Key-dependent cube attack on reduced Frit permutation in Duplex-AE modes

Lingyue Qin[1], Xiaoyang Dong[1], Keting Jia[2*], Rui Zong[1]

[1] Institute for Advanced Study, Tsinghua University, Beijing 100084, China
[2] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
`ktjia@mail.tsinghua.edu.cn`

Abstract. Frit is a new lightweight 384-bit cryptographic permutation proposed by Simon *et al.*, which is designed for resisting fault injection and performs competitively in both hardware and software. Dobraunig *et al.* first studied Frit in EM construction, and left an open problem to explore the security of Frit in a sponge or duplex modes. In this paper, by introducing a new key-dependent cube attack method, we partially answer the open question by Dobraunig *et al.* and give some key-recovery attacks on the rounded-reduced Frit used in duplex authenticated encryption mode (Frit-AE). Our results cover all the versions of Frit-AE and include some practical key-recovery attacks that could recover the key within several minutes.

Keywords: Frit, Duplex authenticated encryption mode, Key-dependent cube attack, Key-recovery, Permutation-based cryptology

1 Introduction

Recently, the permutation-based cryptology becomes a good topic in symmetric-key research groups. On one hand, many dedicated ciphers are permutation-based, including Keccak [1], Keyak [2], Ketje [3], Chaskey [4], Salsa20 [5], Ascon [6] and *et al.* On the other hand, researchers introduced many cryptographic permutations recently, such as Simpira [7], Gimli [8], Xoodoo [9], Frit [10] and *et al.*, whose target is to design one unified cryptographic primitive suitable for many different applications (collision-resistant hashing, preimage-resistant hashing, message authentication, message encryption, etc.). Using these permutations, one could possibly initiate them with Even-Mansour construction [11] to get a block cipher, such as Simpira-EM, Frit-EM. Also, one could use them with Sponge construction [12] to get hash functions, like SHA-3. Another way is to use these permutations with MonkeyDuplex [13] constructions to achieve authenticated encryptions (AE), which is made very popular by CAESAR competition. As far as we know, 4 out of 15 third-round candidates of CAESAR following this strategy to achieve AE, i.e. Keyak [2], Ketje [3], Ascon [6], NORX [14]. Notably, Ascon is selected as one of the finalists.

* Corresponding Author

Frit (Fault-Resistant Iterative Transformation) [10] is a new lightweight 384-bit cryptographic permutation proposed by Simon *et al.* recently. They give a novel approach for designing cryptographic primitives to against fault injection attack, providing a number of lightweight operations for nonlinearity and diffusion. Frit can also be used for designing block ciphers, AE schemes, stream ciphers and MAC functions.

Dobraunig *et al.* [15] first studied the Frit cipher against algebraic attack and gave some key-recovery attacks on Frit in EM constructions, i.e. Frit-EM block cipher. In the end of their paper, they left an open problem that if Frit is used in MonkeyDuplex construction (denoted as Frit-AE, i.e. Frit-based authenticated encryption), what is the security level of Frit-AE against the algebraic attacks, such as cube-like or conditional cube attacks. In this paper, we will focus on this open question.

Our contributions. This paper analyzes the security of the rounded-reduced Frit used in MonkeyDuplex authenticated encryption mode (Frit-AE) against cube-like attack. We first give the brief description that the possible implementations of Frit with MonkeyDuplex. Similar to Ketje [3] and Ascon [6], shown in Figure 2, we place the 16-round Frit in the initialization phase, whose input is a 384-bit concatenation of 128-bit key (one limb) and 256-bit nonce (two limbs). Then, a 128-bit limb is XORed with 128-bit plaintext and output the 128-bit ciphertext. Since there are three limbs (a, b, c) in the state of Frit, nine possible versions for the initialization phase with different limb positions of the 128-bit key and 128-bit ciphertext. We denote them as Frit$_\alpha^\beta$-AE, where $\alpha, \beta \in \{a, b, c\}$ indicate the limb positions of 128-bit key and 128-bit ciphertext, respectively. For detailed information, please refer to Sect. 2.

At EUROCRYPT 2017, Huang *et al.* [16] introduced the conditional cube attacks on Keccak sponge function [1]. Then, several cube-like attacks [17–20] were proposed on permutation based AE schemes, i.e. Ketje, Keyak, Ascon. By exploring bit conditions, which are related to both public bits and key bits, they could reduce the diffusion of cube variables and construct cube testers for Keccak. In this paper, we introduce a new key-dependent cube attack on Frit$_\alpha^\beta$-AE. Similar to conditional cube attacks [16], the key-dependent cube attack also exploit cube testers with constraints. However, the difference with conditional cube attack is that, the new attacks do not require the conditions to be dependent on public bits. Actually, the idea of assigning (dynamic) constraints to public variables and using them to recover key bits was earlier appeared in conditional differential attacks, which was introduced by Knellwolf, Meier and Naya-Plasencia at ASIACRYPT 2010 [21]. The authors classified the conditions into three types:

- Type 0 conditions only involve public bits;
- Type 1 conditions involve both public bits and secret bits;
- Type 2 conditions only involve secret bits.

The key-dependent cube attack only considers the type 2 conditions, that only involve secret key bits. In our attacks on Frit$_\alpha^\beta$-AE, we find many different cube testers for different key-dependent bit conditions with the help of MILP method. So we could detect many key-dependent equations by exploring different cube testers. Based on this idea, we give some round-reduced attacks on all nine versions of Frit$_\alpha^\beta$-AE. The attacked rounds vary from 8 to 12, which are summarised in Table 1.

Table 1. Summary of cryptanalysis results

α	β	Attacked Round	Time Complexity	Reference
a	a	9	2^{29}	Sect. 6.1
	b	10	2^{29}	
	c	9	2^{29}	
b	a	8	2^{29}	Sect. 5
		9	2^{42}	
		10	2^{63}	
		11	2^{97}	
	b	9	2^{29}	
		10	2^{42}	
		11	2^{63}	
		12	2^{97}	
	c	8	2^{29}	
		9	2^{42}	
		10	2^{63}	
		11	2^{97}	
c	a	10	2^{29}	Sect. 6.2
	b	11	2^{29}	
	c	10	2^{29}	

We also give practical implementations of 9-round attack on Frit$_b^b$-AE in 7 minutes to recover 128-bit key and 10-round Frit$_b^b$-AE in 8 hours to recover 1-bit key. For Frit$_a^b$-AE and Frit$_c^b$-AE, 10-round and 11-round attacks are implemented to recover 128-bit key in 8 minutes. The success rate is 100% corresponding to our analysis, which proves our algorithm is effective. The test code is given in https://github.com/qly14/FritAE.git.

2 Frit

This section gives the used notations in the paper, a brief description of Frit, and the Frit used in duplex authenticated encryption mode.

2.1 Notations

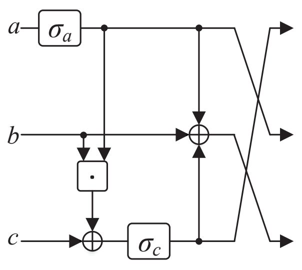
a, b, c	three limbs in $\{0,1\}^{128}$
a_r, b_r, c_r	the three limbs after r-round
K	the 128-bit secret key
v, v'	two 128-bit variable vectors
RC_i	the i-th round constant,$0 \le i \le 15$
v_i, v'_i	the i-th variable vectors of v, v', $0 \le i \le 127$
K_i	the i-th bit of key, $0 \le i \le 127$
$\oplus$	128-bit bitwise XOR
$\odot$	128-bit bitwise AND
$a \lll i$	cycle shift of a to the left by i bits

2.2 The Frit permutation

Frit is a 384-bit cryptographic permutation proposed by Simon *et al.*, which operates on a state of three limbs a, b, c in $\{0,1\}^{128}$ updated in 16-round. Each round the state is updated in 6 bitwise operations: the round constant addition, a mixing operation of limb a, the only nonlinear operation $\odot$ used as a Toffoli gate, a mixing operation of limb c, a switch operation and a transposition. The details are illustrated in Algorithm 1.

Algorithm 1 Frit

```
Input: $a, b, c \in \{0,1\}^{128}$
  for each $i \in [0, 15]$ do
    $c \leftarrow c \oplus RC_i$
    $a \leftarrow a \oplus (a \lll 110) \oplus (a \lll 87)$
    $c \leftarrow c \oplus (a \odot b)$
    $c \leftarrow c \oplus (c \lll 118) \oplus (c \lll 88)$
    $b \leftarrow a \oplus b \oplus c$
    $(a, b, c) \leftarrow (c, a, b)$
  end for
  return $(a, b, c)$
```



Round Constants. The master round constant is generated by the primitive polynomial $X^5 + X^2 + 1$ with the initial states $(1,1,1,1,1)$. Choosing the first 32 bits of the sequence as the master round constant, for the i-th round RC_i is obtained by shifting the master round constant to the left by i bits $(0 \le i \le 15)$.

Mixing operation. The two mixing steps are denoted as $\sigma_a(a) = a \oplus (a \lll 110) \oplus (a \lll 87)$ and $\sigma_c(c) = c \oplus (c \lll 118) \oplus (c \lll 88)$. We refer to the inverses of σ_a, σ_c as $\sigma_a^{-1}, \sigma_c^{-1}$, which are similar rotation-invariants but need more operations.

2.3 Frit used in duplex authenticated encryption mode

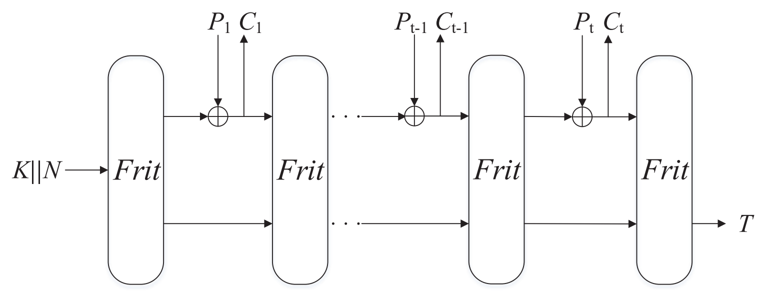


Fig. 1. Frit$_\alpha^\beta$-AE

Similar to Ketje [3] and Ascon [6], we use Frit to design authenticated encryption by using the duplex authenticated encryption mode [13] as shown in Figure 1. We denote it as Frit-AE. Our attack target is the initialization phase of Frit-AE, as shown in Figure 2. In the initialization phase, the input of 16-round Frit is a 384-bit concatenation of 128-bit key (one limb) and 256-bit nonce (two limbs). Then, a 128-bit limb is XORed with 128-bit plaintext and output the 128-bit ciphertext. Since there are three limbs (a, b, c) in the state of Frit, nine possible versions for the initialization phase with different limb positions of the 128-bit key and 128-bit ciphertext. We denote them as Frit$_\alpha^\beta$-AE, where $\alpha, \beta \in \{a, b, c\}$ indicate the limb positions of 128-bit key and 128-bit ciphertext, respectively.

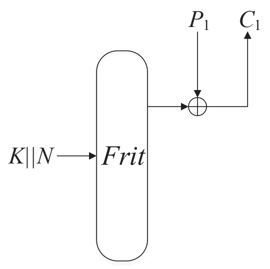


Fig. 2. Initialization phase of Frit-AE

3 Related Work

3.1 Cube attack

The cube attack was proposed by Dinur and Shamir in EUROCRYPT2009 [22]. The output bit of a cryptographic scheme can be denoted as a polynomial $f(k_0, \cdots, k_{n-1}, v_0, \cdots, v_{m-1})$ over $GF(2)$, where $\{k_0, \cdots, k_{n-1}\}$ are the secret variables(the key bits) and $\{v_0, \cdots, v_{m-1}\}$ are the public variables(the IV or nonce bits). We review the basic idea of [22] as Theorem 1.

Theorem 1.

$$f(k_0, \cdots, k_{n-1}, v_0, \cdots, v_{m-1}) = T \cdot P + Q(k_0, \cdots, k_{n-1}, v_0, \cdots, v_{m-1})$$

T is a monomial which is actually the product of some public variables $\{v_0, \cdots, v_{s-1}\}$ $(1 \leq s \leq m)$, denoted as cube C_T. None of the monomials in Q is divisible by T. P is called superpoly, which does not involve any variables of C_T. Then the sum of f over all values of the cube C_T(cube sum) is

$$\sum_{v'=(v_0,\cdots,v_{s-1})\in C_T} f(k_0, \cdots, k_{n-1}, v', v_0, \cdots, v_{m-1}) = P$$

where C_T contains all binary vectors of the length s, and $\{v_s, \cdots, v_{m-1}\}$ are fixed to constant.

The basic idea is to find enough T whose P is linear and not a constant, so as to recover key through solving a system of linear equations.

3.2 Dynamic Cube attack

Dynamic cube attack was introduced by Dinur and Shamir in FES2011 [23]. The basic idea is to simply a complex polynomial P:$P = P_1P_2 + P_3$ to the simple P_3. The P_1 contains a linear public term called a dynamic variable, which can be 0 if the dynamic variable is assigned with a function of some secret variables and cube variables. Thus P is simplified to a lower degree. The right guess of key bits in dynamic variable will lead to zero cube sums of P with high probability, otherwise the cube sums will be random.

3.3 Conditional Differential Cryptanalysis

Knellwolf, Meier and Naya-Plasenciaa [21] applied conditional differential characteristic to NFSR-based constructions and extended to higher order differential attacks at ASIACRYPT 2010. The input of a synchronous stream cipher is an IV and a key. Suppose that the keystream for many chosen IVs under the same secret key can be observed. By imposing specific conditions on certain bits of the IV, the attacker can control the propagation of a difference through the first few-round of the initialization process. Taking IV pairs conformed to these conditions as input, the resulting keystream differences will present a bias. Additionally, conditions upon key define classes of weak keys. The authors classified the conditions into three types:

– Type 0 conditions only involve public bits;
– Type 1 conditions involve both public bits and secret bits;
– Type 2 conditions only involve secret bits.

3.4 Conditional Cube Attack

Conditional cube attack [16] was proposed by Huang *et al.* at EUROCRYPT 2017 to attack Keccak keyed mode. Inspired by dynamic cube attack [22], which reduces the degree of output polynomials of cube variables by adding some bit conditions on the initial value (IV), they reduce the degree by appending key bit conditions. The conditions used by Huang et al. are the Type 1 conditions from Sect. 3.3, which involve both public bits and secret bits.

4 Key-dependent cube attack

Different from conditional cube attack, the key-dependent cube attack do not require the conditions to be dependent on public bits, it only involves Type 2 conditions. In duplex authenticated encryption mode, such as Ketje, Ascon and Frit$_\alpha^\beta$-AE, the initialization phase produces l-bit output. Each of the output bits is written as a polynomial $f_i(k_0,...,k_{n-1},v_0,...,v_{m-1})$, $i=0,1,...,l-1$. Choose a common cube C_T, e.g $(v_0,...,v_{s-1})$, $1\leq s\leq m$, then $f_i=T\cdot P_i+Q_i$, $i=0,1,...,l-1$. In our key-dependent cube attack, a common divisor of P_i is found, which is a polynomial $g(k_0,...,k_{n-1})$ that only involved some key bits. The cube sum of f_i over all values of the cube C_T is $P_i=g(k_0,...,k_{n-1})\cdot P_i'.$. Then the Corollary 1 is given.

Corollary 1. *Given a series of polynomials f_i $(i\in\{0,1,...,l-1\})$:$\{0,1\}^n\rightarrow\{0,1\}$.*

$$\begin{cases} f_0(k_0,...,k_{n-1},v_0,...,v_{m-1})=T\cdot g(k_0,...,k_{n-1})\cdot P_0'+Q_0\\ f_1(k_0,...,k_{n-1},v_0,...,v_{m-1})=T\cdot g(k_0,...,k_{n-1})\cdot P_1'+Q_1\\ ...\\ f_{l-1}(k_0,...,k_{n-1},v_0,...,v_{m-1})=T\cdot g(k_0,...,k_{n-1})\cdot P_{l-1}'+Q_{l-1}\end{cases}\tag{1}$$

where none of the monomials in $Q_i(x)$ is divisible by T. Then the sums of f_i $(i\in\{0,1,...,l-1\})$ over all values of the cube (cube sum) are

$$\begin{cases} \sum_{v'\in C_T} f_0(k_0,...,k_{n-1},v',v_s,...,v_{m-1})=g(k_0,...,k_{n-1})\cdot P_0'\\ \sum_{v'\in C_T} f_1(k_0,...,k_{n-1},v',v_s,...,v_{m-1})=g(k_0,...,k_{n-1})\cdot P_1'\\ ...\\ \sum_{v'\in C_T} f_{l-1}(k_0,...,k_{n-1},v',v_s,...,v_{m-1})=g(k_0,...,k_{n-1})\cdot P_{l-1}'\end{cases}\tag{2}$$

where the C_T contains all binary vectors of the length s, other public variables $v_j, j\in\{s,s+1,...,m-1\}$ are constants.

The following Property 1 is easy to get.

Property 1 *If $g = 0$, cube sums of f_i ($i \in \{0, 1, ..., l-1\}$) will be all 0 with probability 1.*

Assumption 1 *If $g = 1$, cube sums of f_i ($i \in \{0, 1, ..., l-1\}$) will be determined by P'_i ($i \in \{0, 1, ..., l-1\}$), the cube sums of f_i ($i \in \{0, 1, ..., l-1\}$) all equal to 0 with probability about 2^{-l} if f_i ($i \in \{0, 1, ..., l-1\}$) is a random oracle.*

According to Property 1 and Assumption 1, we introduce the cube tester, which has the Property 2 and Assumption 2.

Property 2 *If at least one nonzero cube sum occurs among the cube sums of f_i ($i \in \{0, 1, ..., l-1\}$), we will determine that $g = 1$. It is guaranteed to be right.*

Assumption 2 *If the cube sums of f_i ($i \in \{0, 1, ..., l-1\}$) all equal to 0, we will determine that $g = 0$. Note that, in a random oracle, $g = 0$ is wrong with probability of 2^{-l}, because P'_i is zero with probability of about $\frac{1}{2}$.*

In our paper, with the help of MILP method, we could find many different key-dependent gs corresponding to different cubes, which are all linear with key bits. At last, we could recover the full key by solving a set of linear equations on key bits.

5 Key-dependent cube attack on Frit$_b^\beta$-AE

In this section, we first review the algebraic property of Frit analyzed in [10,15]. Then according to our observation of some properties, we give key-dependent cube attack on three versions of rounded-reduced Frit$_b^\beta$-AE.

5.1 Algebraic property of Frit

The only nonlinear operation of Frit is a bitwise $\odot$, so the round function's degree is 2. Let Frit$_r$ denote the r-round Frit and

$$(a_r, b_r, c_r) = \text{Frit}_1(a_{r-1}, b_{r-1}, c_{r-1}) = \text{Frit}_r(a_0, b_0, c_0).$$

We obtain the following properties:

$$\begin{aligned}
&\deg a_r \leq max(\deg c_{r-1}, \deg a_{r-1} + \deg b_{r-1}),\\
&\deg c_r \leq max(\deg a_r, \deg b_{r-1}, \deg c_{r-1}),\\
&\deg b_r = \deg a_{r-1}.
\end{aligned}$$

Setting the $\deg a_0 = \deg b_0 = \deg c_0 = 1$, we can observe that the degrees of a_r, b_r, c_r follow the Fibonacci sequence $F_r = F_{r-1} + F_{r-2}$ ($F_0 = 0, F_1 = 1$). By induction that $\deg a_{r-1} \le F_{r+1}$, $\deg b_{r-1} \le F_r$ and $\deg c_{r-1} \le F_{r+1}$, we deduce that

$$\begin{aligned}
&\deg a_r \le \deg a_{r-1} + \deg b_{r-1} \le F_{r+1} + F_r = F_{r+2},\\
&\deg c_r \le \deg a_r \le F_{r+2},\\
&\deg b_r = \deg a_{r-1} \le F_{r+1}.
\end{aligned}$$

The degrees of limbs a_r, b_r, c_r for first 10-round are listed in Table 2.

Table 2. Degrees of limbs a_r, b_r, c_r

r	0	1	2	3	4	5	6	7	8	9	10
$\deg a_r$	1	2	3	5	8	13	21	34	55	89	144
$\deg b_r$	1	1	2	3	5	8	13	21	34	55	89
$\deg c_r$	1	2	3	5	8	13	21	34	55	89	144

5.2 New attacks on Frit$_b^\beta$-AE

Consider the 128-bit key K putting in limb b_0 as Figure 3, which is denoted as Frit$_b^\beta$-AE. Then the 256-bit nonce can be put in limbs a_0 and c_0. It is easy to find that the output expressions of a_2, b_2, c_2 of 2-round Frit are linear if we keep the limb a_0 to constants and set variables to limb c_0.

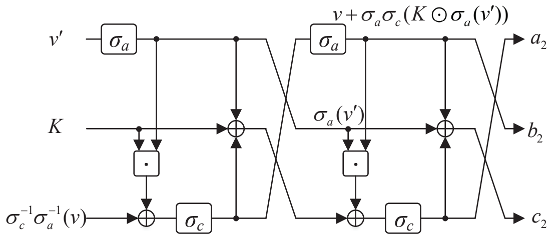


Fig. 3. 2-round initial structure of Frit$_b^\beta$-AE

Set variable vector v' to limb a_0 and $\sigma_c^{-1}\sigma_a^{-1}(v)$ to limb c_0. It is clear that the expressions of a_1,b_1,c_1 of 1-round Frit are linear, and $b_2 = v + \sigma_a\sigma_c(K \odot \sigma_a(v'))$ is linear too. To linearize a_2 and c_2, we need to keep that the expression $b_1 \odot b_2 = \sigma_a(v') \odot v + \sigma_a(v') \odot \sigma_a\sigma_c(K \odot \sigma_a(v'))$ doesn't have quadratic terms. That is,

1. For expression $\sigma_a(v') \odot v$, we need to keep that each v_i $(0 \le i \le 127)$ is not multiplied by v'_j $(0 \le j \le 127)$ after mixing operation σ_a. So if v'_j is

chosen as a cube variable, variables v_j, $v_{(j+18)\%128}$ and $v_{(j+41)\%128}$ need to be constants due to the diffusion property of σ_a.
2. For expression $\sigma_a(v') \odot \sigma_a\sigma_c(K \odot \sigma_a(v'))$, the quadratic term $g_{i,j}(K)v'_iv'_j(i \neq j)$ depends on some relative bits of K. For a certain K, if all $g_{i,j}(K) = 0$, the expression is linear. In the attack procedure, we can set some v'_is to constants to reduce the num of bit conditions $g_{i,j}(K)$.
3. By carefully choosing some variables v_i and v'_j and setting others to constants, we ensure that there are no quadratic terms $v_iv'_j$ in $b_1 \odot b_2$. For all the quadratic terms $g_{i,j}(K)v'_iv'_j$: if $g_{i,j}(K) = 0$ or at least one of v'_i, v'_j is constant, the degree of a_2, c_2 is 1; otherwise the degree is 2.

According to the above observation, assigning variables v_i,$v_{(i+18)\%128}$,$v_{(i+41)\%128}$, $v_{(i+1)\%128}$,$v_{(i+19)\%128}$, $v_{(i+42)\%128}$ and $v'_i(0 \leq i \leq 127)$ to constants except for v'_i and $v'_{(i+1)\%128}$, the only quadratic term of a_2, c_2 is $K_{(i+1)\%128}v'_iv'_{(i+1)\%128}$. Adding r-round after the 2-round initial structure of Figure 3, we try to attack the $(r + 2)$-round Frit$_b^b$-AE as an example. We choose v'_i, $v'_{(i+1)\%128}$ and other $F_{r+1} - 1$ variables in v as a $(F_{r+1} + 1)$-dimension cube C_i. If $K_{(i+1)\%128} = 0$, the expressions of a_2, b_2 and c_2 are linear, and the degree of b_{r+2} is F_{r+1} according to Table 2. If $K_{(i+1)\%128} = 1$, the expressions of a_2 and c_2 have only one quadratic term $K_{(i+1)\%128}v'_iv'_{(i+1)\%128}$. According to our experimental attacks on 9-round Frit$_b^b$-AE in Sect. 5.3, the expression of b_{r+2} has terms of degree $F_{r+1} + 1$, which must involve $K_{(i+1)\%128}v'_iv'_{(i+1)\%128}$. By calculating the sums of all bit positions of the output limb after $(r + 2)$-round Frit over all values of the cube C_i (cube sum), we can recover the value of $K_{(i+1)\%128}$: if the cube sums of all bit positions of the output limb are 0, $K_{(i+1)\%128} = 0$; otherwise $K_{(i+1)\%128} = 1$. For Frit$_b^a$-AE and Frit$_b^c$-AE, we can test the terms of degree $F_{r+2} + 1$ to recover the key. The key-dependent attack on $r + 2$-round Frit$_b^\beta$-AE is concluded as follows:

1. First set the cube's dimension $d = F_{r+1} + 1$ $(\beta = b)$ or $F_{r+2} + 1$ $(\beta = a, c)$ and cube variables set $C_i = \{v'_i, v'_{(i+1)\%128}, v_{j_0}, \cdots, v_{j_{d-3}}\}$, where set $\{j_0, \cdots, j_{d-3}\}$ doesn't have any elements of $\{i, (i+18)\%128, (i+41)\%128, (i+1)\%128, (i+19)\%128, (i+42)\%128\}$.
2. Assign the other variables of v, v' except for the cube C_i to constants 0 and calculate the cube sums of the whole 128 bit positions of the output limb after $r + 2$-round Frit over all values of the cube C_i. If all the 128 cube sums are 0, we take the $K_{(i+1)\%128}$ as 0, otherwise $K_{(i+1)\%128} = 1$.
3. The time complexity of recovering 1-bit key is 2^d, and the time to get the whole 128-bit key is $2^d \times 2^7 = 2^{7+d}$ by traversing i from 0 to 127.

According to Table 2, we can apply key-dependent attack to no more than 12-round Frit$_b^b$-AE, 11-round Frit$_b^a$-AE and Frit$_b^c$-AE. We give the experiments on 9-round Frit$_b^b$-AE and 10-round Frit$_b^b$-AE with time complexity 2^{29} and 2^{42}. Then the cube variables for attacking 11-round Frit$_b^b$-AE and 12-round Frit$_b^b$-AE with time complexity 2^{63} and 2^{97} are given in Table 6 and Table 7.

5.3 Experiments on 9-round Frit$_b^b$-AE

We do experiments on the 9-round Frit$_b^b$-AE to verity our attack results. Using the 2-round initial structure in Figure 3, we can use a (F_8+1)-dimension (22-dimension) cube to recover 1-bit K. According to the attack procedure in Sect. 5.2, the cube variables for recovering K_1 are listed in Table 3. To recover K_i $(0 \leq i \leq 127)$, the cube variables needed are the variables in Table 3 by adding $i-1$ to the indexes in $GF(2^7)$. We give several examples of the recovered 1-bit key and corresponding 128-bit cube sums for some random keys in Table 4, using the cube variables in Table 3. The details of the experiments refer to https://github.com/qly14/FritAE.git.

Table 3. Cube variables of 9-round Frit$_b^b$-AE

Key	Deg	Cube variables
K_1	22	$v_0', v_1', v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_{10}, v_{11}, v_{12}$ $v_{13}, v_{14}, v_{15}, v_{16}, v_{17}, v_{20}, v_{21}, v_{22}, v_{23}, v_{24}$

We test about 100 random keys, and the success rate of recovering the whole 128-bit key is 100%. The time complexity of our attack on 9-round Frit$_b^b$-AE is 2^{29}, which only needs about 7 minutes on a personal computer.

Experiments on 8-round Frit$_b^a$-AE and Frit$_b^c$-AE. Due to the property that $a_8 = \sigma_a^{-1}(b_9)$ and $c_8 = a_8 + a_7 + b_7$, the terms of degree 22 that we test in b_9 are also existed in the expressions of a_8 and c_8. So using the same cube as Table 3, we can make key-recovery attack on 8-round Frit$_b^a$-AE and Frit$_b^c$-AE. The only difference is that the cube sums are calculated by all bit positions of limb a_8 or c_8. So with time complexity 2^{29}, we can make key-recovery attack on 8-round Frit$_b^a$-AE and Frit$_b^c$-AE with success rate 100%.

Table 4. Experimental results of 9-round Frit$_b^b$-AE

1-bit key	128-bit random key	Cube sums
$K_1 = 0$	0x1c93b7ae 81cf5ca8 644a0463 0c41db9e	0x00000000 00000000 00000000 00000000
$K_1 = 1$	0xe58ec52a 3b3fccf2 17d04d42 4618e031	0x0800c010 20000040 00000000 00802020
$K_2 = 0$	0x05ab60a7 fe41288e 69983eed 4ae9fe4c	0x00000000 00000000 00000000 00000000
$K_2 = 1$	0xe96f359e 26ace184 1565c5cb 0fe1b095	0x04006008 10000020 00000000 00401010
$K_3 = 0$	0x8047f929 e59445dc 0d13ea46 60acb0ec	0x00000000 00000000 00000000 00000000
$K_3 = 1$	0xb3e808b5 a9094cb4 1064fa84 339eac56	0x02003004 08000010 00000000 00200808

5.4 Experiments on 10-round Frit$_b^b$-AE

Adding 8-round Frit after the 2-round initial structure, we can attack 10-round Frit$_b^b$-AE using the (F_9+1)-dimension (35-dimension) cube. Similar to the attack on 9-round Frit$_b^b$-AE, we give the cube variables for recovering the K_1 of the 10-round Frit$_b^b$-AE in Table 5.

Table 5. Cube variables of 10-round Frit$_b^b$-AE to recover K_1

Key	Deg	Cube variables
K_1	35	$v_0', v_1', v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}, v_{16}, v_{17}$ $v_{20}, v_{21}, v_{22}, v_{23}, v_{24}, v_{25}, v_{26}, v_{27}, v_{28}, v_{30}, v_{31}, v_{32}, v_{33}, v_{34}, v_{35}, v_{36}, v_{37}$

The time complexity is 2^{35} for recovering 1-bit key and 2^{42} for all 128-bit key. Limited to the personal computer power, we only try to recover K_1 for a certain key as an example. The success rate of testing 10 random keys is 100%, and recovering each 1-bit key needs about 8 hours. We notice that the same cube can be used to attack 9-round Frit$_b^a$-AE and Frit$_b^c$-AE. The time complexity and success rate is same with the case of 10-round Frit$_b^b$-AE.

5.5 Attack on 11-round Frit$_b^b$-AE

Using the 2-round initial structure we can choose the 56-dimension cube to attack the 11-round Frit$_b^b$-AE. The time complexity of recovering 128-bit key is $2^{56} \times 2^7 = 2^{63}$. The cube variables to recover K_1 for 11-round Frit$_b^b$-AE are given in Table 6. We can apply same attack procedure to 10-round Frit$_b^a$-AE and 10-round Frit$_b^c$-AE with complexity 2^{63}.

Table 6. Cube variables of 11-round Frit$_b^b$-AE to recover K_1

Key	Deg	Cube variables
K_1	56	$v_0', v_1', v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}, v_{16}, v_{17}, v_{20}, v_{21}$ $v_{22}, v_{23}, v_{24}, v_{25}, v_{26}, v_{27}, v_{28}, v_{29}, v_{30}, v_{31}, v_{32}, v_{33}, v_{34}, v_{35}, v_{36}, v_{37}, v_{38}, v_{39},$ $v_{43}, v_{44}, v_{45}, v_{46}, v_{47}, v_{48}, v_{49}, v_{50}, v_{51}, v_{52}, v_{53}, v_{54}, v_{56}, v_{57}, v_{59}, v_{60}, v_{61}, v_{62}$

5.6 Attack on 12-round Frit$_b^b$-AE

Similar to the previous attack, the 90-dimension cube can be used to attack 12-round Frit$_b^b$-AE, 11-round Frit$_b^a$-AE and 11-round Frit$_b^c$-AE with complexity $2^{90} \times 2^7 = 2^{97}$. The cube variables to recover K_1 for 12-round Frit$_b^b$-AE are given in Table 7 as an example.

Table 7. Cube variables of 12-round Frit$_b^b$-AE to recover K_1

Key	Deg	Cube variables
K_1	90	$v_0', v_1', v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}, v_{14}, v_{15}, v_{16}, v_{17}, v_{20}, v_{21}$ $v_{22}, v_{23}, v_{24}, v_{25}, v_{26}, v_{27}, v_{28}, v_{29}, v_{30}, v_{31}, v_{32}, v_{33}, v_{34}, v_{35}, v_{36}, v_{37}, v_{38}, v_{39},$ $v_{40}, v_{43}, v_{44}, v_{45}, v_{46}, v_{47}, v_{48}, v_{49}, v_{50}, v_{51}, v_{52}, v_{53}, v_{54}, v_{56}, v_{57}, v_{58}, v_{59}, v_{60}$ $v_{61}, v_{62}, v_{63}, v_{64}, v_{65}, v_{66}, v_{67}, v_{68}, v_{69}, v_{70}, v_{71}, v_{72}, v_{73}, v_{74}, v_{75}, v_{76}, v_{77}, v_{78}$ $v_{79}, v_{80}, v_{81}, v_{82}, v_{83}, v_{84}, v_{85}, v_{86}, v_{87}, v_{88}, v_{89}, v_{90}, v_{91}, v_{92}, v_{93}, v_{94}$

6 Key-dependent cube attack on Frit$_a^\beta$-AE and Frit$_c^\beta$-AE

In this section, we discuss the key-dependent cube attack on Frit$_a^\beta$-AE and Frit$_c^\beta$-AE.

6.1 New attacks on Frit$_a^\beta$-AE

The cipher Frit$_a^\beta$-AE sets the 128-bit key K to limb a_0 as Figure 4 and the 256-bit nonce to limbs b_0 and c_0. We give a 3-round initial structure by keeping the limb b_0 to constants 0 and setting $\sigma_c^{-1}\sigma_a^{-1}(v)$ to limb c_0. After 2-round Frit the output expressions of a_2, b_2, c_2 are linear with v. To linearize the output expressions of a_3, c_3, the expression $b_2 \odot b_3 = \sigma_a\sigma_c(\sigma_a(K) \odot v + \sigma_a^{-1}(v)) \odot v$ should not involve quadratic terms.

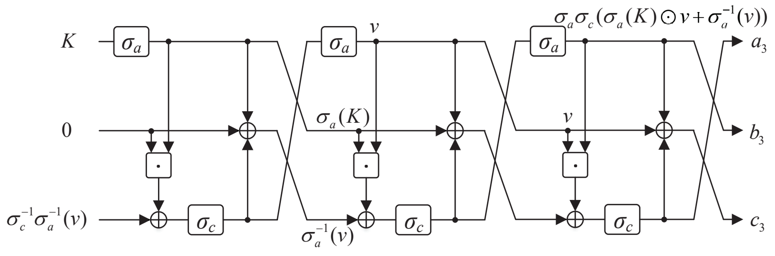


Fig. 4. 3-round initial structure of Frit$_a^\beta$-AE

We notice that the mixing operation σ_a^{-1} is much more complicated than σ_a, where σ_a^{-1} has 65 rotations but σ_a only has 3 rotations. Both the mixing operation σ_a and σ_c can be regarded as cyclic matrices, which are also commutative matrices. So it is clear that

$$\begin{aligned}\sigma_a\sigma_c(\sigma_a(K) \odot v + \sigma_a^{-1}(v)) \odot v &= \sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_a\sigma_c(\sigma_a^{-1}(v)) \odot v \\ &= \sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_c\sigma_a(\sigma_a^{-1}(v)) \odot v \\ &= \sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_c(v) \odot v.\end{aligned}$$

Without the complicated mixing operation σ_a^{-1}, it's easier to guarantee there are no quadratic terms in $b_2 \odot b_3 = \sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_c(v) \odot v$. We use the MILP(mixed-integer linear programming) to solve the problem of finding variables of v which don't multiply with each other as many as possible. The successful applications of MILP involve counting active Sboxes of word-based block ciphers introduced by Mouha *et al.* [24] and searching differential and linear trails introduced by Sun *et al.* [25], etc. Then Li *et al.* give a new MILP model to improve the key-recovery attack on Keccak [18]. In our MILP model, each variable v_i $(i \in [0, 127])$ is assigned with a variable $x_i \in \{0, 1\}$. Then the case $x_i = 1$ represents that v_i can be chosen as a cube variables candidate. We generate the constraints set F of $\{x_i\}$ to guarantee there are no quadratic terms in a_3, c_3 as Algorithm 2. For each term v_iv_j in expression $\sigma_a\sigma_c(\sigma_a(K)\odot v)\odot v+\sigma_c(v)\odot v$, if the coefficient $g_{i,j}(K)$ of $v_iv_j(i \neq j)$ is not 0, we add a constraint $x_i + x_j \leq 1$ to F. (Notice that the coefficient of $v_iv_j(i \neq j)$ can not be constant 1.)

Algorithm 2 Generating Constraints on v to linearize a_3, c_3

```
Input: Variables set $v = \{v_i\}$ $(i \in [0, 127])$
Output: A set $F$ of constraints
  $F = \emptyset$
  $Exp = \sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_c(v) \odot v$
  for each $i \in [0, 127]$ do
    for each $j \in [i + 1, 127]$ do
      if $g_{i,j}(K)v_iv_j \in Exp$ and $g_{i,j}(K) \neq 0$ then
        $F \leftarrow F \cup \{x_i + x_j \leq 1\}$
      end if
    end for
  end for
  return $F$
```

Our problem is modeled into a binary linear programming problem:

$$\begin{aligned} &Maximize \sum_{i=0}^{127} x_i \\ &s.t.\ AX \leq b,\ X = \{x_i | x_i \in \{0, 1\}, 0 \leq i \leq 127\} \end{aligned}$$

where the $AX \leq b$ describe the constraints set F. Using the Gurobi Optimizer [26] to solve the problem, we get the first two optimum solutions and the corresponding index sets of v are listed in Table 8. Every variable v_i in each set will not multiply with each other in the same set. In the following we will use the $Index_0$ to introduce the basic idea of our attack. (The $Index_0$ can be replaced with $Index_1$ to get different bit conditions of K.)

The output limbs a_3 and c_3 can be linear by assigning the other variables $\{v_i\}$ to constants 0 if i $(0 \leq i \leq 127)$ is not involved in $Index_0$. Then setting one variable v_j $(j \notin Index_0)$ to be a cube variable(not a constant), we can get

Table 8. Index sets of independent variables

Set	Num	Values
$Index_0$	29	$0, 1, 7, 8, 15, 16, 23, 30, 31, 38, 39, 45, 46, 53, 60, 61, 68, 69, 75, 76, 83, 91, 98, 99, 105, 106, 113, 114, 121$
$Index_1$	28	$0, 1, 2, 9, 16, 23, 24, 25, 31, 32, 39, 46, 54, 55, 61, 62, 69, 84, 85, 91, 92, 98, 99, 107, 114, 115, 121, 122$

some quadratic terms $g_{i,j}(K)v_iv_j(i \in Index_0)$, where $g_{i,j}(K)$ is not a constant. The two cases $g_{i,j}(K) = 0$ and $g_{i,j}(K) = 1$ can be distinguished by some cube testers, which are similar to the attack on Frit$_b^\beta$-AE. So we can get some bit conditions to recover the secret key. By testing different cube sums to get 128 linearly independent bit conditions we can recover the 128-bit key.

Algorithm 3 Generating bit conditions and corresponding cube variables

```
**Input:** A set $Index$, the dimension $d$
**Output:** A list $B_c$ of bit conditions and a list $C_T$ of corresponding cube variables
  $B_c = [\,]$
  $C_T = [\,]$
  $Exp = \sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_c(v) \odot v$
  for each $j \in [0, 127] \setminus Index$ do
    $V_0 = \emptyset$
    $V_1 = [\,]$
    for each $i \in Index$ do
      if $g_{i,j}(K)v_iv_j \in Exp$ and $g_{i,j}(K) \neq 0$ then
        $V_0 \leftarrow V_0 \cup \{i\}$
        if $g_{i,j}(K)$ and $(g_{i,j}(K) + 1)$ not in $B_c$ then
          Add $i$ to $V_1$
          Add $g_{i,j}(K)$ to $B_c$
        end if
      end if
    end for
    for each $i \in V_1$ do
      $cube = \{j, i\} \cup \{k_m | k_m \in Index \setminus V_0, 0 \leq m \leq d - 3\}$
      Add $cube$ to $C_T$
    end for
  end for
  return $B_c$, $C_T$
```

The procedure to attack $r + 3$-round Frit$_a^\beta$-AE is concluded as follows.

1. First set the cube's dimension $d = F_{r+1} + 1(\beta = b)$ or $F_{r+2} + 1(\beta = a, c)$. Adding $v_j(j \notin Index_0)$ to the cube variables set, we can choose one quadratic term $g_{i,j}(K)v_iv_j(i \in Index_0)$ from c_3 and add v_i to the cube variables set. The other $d - 2$ cube variables are choosing from $Index_0$, which are not

multiplied with v_j. That is, we obtain a d-dimension cube to recover one bit condition $g_{i,j}(K)$.

2. Assign the other variables of v to constants 0 except for the cube variables and calculate the cube sum of the whole 128 bits output after $r+3$-round Frit. If all the 128 cube sums are 0, we take the $g_{i,j}(K)$ as 0, otherwise $g_{i,j}(K)=1$.
3. The time complexity of recovering 1 bit condition of K is 2^d. By changing the value of j and relative quadratic term $g_{i,j}(K)v_iv_j$, we can generate different cube variables to recover different $g_{i,j}(K)$. We can get 128 linearly independent bit conditions and solve the set of equations to recover the 128-bit key. We introduce the details to choose different bit conditions and corresponding cube variables in Algorithm 3. The time complexity is $2^d \times 2^7 = 2^{7+d}$.(The time to solving the linear system can be omitted.)

We notice that the cube's dimension d needs to be less than the size of set $Index_0$ (or $Index_1$). So our attack can be applied to no more than 10-round Frit$_a^b$-AE and 9-round Frit$_a^a$-AE or Frit$_a^c$-AE with $d=22$. We give the details of our attack on Frit$_a^b$-AE as an example, and the attack procedures for Frit$_a^a$-AE or Frit$_a^c$-AE are similar.

Experiments on 10-round Frit$_a^b$-AE. Applying the 3-round initial structure in Figure 4 to the 10-round Frit$_a^b$-AE, we can use the 22-dimension cube to get some bit conditions of K. For example, setting $j=4$ (v_4 is a cube variable), there are three quadratic terms in the expressions of c_3 and a_3:

$$(K_4+K_{91}+K_{114})v_4v_{45}, (K_{50}+K_{73}+K_{91})v_4v_{91}, (K_{73}+K_{96}+K_{114})v_4v_{114}.$$

Keeping only one variable in set $\{v_{45}, v_{91}, v_{114}\}$ to be a cube variable, there is only one quadratic term in the expressions of c_3 and a_3. We can get 1 bit condition of K by testing one cube. The examples of the bit conditions and relative cube variables are listed in Table 9.

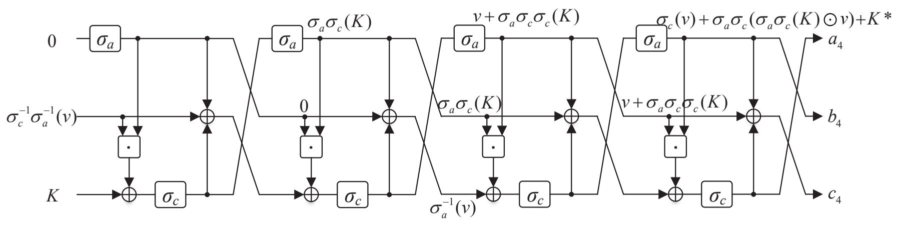
Table 9. Bit conditions and cube variables of 10-round Frit$_a^b$-AE

Bit conditions	Deg	Cube variables
$K_4+K_{91}+K_{114}$	22	$v_0, v_1, v_4, v_7, v_8, v_{15}, v_{16}, v_{23}, v_{30}, v_{31}, v_{38}, v_{39}$ $v_{45}, v_{46}, v_{53}, v_{60}, v_{61}, v_{68}, v_{69}, v_{75}, v_{76}, v_{83}$
$K_{50}+K_{73}+K_{91}$	22	$v_0, v_1, v_4, v_7, v_8, v_{15}, v_{16}, v_{23}, v_{30}, v_{31}, v_{38}, v_{39}$ $v_{46}, v_{53}, v_{60}, v_{61}, v_{68}, v_{69}, v_{75}, v_{76}, v_{83}, v_{91}$
$K_{73}+K_{96}+K_{114}$	22	$v_0, v_1, v_4, v_7, v_8, v_{15}, v_{16}, v_{23}, v_{30}, v_{31}, v_{38}, v_{39}$ $v_{46}, v_{53}, v_{60}, v_{61}, v_{68}, v_{69}, v_{75}, v_{76}, v_{83}, v_{114}$

All the 128 bit conditions and corresponding cube variables can be found by Algorithm 3 using SageMath [27]. Then solving a set of 128 linear equations we can recover the 128-bit key. Testing about 100 random keys has a success rate of 100%, and recovering each key needs about 8 minutes with time complexity 2^{29}.

6.2 New attacks on Frit$_c^\beta$-AE

Set the 128-bit key K to limb c_0 and the 256-bit nonce to limbs a_0 and b_0 as Figure 5. We give a 4-round initial structure of Frit$_c^\beta$-AE by keeping the limb a_0 to constants 0 and setting $\sigma_c^{-1}\sigma_a^{-1}(v)$ to limb b_0. After 3-round Frit the expressions of a_3, b_3, c_3 are linear with v.



0
$\sigma_c^{-1}\sigma_a^{-1}(v)$
K
σ_a
σ_c
$\sigma_a\sigma_c(K)$
0
$v+\sigma_a\sigma_c\sigma_c(K)$
$\sigma_a\sigma_c(K)$
$\sigma_a^{-1}(v)$
$\sigma_c(v)+\sigma_a\sigma_c(\sigma_a\sigma_c(K)\odot v)+K*$
$v+\sigma_a\sigma_c\sigma_c(K)$
a_4
b_4
c_4

Fig. 5. 4-round initial structure of Frit$_c^\beta$-AE, $K^* = \sigma_a\sigma_c(\sigma_a\sigma_c\sigma_c(K) \odot \sigma_a\sigma_c(K))$

To linearize the output expressions of a_4 and c_4, the expression $b_3 \odot b_4$ should not involve quadratic terms. By the conclusion $\sigma_a\sigma_c(\sigma_a^{-1}(v)) = \sigma_c(v)$ of Sect. 6.1, there needs that the expression $\sigma_a\sigma_c(\sigma_a\sigma_c(K) \odot v) \odot v + \sigma_c(v) \odot v$ to be linear if we only consider the linear parts of b_3 and b_4 for simplicity.(It should be noted that the K in the expression actually is $K \oplus RC_0$.) Comparing this expression with $\sigma_a\sigma_c(\sigma_a(K) \odot v) \odot v + \sigma_c(v) \odot v$ in Sect. 6.1, the only difference is the coefficient $g_{i,j}(K)$ of each term v_iv_j. We can apply the same analysis to Frit$_c^\beta$-AE, and the details are not repeated here. The procedure to attack $r + 4$-round Frit$_c^\beta$-AE is similar with the procedure to attack $r + 3$-round Frit$_a^\beta$-AE given in Sect. 6.1. We notice that only the 128 independent equations used to recover the 128-bit key are different. As a result, we can attack 11-round Frit$_c^b$-AE and 10-round Frit$_c^a$-AE and Frit$_c^c$-AE with time complexity 2^{29}. We give the details of the attack procedure on Frit$_c^b$-AE as an example.

Experiments on 11-round Frit$_c^b$-AE. Applying the 4-round initial structure in Figure 5 to the 11-round Frit$_c^b$-AE, we can use a 22-dimension cube to get 1 bit condition of K, which is similar to the experiment on 10-round Frit$_a^b$-AE. The three examples of recovering 1 bit condition are listed in Table 10. It is clear that the only differences are the bit conditions, which are recovered by the same cube variables. Recovering 128-bit K needs to solve a set of 128 linear equations, which are also can be calculated by Algorithm 3. After get 128-bit key, the last step is calculating $K \oplus RC_0$ to get the original secret key. Testing about 100 random keys also has a success rate of 100% in about 8 minutes each.

Table 10. Bit conditions and cube variables of 11-round Frit$_c^b$-AE

Bit conditions	Deg	Cube variables
$K_4 + K_{51} + K_{74} + K_{81} + K_{91}$ $+K_{92} + K_{104} + K_{114} + K_{122}$	22	$v_0, v_1, v_4, v_7, v_8, v_{15}, v_{16}, v_{23}, v_{30}, v_{31}, v_{38}, v_{39}$ $v_{45}, v_{46}, v_{53}, v_{60}, v_{61}, v_{68}, v_{69}, v_{75}, v_{76}, v_{83}$
$K_{10} + K_{33} + K_{40} + K_{50} + K_{51}$ $+K_{63} + K_{73} + K_{81} + K_{91}$	22	$v_0, v_1, v_4, v_7, v_8, v_{15}, v_{16}, v_{23}, v_{30}, v_{31}, v_{38}, v_{39}$ $v_{46}, v_{53}, v_{60}, v_{61}, v_{68}, v_{69}, v_{75}, v_{76}, v_{83}, v_{91}$
$K_{33} + K_{56} + K_{63} + K_{73} + K_{74}$ $+K_{86} + K_{96} + K_{104} + K_{114}$	22	$v_0, v_1, v_4, v_7, v_8, v_{15}, v_{16}, v_{23}, v_{30}, v_{31}, v_{38}, v_{39}$ $v_{46}, v_{53}, v_{60}, v_{61}, v_{68}, v_{69}, v_{75}, v_{76}, v_{83}, v_{114}$

7 Conclusion

In this paper, we partially answer the open question by Dobraunig *et al.* and give some key-recovery attacks on the rounded-reduced Frit used in duplex authenticated encryption mode (Frit$_\alpha^\beta$-AE). Our results cover all the versions of Frit$_\alpha^\beta$-AE and include some practical key-recovery attacks that could recover the key within several minutes. In the future, we will try to study the hash function mode of Frit, i.e. Frit with sponge.

References

1. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The keccak sha-3 submission. Submission to NIST (Round 3), 6(7):16, 2011.
2. Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer. Keyak. submission to the caesar competition, 2014.
3. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Caesar submission: Ketje v2. online at http://ketje. noekeon. org/ketje-1.1. pdf, 2014.
4. Nicky Mouha, Bart Mennink, Anthony Van Herrewege, Dai Watanabe, Bart Preneel, and Ingrid Verbauwhede. Chaskey: an efficient mac algorithm for 32-bit microcontrollers. In International Workshop on Selected Areas in Cryptography, pages 306-323. Springer, 2014.
5. Daniel J Bernstein. The salsa20 family of stream ciphers. In New stream cipher designs, pages 84-97. Springer, 2008.
6. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1. 2. Submission to the CAESAR Competition, 2016.
7. Shay Gueron and Nicky Mouha. Simpira v2: a family of efficient permutations using the aes round function. In International Conference on the Theory and Application of Cryptology and Information Security, pages 95-125. Springer, 2016.
8. Daniel J Bernstein, Stefan Kölbl, Stefan Lucks, Pedro Maat Costa Massolino, Florian Mendel, Kashif Nawaz, Tobias Schneider, Peter Schwabe, Fran?cois-Xavier Standaert, Yosuke Todo, et al. Gimli: a cross-platform permutation. In International Conference on Cryptographic Hardware and Embedded Systems, pages 299-320. Springer, 2017.
9. Joan Daemen, Seth Hoffert, Gilles Van Assche, and Ronny Van Keer. Xoodoo cookbook. Technical report, Cryptology ePrint Archive: Report 2018/767, 2018. https://eprint. iacr. org/2018/767, 2018.

10. Thierry Simon, Lejla Batina, Joan Daemen, Vincent Grosso, Pedro Maat Costa Massolino, Kostas Papagiannopoulos, Francesco Regazzoni, and Niels Samwel. Towards lightweight cryptographic primitives with built-in fault-detection. Cryptology ePrint Archive, Report 2018/729, 2018. https://eprint.iacr.org/2018/729.
11. Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. In International Conference on the Theory and Application of Cryptology, pages 210-224. Springer, 1991.
12. Guido Bertoni, Joan Daemen, and Gilles Van Assche. On the indifferentiability of the sponge construction. In International Conference on the Theory and Applications of Cryptographic Techniques, pages 181-197, 2008.
13. Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Duplexing the sponge: Single-pass authenticated encryption and other applications. In International Conference on Selected Areas in Cryptography, pages 320-337, 2011.
14. Jean-Philippe Aumasson, Philipp Jovanovic, and Samuel Neves. Norx: parallel and scalable aead. In European Symposium on Research in Computer Security, pages 19-36. Springer, 2014.
15. Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Markus Schofnegger. Algebraic cryptanalysis of frit. Cryptology ePrint Archive, Report 2018/809, 2018. https://eprint.iacr.org/2018/809.
16. Senyang Huang, Xiaoyun Wang, Guangwu Xu, Meiqin Wang, and Jingyuan Zhao. Conditional cube attack on reduced-round keccak sponge function. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 259-288. Springer, 2017.
17. Xiaoyang Dong, Zheng Li, Xiaoyun Wang, Ling Qin. Cube-like attack on round-reduced initialization of Ketje Sr. IACR Trans Symmetric Cryptol, 2017(1): 259-280.
18. Zheng Li, Wenquan Bi, Xiaoyang Dong, Xiaoyun Wang. Improved conditional cube attacks on keccak keyed modes with MILP method. In: Takagi T and Peyrin T, eds. Advances in Cryptology - ASIACRYPT 2017, Part I. LNCS, Vol 10624. Springer, Cham, 2017. 99-127.
19. Zheng Li, Xiaoyang Dong, Xiaoyun Wang. Conditional cube attack on round-reduced ASCON. IACR Trans Symmetric Cryptol, 2017(1): 175-202.
20. Wenquan Bi, Xiaoyang Dong, Zheng Li, Xiaoyun Wang. MILP-aided cube-attack-like cryptanalysis on keccak keyed modes. Des Codes Cryptogr, (2018). https://doi.org/10.1007/s10623-018-0526-x
21. Simon Knellwolf, Willi Meier, and María Naya-Plasencia. Conditional differential cryptanalysis of nlfsr-based cryptosystems. In International Conference on the Theory and Application of Cryptology and Information Security, pages 130-145. Springer, 2010.
22. Itai Dinur and Adi Shamir. Cube attacks on tweakable black box polynomials. In Advances in Cryptology - EUROCRYPT 2009, International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings, pages 278-299, 2008.
23. Itai Dinur and Adi Shamir. Breaking grain-128 with dynamic cube attacks. In International Conference on FAST Software Encryption, pages 167-187, 2011.
24. Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and Linear Cryptanalysis Using Mixed-Integer Linear Programming. In: Wu CK., Yung M., Lin D. (eds) Information Security and Cryptology. Inscrypt 2011. Lecture Notes in Computer Science, vol 7537. Springer, Berlin, Heidelberg.

25. Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic Security Evaluation and (Related-key) Differential Characteristic Search: Application to SIMON, PRESENT, LBlock, DES(L) and Other Bit-Oriented Block Ciphers. In: Sarkar P., Iwata T. (eds) Advances in Cryptology C ASIACRYPT 2014. ASIACRYPT 2014. Lecture Notes in Computer Science, vol 8873. Springer, Berlin, Heidelberg.
26. http://www.gurobi.com/
27. http://www.sagemath.org/