Cube-Based Cryptanalysis of Subterranean-SAE

Fukang Liu[1,3], Takanori Isobe[2,3], Willi Meier[4]

[1]East China Normal University, China
[2]NICT, Japan
[3]University of Hyogo, Japan
[4]FHNW, Windisch, Switzerland

Oct. 23, 2020

Background

♢ NIST Lightweight Cryptography Standardization process

- Start: 2013
- Call For Submissions: 2018
- Public (the first round): April 18, 2019
- Number (the first round): 56 candidates
- Public (the second round): Aug. 31, 2019
- Number (the second round): 32 candidates

♢Third-party cryptanalysis is essential

Target

- Subterranean-SAE (the AEAD scheme based on Subterranean 2.0)
- Designers
  - Joan Daemen
  - Pedro Maat Costa Massolino
  - Yann Rotella

Results:

Table: The analytical results of reduced Subterranean-SAE

Attack Type	Blank rounds	Data	Time	Nonce-misuse
State-recovery	arbitrary	1177	2^{16}	Yes
Key-recovery	arbitrary	1177	2^{35}	Yes
Key-recovery	4/8	$2^{69.5}$	2^{122}	No
Distinguisher	4/8	2^{33}	2^{33}	No

Sponge-Based AEAD: Subterranean-SAE

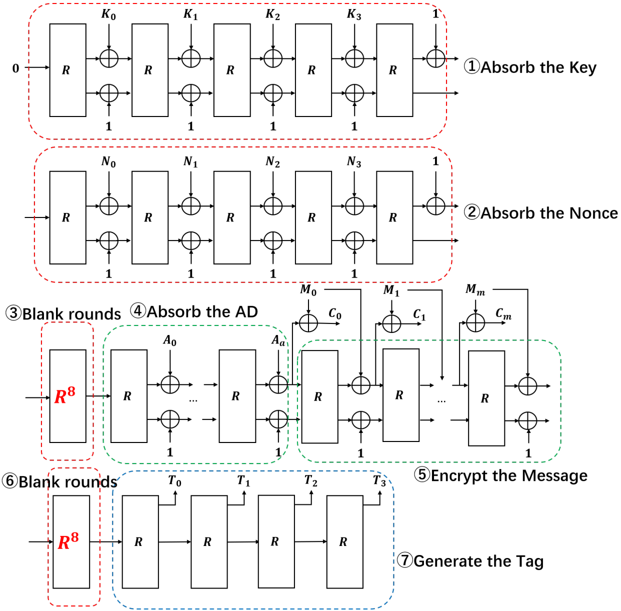
Input:

- 128-bit key (K_0, K_1, K_2, K_3), $K_i \in (F_2^{32})^4$
- 128-bit Nonce (N_0, N_1, N_2, N_3), $N_i \in (F_2^{32})^4$
- Associated data $(A_0, \cdots, A_a)$, $A_i \in F_2^{32}$
- Message $(M_0, \cdots, M_m)$, $M_i \in F_2^{32}$

Output:

- Ciphertext $(C_0, \cdots, C_m)$, $C_i \in F_2^{32}$
- 128-bit Tag (T_0, T_1, T_2, T_3), $T_i \in (F_2^{32})^4$

Sponge-Based AEAD: Subterranean-SAE



Subterranean-SAE: Round Function *R*

The one-round permutation $R = \pi \circ \theta \circ \iota \circ \chi$:

$$\begin{aligned} \chi &: \quad s[i] \leftarrow s[i] \oplus \overline{s[i+1]} s[i+2], \\ \iota &: \quad s[0] \leftarrow s[0] \oplus 1, \\ \theta &: \quad s[i] \leftarrow s[i] \oplus s[i+3] \oplus s[i+8], \\ \pi &: \quad s[i] \leftarrow s[12i], \end{aligned}$$

where $0 \leq i \leq 256$.

Inject the Message

Table: Injected position

i	0	1	2	3	4	5	6	7	8
IN[i]	1	176	136	35	249	134	197	234	64
i	9	10	11	12	13	14	15	16	17
IN[i]	213	223	184	2	95	15	70	241	11
i	18	19	20	21	22	23	24	25	26
IN[i]	137	211	128	169	189	111	4	190	30
i	27	28	29	30	31	32	–	–	–
IN[i]	140	225	22	17	165	256	–	–	–

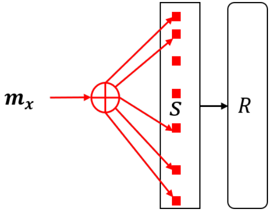


Figure: Inject the message, where $s[\mathrm{IN}[i]] = s[\mathrm{IN}[i]] \oplus m_x[i]$

Extract the State

Table: Extracted position

i	0	1	2	3	4	5	6	7	8
EX[i]	256	81	121	222	8	123	60	23	193
i	9	10	11	12	13	14	15	16	17
EX[i]	44	34	73	255	162	242	187	16	246
i	18	19	20	21	22	23	24	25	26
EX[i]	120	46	129	88	68	146	253	67	227
i	27	28	29	30	31	–	–	–	–
EX[i]	117	32	235	240	92	–	–	–	–

$z = \mathit{extract}(s)$, where $z[i] = s[\mathrm{IN}[i]] \oplus s[\mathrm{EX}[i]]$.

Attack Scenarios

- The state-recovery attack in the nonce-misuse setting

[Subterranean paper] In nonce-misuse scenario or when unwrapping invalid cryptograms returns more information than a simple error, we make no security claims and an attacker may even be able to reconstruct the secret state. Nevertheless we believe that this would probably a non-trivial effort, both in attack complexity as in ingenuity.

- Attacks in the nonce-respecting setting by reducing blank rounds

Reason: the blank rounds in Subterranean-SAE are used to separate the controllable input and output and the designers choose 8 blank rounds.

Simple Properties of the Quadratic Function

Simple properties of $y_i = x_i \oplus \overline{x_{i+1}} x_{i+2}$:

$$\begin{aligned} x_{i+1} = 1 || x_{i+2} = 0 &\rightarrow y_i = x_i \\ x_{i+2} = 1 &\rightarrow y_i = x_i \oplus x_{i+1} \oplus 1 \\ x_{i+1} = 0 &\rightarrow y_i = x_i \oplus x_{i+2} \end{aligned}$$

1. x_i always appears in the expression of y_i.
2. x_{i+1} appears in the expression of y_i only when $x_{i+2} = 1$.
3. x_{i+2} appears in the expression of y_i only when $x_{i+1} = 0$.

Simple Properties of the Quadratic Function

If any of (x_i, x_{i+1}, x_{i+2}) is set as a variable, we have the following simple observations:

1. If x_i is set as a variable, y_i must be linear in this variable.
2. If x_{i+1} is set as a variable, y_i must be linear in it iff $x_{i+2} = 1$. y_i is constant iff $x_{i+2} = 0$.
3. If x_{i+2} is set as a variable, y_i must be linear in it iff $x_{i+1} = 0$. y_i is constant iff $x_{i+1} = 1$.

Break Subterranean-SAE in the Nonce-misuse Setting

The nonce misuse setting:

- The same (nonce, key) can be used to encrypt different messages.

Main idea

Choose a difference in the message blocks and trace its propagation. Recover the secret state bits from the observed propagation in the ciphertext.

Break Subterranean-SAE in the Nonce-misuse Setting

Type-1: Recover the state bits next to the injected postions

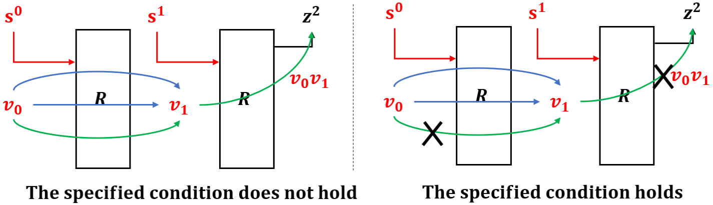
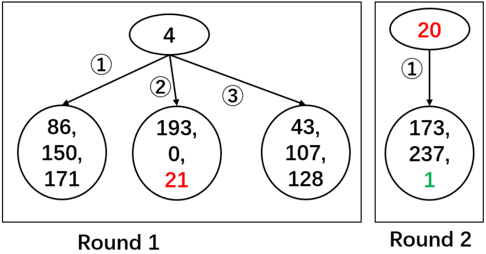


Figure: The type-1 conditional cube tester

Break Subterranean-SAE in Nonce-misuse Setting



①: Core bit
②: zero-condition bit
③: one-condition bit

Cube variables (v_0, v_1) are set at $(s^0[4],\ s^1[22])$.

Figure: An example of type-1 conditional cube tester

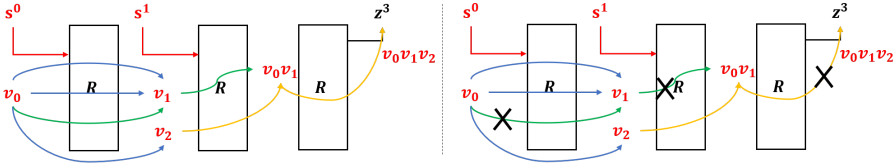
Break Subterranean-SAE in Nonce-misuse Setting

Table: Parameters for Type-1 conditional cube tester

Position of v_0	2	4	11	15	22	64	64	70	95	95	111	128	128	134	136	165	169	197	197	211	213	225	234	241
Position of v_1	213	22	128	128	2	197	111	176	30	137	136	95	140	95	140	184	184	165	17	211	190	189	189	190
Position of condition	3	5	10	16	21	65	63	69	96	94	112	129	127	133	135	166	168	198	196	212	214	226	233	240
Value of condition	0	0	1	0	1	0	1	1	0	1	0	0	1	1	1	0	1	0	1	0	0	0	1	1

Break Subterranean-SAE in the Nonce-misuse Setting

Type-2: Recover more state bits next to the injected postions



The specified condition does not hold

The specified condition holds

Figure: The type-2 conditional cube tester

Difference from Type-1: Choose 2 variables in s^1.

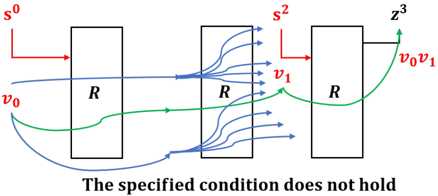
Break Subterranean-SAE in Nonce-misuse Setting

Table: Parameters for Type-2 conditional cube tester

Position of v_0	1	2
Position of (v_1, v_2)	(1,11)	(1,11)
Position of condition	2	1
Value of condition	0	1

Break Subterranean-SAE in the Nonce-misuse Setting

Type-3: Recover more state bits next to the injected postions



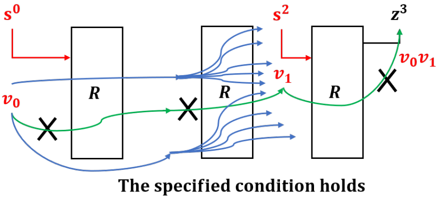
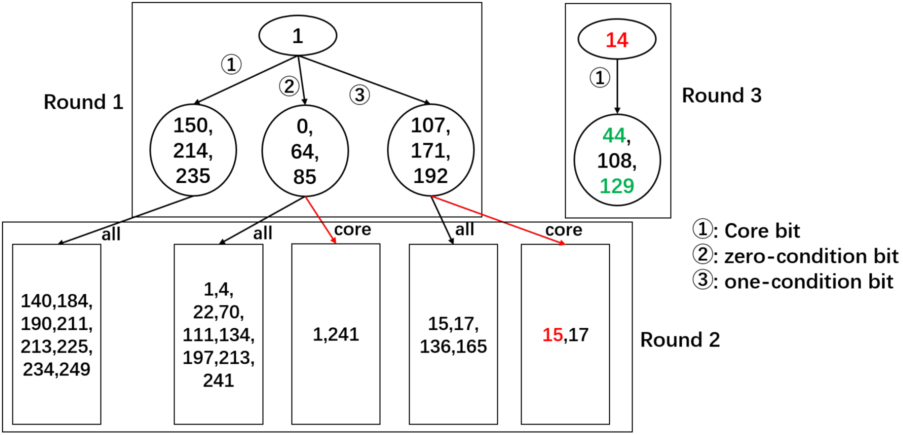


Figure: The type-3 conditional cube tester

Difference from Type-1 and Type-2: Choose variables in (s^0, s^2) rather than (s^0, s^1).

Break Subterranean-SAE in Nonce-misuse Setting



Cube variables (v_0, v_1) are set at $(s^0[1],\ s^2[15])$.

Figure: An example of type-3 conditional cube tester

Break Subterranean-SAE in Nonce-misuse Setting

Table: Parameters for Type-3 conditional cube tester

Position of v_0	1	11	15	17	22	30	30	35	35	70	111	136	137	140	140	165	169	176	176	184	190	211	223	234	241	249	249	–
Position of v_1	15	111	35	35	35	197	11	1	11	140	35	1	1	223	169	11	30	95	211	2	11	70	189	22	2	95	2	–
Position of condition	0	12	14	18	23	31	29	36	34	71	110	137	136	141	139	164	170	177	175	185	191	210	224	235	242	248	250	–
Value of condition	1	0	1	0	0	0	1	0	1	0	1	0	1	0	1	1	0	0	1	0	0	1	0	0	0	1	0	–

Break Subterranean-SAE in the Nonce-misuse Setting

Type-4: Recover more state bits **NOT** next to the injected postions

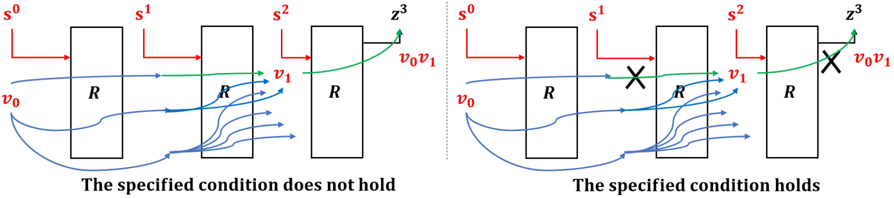


Figure: The type-4 conditional cube tester

Difference from Type-1, Type-2 and Type 3:

1. Recover the state bits in s^1 rather than the state bits in s^0.
2. Recover the state bits not next to the injected positions.

Break Subterranean-SAE in the Nonce-misuse Setting

Table: Parameters for Type-4 conditional cube tester

Position of v_0	1	1	2	4	11	15	15	17	17	22	35
Position of v_1	190	211	136	70	17	165	15	190	111	211	95
Position of condition	213	236	106	85	194	195	193	238	45	217	109
Value of condition	1	0	1	1	0	0	1	0	0	0	1

Position of v_0	35	64	64	70	95	111	111	128	128	136	140
Position of v_1	184	137	70	197	165	165	15	249	190	35	249
Position of condition	173	92	90	49	178	203	201	183	245	160	182
Value of condition	1	0	1	0	1	0	1	0	1	1	1

Position of v_0	165	169	169	184	184	184	189	190	190	197	197
Position of v_1	176	189	234	95	30	184	134	4	70	234	70
Position of condition	77	229	227	102	100	166	79	38	59	251	58
Value of condition	1	0	1	0	1	0	1	0	0	1	1

Position of v_0	213	213	223	225	225	225	234	234	249	249	–
Position of v_1	70	225	197	70	225	184	11	189	70	11	–
Position of condition	83	147	41	82	146	169	149	234	86	148	–
Value of condition	0	0	0	1	1	0	0	0	0	1	–

Break Subterranean-SAE in the Nonce-misuse Setting

Send an encryption query (N, A, M) and obtain (C, T).

The goal is recover the secret state (MS_1^{in}, MS_2^{in}, MS_3^{in}) in this query.

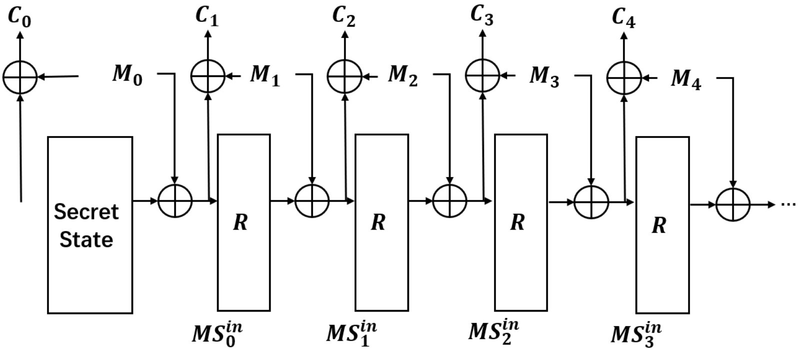


Figure: Encryption phase

Break Subterranean-SAE in the Nonce-misuse Setting

1. Treat MS_0^{in}, MS_1^{in} and MS_2^{in} as s^0, s^1 and s^2 respectively. Recover 43 secret bits of MS_1^{in} using Type-4.
2. The first message block has to be kept the same with that in the very first query. Treat MS_1^{in}, MS_2^{in} and MS_3^{in} as s^0, s^1 and s^2 respectively. Recover 53 extra secret bits of MS_1^{in} and 43 secret bits of MS_2^{in} using Type-1/2/3/4.
3. The first two message blocks have to be kept the same with those in the very first query. Treat MS_2^{in}, MS_3^{in} and MS_4^{in} as s^0, s^1 and s^2 respectively. Recover 53 extra secret bits of MS_2^{in} and 43 secret bits of MS_3^{in} using Type-1/2/3/4.
4. The first three message blocks have to be kept the same with those in the very first query. Treat MS_3^{in}, MS_4^{in} and MS_5^{in} as s^0, s^1 and s^2 respectively. Recover 53 extra secret bits of MS_3^{in} using Type-1/2/3.

Break Subterranean-SAE in the Nonce-misuse Setting

The recovered information:

- 111 secret bits (red) and 16 linear equations of MS_i^{in} $(i = 1, 2, 3)$.

0	1	2	3	4	5	6	7	8	9	10	11	12	13
14	15	16	17	18	19	20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35	36	37	38	39	40	41
42	43	44	45	46	47	48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79	80	81	82	83
84	85	86	87	88	89	90	91	92	93	94	95	96	97
98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122	123	124	125
126	127	128	129	130	131	132	133	134	135	136	137	138	139
140	141	142	143	144	145	146	147	148	149	150	151	152	153
154	155	156	157	158	159	160	161	162	163	164	165	166	167
168	169	170	171	172	173	174	175	176	177	178	179	180	181
182	183	184	185	186	187	188	189	190	191	192	193	194	195
196	197	198	199	200	201	202	203	204	205	206	207	208	209
210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	225	226	227	228	229	230	231	232	233	234	235	236	237
238	239	240	241	242	243	244	245	246	247	248	249	250	251
252	253	254	255	256									

Break Subterranean-SAE in the Nonce-misuse Setting

Recover the remaining unknown state bits via solving equations.

Variables and leaked equations:

1. $257 - 111 = 146$ variables in MS_1^{in}.
2. 16 leaked linear Boolean equations in these 146 variables from MS_1^{in}.
3. $111 + 16 = 127$ leaked (quadratic) Boolean equations in these 146 variables from MS_2^{in}.
4. 51 leaked quadratic Boolean equations in these 146 variables from MS_3^{in} (carefully consider the relations between MS_3^{in} and MS_2^{in}).

Break Subterranean-SAE in the Nonce-misuse Setting

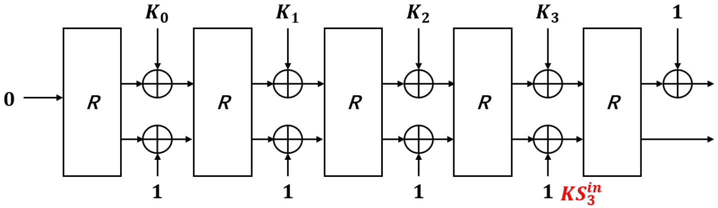
Solve equations:

1. Guess 16 out of the 146 variables (as marked in blue).
2. There will be $146 - 16 = 130$ variables and in total 54 possible quadratic terms in terms of these 130 variables.
3. The number of variables after linearization is $130 + 54 = 184$.
4. The number of equations is $16 + 127 + 51 = 194 > 184$.

Time complexity: 2^{16}

Break Subterranean-SAE in the Nonce-misuse Setting

Recover the secret key:



1. Guess K_0 and 3 bits of K_1 injected at positions $(1, 136, 189)$.
2. Use the $257 - 32 = 225$ known bits of KS_3^{in} to construct the equation system in terms of (K_1, K_2).
3. There are $29 + 32 = 61$ variables and 128 possible quadratic terms $(61 + 128 < 225)$.

Time complexity: 2^{35}

Key-Recovery Attacks in the Nonce-respecting Setting

Main idea

1. Use the degree of freedom of (N_0, N_1, N_2, N_3).
2. Recover some state bits of NS_1^{in}.
3. Guess some key bits and compute other key bits by solving a linear equation system.

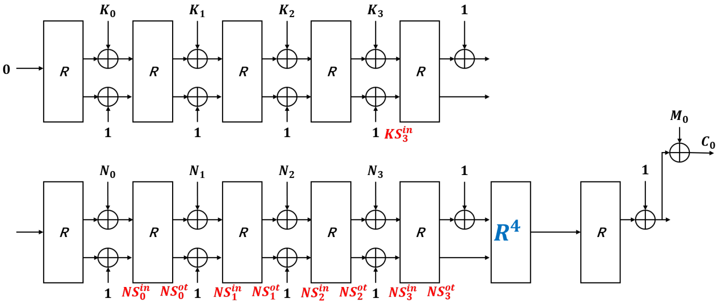
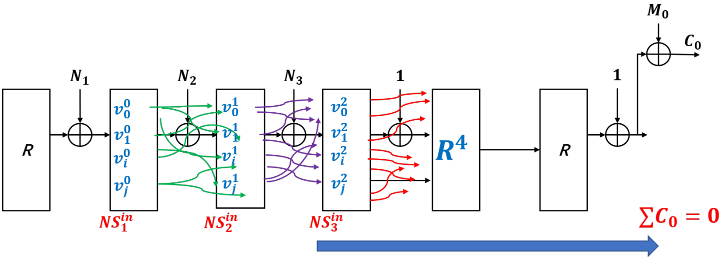


Figure: Subterranean-SAE with 4 blank rounds

Key-Recovery Attacks in the Nonce-respecting Setting


M_0
C_0
N_1
N_2
N_3
1
1
R
v_0^0 v_1^0 v_i^0 v_j^0
v_0^1 v_1^1 v_i^1 v_j^1
v_0^2 v_1^2 v_i^2 v_j^2
R^4
R
NS_1^{in}
NS_2^{in}
NS_3^{in}
$\sum C_0 = 0$

$$|v^0| + |v^1| + |v^2| = 65$$

Degree is at most 64

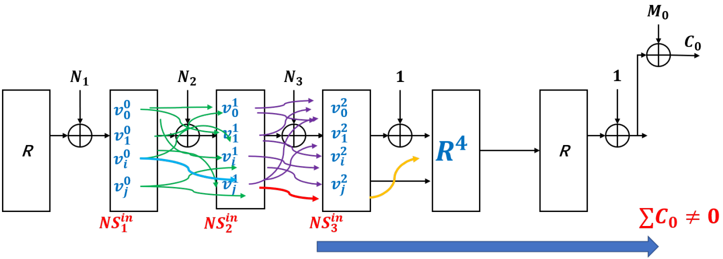
⟶ v^0 will not multiply with each other nor v^1

⟶ v^1, v^0 will not multiply with each other

⟶ **quadratic term**

Figure: The sum of C_0 is zero if a bit of NS_1^{in} takes a specified value

Key-Recovery Attacks in the Nonce-respecting Setting



$|v^0| + |v^1| + |v^2| = 65$

Degree is at most 64

→ v^0 will not multiply with each other nor v^1

→ v^1, v^0 will not multiply with each other

→ v^0 will multiply with v^1

→ **quadratic term**

→ **cubic term**

Figure: The sum of C_0 is nonzero if a bit of NS_1^{in} takes a specified value

Key-Recovery Attacks in the Nonce-respecting Setting

We have recovered 22 secret state bits of NS_0^{ot}.

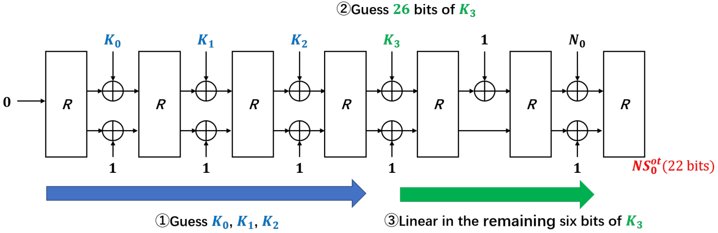


Figure: The procedure to recover the 128-bit key

Guess 122 bits of (K_0, K_1, K_2, K_3)

$\Downarrow$

Construct 22 quadratic equations in terms of the remaining 6 key bits.

$\Downarrow$

Recover the 128-bit key with 2^{122} time complexity (faster than 2^{128})

Distinguishing Attacks in the Nonce-respecting Setting

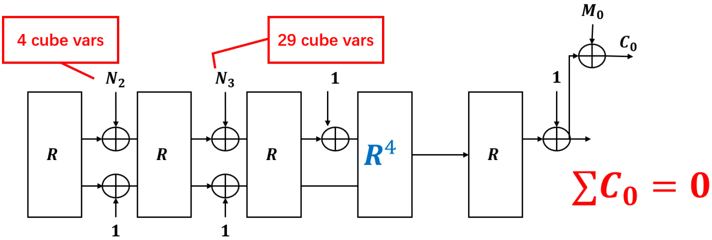


Figure: The sum of C_0 is 0 by properly choosing 33 cube variables

Summary

Table: The analytical results of reduced Subterranean-SAE

Attack Type	Blank rounds	Data	Time	Nonce-misuse
State-recovery	arbitrary	1177	2^{16}	Yes
Key-recovery	arbitrary	1177	2^{35}	Yes
Key-recovery	4/8	$2^{69.5}$	2^{122}	No
Distinguisher	4/8	2^{33}	2^{33}	No

Thank you