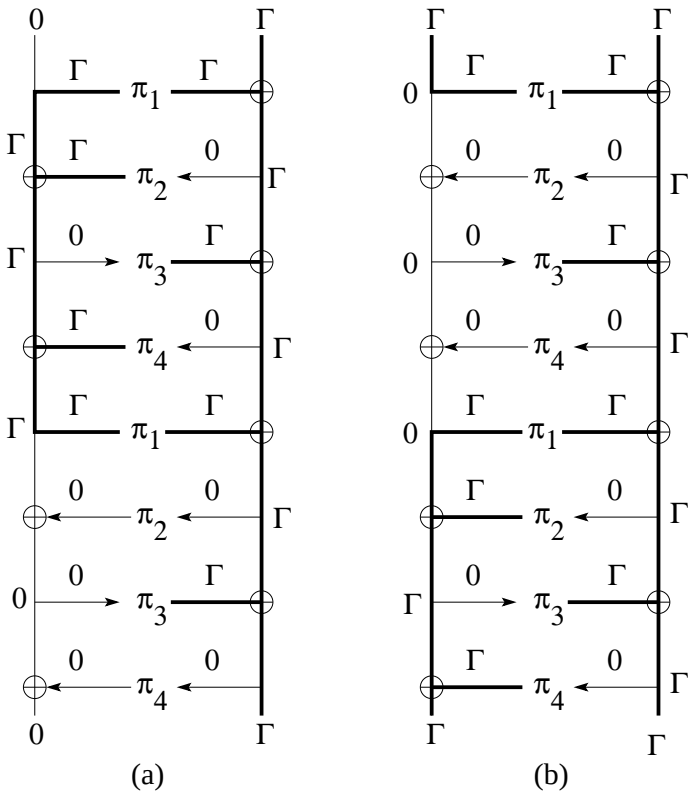

Fig. 3. Eight-round iterative linear trails.

We consider each round function as operating over 32-bit data (for a fixed, unknown key). Instead of deriving the bias for a given output bitmask, we have searched for promising bitmasks (with high bias), by exhaustive search over the inputs for each round function. From the linear relations listed above, the one in Fig. 2(a) is the most promising, since it involves only two active round functions for every four rounds: π_2 and π_4. This is an indication that the left half of the MULTI2 encryption/decryption framework is weaker than the right half.

Our search (over random keys) started with masks of low Hamming weight. The use of rotation suggests that masks with short repeating patterns tend to show higher biases, which our experiments confirmed: the 32-bit mask

$$\Gamma = \mathtt{AAAAAAAA}_{\mathtt{x}}$$

presented the highest bias: 2^{-1} for π_2, and $2^{-6.7}$ for π_4 (Fig. 2(a)). The overall bias is $p' = 2^{-6.7}$, using the piling-up lemma [10]. This bitmask was independently discovered by Aoki and Kurokawa in [1].

Comparatively, for Fig. 2(b), the best mask we have found is $\Gamma = \mathtt{88888888}_{\mathtt{x}}$, with bias 2^{-1} for π_2, and $2^{-8.6}$ for π_3. The overall bias for four rounds is $2^{-8.6}$.

Using Fig. 2(a), one can distinguish 4-round MULTI2 from a random permutation using $8 \times (p')^{-2} = 2^{16.4}$ known plaintexts (KP), for a high success rate attack; the memory complexity is negligible and the attack effort is essentially $2^{16.4}$ parity computations. For eight rounds, the attack complexity is $8 \times (2 \times (p')^2)^{-2} = 8 \times (2^{-12.4})^{-2} = 2^{27.8}$ KP and equivalent parity computations; for twelve rounds, the data complexity becomes $8 \times (2^2 \times (p')^3)^{-2} = 8 \times (2^{-18.1})^{-2} = 2^{39.2}$ KP; for sixteen rounds, $8 \times (2^{-23.8})^{-2} = 2^{50.6}$ KP. For further rounds, more plaintexts than the codebook are required.

For key-recovery attacks on twenty rounds, we use the 12-round linear relation described in the previous paragraph. Notice that across 20-round MULTI2, the same sequence of four subkeys, $k_1, \ldots, k_4$ repeats at the first and at the last four rounds. Thus, we place the 12-round relation in the middle of 20-round MULTI2 and guess consecutively k_1, k_2 (cost 2^{32} for each of them), then k_3 (cost $2^{30.7}$), and finally k_4 (free). Time complexity is thus about $2^{94.7} + 2^{94.7} = 2^{95.7}$ 4-round decryptions, that is, $1/5 \cdot 2^{95.7} \approx 2^{93.4}$ 20-round encryptions. Storage of $2^{39.2}$ ciphertexts is necessary.

6 Related-Key Slide Attack

We present key-recovery known-plaintext related-key slide attacks [3–5]. These attacks exploit the partial similarity of 4-round sequences, and works for any version of MULTI2 whose number of rounds is a multiple of eight.

Let $F_{1\ldots4}$ stand for 4-round encryption involving $\pi_1, \ldots, \pi_4$ with subkeys $k_1, \ldots, k_4$. Similarly, let $F_{5\ldots8}$ stand for 4-round encryption involving $\pi_1, \ldots, \pi_4$ with subkeys $k_5, \ldots, k_8$; $F'_{1\ldots4}$ stand for $\pi_1, \ldots, \pi_4$ with subkeys $k'_1, \ldots, k'_4$, and $F'_{5\ldots8}$ stand for $\pi_1, \ldots, \pi_4$ with subkeys $k'_5, \ldots, k'_8$.

Given an unknown key pair (s, d), we consider a related-key pair (s', d') that gives k' such that

$$\begin{array}{cccc} k'_1 = k_5 & k'_2 = k_6 & k'_3 = k_7 & k'_4 = k_8 \\ k'_5 = k_1 & k'_6 = k_2 & k'_7 = k_3 & k'_8 = k_4 \end{array} \tag{4}$$

Thus, $F'_{1\ldots4} \equiv F_{5\ldots8}$ and $F'_{5\ldots8} \equiv F_{1\ldots4}$.

For Eq. (4) to hold, it is necessary that the related key (s', d') satisfies

$$\begin{array}{cccc} d'_1 = k_3 & d'_1 \oplus d'_2 = k_4 & & \\ s'_1 = s_5 & s'_2 = s_6 & s'_3 = s_7 & s'_4 = s_8 \\ s'_5 = s_1 & s'_6 = s_2 & s'_7 = s_3 & s'_8 = s_4. \end{array}$$

The conditions $k'_1 = k_5$ and $k'_2 = k_6$ require

$$\begin{aligned} k_3 \oplus \pi_2(k_4, s_5) &= d'_1 \oplus \pi_2(d'_1 \oplus d'_2, s'_1) \\ k_4 \oplus \pi_3(k_5, s_6, s_7) &= d'_1 \oplus d'_2 \oplus \pi_3(k'_1, s'_2, s'_3). \end{aligned}$$

A slid pair gives $P' = F_{1\ldots4}(P)$, which implies $C' = F'_{5\ldots8}(C) = F_{1\ldots4}(C)$, as shown below.

$$\begin{array}{ccccccc} P & \overset{F_{1\ldots4}}{\rightarrow} & X & \overset{F_{5\ldots8}}{\rightarrow} & \cdots & \overset{F_{5\ldots8}}{\rightarrow} & C \\ P' & \overset{F'_{1\ldots4}}{\rightarrow} & \cdots & \overset{F'_{1\ldots4}}{\rightarrow} & Y & \overset{F'_{5\ldots8}}{\rightarrow} & C' \end{array}$$

That is, one get two 64-bit conditions since both the plaintext and ciphertext slid pairs are keyed by the same subkeys. Thus one slid pair is sufficient to identify $k_1, \ldots, k_4$. The attack goes as follows:

1. collect 2^{32} distinct (P_i, C_i) pairs, $i = 1, \ldots, 2^{32}$ encrypted with k
2. collect 2^{32} distinct (P'_i, C'_i) pairs, $i = 1, \ldots, 2^{32}$ encrypted with k'
3. for each $(i, j) \in \{1, \ldots, 2^{32}\}^2$
4. find the value of $k_1, \ldots, k_4$ that satisfy $P'_j = F_{1\ldots4}(P_i)$ and $C'_j = F_{1\ldots4}(C_i)$
5. search exhaustively $k_5, \ldots, k_8$ (there are 2^{96} choices, exploiting the non-surjectivity of π_2 and π_4)

We cannot filter the wrong slid pairs, so we try all possible 2^{64} pairs (P_i, P_j). But each potential slid pairs provides 128-bit condition, because both the plaintext and ciphertext pairs are keyed by the same unknown subkeys. Thus, we can filter the wrong subkeys at once.

To recover $k_1, \ldots, k_4$ we use the potential slid pair (P'_j, P_i). Guess k_1 (2^{32} choices). Then, guess k_2 (2^{32} choices), and find the k_3 that yields the (known) output of π_3. Deduce k_4, as $k_2 \oplus k_3$ and finally, test whether the current choice of $k_1, \ldots, k_4$ is consistent with the second potential slid pair (C'_j, C_i).

Finding k_3 from k_2, the input of π_3, and its output one has to solve an equation of the form $(x \lll 16) \oplus (x \vee L) = b$, then an equation $(t \lll 1) - t = x$, where x and t are the unknowns. The first can be solved bit per bit, by iteratively storing the solutions for each pair (x_i, x_{i+16}). There are 16 such pairs, and for

each pair there are at most two solutions. Hence in the worst case there will be 2^{16} solutions. The effort up to this point is roughly $2^{32+32} = 2^{64}$ π_2 and π_3-computations.

In total, up to this point, there are $2^{32+32+16} = 2^{80}$ possible values for (k_1, k_2, k_3, k_4). The value of $k_4 = k_2 \oplus k_3$ can be further checked using the (P_i, P'_j) pair. Let $P_i = (P_L, P_R)$ and $P'_j = (P'_L, P'_R)$. Then, $P'_L \oplus P_L \oplus \pi_2(P_R \oplus P_L, k_1) = \pi_4(P'_R, k_4)$, which is a condition on 26.7 bits, since the output of π_2 and π_4 intersect in $2^{26.7}$ distinct values. Thus, we expect only $2^{80}/2^{26.7} = 2^{53.3}$ tuples (k_1, k_2, k_3, k_4) to survive. Using (C_i, C'_j), a 64-bit condition, reduces the number of wrong key candidates to $2^{53.3}/2^{64} < 1$.

The final attack complexity is thus about $2^{64} \times 2^{64}$ 1-round computations, or $\left(2^{128}/r\right)$ r-round computations to recover $k_1, \ldots, k_4$. Further, to recover k_5, ..., k_8, we run a similar procedure, but over $r - 8$ rounds (after decrypting the top and bottom four rounds). The complexity is 2^{96} $(r - 8)$-round computations. Normalizing the complexity figures, the overall attack complexity is dominated by $\left(2^{128}/r\right)$ r-round computations. The memory complexity is 2^{33} plaintext/ciphertext pairs.

7 Conclusions

We showed that the 320-bit key of MULTI2 can be recovered in about 2^{185} trials instead of 2^{320} ideally, for any number of rounds, and using only three plaintext/ciphertext pairs. This weakness is due to the loss of entropy induced by the key schedule and the non-surjective round functions. We also described a linear (key-recovery) attack on up to 20 rounds, and a related-key slide attack in $\left(2^{128}/r\right)$ r-round computations for any number r of rounds that is a multiple of eight (thus including the recommended 32 rounds).

Although our results do not represent any practical threat when the 32-round recommendation is followed, they show that the security of MULTI2 is not as high as expected, and raise concerns on its long-term reliability. A practical break of MULTI2 would have dramatic consequences: millions of receivers would have to be replaced, a new technology and new standards would have to be designed and implemented.

Finally, note that the Common Scrambling Algorithm (CSA), used in Europe through the digital-TV standard DVB[4] also underwent some (non-practical) attacks [15, 17]. For comparison, the American standard ATSC uses Triple-DES in CBC mode[5].

Acknowledgments

We wish to thank Kazumaro Aoki for communicating us a copy of his article, and also Tim Sharpe (NTT communication) and Jack Laudo for their assistance.

[4] See `http://www.dvb.org/`.

[5] See `http://www.atsc.org/standards/a_70a_with_amend_1.pdf`.

We are also grateful to the reviewers of FSE 2009 for their valuable comments, and for pointing out reference [9].

References

1. Kazumaro Aoki and Kazuhiro Kurokawa. A study on linear cryptanalysis of Multi2 (in Japanese). In *The 1995 Symposium on Cryptography and Information Security, SCIS95*, 1995.
2. ARIB. *STD B25 v. 5.0*, 2007. `http://www.arib.or.jp/`.
3. Eli Biham. New types of cryptanalytic attacks using related keys. *Journal of Cryptology*, 7(4):229–246, 1994.
4. Alex Biryukov and David Wagner. Slide attacks. In Lars R. Knudsen, editor, *FSE*, volume 1636 of *LNCS*, pages 245–259. Springer, 1999.
5. Alex Biryukov and David Wagner. Advanced slide attacks. In Bart Preneel, editor, *EUROCRYPT*, volume 1807 of *LNCS*, pages 589–606. Springer, 2000.
6. BS Conditional Access Systems Co., Ltd. `http://www.b-cas.co.jp/`.
7. Hitachi. Japanese laid-open patent application no. H1-276189, 1998.
8. ISO. Algorithm registry entry 9979/0009, 1994.
9. Tomoari Katagi, Takaya Inoue, Takeshi Shimoyama, and Shigeo Tsujii. A correlation attack on block ciphers with arithmetic operations (in Japanese). In *SCIS*, 2003. reference no. SCIS2003 5D-2.
10. Mitsuru Matsui. Linear cryptoanalysis method for DES cipher. In Tor Helleseth, editor, *EUROCRYPT*, volume 765 of *LNCS*, pages 386–397. Springer, 1993.
11. Mitsuru Matsui and Atsushiro Yamagishi. On a statistical attack of secret key cryptosystems. *Electronics and Communications in Japan, Part III: Fundamental Electronic Science (English translation of Denshi Tsushin Gakkai Ronbunshi)*, 77(9):61–72, 1994.
12. Alfred J. Menezes, Paul C. van Oorschot, and Scot A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.
13. Kazuo Takaragi, Fusao Nakagawa, and Ryoichi Sasaki. U.S. patent no. 4982429, 1989.
14. Kazuo Takaragi, Fusao Nakagawa, and Ryoichi Sasaki. U.S. patent no. 5103479, 1990.
15. Ralf-Phillip Weinmann and Kai Wirt. Analysis of the DVB common scrambling algorithm. In *8th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS)*. Springer, 2004.
16. Wikipedia. Mobaho! Accessed 05-February-2009.
17. Kai Wirt. Fault attack on the DVB common scrambling algorithm. In Osvaldo Gervasi et al., editor, *ICCSA (2)*, volume 3481 of *LNCS*, pages 577–584. Springer, 2005.
18. Toshiro Yoshimura. Conditional access system for digital broadcasting in Japan. *Proceedings of the IEEE*, 94(1):318–322, 2006.