Wang's sufficient conditions of MD5 are not sufficient

Jun Yajima and Takeshi Shimoyama

FUJITSU LABORATORIES LTD.
{jyajima,shimo}@labs.fujitsu.com

Abstract

In this paper, we report that the "sufficient conditions" of MD5 [1] of the modification technique for the collision search algorithm described by Wang [2] are not sufficient. In our analysis, we show at least 4 extra-conditions for the message modification in the first block and corrections of the several conditions which are correspond to the highest (32nd) bit of the sufficient conditions in the second block should be needed. And we show the new collision message which is completely different from the message pairs showed in [2] [3] by using our extended sufficient conditions.

1 Introduction

Hash functions are one of the primitive functions for the digital signature to generate the digest value from arbitrary length message data. In order to authenticate the digital data, the verifier confirms the received data by comparing the hash value. Then, the collision resistance is one of the necessary property for every secure hash functions. Then, it will be important question for the security of hash function that a collision pair can be generated easily or not.

MD5 is a hash function developed by Rivest in 1992 and is selected in RFC1321 [1]. MD5 has been used in many applications for digital signature. The input of the compression function of MD5 is 512-bit, and the digest value is 128-bit.

In 2004, Wang et al. presented the concrete message pair with 2 message blocks which occurs the collision of MD5 [2]. They used "message modification technique", that is, they controlled message blocks for satisfying the "sufficient conditions" of the output bits of every steps in the compression functions, and they succeeded to derive the collision message pairs. After Wang's work, Klima proposed the improvement algorithm [3]. He also showed different collision messages. It seems, however, his search were based on the message pairs found by Wang. Actually, in the first message block the message words from 6th to 15th in his collision message pairs are as same as Wang's collision message pairs.

We tried to trace their collision search algorithm in order to derive the new collision messages. In our experiment, however, we could not find new collision messages except for data close to one showed by Wang or Klima. Then, by executing careful analysis, we found some missing conditions in the "sufficient conditions" described in [2].

In this paper, we show the additional "sufficient conditions" which should be required for the message modificatin from the given any random seed. And we show the new collision

message by using our modified sufficient conditions, which is completely different from the message pairs showed in [2] [3].

2 Collision search algorithm by Wang's et al.

Wang et al. showed an algorithm to find the collision message pair of MD5 [2]. In their method, they search some message pairs which output arithmetic differentials of each steps become the desired value. To do this, they used the following search algorithms.

Procedure 1. For all steps in the first round (1-16 step) and the first 3 or 4 steps in the second round (17-19 or 17-20 step), the message words are modified in order to satisfy the bitwise equations, so called "sufficient condition", related to input words (a_i, b_i, c_i, d_i) and an output word in each steps.

Procedure 2. For the remaining steps, the output is checked whether the equations for the differentials of each step are satisfied or not. If one of the equations does not hold, the procedure go back to the last part of the Procedure 1.

3 Missing conditions

After the execution of the Procedure 1 in the previous section, the output of each steps must satisfy the condition of the differentials described in the Table 3 and Table 5 in [2] with high probability. In our experiment, however, we came across that the differential condition of the output of some steps were not hold, although all sufficient conditions were satisfied, especially the 7th step of the first message block, and the 3rd step of the second message block. (See Example 3 and Example 4 in Appendix 1.) Then, we conjectured that it means some essential conditions were not listed in the Table 4 and 6 in [2]. Then, we executed the careful analysis at the steps which was not satisfied the differential condition, and we found the necessary extra-conditions, as follows.

(i) Missing condition $a_{2,31} = 0$, $a_{2,30} = 0$, $a_{2,29} = 0$, $a_{2,27} = 0$
In the single-message modification procedure based on Wang's approach for the first block, the procedure success with extremely low probability at the 7th step (the step generating c_2). Then, we conjectured that extra-conditions are required which are related to the input bits of the 7th step. In order to seek the missing condition, we execute the following test at the 7th step.

1. Set the 7th step input value for satisfying the sufficient conditions.
2. Set the random bit value at the 50 bits which does not appear the list of the sufficient conditions.
3. Examine the single modification procedure and check that the output difference come to the expected value or not, exhaustively.
4. Extract the common condition of the input value for all succeeded data.

Then we found that the following 9 conditions were satisfied in all success data in the 7th step.

$$a_{2,31} = 0, a_{2,30} = 0, a_{2,29} = 0, a_{2,27} = 0,$$
$$d_{2,31} = 0, d_{2,30} = 0, d_{2,29} = 0, d_{2,27} = 0,$$
$$b_{1,6} = 1.$$

Moreover, we found that the following 5 conditions were not necessary $d_{2,31} = 0, d_{2,30} =$

$0, d_{2,29} = 0, d_{2,27} = 0, b_{1,6} = 1$. On the other hand, if we omit one of the conditions $a_{2,31} = 0, a_{2,30} = 0, a_{2,29} = 0, a_{2,27} = 0$, the success rate of the modification become extremely low. Then, we conclude that the following 4 conditions should be needed.

$$a_{2,31} = 0, a_{2,30} = 0, a_{2,29} = 0, a_{2,27} = 0$$

(ii) Correction of the sufficient conditions in the 2nd message block

At the 3rd step in the 2nd message block (generating c_1 word), the output difference of the f function must be equal to 2^{25}. Then the bitwise difference at 32nd bit of the function f ($f_{3,32}$) should be equal to zero. Therefore the condition $a_{1,32} \neq bb_{0,32}$ must be held. On the other hand, the condition $a_{1,32} = 1$ is listed on the Table 6 in [2], therefore the input words of the 2nd block should be satisfied $bb_{0,32} = cc_{0,32} = dd_{0,32} = 0$. In the sufficient conditions, however, this kind of condition has not not appeared. Then it is possible that although the all of the sufficient conditions and the difference condition in [2] were satisfied in the first message block, "the single-message modification" of the 2nd message block never succeed. Since the collision searches of the first message block and the second message block can be executed independently, from the wrong first message block satisfying all of the sufficient condition but satisfied $bb_{0,32} = 1$, they will never success to find message to hold the differential conditions. (See Example 4.) Therefore, if the collision searches are executed by using the sufficient condition of the 2nd block listed in Table 6 [2] by Wang, the additional condition $bb_{0,32} = 0$ should be needed. This additional condition, however, makes the complexity twice as large as the original one for the message search of the first block. On the other hand, by the careful analysis, we found that the condition $a_{1,32} = 1$ is unnecessary. In order to remove this condition, the sufficient conditions are modified as described in Appendix 2.

4 Collision Search

By executing the collision search algorithm by using the sufficient conditions with our new conditions, we succeeded to find the several original collision message pairs. The collision message pairs shows in Example1 and Example2 in Appendix1.

By our computer experiments through the several days, we found 1319 pairs of the first message blocks which hold the differential properties in each round written in Table 3 of [2]. There are 8 conditions which are related to the first word aa_0, bb_0, cc_0, dd_0 of the second block in Table 4 of [2], and we found 6 message block satisfying 8 conditions. ($1319/2^8 = 5.15$.) In those 6 messages, 3 messages satisfied the condition $bb_{0,32} = 0$, we found the collision pairs of MD5 from those messages. (See Example1.) From the remaining 3 messages satisfied the condition $bb_{0,32} = 1$, we would never find the collision message by using Wang's sufficient condition, however, we succeeded to find the collision pairs of MD5 by using our new sufficient conditions. (See Example2.)

On the average, it took several hours by using PC (Pentium4 3.8GHz) for finding one collision message of MD5.

5 Conclusion

In this paper, we reported that the "sufficient conditions" of MD5 of the modification technique for collision search described by Wang are not sufficient. In our analysis, at least 4 extra-conditions in the 1st block and correction of the sufficient condition in 2nd block should be needed for the message modification. We could trace the Wang's collision search

algorithm and could derive the new collision messages of MD5.

References

[1] R.L.Rivest, "The-MD5 Message-Digest Algorithm", RFC1321, April 1992.

[2] Xiaoyun Wang and Hongbo Yu, "How to Break MD5 and Other Hash Functions", published on the web.
`http://www.infosec.sdu.edu.cn/paper/md5-attack.pdf`

[3] Vlastimil Klima, "Finding MD5 Collisions on a Notebook PC Using Multi-message Modifications", Cryptology ePrint Archive, April 2005.
`http://eprint.iacr.org/2005/102.pdf`

Appendix1 : Examples

Example1(Collision message pairs ($bb_{0,32} = 0$[in 1st block]))

Stream data format

message(input stream for MD5)	message'(input stream for MD5)
8AF3471E D92DB46B A73C6CF2 384C3925	8AF3471E D92DB46B A73C6CF2 384C3925
E14FEA6E 3447B53D 5B322F79 A268209C	E14FEAEE 3447B53D 5B322F79 A268209C
31ED3206 F8A93424 BBFA6676 82270114	31ED3206 F8A93424 BBFA6676 82A70114
E3088992 0E7DAF60 D1097B79 1838BC7C	E3088992 0E7DAF60 D1097BF9 1838BC7C
ADB087C4 FEC56E1B 09F0C324 DD949A39	ADB087C4 FEC56E1B 09F0C324 DD949A39
D5754A47 FE74344C 2465A855 5088F9DD	D5754AC7 FE74344C 2465A855 5088F9DD
51A07A6B 4CD9DA87 36BB1758 AE68DDAE	51A07A6B 4CD9DA87 36BB1758 AEE8DCAE
D00C8EAA 72EA1B6C A7C34F6B 8D7748FC	D00C8EAA 72EA1B6C A7C34FEB 8D7748FC
digest(output stream from MD5)	**digest'(output stream from MD5)**
BF5AFFA4 DF8F186D 3FA7D511 5A3AE28E	BF5AFFA4 DF8F186D 3FA7D511 5A3AE28E

Internal word data format

message word for block1	message' word for block1
0x1E47F38A, 0x6BB42DD9,	0x1E47F38A, 0x6BB42DD9,
0xF26C3CA7, 0x25394C38,	0xF26C3CA7, 0x25394C38,
0x6EEA4FE1, 0x3DB54734,	0xEEEA4FE1, 0x3DB54734,
0x792F325B, 0x9C2068A2,	0x792F325B, 0x9C2068A2,
0x0632ED31, 0x2434A9F8,	0x0632ED31, 0x2434A9F8,
0x7666FABB, 0x14012782,	0x7666FABB, 0x1401A782,
0x928908E3, 0x60AF7D0E,	0x928908E3, 0x60AF7D0E,
0x797B09D1, 0x7CBC3818	0xF97B09D1, 0x7CBC3818
context state after block1	**context state after block1**
aa0 = 0xE9DD5341, dd0 = 0x4123AA64,	aa0' = 0x69DD5341, dd0' = 0xC323AA64,
cc0 = 0x5B8E4409, bb0 = 0x0946EB5C	cc0' = 0xDD8E4409, bb0' = 0x8B46EB5C

message word for block2	message' word for block2
0xC487B0AD, 0x1B6EC5FE,	0xC487B0AD, 0x1B6EC5FE,
0x24C3F009, 0x399A94DD,	0x24C3F009, 0x399A94DD,
0x474A75D5, 0x4C3474FE,	0xC74A75D5, 0x4C3474FE,
0x55A86524, 0xDDF98850,	0x55A86524, 0xDDF98850,
0x6B7AA051, 0x87DAD94C,	0x6B7AA051, 0x87DAD94C,
0x5817BB36, 0xAEDD68AE,	0x5817BB36, 0xAEDCE8AE,
0xAA8E0CD0, 0x6C1BEA72,	0xAA8E0CD0, 0x6C1BEA72,
0x6B4FC3A7, 0xFC48778D	0xEB4FC3A7, 0xFC48778D
context state after block2	**context state after block2**
aa0 = 0xED1AE2D5, dd0 = 0x53CE6144,	aa0' = 0xED1AE2D5, dd0' = 0x53CE6144,
cc0 = 0xAC7788A2, bb0 = 0x06E78521	cc0' = 0xAC7788A2, bb0' = 0x06E78521

(aa0=aa0', dd0=dd0', cc0=cc0', bb0=bb0')

Example2(Collision message pairs ($bb_{0,32} = 1$[in 1st block]))

Stream data format

message(input stream for MD5)	message'(input stream for MD5)
B484742B CC8C3006 A147ACA0 C906B20F	B484742B CC8C3006 A147ACA0 C906B20F
CBA8E79B 21089D5C D8AEA238 30E70F06	CBA8E71B 21089D5C D8AEA238 30E70F06
41AD3305 FC2714A3 37B18681 626B12D7	41AD3305 FC2714A3 37B18681 62EB12D7
DD2CF625 B863392F 852DDF25 9BFD9650	DD2CF625 B863392F 852DDFA5 9BFD9650
93DF6C6A 819A3D1A 0121C1F3 70E16A74	93DF6C6A 819A3D1A 0121C1F3 70E16A74
9471A6A5 922744ED 24959DC6 4CF8FDF0	9471A625 922744ED 24959DC6 4CF8FDF0
6D1734A3 535B9F69 BAD54524 6F702956	6D1734A3 535B9F69 BAD54524 6FF02856
E42475A1 A51B44E3 8ADF50CD 2F17158D	E42475A1 A51B44E3 8ADF504D 2F17158D
digest(output stream from MD5)	**digest'(output stream from MD5)**
B9B32875 F636F1A2 3DA41833 1E7B87A7	B9B32875 F636F1A2 3DA41833 1E7B87A7

Internal word data format

message word for block1	message' word for block1
0x2B7484B4, 0x06308CCC,	0x2B7484B4, 0x06308CCC,
0xA0AC47A1, 0x0FB206C9,	0xA0AC47A1, 0x0FB206C9,
0x9BE7A8CB, 0x5C9D0821,	0x1BE7A8CB, 0x5C9D0821,
0x38A2AED8, 0x060FE730,	0x38A2AED8, 0x060FE730,
0x0533AD41, 0xA31427FC,	0x0533AD41, 0xA31427FC,
0x8186B137, 0xD7126B62,	0x8186B137, 0xD712EB62,
0x25F62CDD, 0x2F3963B8,	0x25F62CDD, 0x2F3963B8,
0x25DF2D85, 0x5096FD9B	0xA5DF2D85, 0x5096FD9B
context state after block1	**context state after block1**
aa0 = 0xC9D4C403, dd0 = 0xB8F90DA1,	aa0' = 0x49D4C403, dd0' = 0x3AF90DA1,
cc0 = 0xF3027284, bb0 = 0xF1C68C8A	cc0' = 0x75027284, bb0' = 0x73C68C8A

message word for block2	message' word for block2
0x6A6CDF93, 0x1A3D9A81,	0x6A6CDF93, 0x1A3D9A81,
0xF3C12101, 0x746AE170,	0xF3C12101, 0x746AE170,
0xA5A67194, 0xED442792,	0x25A67194, 0xED442792,
0xC69D9524, 0xF0FDF84C,	0xC69D9524, 0xF0FDF84C,
0xA334176D, 0x699F5B53,	0xA334176D, 0x699F5B53,
0x2445D5BA, 0x5629706F,	0x2445D5BA, 0x5628F06F,
0xA17524E4, 0xE3441BA5,	0xA17524E4, 0xE3441BA5,
0xCD50DF8A, 0x8D15172F	0x4D50DF8A, 0x8D15172F
context state after block2	**context state after block2**
aa0 = 0xCD75D59D, dd0 = 0x1F23D8E4,	aa0' = 0xCD75D59D, dd0' = 0x1F23D8E4,
cc0 = 0x7230864E, bb0 = 0xBEDAF3B9	cc0' = 0x7230864E, bb0' = 0xBEDAF3B9

(aa0=aa0', dd0=dd0', cc0=cc0', bb0=bb0')

Example3 (the differential condition of the 7th step output of block1 were not hold.($a_{2,27} = 1$, $a_{2,30} = 1$ [in 1st block]))

Internal word data format

message word for block1	message' word for block1
0x5d35049e, 0x2b3c9edc, 0x63ea6960, 0x32a1aa1a, 0x83ba6cfb, 0xabd27635	0x5d35049e, 0x2b3c9edc, 0x63ea6960, 0x32a1aa1a, 0x03ba6cfb, 0xabd27635
input data for 7th step	**input data for 7th step**
c1=0x1c72d438, b1=0xc77adc2d a2=0xad400027, d2=0x277fbc43	c1'=0x1c72d438, b1'=0xc77adc2d a2'=0xad3fffe7, d2'=0xa7ffbc03

Example4 (the differential condition of the 3rd step output of block2 were not hold.($a_{1,32} = bb_{0,32}$ [in 2nd block]))

Internal word data format

message word for block1	message' word for block1
0x2B7484B4, 0x06308CCC, 0xA0AC47A1, 0x0FB206C9, 0x9BE7A8CB, 0x5C9D0821, 0x38A2AED8, 0x060FE730, 0x0533AD41, 0xA31427FC, 0x8186B137, 0xD7126B62, 0x25F62CDD, 0x2F3963B8, 0x25DF2D85, 0x5096FD9B	0x2B7484B4, 0x06308CCC, 0xA0AC47A1, 0x0FB206C9, 0x1BE7A8CB, 0x5C9D0821, 0x38A2AED8, 0x060FE730, 0x0533AD41, 0xA31427FC, 0x8186B137, 0xD712EB62, 0x25F62CDD, 0x2F3963B8, 0xA5DF2D85, 0x5096FD9B
context state after block1	**context state after block1**
aa0 = 0xC9D4C403, dd0 = 0xB8F90DA1, cc0 = 0xF3027284, bb0 = 0xF1C68C8A	aa0' = 0x49D4C403, dd0' = 0x3AF90DA1, cc0' = 0x75027284, bb0' = 0x73C68C8A

message word for block2	message' word for block2
0x590d4d36, 0xec1d7483	0x590d4d36, 0xec1d7483
input data for 3rd step	**input data for 3rd step**
cc0=0xf3027284, bb0=0xf1c68c8a a1=0xb5a23603, d1=0xbd823e19	cc0'=0x75027284, bb0'=0x73c68c8a a1'=0x37a23603, d1'=0x3f823e39

Appendix2 : List of sufficient sonditions (corrected)

First Message Block

c_1	$c_{1,7}=0$, $c_{1,12}=0$, $c_{1,20}=0$
b_1	$b_{1,7}=0$, $b_{1,8}=c_{1,8}$, $b_{1,9}=c_{1,9}$, $b_{1,10}=c_{1,10}$, $b_{1,11}=c_{1,11}$, $b_{1,12}=1$, $b_{1,13}=c_{1,13}$, $b_{1,14}=c_{1,14}$, $b_{1,15}=c_{1,15}$, $b_{1,16}=c_{1,16}$, $b_{1,17}=c_{1,17}$, $b_{1,18}=c_{1,18}$, $b_{1,19}=c_{1,19}$, $b_{1,20}=1$, $b_{1,21}=c_{1,21}$, $b_{1,22}=c_{1,22}$, $b_{1,23}=c_{1,23}$, $b_{1,24}=0$, $b_{1,32}=1$
a_2	$a_{2,1}=1$, $a_{2,3}=1$, $a_{2,6}=1$, $a_{2,7}=0$, $a_{2,8}=0$, $a_{2,9}=0$, $a_{2,10}=0$, $a_{2,11}=0$, $a_{2,12}=0$, $a_{2,13}=0$, $a_{2,14}=0$, $a_{2,15}=0$, $a_{2,16}=0$, $a_{2,17}=0$, $a_{2,18}=0$, $a_{2,19}=0$, $a_{2,20}=0$, $a_{2,21}=0$, $a_{2,22}=0$, $a_{2,23}=1$, $a_{2,24}=0$, $a_{2,26}=0$, $\boxed{a_{2,27}=0}$, $a_{2,28}=1$, $\boxed{a_{2,29}=0}$, $\boxed{a_{2,30}=0}$, $\boxed{a_{2,31}=0}$, $a_{2,32}=1$
d_2	$d_{2,1}=1$, $d_{2,2}=a_{2,2}$, $d_{2,3}=0$, $d_{2,4}=a_{2,4}$, $d_{2,5}=a_{2,5}$, $d_{2,6}=0$, $d_{2,7}=1$, $d_{2,8}=0$, $d_{2,9}=0$, $d_{2,10}=0$, $d_{2,11}=1$, $d_{2,12}=1$, $d_{2,13}=1$, $d_{2,14}=1$, $d_{2,15}=0$, $d_{2,16}=1$, $d_{2,17}=1$, $d_{2,18}=1$, $d_{2,19}=1$, $d_{2,20}=1$, $d_{2,21}=1$, $d_{2,22}=1$, $d_{2,23}=1$, $d_{2,24}=0$, $d_{2,25}=a_{2,25}$, $d_{2,26}=1$, $d_{2,27}=a_{2,27}$, $d_{2,28}=0$, $d_{2,29}=a_{2,29}$, $d_{2,30}=a_{2,30}$, $d_{2,31}=a_{2,31}$, $d_{2,32}=0$
c_2	$c_{2,1}=0$, $c_{2,2}=0$, $c_{2,3}=0$, $c_{2,4}=0$, $c_{2,5}=0$, $c_{2,6}=1$, $c_{2,7}=0$, $c_{2,8}=0$, $c_{2,9}=0$, $c_{2,10}=0$, $c_{2,11}=0$, $c_{2,12}=1$, $c_{2,13}=1$, $c_{2,14}=1$, $c_{2,15}=1$, $c_{2,16}=1$, $c_{2,17}=0$, $c_{2,18}=1$, $c_{2,19}=1$, $c_{2,20}=1$, $c_{2,21}=1$, $c_{2,22}=1$, $c_{2,23}=1$, $c_{2,24}=1$, $c_{2,25}=1$, $c_{2,26}=1$, $c_{2,27}=0$, $c_{2,28}=0$, $c_{2,29}=0$, $c_{2,30}=0$, $c_{2,31}=0$, $c_{2,32}=0$
b_2	$b_{2,1}=0$, $b_{2,2}=0$, $b_{2,3}=0$, $b_{2,4}=0$, $b_{2,5}=0$, $b_{2,6}=0$, $b_{2,7}=1$, $b_{2,8}=0$, $b_{2,9}=1$, $b_{2,10}=0$, $b_{2,11}=1$, $b_{2,12}=0$, $b_{2,14}=0$, $b_{2,16}=0$, $b_{2,17}=1$, $b_{2,18}=0$, $b_{2,19}=0$, $b_{2,20}=0$, $b_{2,21}=1$, $b_{2,24}=1$, $b_{2,25}=1$, $b_{2,26}=0$, $b_{2,27}=0$, $b_{2,28}=0$, $b_{2,29}=0$, $b_{2,30}=0$, $b_{2,31}=0$, $b_{2,32}=0$
a_3	$a_{3,1}=1$, $a_{3,2}=0$, $a_{3,3}=1$, $a_{3,4}=1$, $a_{3,5}=1$, $a_{3,6}=1$, $a_{3,7}=0$, $a_{3,8}=0$, $a_{3,9}=1$, $a_{3,10}=1$, $a_{3,11}=1$, $a_{3,12}=1$, $a_{3,13}=b_{2,13}$, $a_{3,14}=1$, $a_{3,16}=0$, $a_{3,17}=0$, $a_{3,18}=0$, $a_{3,19}=0$, $a_{3,20}=0$, $a_{3,21}=1$, $a_{3,25}=1$, $a_{3,26}=1$, $a_{3,27}=0$, $a_{3,28}=1$, $a_{3,29}=1$, $a_{3,30}=1$, $a_{3,31}=1$, $a_{3,32}=1$
d_3	$d_{3,1}=0$, $d_{3,2}=0$, $d_{3,7}=1$, $d_{3,8}=0$, $d_{3,9}=0$, $d_{3,13}=1$, $d_{3,14}=0$, $d_{3,16}=1$, $d_{3,17}=1$, $d_{3,18}=1$, $d_{3,19}=1$, $d_{3,20}=1$, $d_{3,21}=1$, $d_{3,24}=0$, $d_{3,31}=1$, $d_{3,32}=0$
c_3	$c_{3,1}=0$, $c_{3,2}=1$, $c_{3,7}=1$, $c_{3,8}=1$, $c_{3,9}=0$, $c_{3,13}=0$, $c_{3,14}=0$, $c_{3,15}=d_{3,15}$, $c_{3,16}=1$, $c_{3,17}=1$, $c_{3,18}=0$, $c_{3,19}=0$, $c_{3,20}=0$, $c_{3,31}=0$, $c_{3,32}=0$
b_3	$b_{3,8}=0$, $b_{3,9}=1$, $b_{3,13}=1$, $b_{3,14}=0$, $b_{3,15}=0$, $b_{3,16}=0$, $b_{3,17}=0$, $b_{3,18}=0$, $b_{3,19}=0$, $b_{3,20}=1$, $b_{3,25}=c_{3,25}$, $b_{3,26}=c_{3,26}$, $b_{3,31}=0$, $b_{3,32}=0$
a_4	$a_{4,4}=1$, $a_{4,8}=0$, $a_{4,9}=0$, $a_{4,14}=1$, $a_{4,15}=1$, $a_{4,16}=1$, $a_{4,17}=1$, $a_{4,18}=1$, $a_{4,19}=1$, $a_{4,20}=1$, $a_{4,25}=1$, $a_{4,26}=0$, $a_{4,31}=1$, $a_{4,32}=0$
d_4	$d_{4,4}=1$, $d_{4,8}=1$, $d_{4,9}=1$, $d_{4,14}=1$, $d_{4,15}=1$, $d_{4,16}=1$, $d_{4,17}=1$, $d_{4,18}=1$, $d_{4,19}=0$, $d_{4,20}=1$, $d_{4,25}=0$, $d_{4,26}=0$, $d_{4,30}=0$, $d_{4,32}=0$
c_4	$c_{4,4}=0$, $c_{4,16}=1$, $c_{4,25}=1$, $c_{4,26}=0$, $c_{4,30}=1$, $c_{4,32}=0$
b_4	$b_{4,30}=1$, $b_{4,32}=0$
2R	$a_{5,4}=b_{4,4}$, $a_{5,16}=b_{4,16}$, $a_{5,18}=0$, $a_{5,32}=0$, $d_{5,18}=1$, $d_{5,30}=a_{5,30}$, $d_{5,32}=0$, $c_{5,18}=0$, $c_{5,32}=0$, $b_{5,32}=0$, $a_{6,18}=b_{5,18}$, $a_{6,32}=0$, $d_{6,32}=0$, $c_{6,32}=0$, $b_{6,32}\neq c_{6,32}$,
3R	$\phi_{34,32}=0$, $b_{12,32}=d_{12,32}$,
4R	$a_{13,32}=c_{12,32}$, $d_{13,32}\neq b_{12,32}$, $c_{13,32}=a_{13,32}$, $b_{13,32}=d_{13,32}$, $a_{14,32}=c_{13,32}$, $d_{14,32}=b_{13,32}$, $c_{14,32}=a_{14,32}$, $b_{14,32}=d_{14,32}$, $a_{15,32}=c_{14,32}$, $d_{15,32}=b_{14,32}$, $c_{15,32}=a_{15,32}$, $b_{15,26}=0$, $b_{15,32}\neq d_{15,32}$, $a_{16,26}=1$, $a_{16,27}=0$, $a_{16,32}=c_{15,32}$, $d_{16,32}=b_{15,32}$, $c_{16,32}=d_{16,32}$
OUT	$dd_{0,26}=0$, $cc_{0,26}=1$, $cc_{0,27}=0$, $cc_{0,32}=dd_{0,32}$, $bb_{0,26}=0$, $bb_{0,27}=0$, $bb_{0,6}=0$, $bb_{0,32}=cc_{0,32}$

Second Message Block

a_1	$a_{1,6}=0$, $a_{1,12}=0$, $a_{1,22}=1$, $a_{1,26}=0$, $a_{1,27}=1$, $a_{1,28}=0$, $\boxed{a_{1,32} \neq bb_{0,32}}$
d_1	$d_{1,2}=0$, $d_{1,3}=0$, $d_{1,6}=0$, $d_{1,7}=a_{1,7}$, $d_{1,8}=a_{1,8}$, $d_{1,12}=1$, $d_{1,13}=a_{1,13}$, $d_{1,16}=0$, $d_{1,17}=a_{1,17}$, $d_{1,18}=a_{1,18}$, $d_{1,19}=a_{1,19}$, $d_{1,20}=a_{1,20}$, $d_{1,21}=a_{1,21}$, $d_{1,22}=0$, $d_{1,26}=0$, $d_{1,27}=1$, $d_{1,28}=1$, $d_{1,29}=a_{1,29}$, $d_{1,30}=a_{1,30}$, $d_{1,31}=a_{1,31}$, $\boxed{d_{1,32}=a_{1,32}}$
c_1	$c_{1,2}=1$, $c_{1,3}=1$, $c_{1,4}=d_{1,4}$, $c_{1,5}=d_{1,5}$, $c_{1,6}=1$, $c_{1,7}=1$, $c_{1,8}=0$, $c_{1,9}=1$, $c_{1,12}=1$, $c_{1,13}=0$, $c_{1,17}=1$, $c_{1,18}=1$, $c_{1,19}=1$, $c_{1,20}=1$, $c_{1,21}=1$, $c_{1,22}=0$, $c_{1,26}=1$, $c_{1,27}=1$, $c_{1,28}=1$, $c_{1,29}=1$, $c_{1,30}=1$, $c_{1,31}=0$, $\boxed{c_{1,32}=d_{1,32}}$
b_1	$b_{1,1}=c_{1,1}$, $b_{1,2}=0$, $b_{1,3}=0$, $b_{1,4}=0$, $b_{1,5}=1$, $b_{1,6}=0$, $b_{1,7}=0$, $b_{1,8}=0$, $b_{1,9}=0$, $b_{1,10}=c_{1,10}$, $b_{1,11}=c_{1,11}$, $b_{1,12}=0$, $b_{1,13}=0$, $b_{1,17}=0$, $b_{1,18}=0$, $b_{1,19}=1$, $b_{1,20}=0$, $b_{1,21}=0$, $b_{1,22}=0$, $b_{1,26}=1$, $b_{1,27}=0$, $b_{1,28}=1$, $b_{1,29}=1$, $b_{1,30}=1$, $b_{1,31}=0$, $\boxed{b_{1,32}=c_{1,32}}$
a_2	$a_{2,1}=0$, $a_{2,2}=0$, $a_{2,3}=0$, $a_{2,4}=0$, $a_{2,5}=1$, $a_{2,6}=0$, $a_{2,7}=1$, $a_{2,8}=0$, $a_{2,9}=0$, $a_{2,10}=1$, $a_{2,11}=1$, $a_{2,12}=1$, $a_{2,13}=0$, $a_{2,17}=1$, $a_{2,18}=1$, $a_{2,19}=1$, $a_{2,20}=1$, $a_{2,21}=0$, $a_{2,22}=1$, $a_{2,27}=0$, $a_{2,28}=1$, $a_{2,29}=0$, $a_{2,30}=0$, $a_{2,31}=1$, $\boxed{a_{2,32} \neq b_{1,32}}$
d_2	$d_{2,1}=0$, $d_{2,2}=1$, $d_{2,3}=1$, $d_{2,4}=0$, $d_{2,5}=1$, $d_{2,6}=0$, $d_{2,7}=1$, $d_{2,8}=0$, $d_{2,9}=0$, $d_{2,10}=0$, $d_{2,11}=1$, $d_{2,12}=1$, $d_{2,13}=0$, $d_{2,17}=0$, $d_{2,18}=1$, $d_{2,21}=0$, $d_{2,22}=1$, $d_{2,26}=0$, $d_{2,27}=1$, $d_{2,28}=0$, $d_{2,29}=0$, $\boxed{d_{2,32}=a_{2,32}}$
c_2	$c_{2,1}=1$, $c_{2,7}=0$, $c_{2,8}=0$, $c_{2,9}=0$, $c_{2,10}=1$, $c_{2,11}=1$, $c_{2,12}=1$, $c_{2,13}=1$, $c_{2,16}=d_{2,16}$, $c_{2,17}=1$, $c_{2,18}=0$, $c_{2,21}=0$, $c_{2,22}=0$, $c_{2,24}=d_{2,24}$, $c_{2,25}=d_{2,25}$, $c_{2,26}=1$, $c_{2,27}=1$, $c_{2,28}=0$, $c_{2,29}=1$, $\boxed{c_{2,32} \neq d_{2,32}}$
b_2	$b_{2,1}=0$, $b_{2,2}=c_{2,2}$, $b_{2,7}=1$, $b_{2,8}=1$, $b_{2,9}=1$, $b_{2,10}=1$, $b_{2,16}=1$, $b_{2,17}=0$, $b_{2,18}=1$, $b_{2,21}=1$, $b_{2,22}=1$, $b_{2,24}=0$, $b_{2,25}=0$, $b_{2,26}=0$, $b_{2,27}=1$, $b_{2,28}=0$, $b_{2,29}=0$, $\boxed{b_{2,32}=c_{2,32}}$
a_3	$a_{3,1}=1$, $a_{3,2}=0$, $a_{3,7}=1$, $a_{3,8}=1$, $a_{3,9}=1$, $a_{3,10}=0$, $a_{3,13}=b_{2,13}$, $a_{3,16}=0$, $a_{3,17}=1$, $a_{3,18}=0$, $a_{3,24}=0$, $a_{3,25}=0$, $a_{3,26}=0$, $a_{3,27}=1$, $a_{3,28}=1$, $a_{3,29}=1$, $\boxed{a_{3,32}=b_{2,32}}$
a_3	$d_{3,1}=0$, $d_{3,2}=0$, $d_{3,7}=1$, $d_{3,8}=1$, $d_{3,9}=1$, $d_{3,10}=1$, $d_{3,13}=0$, $d_{3,16}=1$, $d_{3,17}=1$, $d_{3,18}=1$, $d_{3,19}=0$, $d_{3,24}=1$, $d_{3,25}=1$, $d_{3,26}=1$, $d_{3,27}=1$, $\boxed{d_{3,32}=a_{3,32}}$
a_3	$c_{3,1}=1$, $c_{3,2}=1$, $c_{3,7}=1$, $c_{3,8}=1$, $c_{3,9}=1$, $c_{3,10}=1$, $c_{3,13}=0$, $c_{3,14}=d_{3,14}$, $c_{3,15}=d_{3,15}$, $c_{3,16}=1$, $c_{3,17}=1$, $c_{3,18}=0$, $c_{3,19}=1$, $c_{3,20}=d_{3,20}$, $\boxed{c_{3,32}=d_{3,32}}$
a_3	$b_{3,8}=1$, $b_{3,13}=1$, $b_{3,14}=0$, $b_{3,15}=0$, $b_{3,16}=0$, $b_{3,17}=0$, $b_{3,18}=0$, $b_{3,19}=0$, $b_{3,20}=1$, $b_{3,25}=c_{3,25}$, $b_{3,26}=c_{3,26}$, $b_{3,27}=c_{3,27}$, $b_{3,28}=c_{3,28}$, $b_{3,29}=c_{3,29}$, $b_{3,30}=c_{3,30}$, $b_{3,31}=c_{3,31}$ $\boxed{b_{3,32}=c_{3,32}}$
a_4	$a_{4,4}=1$, $a_{4,8}=0$, $a_{4,14}=1$, $a_{4,15}=1$, $a_{4,16}=1$, $a_{4,17}=1$, $a_{4,18}=1$, $a_{4,19}=1$ $a_{4,20}=1$, $a_{4,25}=1$, $a_{4,26}=1$, $a_{4,27}=1$, $a_{4,28}=1$, $a_{4,29}=1$, $a_{4,30}=1$, $a_{4,31}=0$, $\boxed{a_{4,32} \neq b_{3,32}}$
d_4	$d_{4,4}=1$, $d_{4,8}=1$, $d_{4,14}=1$, $d_{4,15}=1$, $d_{4,16}=1$, $d_{4,17}=1$, $d_{4,18}=1$, $d_{4,19}=0$, $d_{4,20}=1$, $d_{4,25}=0$, $d_{4,26}=0$, $d_{4,27}=0$, $d_{4,28}=0$, $d_{4,29}=0$, $d_{4,30}=0$, $d_{4,31}=1$, $\boxed{d_{4,32}=a_{4,32}}$
c_4	$c_{4,4}=0$, $c_{4,16}=0$, $c_{4,25}=1$, $c_{4,26}=0$, $c_{4,27}=1$, $c_{4,28}=1$, $c_{4,29}=1$, $c_{4,30}=1$, $c_{4,31}=1$, $\boxed{c_{4,32}=d_{4,32}}$
b_4	$b_{4,30}=1$, $\boxed{b_{4,32}=c_{4,32}}$
2R	$a_{5,4}=b_{4,4}$, $a_{5,16}=b_{4,16}$, $a_{5,18}=0$, $\boxed{a_{5,32}=b_{4,32}}$, $d_{5,18}=1$, $d_{5,30}=a_{5,30}$, $\boxed{d_{5,32}=a_{5,32}}$, $c_{5,18}=0$, $\boxed{c_{5,32}=d_{5,32}}$, $\boxed{b_{5,32}=c_{5,32}}$, $a_{6,18}=b_{5,18}$, $\boxed{a_{6,32}=b_{5,32}}$, $\boxed{d_{6,32}=a_{6,32}}$, $\boxed{c_{6,32}=d_{6,32}}$, $\boxed{b_{6,32} \neq c_{6,32}}$
3R	$\phi_{34,32}=1$, $b_{12,32}=d_{12,32}$
4R	$a_{13,32}=c_{12,32}$, $d_{13,32} \neq b_{12,32}$, $c_{13,32}=a_{13,32}$, $b_{13,32}=d_{13,32}$, $a_{14,32}=c_{13,32}$, $d_{14,32}=b_{13,32}$, $c_{14,32}=a_{14,32}$, $b_{14,32}=d_{14,32}$, $a_{15,32}=c_{14,32}$, $d_{15,32}=b_{14,32}$, $c_{15,32}=a_{15,32}$, $b_{15,32} \neq d_{15,32}$, $a_{16,26}=1$, $a_{16,32}=c_{15,32}$, $d_{16,26}=1$, $d_{16,32}=b_{15,32}$, $c_{16,26}=1$, $c_{16,32}=a_{16,32}$, $b_{16,26}=1$