Message Recovery and Pseudo-Preimage Attacks on the Compression Function of Hamsi-256

Çağdaş Çalık and Meltem Sönmez Turan

Institute of Applied Mathematics, Middle East Technical University, Ankara, Turkey
Computer Security Division, National Institute of Standards and Technology, USA

Overview

Hamsi

- Hash function designed by Özgül Küçük from K.U. Leuven.

Hamsi

- Hash function designed by Özgül Küçük from K.U. Leuven.
- One of the 14 second round finalists of NIST SHA-3 competition.

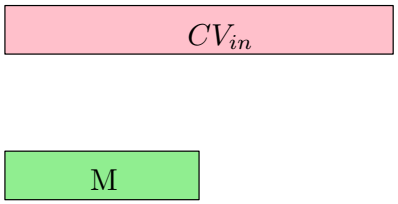
Hamsi

- Hash function designed by Özgül Küçük from K.U. Leuven.
- One of the 14 second round finalists of NIST SHA-3 competition.
- Supports output sizes of 224, 256, 384, 512 bits.

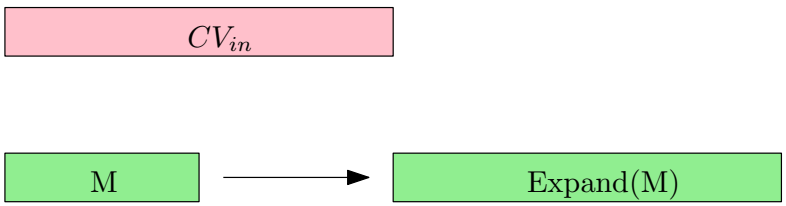
Hamsi

- Hash function designed by Özgül Küçük from K.U. Leuven.
- One of the 14 second round finalists of NIST SHA-3 competition.
- Supports output sizes of 224, 256, 384, 512 bits.
- This work is focused on the compression function of Hamsi-256.

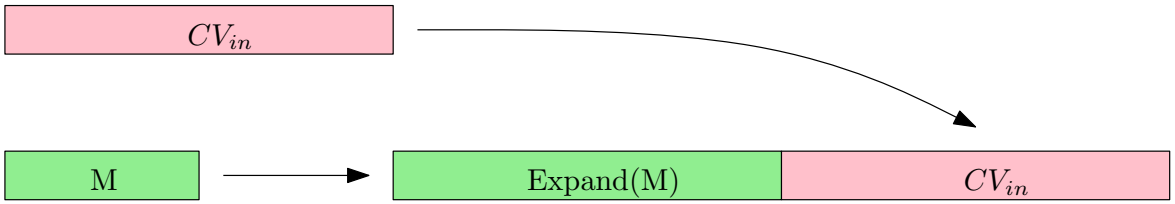
Compression Function of Hamsi-256



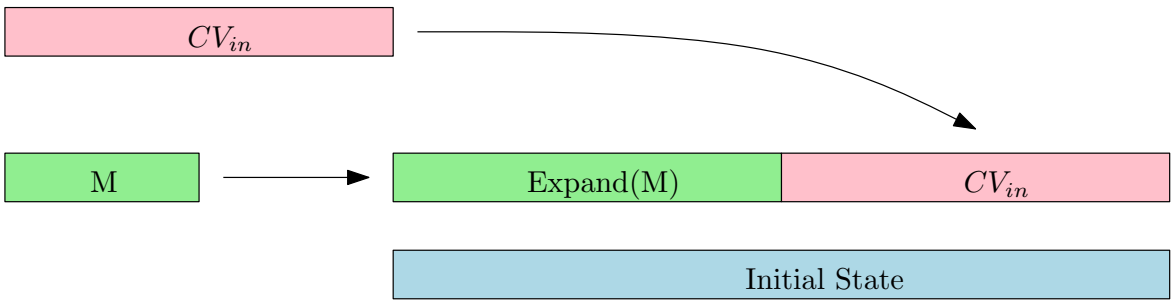
Compression Function of Hamsi-256



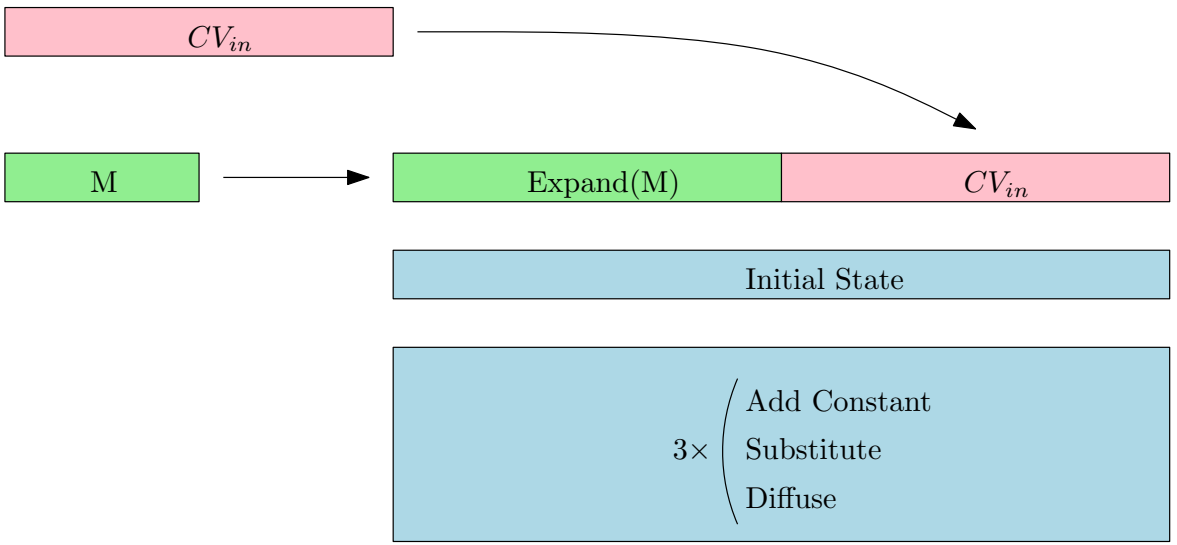
Compression Function of Hamsi-256



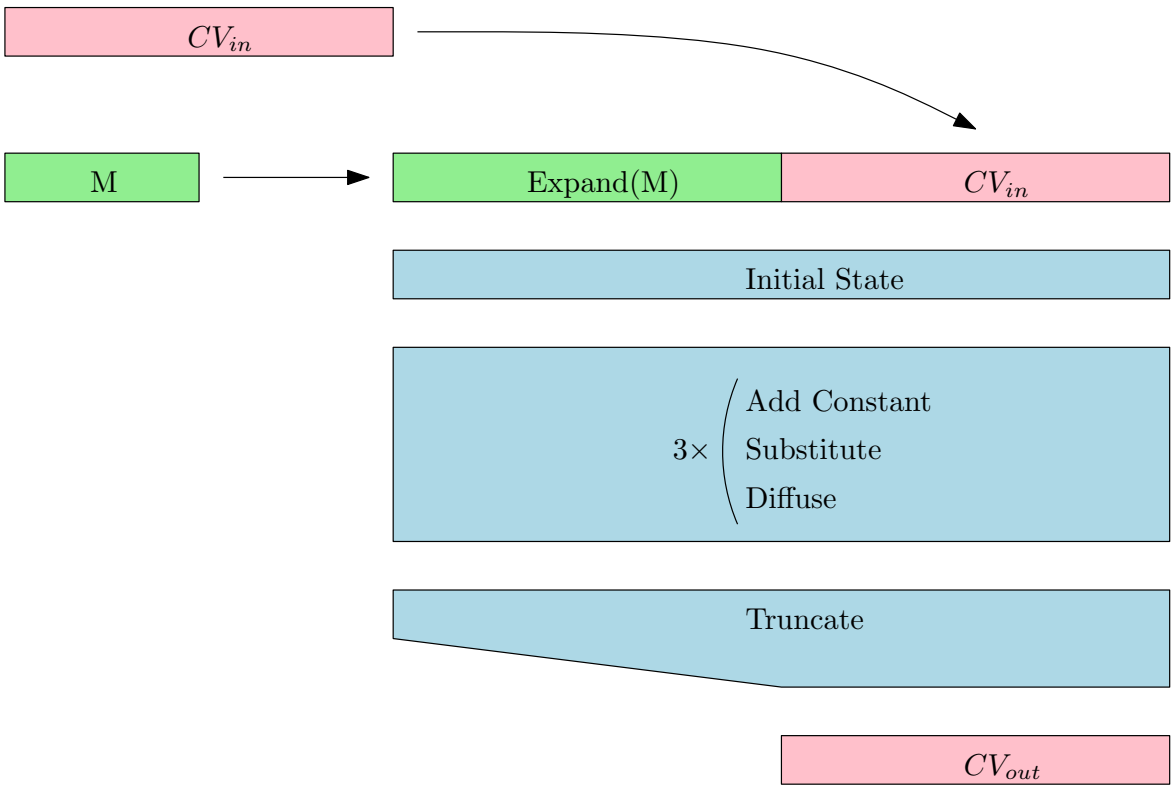
Compression Function of Hamsi-256



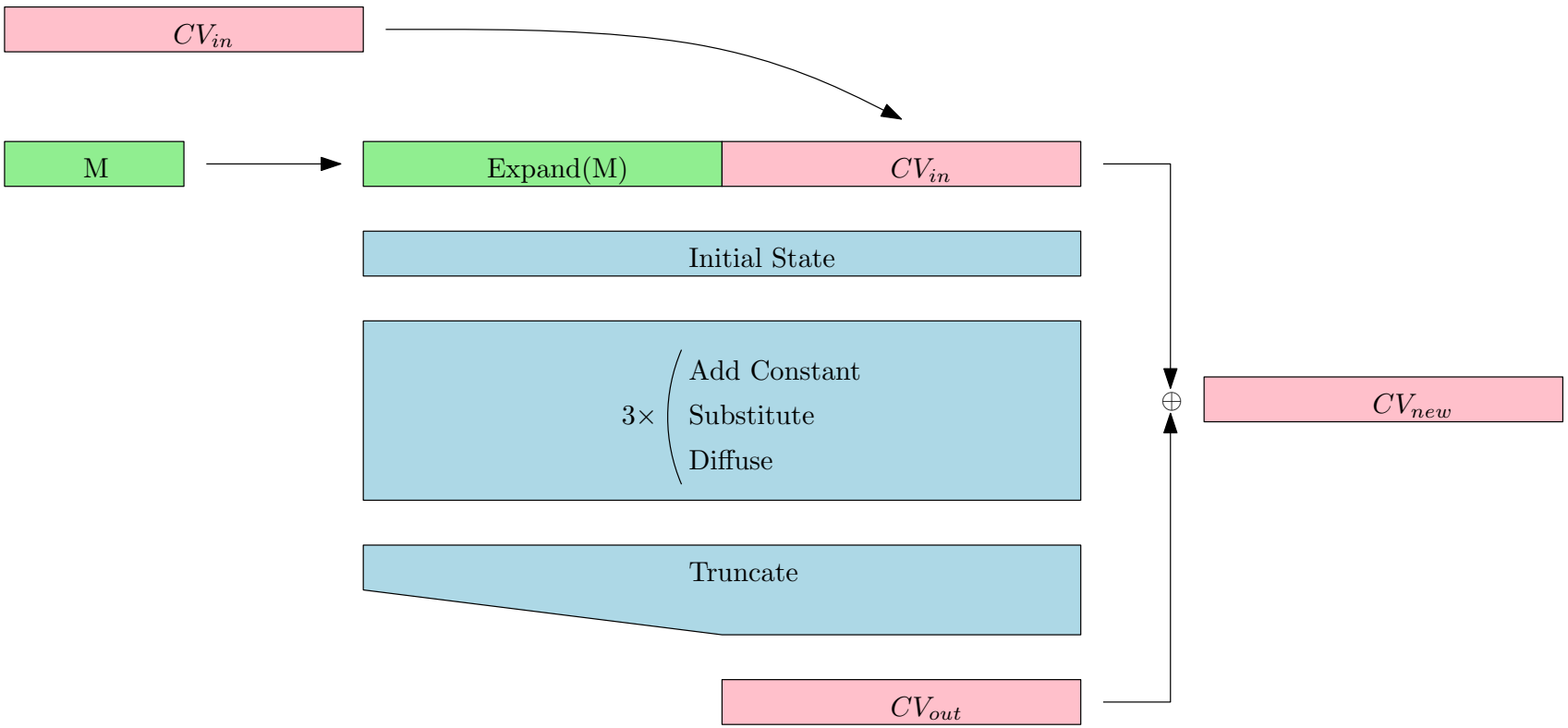
Compression Function of Hamsi-256



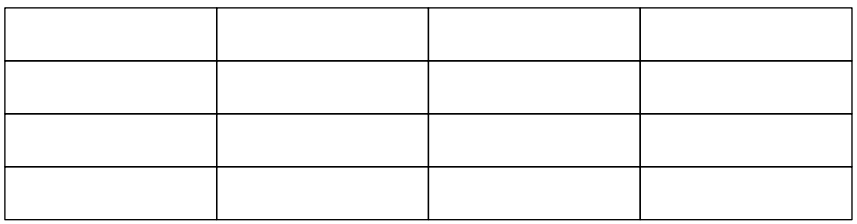
Compression Function of Hamsi-256



Compression Function of Hamsi-256



Initialization of the State



- Hamsi-256 state consists of a 4x4 array of 32-bit words.
- It can also be regarded as 128 columns, each containing a 4-bit value.
- Each column contains 2-bits from expanded message and 2-bits from the CV.

Initialization of the State

m_0	m_1		
		m_2	m_3
m_4	m_5		
		m_6	m_7

- Hamsi-256 state consists of a 4x4 array of 32-bit words.
- It can also be regarded as 128 columns, each containing a 4-bit value.
- Each column contains 2-bits from expanded message and 2-bits from the CV.

Initialization of the State

m_0	m_1	cv_0	cv_1
cv_2	cv_3	m_2	m_3
m_4	m_5	cv_4	cv_5
cv_6	cv_7	m_6	m_7

- Hamsi-256 state consists of a 4x4 array of 32-bit words.
- It can also be regarded as 128 columns, each containing a 4-bit value.
- Each column contains 2-bits from expanded message and 2-bits from the CV.

Initialization of the State

m_0	m_1	cv_0	cv_1
cv_2	cv_3	m_2	m_3
m_4	m_5	cv_4	cv_5
cv_6	cv_7	m_6	m_7

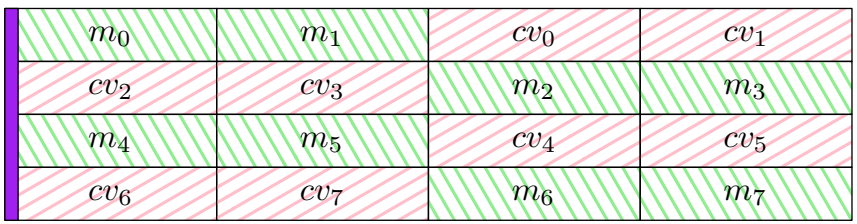
- Hamsi-256 state consists of a 4x4 array of 32-bit words.
- It can also be regarded as 128 columns, each containing a 4-bit value.
- Each column contains 2-bits from expanded message and 2-bits from the CV.

Initialization of the State

m_0	m_1	cv_0	cv_1
cv_2	cv_3	m_2	m_3
m_4	m_5	cv_4	cv_5
cv_6	cv_7	m_6	m_7

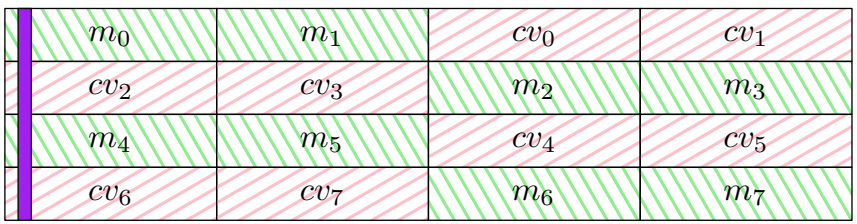
- Hamsi-256 state consists of a 4x4 array of 32-bit words.
- It can also be regarded as 128 columns, each containing a 4-bit value.
- Each column contains 2-bits from expanded message and 2-bits from the CV.

Substitution



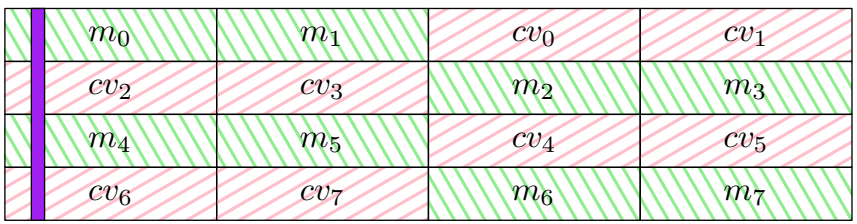
- Each column is subsituted by the 4x4 Hamsi s-box.

Substitution



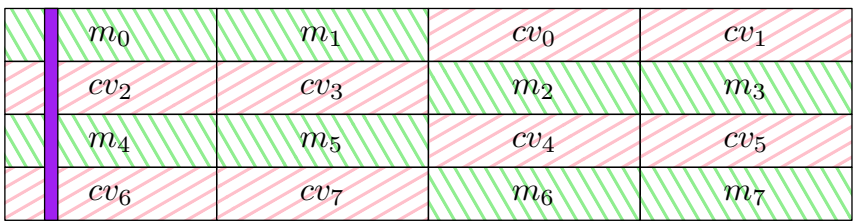
- Each column is subsituted by the 4x4 Hamsi s-box.

Substitution



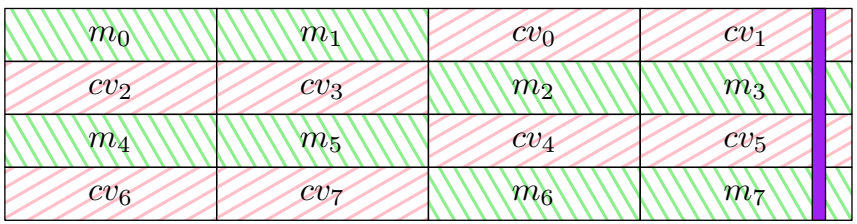
- Each column is subsituted by the 4x4 Hamsi s-box.

Substitution



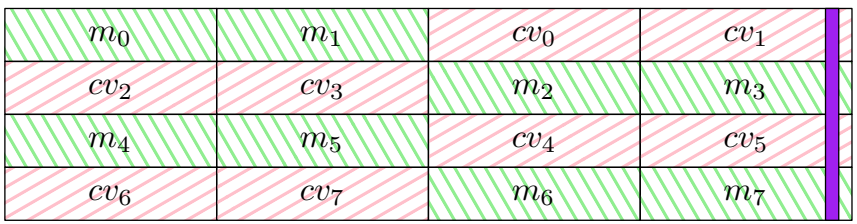
- Each column is subsituted by the 4x4 Hamsi s-box.

Substitution



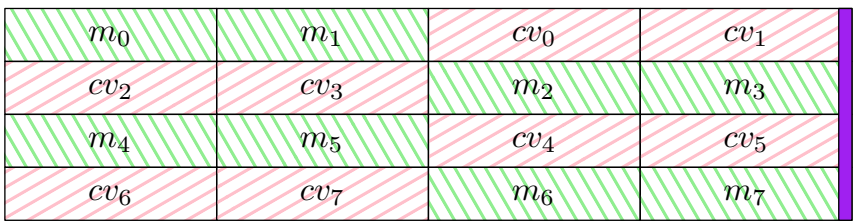
- Each column is subsituted by the 4x4 Hamsi s-box.

Substitution



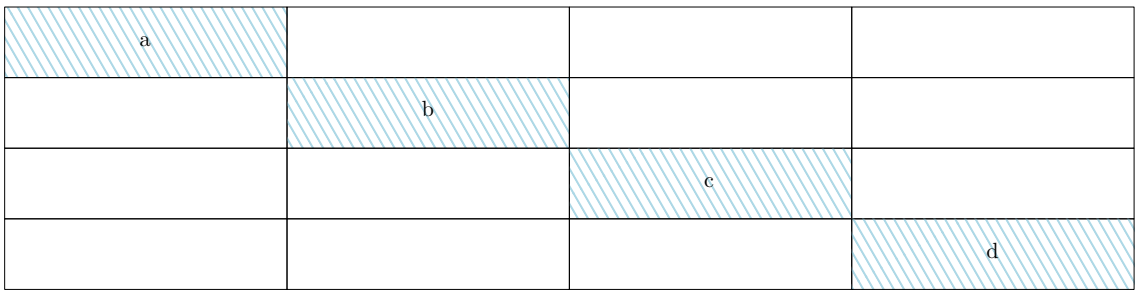
- Each column is subsituted by the 4x4 Hamsi s-box.

Substitution



- Each column is subsituted by the 4x4 Hamsi s-box.

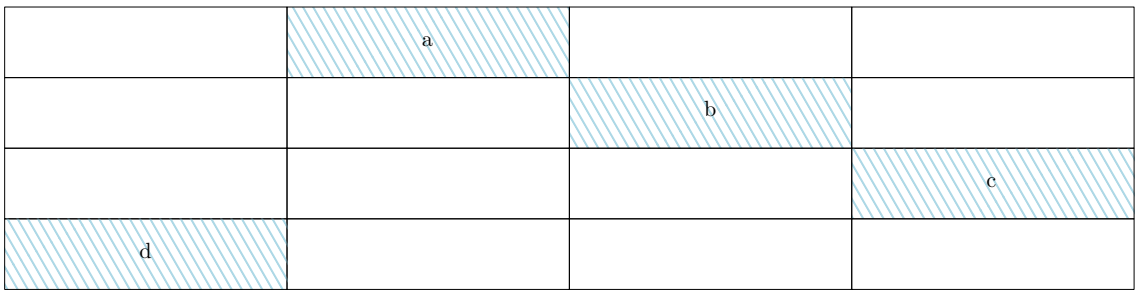
Diffusion



L

$$
\begin{aligned}
a &:= a \lll 13 \\
c &:= c \lll 3 \\
b &:= b \oplus a \oplus c \\
d &:= d \oplus c \oplus (a \ll 3) \\
b &:= b \lll 1 \\
d &:= d \lll 7 \\
a &:= a \oplus b \oplus d \\
c &:= c \oplus d \oplus (b \ll 7) \\
a &:= a \lll 5 \\
c &:= c \lll 22
\end{aligned}
$$

Diffusion



L

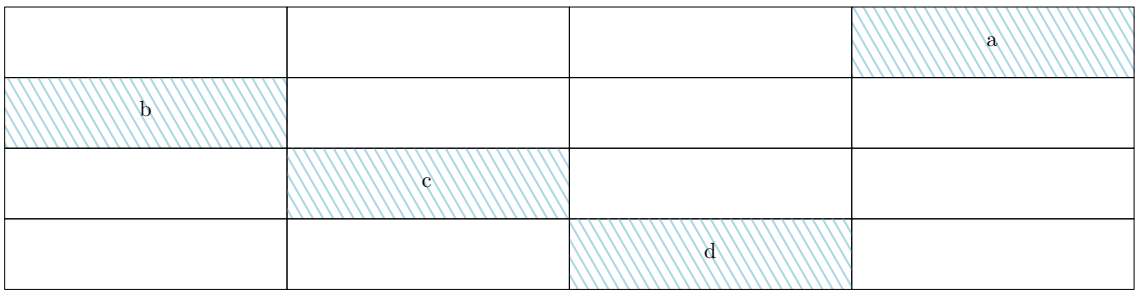
$$
\begin{aligned}
a &:= a \lll 13 \\
c &:= c \lll 3 \\
b &:= b \oplus a \oplus c \\
d &:= d \oplus c \oplus (a \ll 3) \\
b &:= b \lll 1 \\
d &:= d \lll 7 \\
a &:= a \oplus b \oplus d \\
c &:= c \oplus d \oplus (b \ll 7) \\
a &:= a \lll 5 \\
c &:= c \lll 22
\end{aligned}
$$

Diffusion

L

$$
\begin{aligned}
a &:= a \lll 13 \\
c &:= c \lll 3 \\
b &:= b \oplus a \oplus c \\
d &:= d \oplus c \oplus (a \ll 3) \\
b &:= b \lll 1 \\
d &:= d \lll 7 \\
a &:= a \oplus b \oplus d \\
c &:= c \oplus d \oplus (b \ll 7) \\
a &:= a \lll 5 \\
c &:= c \lll 22
\end{aligned}
$$

Diffusion



L

$$
\begin{aligned}
a &:= a \lll 13 \\
c &:= c \lll 3 \\
b &:= b \oplus a \oplus c \\
d &:= d \oplus c \oplus (a \ll 3) \\
b &:= b \lll 1 \\
d &:= d \lll 7 \\
a &:= a \oplus b \oplus d \\
c &:= c \oplus d \oplus (b \ll 7) \\
a &:= a \lll 5 \\
c &:= c \lll 22
\end{aligned}
$$

Substitution

Table: Difference distribution table of the Hamsi s-box.

δ_i / δ_j	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	2	0	2	0	0	2	2	2	0	4	2
2	0	0	0	4	0	4	0	0	0	4	0	0	0	0	0	4
3	0	4	2	0	0	0	2	0	0	2	0	0	2	0	2	2
4	0	0	0	0	0	0	4	0	0	0	4	4	0	4	0	0
5	0	4	0	2	2	2	2	0	2	0	0	0	2	0	0	0
6	0	0	2	2	2	2	0	0	2	2	0	0	0	0	2	2
7	0	0	0	0	4	2	0	2	0	0	2	2	2	0	0	2
8	0	0	0	2	0	2	0	4	0	2	0	0	0	4	0	2
9	0	0	0	2	0	0	0	2	4	2	2	2	2	0	0	0
a	0	0	2	0	2	0	4	0	2	0	4	0	0	0	2	0
b	0	4	0	0	2	0	2	0	2	2	0	0	2	0	0	2
c	0	0	2	0	2	0	0	0	2	0	0	4	0	4	2	0
d	0	4	2	2	0	2	2	0	0	0	0	0	2	0	2	0
e	0	0	2	0	2	0	0	4	2	0	0	0	0	4	2	0
f	0	0	4	2	0	0	0	2	0	2	2	2	2	0	0	0

Substitution

Table: Difference distribution table of the Hamsi s-box.

δ_i / δ_j	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	2	0	2	0	0	2	2	2	0	4	2
2	0	0	0	4	0	4	0	0	0	4	0	0	0	0	0	4
3	0	4	2	0	0	0	2	0	0	2	0	0	2	0	2	2
4	0	0	0	0	0	0	4	0	0	0	4	4	0	4	0	0
5	0	4	0	2	2	2	2	0	2	0	0	0	2	0	0	0
6	0	0	2	2	2	2	0	0	2	2	0	0	0	0	2	2
7	0	0	0	0	4	2	0	2	0	0	2	2	2	0	0	2
8	0	0	0	2	0	2	0	4	0	2	0	0	0	4	0	2
9	0	0	0	2	0	0	0	2	4	2	2	2	2	0	0	0
a	0	0	2	0	2	0	4	0	2	0	4	0	0	0	2	0
b	0	4	0	0	2	0	2	0	2	2	0	0	2	0	0	2
c	0	0	2	0	2	0	0	0	2	0	0	4	0	4	2	0
d	0	4	2	2	0	2	2	0	0	0	0	0	2	0	2	0
e	0	0	2	0	2	0	0	4	2	0	0	0	0	4	2	0
f	0	0	4	2	0	0	0	2	0	2	2	2	2	0	0	0

Diffusion

- Each bit in L affects at least 2 output bits.
- Each bit in L affects at most 7 output bits.

Message Expansion

- (128,16,70) linear code over F_4.
- Any non-zero difference in a message block spreads to at least 70 columns.
- Probability of a 1-round characteristic is at least $2^{-2.70} = 2^{-140}$.

Finding unaffected bits

Aim

- Given an input CV difference, find (if there are any) the output bits that will not be affected with probability 1.

Finding unaffected bits

Aim

- Given an input CV difference, find (if there are any) the output bits that will not be affected with probability 1.

Substitution

- If any of the bits in a column is affected, mark the whole column bits as affected, i.e. assign the value $0xf$ to that column.
- Exception: In the first Substitution operation, we can use the truncated differential property of the s-box. If the input difference is $0xa$, assign the value $0xe$ to that column, i.e., the least significant bit will not be affected.

Finding unaffected bits

Aim

- Given an input CV difference, find (if there are any) the output bits that will not be affected with probability 1.

Substitution

- If any of the bits in a column is affected, mark the whole column bits as affected, i.e. assign the value $0xf$ to that column.
- Exception: In the first Substitution operation, we can use the truncated differential property of the s-box. If the input difference is $0xa$, assign the value $0xe$ to that column, i.e., the least significant bit will not be affected.

Diffusion

- Precomputation: For each state bit, construct a list that contains the indices which affects that bit of the state.
- Using this list, mark a state bit affected if any of the bits it depends is affected.

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Substitute

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Substitute

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Diffuse

00000000	00000010	00000082	40000000
02000000	00000000	00000000	10000000
01020000	00000000	00000000	00200000
80000000	00000004	00000000	00000000

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

↓ Round 1 Substitute

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

↓ Round 1 Diffuse

00000000	00000010	00000082	40000000
02000000	00000000	00000000	10000000
01020000	00000000	00000000	00200000
80000000	00000004	00000000	00000000

↓ Round 2 Substitute

83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Substitute

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Diffuse

00000000	00000010	00000082	40000000
02000000	00000000	00000000	10000000
01020000	00000000	00000000	00200000
80000000	00000004	00000000	00000000

Round 2 Substitute

83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000

Round 2 Diffuse

08771d1a	60f028b1	2e19419f	c58be9a0
06041549	800028e8	02050105	b0608008
831e5403	68008abd	0e0415ff	f0e2e020
8b000141	49000a0c	00285100	10438828

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Substitute

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Diffuse

00000000	00000010	00000082	40000000
02000000	00000000	00000000	10000000
01020000	00000000	00000000	00200000
80000000	00000004	00000000	00000000

Round 2 Substitute

83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000

Round 2 Diffuse

08771d1a	60f028b1	2e19419f	c58be9a0
06041549	800028e8	02050105	b0608008
831e5403	68008abd	0e0415ff	f0e2e020
8b000141	49000a0c	00285100	10438828

Round 3 Substitute

8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8
8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8
8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8
8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8

Compression Function

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Substitute

00000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000
01000000	00000000	00000000	00000000

Round 1 Diffuse

00000000	00000010	00000082	40000000
02000000	00000000	00000000	10000000
01020000	00000000	00000000	00200000
80000000	00000004	00000000	00000000

Round 2 Substitute

83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000
83020000	00000014	00000082	50200000

Round 2 Diffuse

08771d1a	60f028b1	2e19419f	c58be9a0
06041549	800028e8	02050105	b0608008
831e5403	68008abd	0e0415ff	f0e2e020
8b000141	49000a0c	00285100	10438828

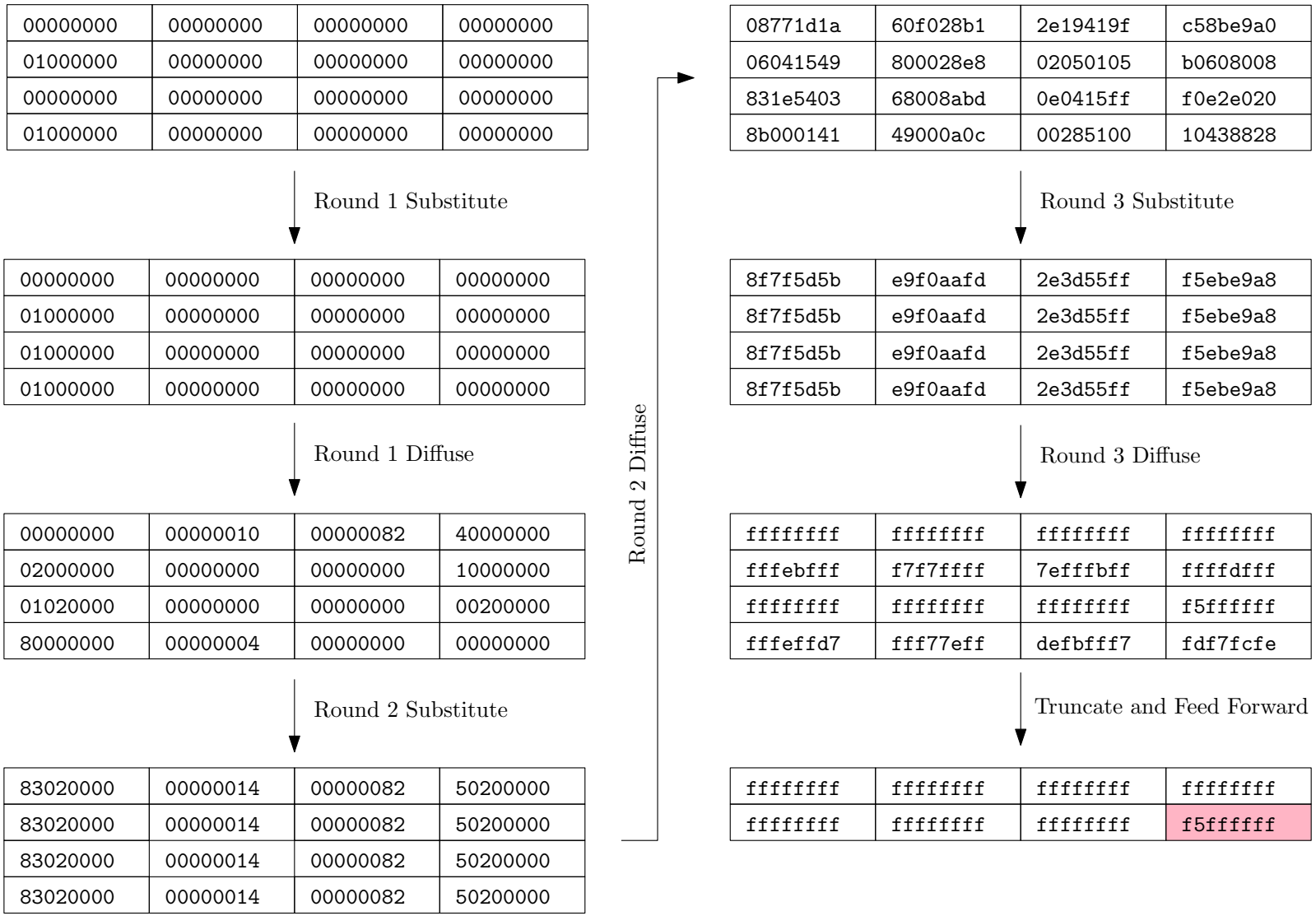
Round 3 Substitute

8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8
8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8
8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8
8f7f5d5b	e9f0aafd	2e3d55ff	f5ebe9a8

Round 3 Diffuse

ffffffff	ffffffff	ffffffff	ffffffff
fffebfff	f7f7ffff	7efffbff	ffffdfff
ffffffff	ffffffff	ffffffff	f5ffffff
fffeffd7	fff77eff	defbfff7	fdf7fcfe

Compression Function



Unaffected bit list for 1-bit input diff.

Column	CV bit	Unaffected output bits	Condition
14	78	78,150	$m_{14} = 1, m_{142} = 1$
15	79	79,151	$m_{15} = 1, m_{143} = 1$
46	110	110, 182	$m_{46} = 1, m_{174} = 1$
47	111	111, 183	$m_{47} = 1, m_{175} = 1$
78	130	14, 214	$m_{78} = 0, m_{206} = 0$
79	131	15, 215	$m_{79} = 0, m_{207} = 0$
110	184	46, 246	$m_{110} = 0, m_{138} = 0$
111	185	47, 247	$m_{111} = 0, m_{139} = 0$

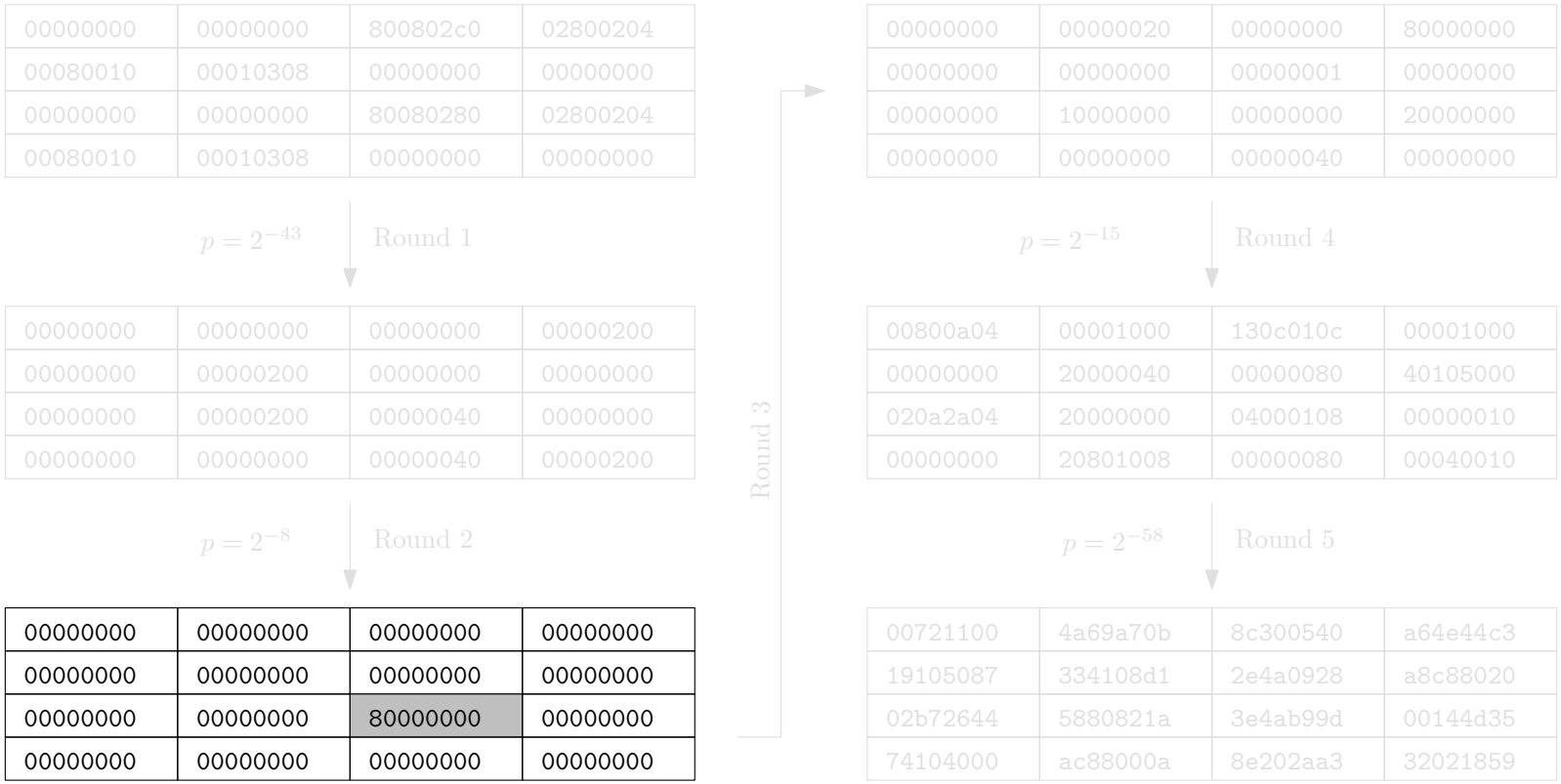
Unaffected bit list for 2-bit input diff.

Column	CV bits	Unaffected bits	Column	CV bits	Unaffected bits
0	64,192	165, 201	32	96,224	197, 233
1	65,193	166	33	97,225	198
2	66,194	167	34	98,226	199
3	67,195	168	35	99,227	200
4	68,196	169	36	100,228	201
6	70,198	227	38	102,230	131
7	71,199	228, 230	39	103,231	132, 134
8	72,200	229	40	104,232	133
9	73,201	131, 230	41	105,233	134, 163
10	74,202	132, 231	42	106,234	135, 164
11	75,203	133, 232	43	107,235	136, 165
12	76,204	134, 233	44	108,236	137, 166
13	77,205	135	45	109,237	167
14	78,206	136	46	110,238	168
15	79,207	137	47	111,239	169
26	90,218	195	58	122,250	227
27	91,219	196	59	123,251	228
28	92,220	197	60	124,252	229
29	93,221	198	61	125,253	230
30	94,222	163, 199	62	126,254	195, 231
31	95,223	164, 200	63	127,255	196, 232

Attacks

- Distinguisher
- Message Recovery Attack
- Pseudo-Preimage Attack

Distinguisher



- Start with a low weight difference in the state.

Distinguisher

00000000	00000000	800802c0	02800204
00080010	00010308	00000000	00000000
00000000	00000000	80080280	02800204
00080010	00010308	00000000	00000000

$p = 2^{-43}$ Round 1

00000000	00000000	00000000	00000200
00000000	00000200	00000000	00000000
00000000	00000200	00000040	00000000
00000000	00000000	00000040	00000200

$p = 2^{-8}$ Round 2

00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000
00000000	00000000	80000000	00000000
00000000	00000000	00000000	00000000

Round 3

00000000	00000020	00000000	80000000
00000000	00000000	00000001	00000000
00000000	10000000	00000000	20000000
00000000	00000000	00000040	00000000

$p = 2^{-15}$ Round 4

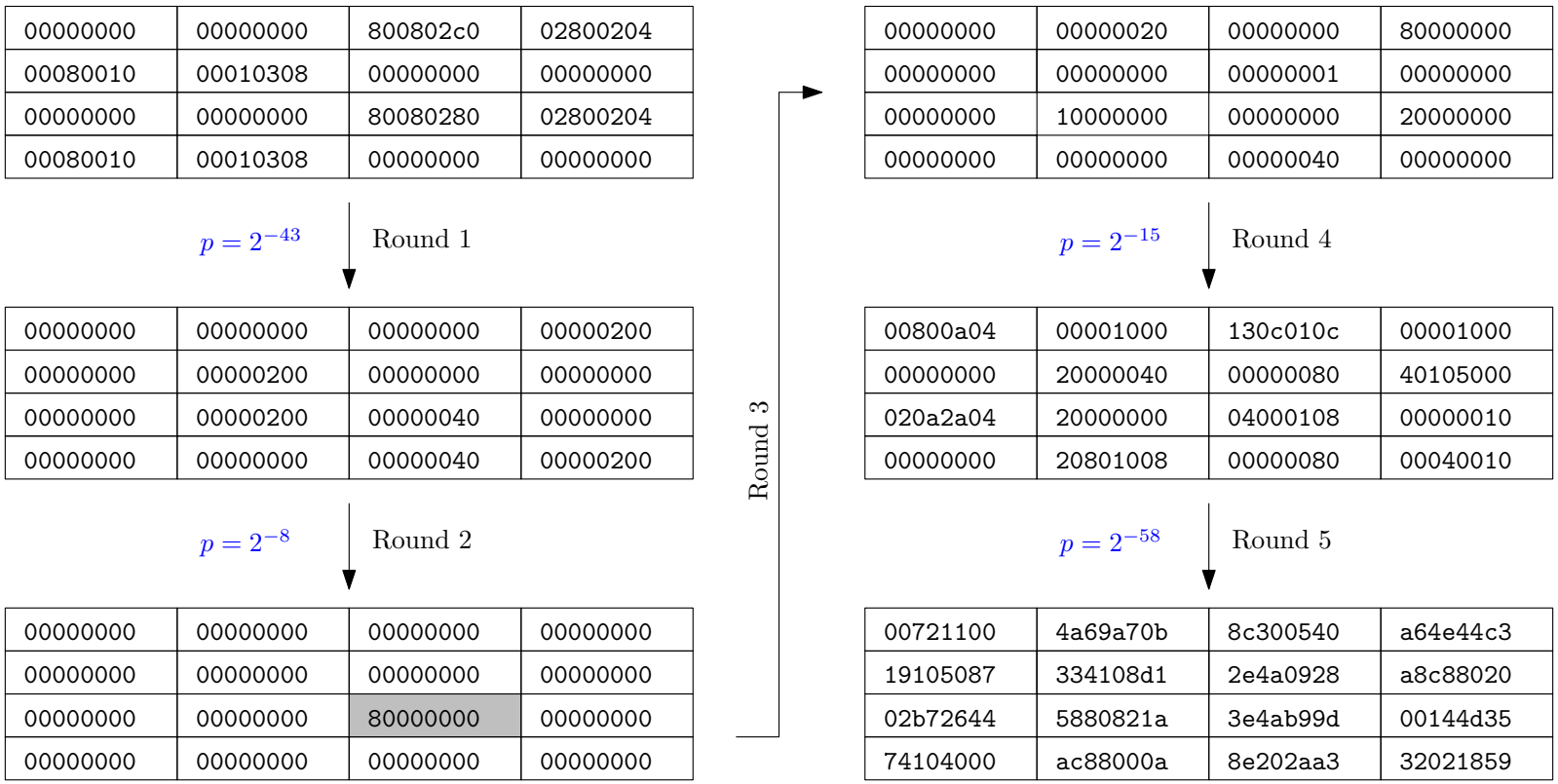
00800a04	00001000	130c010c	00001000
00000000	20000040	00000080	40105000
020a2a04	20000000	04000108	00000010
00000000	20801008	00000080	00040010

$p = 2^{-58}$ Round 5

00721100	4a69a70b	8c300540	a64e44c3
19105087	334108d1	2e4a0928	a8c88020
02b72644	5880821a	3e4ab99d	00144d35
74104000	ac88000a	8e202aa3	32021859

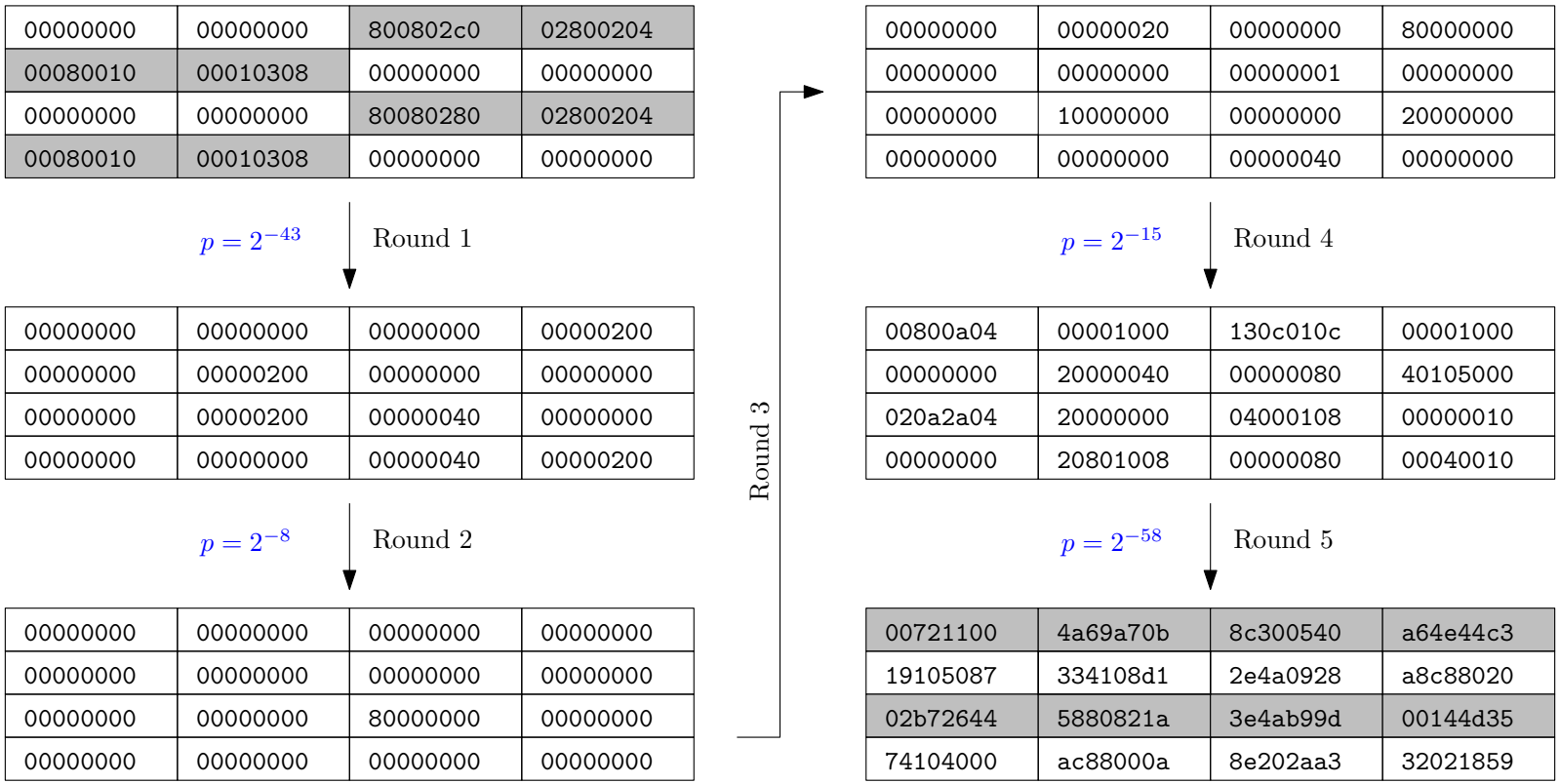
- Trace the difference back for 2 rounds. ($P = 2^{-51}$)

Distinguisher



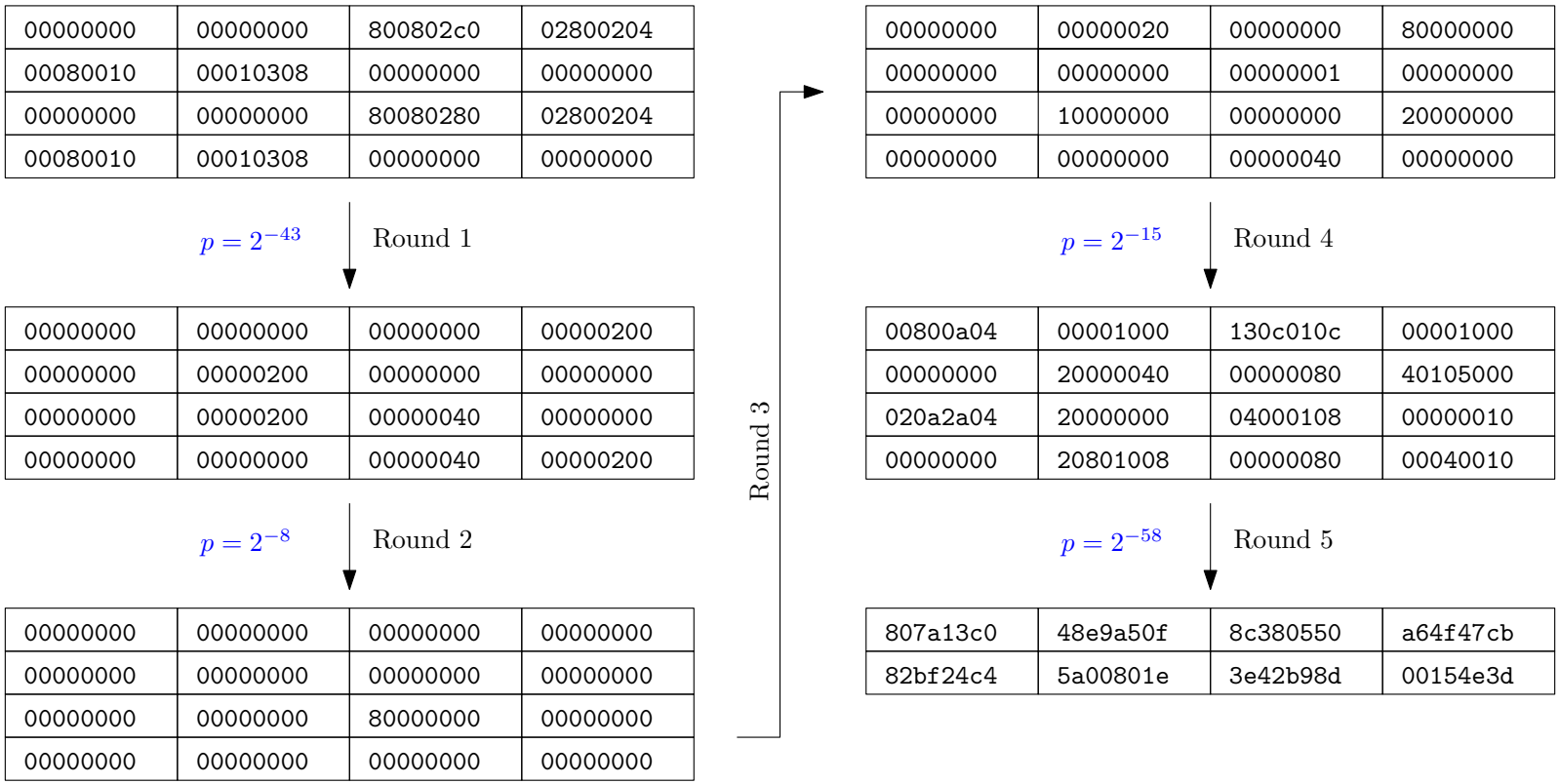
- Trace the difference forward for 3 rounds. ($P = 2^{-75}$)

Distinguisher



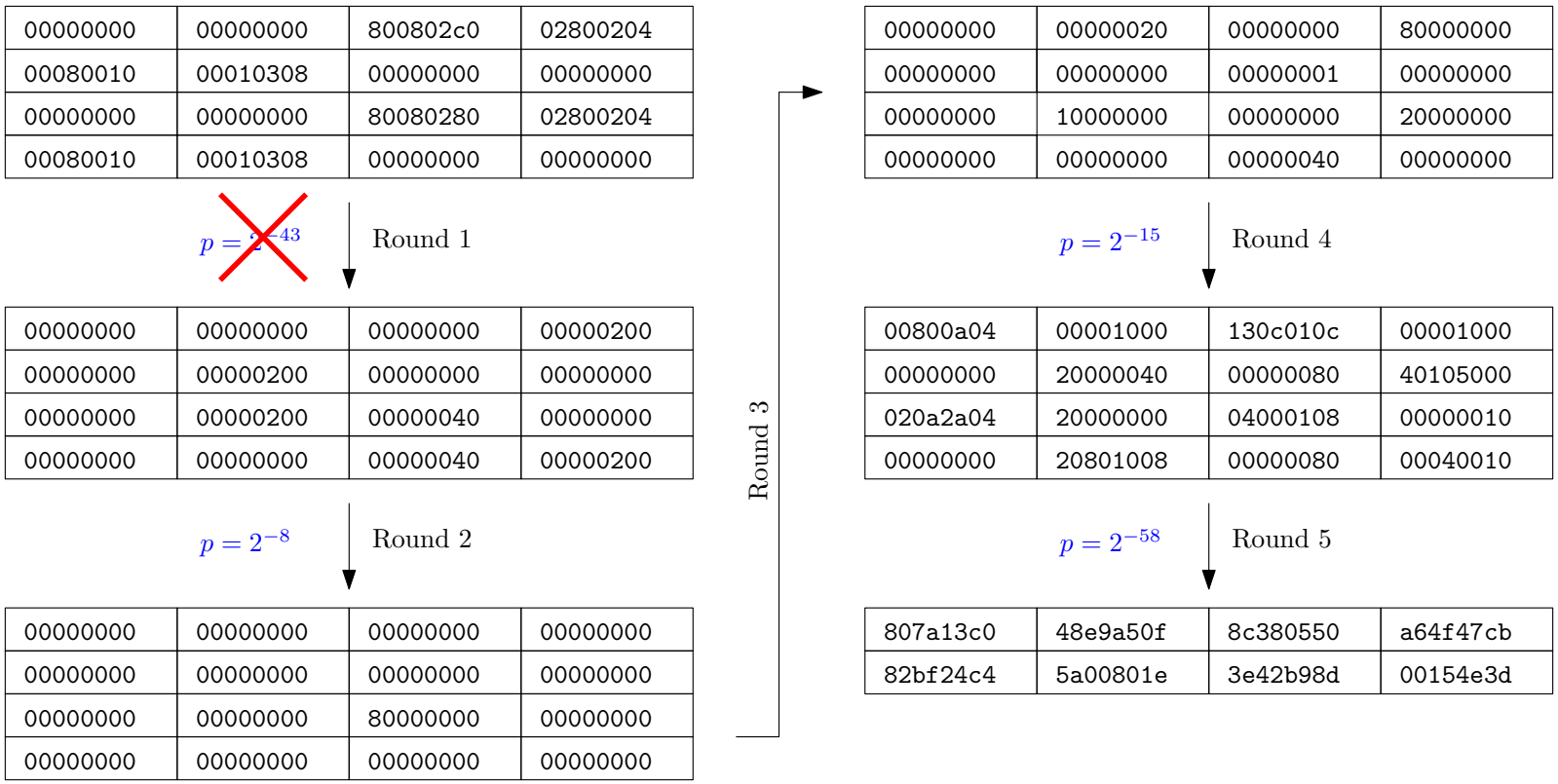
- Feed forward.

Distinguisher



- 5 round characteristic with $P = 2^{-126}$.

Distinguisher



- Choose an appropriate message and CV to satisfy first round transitions for free. ($P = 2^{-83}$)

Message Recovery Attack

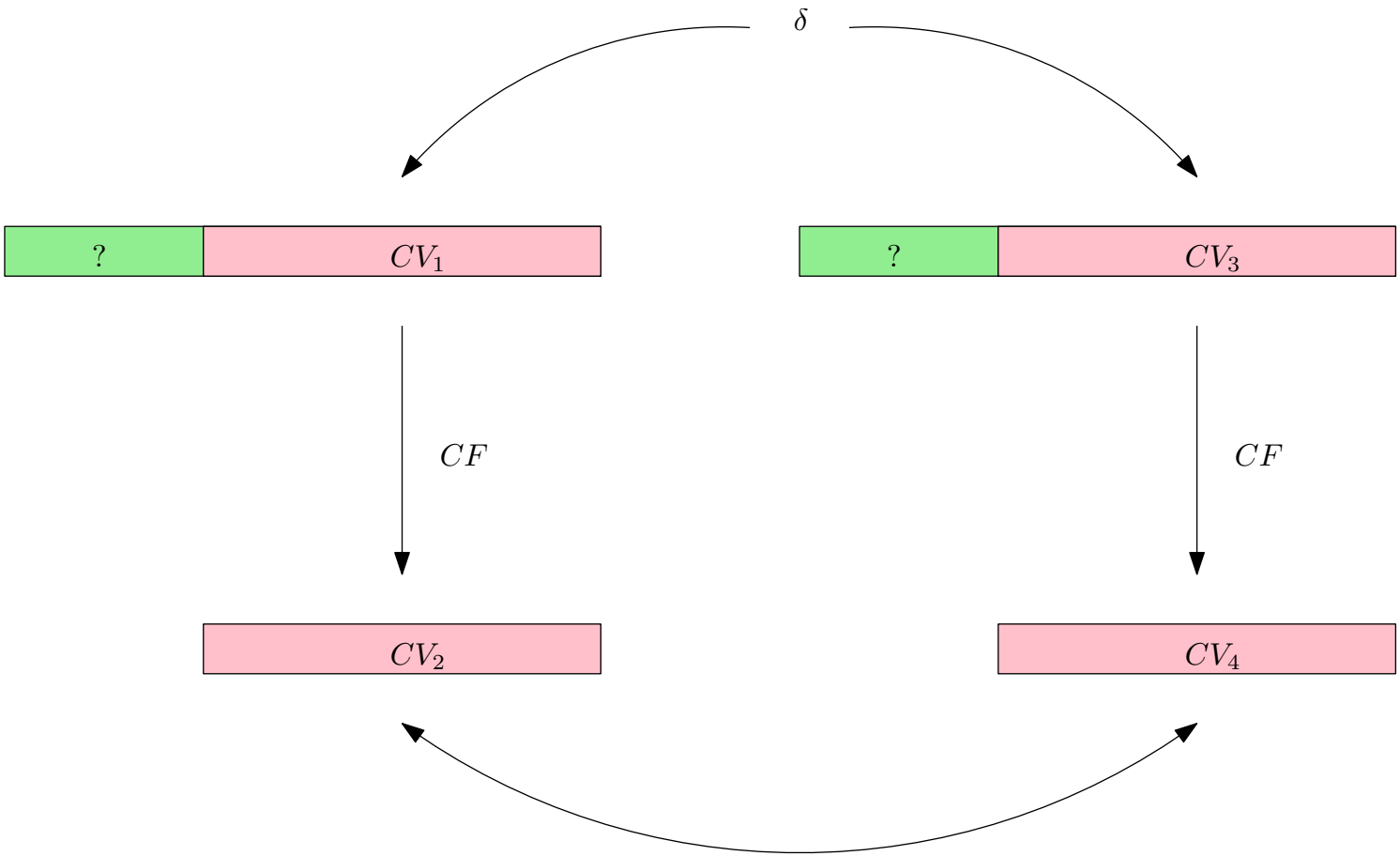
Challenge

Let $CF : M \times CV \rightarrow CV$, and $m \in M$ be secret. Find m by observing outputs of $CV(m, cv)$ for $cv's$ of our choice.

i.e., we can compute $F(cv) = CF(m, cv)$.

If CF would be ideally secure, we had to try 2^{32} possible messages.

Message Recovery Attack



Message Recovery Attack

Table: Unaffected output bits given the differences a_x and 2_x to column 18

Column	Difference	Message bits	Unaffected output bits
18	a_x	0_x	26 109 131 140 154 232
18	a_x	1_x	1 3 10 12 26 30 36 39 49 82 84 109 131 133 134 140 148 154 156 169 196 210 212 232 235 239
18	a_x	2_x	26 109 131 140 154 232
18	a_x	3_x	113 169 198 241
18	2_x	0_x	192
18	2_x	1_x,2_x,3_x	-

Message Recovery Attack

Table: List of chaining value bit indices used for message recovery

Column	$m = 1_x$	$m = 3_x$	$m = 2_x$
16	1	196	-
17	0	197	-
18	1	198	192
19	2	199	163
20	0	200	164
21	1	201	165
22	2	243	166
23	3	244	167
24	4	245	168
25	5	147	169
26	6	148	168
27	7	149	169
28	8	150	168
29	9	151	169
30	10	152	-
31	11	153	-

Message Recovery Attack

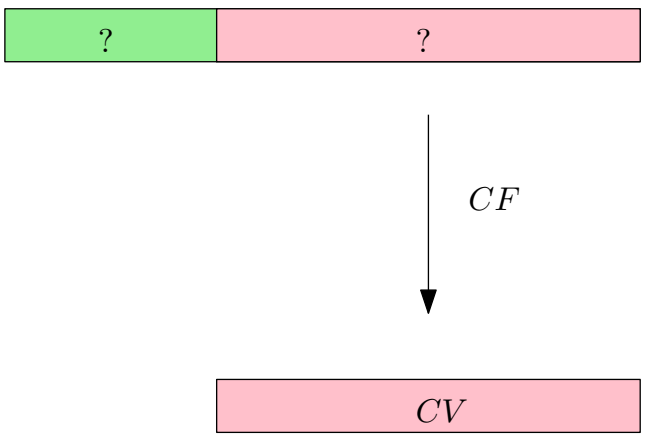
Complexity (worst case)

- According to the experiments we ran, 2^4 tries seems sufficient to decide whether a CV bit changes or not.
- 2 checks/column for the first 2 and last 2 columns: $2 \times 4 = 8$
- 3 checks/column for the remaining 12 columns: $3 \times 12 = 36$
- Exhaustive search for the 4 undetermined bits: 2^4
- Total complexity of the attack: $2^4.(36 + 8).2 + 2^4 = 2^{10.48}$

Pseudo-Preimage Attack

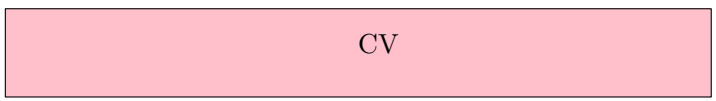
- Given a target CV.
- Find (M, CV_0) such that $CF(M, CV_0) = CV$.

Pseudo-Preimage Attack

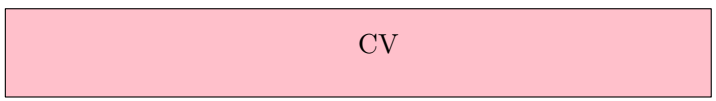


- Given a target CV.
- Find (M, CV_0) such that $CF(M, CV_0) = CV$.

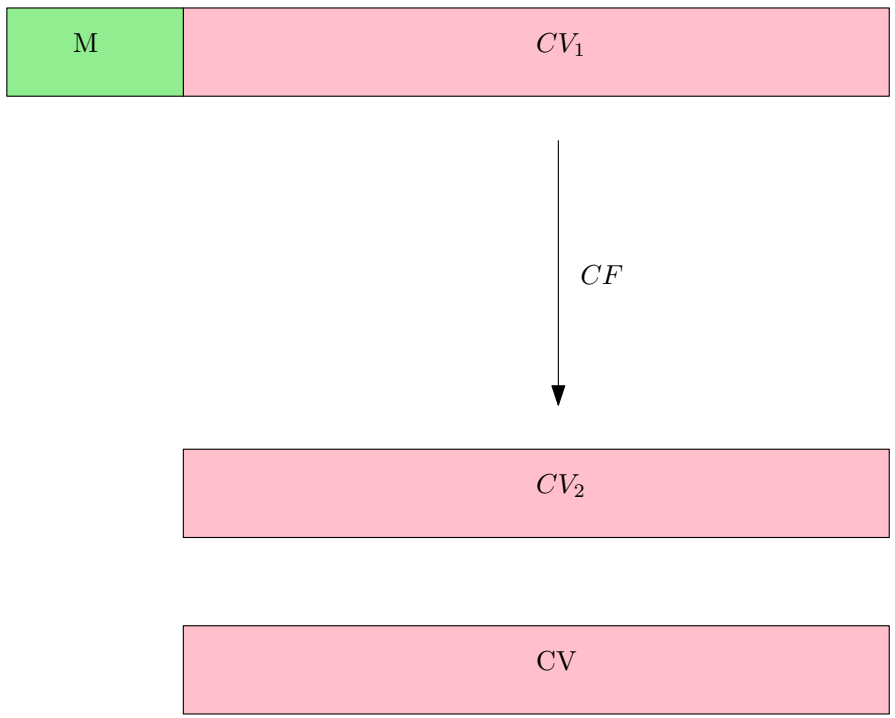
Pseudo-Preimage Attack



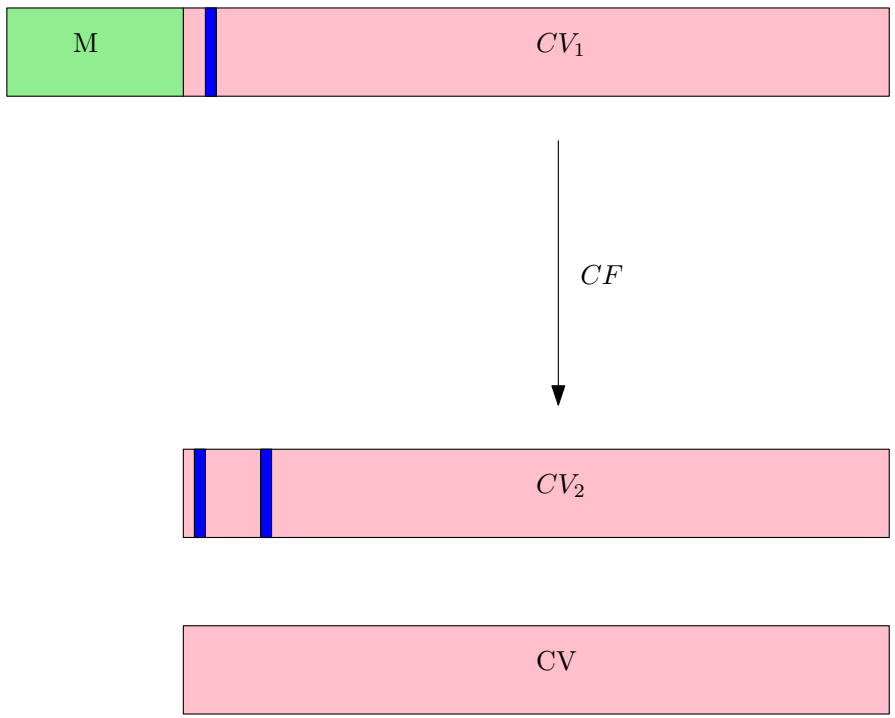
Pseudo-Preimage Attack



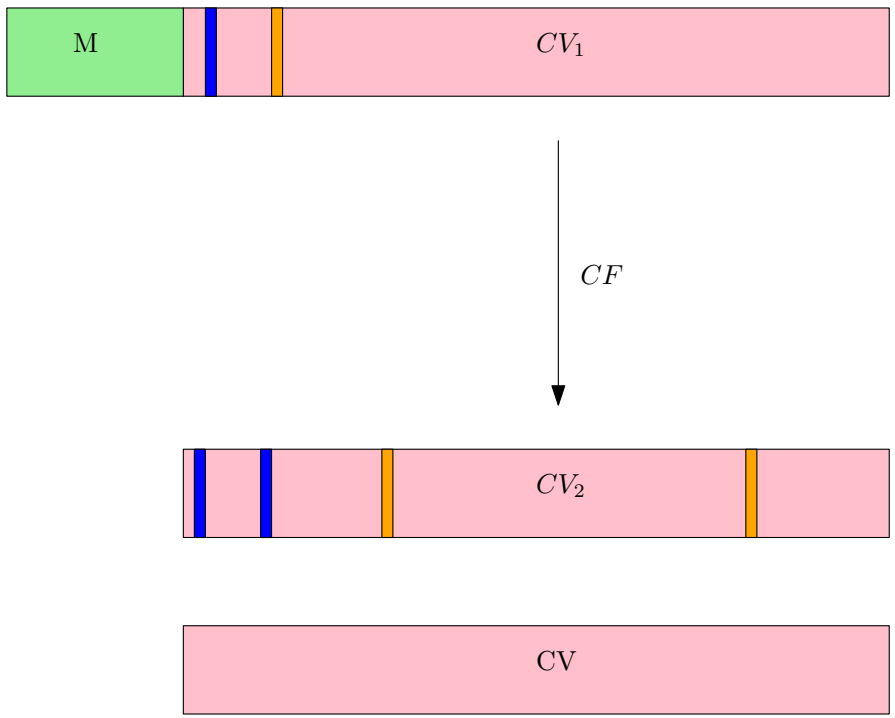
Pseudo-Preimage Attack



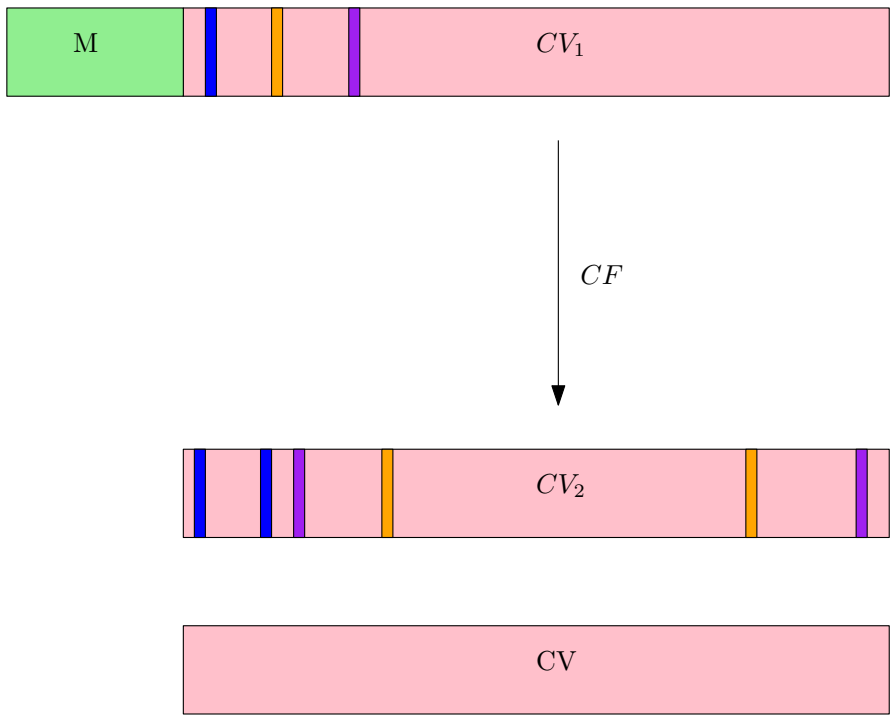
Pseudo-Preimage Attack



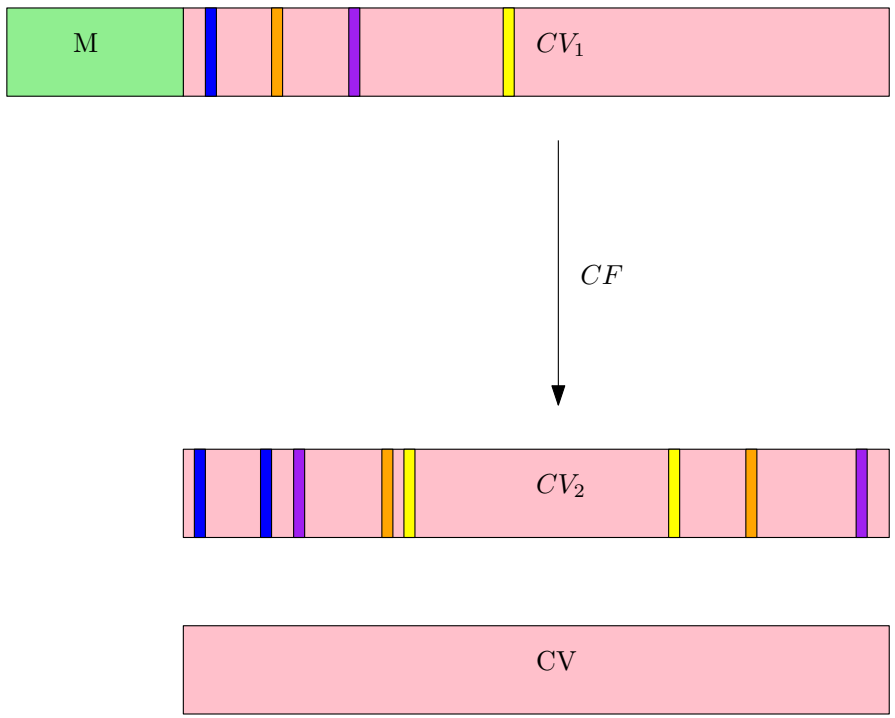
Pseudo-Preimage Attack



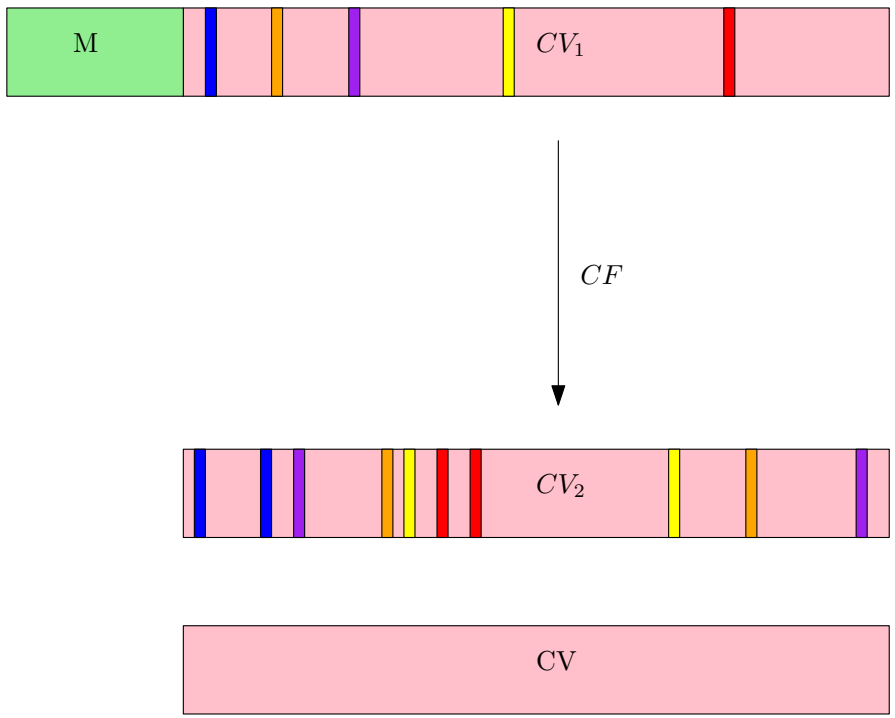
Pseudo-Preimage Attack



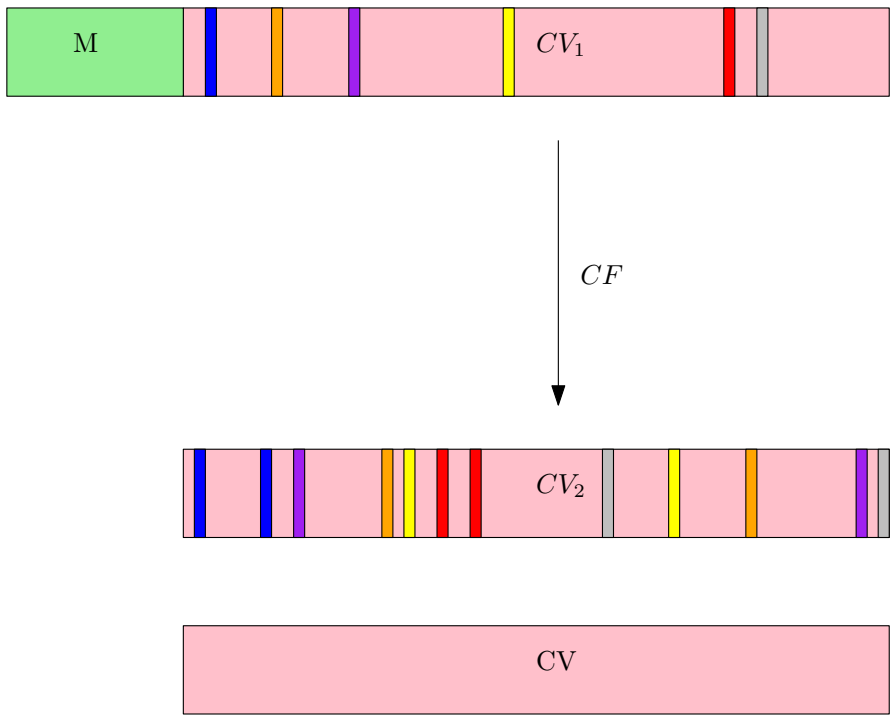
Pseudo-Preimage Attack



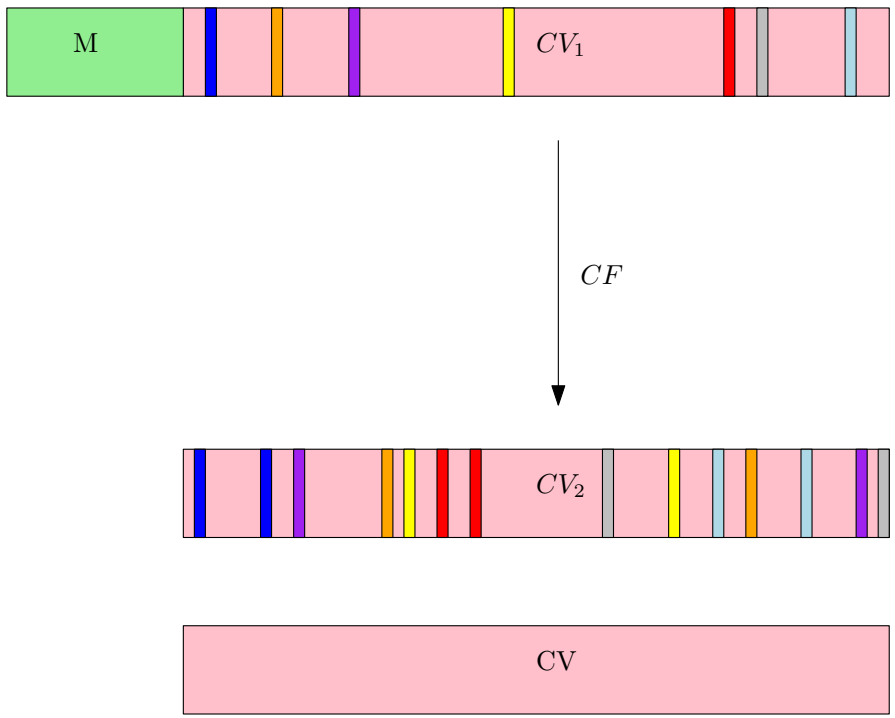
Pseudo-Preimage Attack



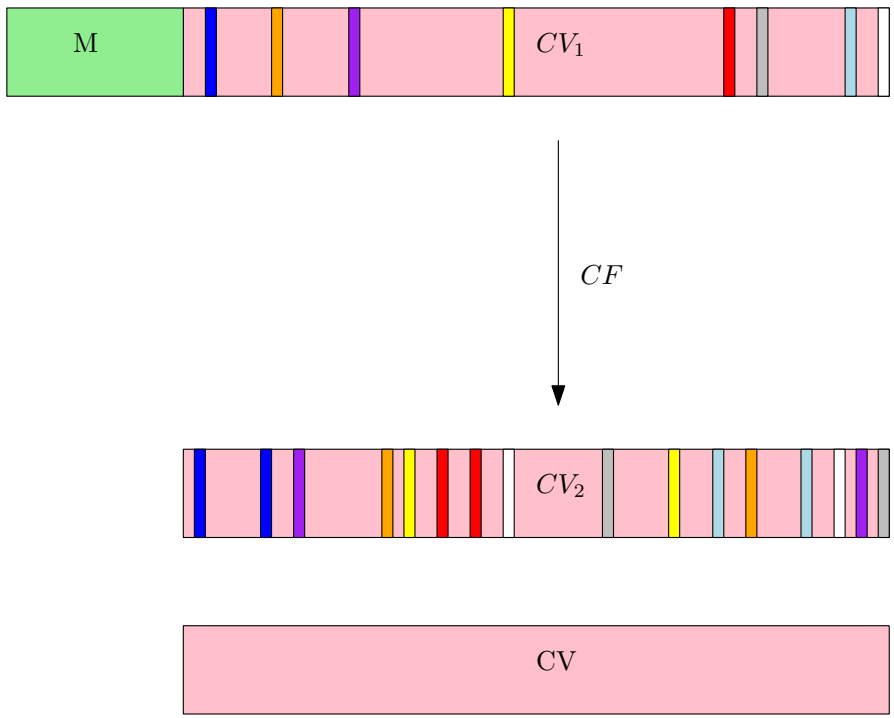
Pseudo-Preimage Attack



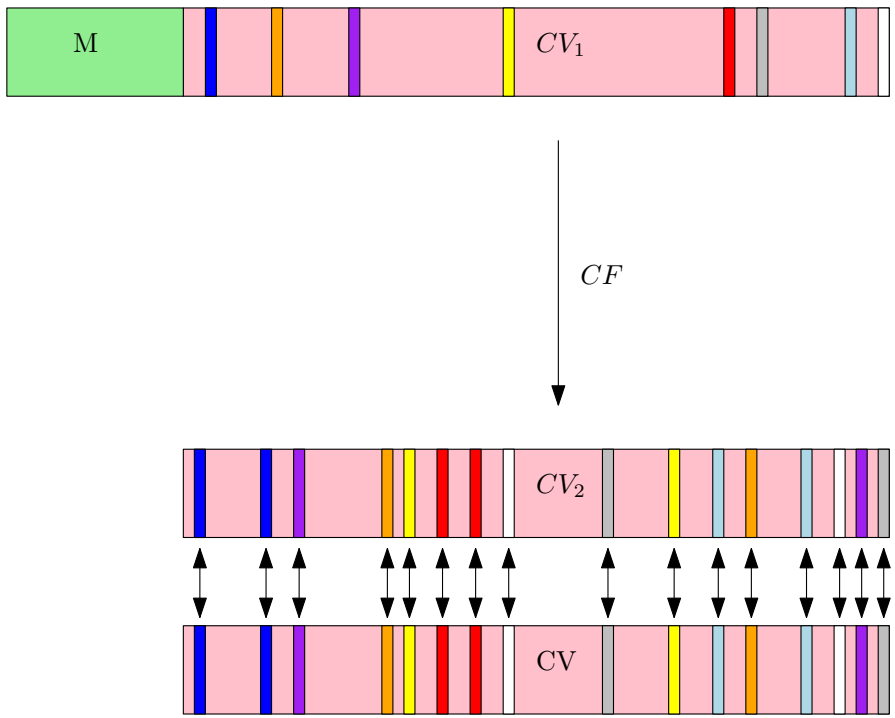
Pseudo-Preimage Attack



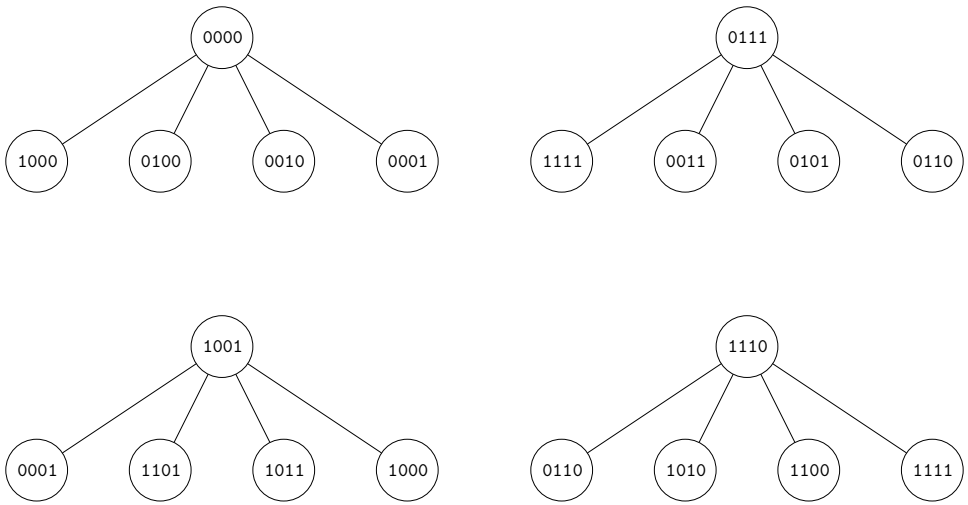
Pseudo-Preimage Attack



Pseudo-Preimage Attack



Example for $S = 2^4$



Vector	Covered by	Probability	Vector	Covered by	Probability
0000	-	1	1000	2	2^{-4}
0001	2	2^{-4}	1001	-	1
0010	1	2^{-2}	1010	1	2^{-2}
0011	1	2^{-2}	1011	1	2^{-4}
0100	1	2^{-2}	1100	1	2^{-2}
0101	1	2^{-2}	1101	1	2^{-2}
0110	2	2^{-4}	1110	-	1
0111	-	1	1111	2	2^{-4}

Pseudo-Preimage Attack

Table: The effect of size of S on the complexity gain

k	Size of S	Expected number evaluations	Gain
2	2^1	$2^{1.09}$	$2^{0.91}$
3	2^1	$2^{1.81}$	$2^{1.19}$
4	2^2	$2^{2.64}$	$2^{1.36}$
5	2^3	$2^{3.60}$	$2^{1.40}$
6	2^4	$2^{4.25}$	$2^{1.75}$
7	2^4	$2^{5.46}$	$2^{1.54}$
8	2^5	$2^{6.36}$	$2^{1.64}$

- Using 6 CV indices, we can search the whole CV space with $2^{250}.2^{4.25} = 2^{254.25}$ CF calls.
- Probability of success for 1 instance of the attack is approximately 63%.
- For 2 and 3 instances, corresponding probability of success and attack complexities are $(86\%, 2^{255.25})$ and $(95\%, 2^{255.83})$.

Summary

- We have found truncated differentials for the compression function of Hamsi-256 with probability 1.

Summary

- We have found truncated differentials for the compression function of Hamsi-256 with probability 1.
- Message recovery attack with complexity $2^{10.48}$ CF calls..

Summary

- We have found truncated differentials for the compression function of Hamsi-256 with probability 1.
- Message recovery attack with complexity $2^{10.48}$ CF calls..
- Preimage attack with complexity $2^{254.25}$ CF calls.

Summary

- We have found truncated differentials for the compression function of Hamsi-256 with probability 1.
- Message recovery attack with complexity $2^{10.48}$ CF calls..
- Preimage attack with complexity $2^{254.25}$ CF calls.
- 5-round distinguisher with complexity 2^{83}.

Thank You.