IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 2026, No. 1, pp. 119–147. DOI:10.46586/tosc.v2026.i1.119-147

Cryptanalysis of TFHE-Friendly Cipher FRAST

Antoine Bak[1,2], Shibam Ghosh[1], Fukang Liu[3], Willi Meier[4], Jianqiang Ni[5] and Léo Perrin[1]

[1] Inria, Paris, France
{antoine.bak,shibam.ghosh,leo.perrin}@inria.fr
[2] Direction Générale de l'Armement (DGA), Paris, France
[3] Institute of Science Tokyo, Tokyo, Japan
liu.f.ad@m.titech.ac.jp
[4] University of Applied Sciences and Arts Northwestern Switzerland, Windisch, Switzerland
willimeier48@gmail.com
[5] East China Normal University, Shanghai, China
jianqiangni0213@163.com

Abstract. FRAST is a TFHE-friendly stream cipher that was published at FSE 2025. The cipher is defined over $\mathbb{Z}_{16}$, and makes extensive use of negacyclic S-boxes over $\mathbb{Z}_{16}$ as they are less costly in TFHE. Like many FHE-friendly ciphers, FRAST randomizes some of its components to increase its security against statistical attacks. In the case of FRAST, some S-boxes are randomized using an XOF that takes a nonce as input. In this work, we point out a strong structural property of the full FRAST permutation, which leads to a much simpler alternative representation of the primitive. We study the consequences of this representation and find a weak key space of non-negligible size (i.e., much larger than 2^{128}) on which every ciphertext leaks one bit of plaintext. This corresponds to a distinguishing attack on the full FRAST in the weak-key setting. In particular, we emphasize that, apart from the structural property, the usage of negacyclic S-boxes further leads to a much larger weak-key space for our attack.
Finally, we provide a general framework to mount a linear attack on FRAST in the average key setting. We briefly describe our approach in the end of the paper, and observe that standard assumptions expected to work in the context of linear cryptanalysis do not hold in the case of FRAST: our experiments indicate that a linear attack in the average key setting does not work as expected.

Keywords: HE-friendly cipher · stream cipher · FRAST · weak-key attack

1 Introduction

Fully homomorphic encryption (FHE) allows to perform computations (addition and multiplication) over encrypted messages, which makes it quite appealing in privacy-preserving applications. Although the poor performance of FHE schemes is still the bottleneck towards wide deployments in real-world applications, it has been substantially improved since the first proposal by Gentry [Gen09]. Specifically, computations over encrypted messages with existing FHE schemes (e.g., BFV [FV12, Bra12], BGV [BGV14], CKKS [CKKS17], FHEW [DM15], and TFHE [CGGI20]) are still time-consuming, and the ciphertext expansion is also a problem since these FHE schemes are lattice-based.

To address the issue of ciphertext expansion, the so-called *hybrid homomorphic encryption (HHE)* or *transciphering* framework was proposed [NLV11]. In HHE, the client side performs relatively easy computations while the server performs intensive computations. The communication cost (i.e., the ciphertext expansion) between the client and server is reduced by using a symmetric-key primitive rather than a lattice-based encryption

Received: 2025-09-01 Revised: 2025-11-23 Accepted: 2026-01-23 Published: 2026-03-16

algorithm to encrypt plain messages. Specifically, on the client side, a symmetric-key primitive is used to encrypt the plain messages, and a FHE scheme is used to encrypt the secret key once. Then, the corresponding ciphertexts as well as the encrypted secret key are sent to the server. After receiving these, the server first performs homomorphic decryption of the ciphertexts with the encrypted secret key. Then, it performs the required homomorphic evaluation functions over the homomorphically decrypted results. Hence, as the length of plain messages increases, there is almost no ciphertext expansion since the length of the plain messages and the encrypted data sent to the server are almost the same.

Although the issue of ciphertext expansion can be addressed with HHE, performing homomorphic decryption of the received ciphertexts is necessary before starting the homomorphic evaluations. Hence, the homomorphic decryption has to be as efficient as possible in order to improve the overall performance. This leads to the emergence of the so-called FHE-friendly symmetric-key encryption algorithms, and many have been proposed in recent years. Examples include block ciphers such as LowMC [ARS+15], Chaghri [AMT22], and stream ciphers such as Kreyvium [CCF+18], FLIP [MJSC16], Rasta [DEG+18] and its variants Fasta [CIR22], Dasta [HL20], Pasta [DGH+23] and Pasta$_{v2}$ [GLR+24], HERA [CHK+21], Rubato [HKL+22], Elisabeth-4 [CHMS22], FRAST [CCH+24] and Transistor [BBB+25].

Many such designs use randomized components, i.e., either randomized affine layers, or randomized key schedules, or randomized S-boxes. Although these FHE-friendly ciphers do significantly outperform conventional symmetric-key primitives (e.g., AES) in FHE protocols, some do not stand the test of time or have potential weaknesses violating the security claims. A key line of cryptanalysis in this domain is the line of algebraic attacks, which exploits the low algebraic complexity of the cipher. Notable examples include attacks on Rasta and Dasta [LSMI21, LSW+22], LowMC [LSMI22], Elisabeth-4 [GBJR23] and Chaghri [LAW+23], all of which leverage the low algebraic degree of these primitives or the sparsity of their polynomial representation. Further algebraic attacks have been mounted on Rubato [GAH+23] and HERA [LKSM24]. Additionally, structural weaknesses have been found in the filter function of the stream cipher FLIP [DLR16],

In this work, we continue the line of research to study the security of FHE-friendly ciphers. In particular, we focus on the TFHE-friendly stream cipher FRAST recently proposed at ToSC 2024 [CCH+24]. This is the first FHE-friendly symmetric-key primitive to use randomized S-boxes in the literature. Apart from this feature, it also has a quite large number of rounds (i.e., 40 rounds) compared with other such primitives, and it is defined over the integer ring $\mathbb{Z}_{16}$ rather than a finite field. It shares this property only with Elisabeth-4, which is the first FHE-friendly stream cipher defined over $\mathbb{Z}_{16}$. However, the latter was soon broken in [GBJR23] by exploiting the fact that the least significant bits of x, y, z are linear over $\mathbb{F}_2$ for the modular addition $z = x + y \pmod{2^i}$ where i is a positive integer.

1.1 Notations

In this paper, $\boxplus$ and $\boxminus$ represent addition and subtraction modulo 16, respectively. We sometimes also use $+$, and when necessary, the modulo will be explicitly stated. Additionally, we use $a\%b$ to denote $a \pmod{b}$. Throughout this paper, vectors and matrices are written in **bold**. For any vector $\mathbf{v} \in \mathbb{Z}_{16}^n$, the i-th entry of $\mathbf{v}$ is denoted by v_i, i.e., $\mathbf{v} = (v_1, \ldots, v_n)$. We also denote $(e^1, ..., e^{32})$ the canonical basis of $\mathbb{Z}_{16}^{32}$, where $e^i_j = 1$ if $i = j$, and 0 otherwise.

1.2 Description of FRAST

Before proceeding, we briefly review the inner workings of FRAST, as introduced in [CCH+24]. FRAST is a stream cipher constructed using a custom block cipher operating in a variant

of the counter mode. The cipher takes as input a 256-bit master key $\boldsymbol{k} \in \mathbb{Z}_{16}^{64}$ and is initialized with a 64-bit nonce $\text{NC} \in \{0,1\}^{64}$ and a 64-bit counter $\text{CTR} \in \{0,1\}^{64}$. The counter is incremented with each invocation of the cipher, while the nonce remains fixed for the duration of its use. The combined value $\text{NC}||\text{CTR} \in \{0,1\}^{128}$ is used to derive most of the cipher's S-boxes.

The internal block cipher consists of 40 rounds, where the r-th round is denoted by $\mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$ for $1 \leq r \leq 40$, i.e.,

$$\mathsf{FRAST}[\boldsymbol{k}, \text{NC}||\text{CTR}] = \mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, 40] \circ \mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, 39] \circ \cdots \circ \mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, 1] \ .$$

A 128-bit keystream block is generated by encrypting a fixed input string defined as $\text{IC} = (0, 1, ..., 15, 0, 1, ..., 15) \in \mathbb{Z}_{16}^{32}$, i.e., the keystream is given by $z = \mathsf{FRAST}[\boldsymbol{k}, \text{NC}||\text{CTR}](\text{IC})$.

Round Functions of FRAST. A schematic illustration of the r-th round function $\mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$ is shown in Figure 1. Let the input and output of the r-th round be $(y_1^{(r-1)}, \ldots, y_{32}^{(r-1)})$ and $(y_1^{(r)}, \ldots, y_{32}^{(r)})$, respectively. The round function $\mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$ is defined as follows:

$$\begin{aligned} y_i^{(r)} &= y_i^{(r-1)} \boxplus S_{\text{erf}}^{(r)}(y_1^{(r-1)} \boxplus \text{rk}_j^{(r)}) \text{ for } j = 2, 3, \ldots, 32, \\ y_1^{(r)} &= y_1^{(r-1)} \boxplus S_{\text{crf}}^{(r)}(\text{rk}_1^{(r)} \boxplus y_2^{(r)} \boxplus y_3^{(r)} \boxplus \cdots \boxplus y_{32}^{(r)}). \end{aligned}$$

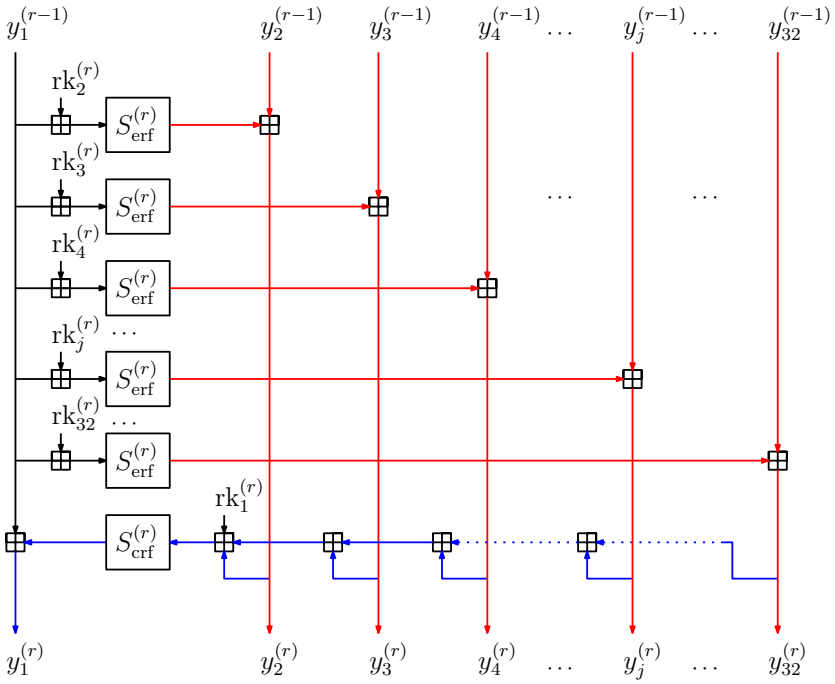

Figure 1: The round function of FRAST

S-boxes of FRAST. Crucially, the 4-bit S-boxes $S_{\text{erf}}^{(r)}$ and $S_{\text{crf}}^{(r)}$ are not fixed for 32 out of 40 rounds. Specifically, for rounds where $r \neq 0 \pmod 5$, they are generated randomly using an eXtendable Output Function (XOF) seeded with the value $\text{NC}||\text{CTR}$. Since CTR increments on each call to FRAST, a unique set of S-boxes is used in each call. For rounds where $r = 0 \pmod 5$, the S-boxes are fixed and identical across those rounds. They are

denoted by S_{fixed}, as defined in Table 1. Rounds that use S-boxes derived from NC||CTR are referred to as *random rounds*, while those using S_{fixed} are referred to as *fixed rounds*. In total, there are 32 random rounds and 8 fixed rounds.

Table 1: The fixed S-box S_{fixed} used in the fixed rounds of FRAST

y	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(y)$	0	3	5	8	6	9	12	7	13	10	14	4	1	15	11	2

Although $S_{\text{erf}}^{(r)}$ and $S_{\text{crf}}^{(r)}$ are randomly generated for each random round, only $4 \times 8 = 32$ random bits are required to instantiate each S-box. This is enabled by the property of *negacyclicity*, designed to optimize the cipher's implementation in TFHE. Negacyclicity enforces that $S(x \boxplus 8) = \boxminus S(x)$ for all $x \in \mathbb{Z}_{16}$, which implies that only values for $x \in \{0, \ldots, 7\}$ need to be explicitly sampled. We emphasize that there is no rejection when sampling these S-boxes, i.e., there are no constraints on the 32 random bits to instantiate each random S-box.

Key Schedule of FRAST. Each round key $\mathbf{rk}^{(r)} = (\mathrm{rk}_1^{(r)}, \mathrm{rk}_2^{(r)}, \ldots, \mathrm{rk}_{32}^{(r)}) \in \mathbb{Z}_{16}^{32}$ is a linear function of the 256-bit master key $\boldsymbol{k}$. Specifically, the round keys $\mathbf{rk}^{(2i-1)}$ and $\mathbf{rk}^{(2i)}$ for $i = 1, 2, \ldots, 20$ are defined by $\mathbf{rk}^{(2i-1)} || \mathbf{rk}^{(2i)} = \boldsymbol{M}^i \cdot \boldsymbol{k}$, where $\boldsymbol{M} \in \mathbb{Z}_{16}^{64 \times 64}$ is a fixed invertible matrix. For the detailed specification of $\boldsymbol{M}$, and for a full description of the initial security analysis of this primitive, we refer to [CCH+24].

1.3 Our Contributions

In this paper, we present a third-party cryptanalysis of FRAST. Our main contribution is the discovery of a strong distinguishing property in the inner block cipher of FRAST— on which all our other results are based. Indeed, it turns out that any input difference taken in a vector space of co-dimension 2 (meaning, in a space of dimension 30) will go through an arbitrary number of rounds with probability 1.

Needless to say, this property has strong consequences in terms of security. However, the mode in which the inner block cipher is used to construct a stream cipher successfully prevents some trivial attacks by injecting the counter alongside the nonce rather than through the plaintext. As a consequence, exploiting the differential patterns we identified turned out to require sophisticated techniques.

We first demonstrate that these differential patterns give rise to a large number of alternative representations of (restrictions of) the round function. These structural redundancies can be exploited in a meaningful way. Indeed, using these alternative forms, we identify extensive families of weak keys. Specifically, out of all possible 256-bit master keys, at least 2^{135+w} keys lead to deterministic relations between bits in the keystream, using an online complexity of $2^{31.9}$ and a choice of nonces that induces an offline complexity of $2^{7(w+1)}$. Using this weak key space, we propose a distinguishing attack in FRAST keystream generator with advantage better than generic attack. This distinguishing attack has been implemented on reduced round versions as a proof of concept.[1]

Then, we investigate the consequences of these alternative representations in terms of linear cryptanalysis. However, the linear behaviour of FRAST turns out to be poorly explained by state-of-the-art techniques. In the hopes to foster further research in this direction, we present an attempt at linear cryptanalysis, and describe the mismatch between the correlations expected based on standard arguments, and the ones we actually observed in practice.

[1]The verification codes are available at https://github.com/ShibamCrS/Cryptanalysis_FRAST.git.

Outline of this paper. The alternative representation of FRAST is detailed in Section 2. Then, we describe the full-round attack in the weak key setting in Section 3. Our attempt at linear cryptanalysis in the average key setting can be found in Section 4. Finally, this paper is concluded in Section 5

2 Alternative Representations of FRAST

It turns out that the round function of FRAST admits many functionally equivalent representations, i.e., there exist multiple distinct circuit structures, in the style of Figure 1, that compute the same round function. These are consequences of a particularly strong differential property that holds with probability 1, which we detail in Section 2.1. Then, we present different alternative representations of FRAST, and in fact of the restriction of FRAST to some specific subspaces, in Section 2.2. Finally, we deduce some probability 1 linear approximations in a weak key setting in Section 2.3.

2.1 A Powerful Differential Distinguisher

FRAST is built as a generalized Feistel network operating on 32 branches. However, the specific variant used in FRAST exhibits an unusually strong property. Using the notation from Figure 1, consider two plaintexts: $(y_1, \ldots, y_{32})$ and $(y_1, y_2 \boxplus \delta, y_3 \boxminus \delta, y_4, \ldots, y_{32})$, meaning a pair corresponding to the input differential $(0, \delta, \boxminus\delta, 0, \ldots, 0)$. This differential pattern passes through the first part of the round function, namely the layer involving $S^{(r)}_{\text{erf}}$, since the difference does not affect any of the S-box inputs. However, it also remains unchanged in the second part. Indeed, the input to $S^{(r)}_{\text{crf}}$ is the same for both plaintexts because the summed difference is equal to 0: $\delta - \delta = 0$. As a consequence, for any δ, we have an iterated probability 1 differential.

This pattern can be greatly generalized. The choice to place the non-zero differences in the second and third branches is arbitrary, and many other such configurations are possible. In fact, the following result holds.

Theorem 1 (Probability 1 iterated differentials)**.** *Let $\Delta = (0, \Delta_2, \Delta_3, \ldots, \Delta_{32})$ be a vector of $\mathbb{Z}_{16}^{32}$ such that $\sum_{i=2}^{32} \Delta_i = 0$. Then, for any input $x \in \mathbb{Z}_{16}^{32}$, key $\boldsymbol{k}$, nonce NC, and round number r, the following equation holds:*

$$\mathcal{R}[\boldsymbol{k}, NC||CTR, r](x + \Delta) \;=\; \mathcal{R}[\boldsymbol{k}, NC, r](x) + \Delta \;. \tag{1}$$

In other words, the round function preserves the difference Δ with probability 1, as long as two linear constraints are satisfied: $\Delta_1 = 0$ and $\sum_{i=2}^{32} \Delta_i = 0$. The set of such valid differences forms a subspace of dimension 30 over $\mathbb{Z}_{16}$, meaning Theorem 1 applies to 2^{120} distinct input differences out of all 2^{128} possible.

2.2 Describing Alternative Representations

Representing a Restriction. To get a better understanding of Theorem 1, we describe in this section an alternative representation of the round function of FRAST. For this purpose, we divide one round of FRAST into two steps: the *expanding* step and the *compressing* step. Then, we apply each step on a state $(x_1, \ldots, x_{32})$, and study how it modifies the specific pair $(x_1, \sum_{k=2}^{32} x_k)$.

- **Expanding step.** This step is defined as

$$(x_1, \ldots, x_{32}) \mapsto (x_1, x_2 \boxplus S_{\text{erf}}(x_1 \boxplus \text{rk}_2), \ldots, x_{32} \boxplus S_{\text{erf}}(x_1 \boxplus \text{rk}_{32})) \;.$$

After this step, the pair $(x_1, \sum_{k=2}^{32} x_k)$ becomes

$$\left(x_1, \sum_{i=2}^{32}(x_i \boxplus S_{\text{erf}}(x_1 \boxplus \text{rk}_i))\right) = \left(x_1, \sum_{i=2}^{32} x_i \boxplus \sum_{j=2}^{32} S_{\text{erf}}(x_1 \boxplus \text{rk}_j)\right) .$$

- **Compressing step.** This step is defined as

$$(x_1, \ldots, x_{32}) \mapsto \left(x_1 \boxplus S_{\text{crf}}\left(\sum_{k=2}^{32} x_k \boxplus \text{rk}_1\right), x_2, \ldots, x_{32}\right) .$$

After this step, the pair $(x_1, \sum_{k=2}^{32} x_k)$ becomes

$$\left(x_1 \boxplus S_{\text{crf}}\left(\sum_{k=2}^{32} x_k \boxplus \text{rk}_1\right), \sum_{k=2}^{32} x_k\right) .$$

Now, if we consider $F_1 : x \mapsto \sum_{j=2}^{32} S_{\text{erf}}(x \boxplus \text{rk}_j)$ and $F_2 : x \mapsto S_{\text{crf}}(x \boxplus \text{rk}_1)$, then it is clear from the above discussion that one round $\mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$ of FRAST acts on $(x_1, \sum_{k=2}^{32} x_k)$ as two rounds of a two-branch Feistel with round functions F_1 and F_2. If we denote this two branch Feistel as $g[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$, and if we let $\mathcal{L}(x_1, \ldots, x_{32}) = (x_1, \sum_{k=2}^{32} x_k)$, then

$$\mathcal{L} \circ \mathcal{R}[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, \ldots, x_{32}) = g[\boldsymbol{k}, \text{NC}||\text{CTR}, r] \circ \mathcal{L}(x_1, \ldots, x_{32}) .$$

This two-branch Feistel structure is represented as the two leftmost branches in Figure 2. Recall that $(e^1, ..., e^{32})$ is the canonical basis of $\mathbb{Z}_{16}^{32}$. Using it, we reframe this finding as follows: the subspace with basis $(e^1, \sum_{k>1} e^k)$ is stable under the round function of FRAST, and said restriction can be seen as a two-branched Feistel network.

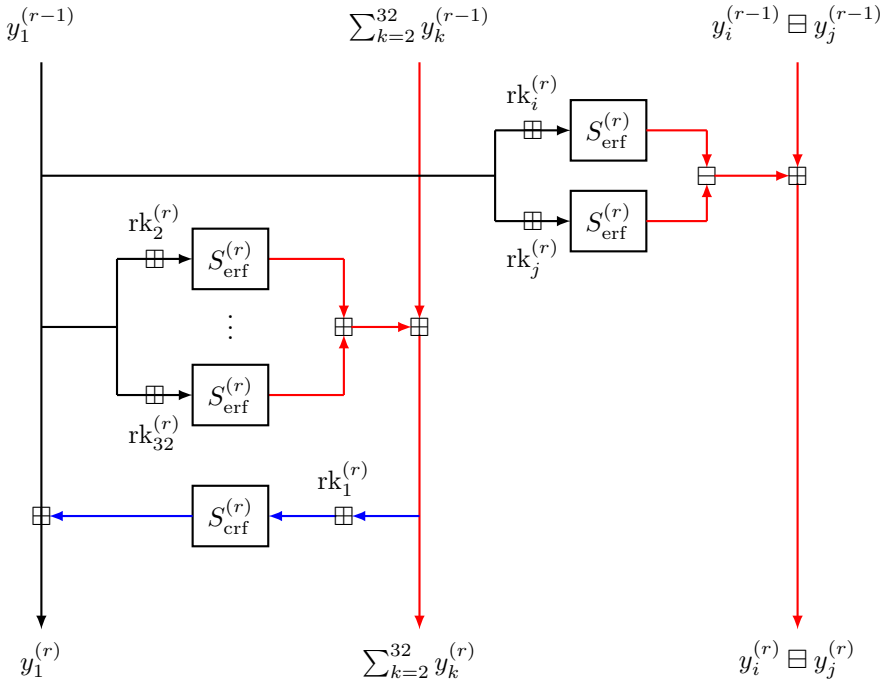

Figure 2: An alternative representation of a restriction of the FRAST round function.

Considering Other Restrictions. Another important observation is that after one round of FRAST, for $2 \leq i \leq 32$, x_i is mapped to $x_i \boxplus S^{(r)}_{\text{erf}}(x_1 \boxplus \text{rk}^{(r)}_i)$, which only depends on the coordinates x_1 and x_i of the input. We will show that this observation allows the description of highly structured alternative representations of the whole FRAST, and also of its restrictions to other spaces.

In its original definition, the round functions are defined using the canonical basis $(e^1, \ldots, e^{32})$ of $\mathbb{Z}_{16}^{32}$. Let us write the equations defining FRAST over other bases. For all $2 \leq i < j \leq 32$, we introduce a subspace of $\mathbb{Z}_{16}^{32}$ defined by the basis

$$\mathcal{B}_{(i,j)} = \left(e^1, \sum_{k=2}^{32} e^k, e^i - e^j \right) .$$

Over the corresponding subspace of dimension 3, the input differences Δ that satisfy the conditions of Theorem 1 take the form $(0, 0, \delta)$, meaning that the non-zero coordinates lie only in the last coordinate. We then define an alternative representation of the round function as follows:

$$\mathcal{R}^{\text{alt}}[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_2, x_3) = (g[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_2), f_{(i,j)}[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_3)) . \tag{2}$$

Here, the function $g[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$ corresponds to the two-branch Feistel structure we defined earlier, i.e. it can be decomposed into two parts as follows

$$g[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_2) \;=\; g_2[\boldsymbol{k}, \text{NC}||\text{CTR}, r] \;\circ\; g_1[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_2) ,$$

where g_1 and g_2 are such that

$$g_1[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_2) = \left(x_1, x_2 \boxplus \sum_{i=2}^{32} S^{(r)}_{\text{erf}}(x_1 \boxplus \text{rk}^{(r)}_i) \right)$$
$$g_2[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_2) = \left(x_1 \boxplus S^{(r)}_{\text{crf}}(x_2 \boxplus \text{rk}^{(r)}_1), x_2 \right) .$$

The last output coordinate (the rightmost one in Figure 2) is then defined as:

$$f_{(i,j)}[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1, x_3) \;=\; x_3 \boxplus S^{(r)}_{\text{erf}} \left(x_1 \boxplus \text{rk}^{(r)}_i \right) \boxminus S^{(r)}_{\text{erf}} \left(x_1 \boxplus \text{rk}^{(r)}_j \right) . \tag{3}$$

This structure mirrors the choice of the basis $\mathcal{B}_{(i,j)}$, where the last vector is expressed as a difference between two standard basis vectors. Thus, the relationship between the original round function $\mathcal{R}$ and its alternative representation $\mathcal{R}^{\text{alt}}$ is captured by the fact that they commute with a specific linear mapping:

$$\mathcal{L} \circ \mathcal{R}[\boldsymbol{k}, \text{NC}, r](x_1, \ldots, x_{32}) = \mathcal{R}^{\text{alt}}[\boldsymbol{k}, \text{NC}, r] \circ \mathcal{L}(x_1, \ldots, x_{32}) , \tag{4}$$

where $\mathcal{L}$ is the linear transformation mapping the canonical representation of vectors to the alternative representation. As a result, the subset $(z_1, \sum_{k=2}^{32} z_k, z_i \boxminus z_j)$ of the output of the full FRAST cipher can be equivalently expressed using:

$$\mathcal{L} \circ \text{FRAST}[\boldsymbol{k}, \text{NC}||\text{CTR}] = \mathcal{R}^{\text{alt}}[\boldsymbol{k}, \text{NC}||\text{CTR}, 40] \circ \cdots \circ \mathcal{R}^{\text{alt}}[\boldsymbol{k}, \text{NC}||\text{CTR}, 1] \circ \mathcal{L} .$$

Remark 1. If the round function had included a final swap between branches — such as the branch rotation commonly used in many generalized Feistel networks — then the iterative structure underlying these alternative representations would no longer hold. It is therefore reasonable to attribute this structural vulnerability to the absence of such a swap. However, whether introducing a swap would be sufficient to eliminate the issue entirely remains an open question.

2.3 Probability 1 Linear Approximations

As a consequence of the observation above, the alternative representation can be used to construct a linear approximation of the round function. Consider the structure depicted in Figure 2 with respect to $\mathcal{B}_{(i,j)}$. Let $\mathbf{x}^{(r-1)}, \mathbf{x}^{(r)} \in \mathbb{Z}_{16}^3$ denote the input and output vectors of $\mathcal{R}^{\text{alt}}$ at round r. From the definition of $f_{(i,j)}$ in Equation (3), we have:

$$\begin{aligned} x_3^{(r)} &= f_{(i,j)}[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1^{(r-1)}, x_3^{(r-1)}) \\ &= x_3^{(r-1)} \boxplus \left(S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_i^{(r)}) \boxminus S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_j^{(r)}) \right). \end{aligned}$$

Now suppose the round key satisfies $\text{rk}_i^{(r)} = \text{rk}_j^{(r)}$. In that case, the two S-box inputs are identical, and the subtraction cancels out, leading to: $x_3^{(r)} = x_3^{(r-1)}$. If this condition holds for every round $r \in [1, 40]$, we obtain the following invariant over 40 rounds of the alternative representation: $x_3^{(40)} = x_3^{(0)}$. Translating this result back to the canonical basis (i.e., undoing the change of basis), if $\mathbf{x}^{(40)} = \mathcal{L}(\mathbf{y}^{(40)})$ and $\mathbf{x}^{(0)} = \mathcal{L}(\mathbf{y}^{(0)})$, then we find the following relation in the input and output values of 40-round FRAST:

$$y_i^{(40)} \boxminus y_j^{(40)} = \text{IC}_i \boxminus \text{IC}_j \ ,$$

where $(\text{IC}_i, \text{IC}_j)$ are the corresponding coordinates of the initial input, which is public. In particular, we have the following linear relation:

$$y_i^{(40)} + y_j^{(40)} = \text{IC}_i + \text{IC}_j \pmod 2 \ , \tag{5}$$

which we will use in the attack. This relation holds for any choice of NC||CTR, provided the condition $\text{rk}_i^{(r)} = \text{rk}_j^{(r)}$ is satisfied for all rounds. For the remainder of the paper, given an alternative representation (i.e., a fixed basis $\mathcal{B}_{(i,j)}$), we refer to any master key that satisfies the corresponding conditions as a *weak key* with respect to that representation.

3 Refined weak key Attack on Full FRAST

As established in Section 2.3, any master key satisfying the weak key conditions leads to a deterministic full-round distinguisher. The central goal of this section is therefore to demonstrate that the space of such weak keys is large enough so that the probability of a randomly chosen key being weak exceeds 2^{-128}. We believe that if the number of weak keys exceeds 2^{128}, the attack becomes more practically relevant and exposes a meaningful structural vulnerability in the cipher, as discussed at the end of this section.

The above analysis given in Section 2.3 indicates that whatever S_{erf} is, it is always possible to add 4 bit conditions on round keys at each round to construct a deterministic full-round distinguisher. However, this attack will require the $4 \times 40 = 160$ bit conditions to be satisfied by the round keys. First, assume that these conditions are independent. As there are $\binom{31}{2}$ possible choices of $\mathcal{B}$, we can have $\binom{31}{2}$ alternative representations. Hence, there are $\binom{31}{2}$ different sets of weak keys, and each set is about $2^{256-160} = 2^{96}$. If the intersection of each set is small enough compared with 2^{96}, the weak key space size is estimated as

$$\binom{31}{2} \cdot 2^{256-160} = \binom{31}{2} \cdot 2^{96} < 2^{128}.$$

How to further enlarge the weak key space so that a weak key is used with probability higher than 2^{-128}? In the following, we will describe three methods to do this.

3.1 Exploiting the Negacyclic Property of Random S-boxes

As already stated, there are 32 rounds of FRAST where random S-boxes are used. In particular, each random S-box is a negacyclic function over $\mathbb{Z}_{16}$. Hence, Lemma 1 will be useful to reduce the number of conditions on round keys for the random rounds.

Lemma 1. *For a negacyclic function $S_{\text{erf}}^{(r)}$ over $\mathbb{Z}_{16}$ which satisfies*

$$\forall x \in \mathbb{Z}_{16}: \ S_{\text{erf}}^{(r)}(x) = \boxminus S_{\text{erf}}^{(r)}(x \boxplus 8) \ ,$$

and for any $\delta \in \{0, 8\}$, the following holds:

$$\forall x \in \mathbb{Z}_{16}: \ S_{\text{erf}}^{(r)}(x \boxplus \delta) - S_{\text{erf}}^{(r)}(x) = 0 \pmod 2 \ . \tag{6}$$

Proof. The proof is rather simple:

- If $\delta = 0$, then it is clear that $S_{\text{erf}}^{(r)}(x \boxplus \delta) \boxminus S_{\text{erf}}^{(r)}(x) = 0$. In particular, the equality also holds modulo 2.
- If $\delta = 8$, then by negacyclicity we have $S_{\text{erf}}^{(r)}(x \boxplus \delta) \boxminus S_{\text{erf}}^{(r)}(x) = \boxminus 2 \cdot S_{\text{erf}}^{(r)}(x) = 0 \pmod 2$.

□

Specifically, for any round $\mathcal{R}^{\text{alt}}[\boldsymbol{k}, \text{NC}||\text{CTR}, r]$ and for any $\mathcal{B}_{(i,j)}$, we have

$$\begin{aligned} x_3^{(r)} &= f_{(i,j)}[\boldsymbol{k}, \text{NC}||\text{CTR}, r](x_1^{(r-1)}, x_3^{(r-1)}) \\ &= x_3^{(r-1)} \boxplus \left(S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_i^{(r)}) \boxminus S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_j^{(r)}) \right) \\ &= x_3^{(r-1)} + \left(S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_i^{(r)}) + S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_j^{(r)}) \right) \pmod 2 \ . \end{aligned} \tag{7}$$

Thus, if $\text{rk}_i^{(r)} \boxminus \text{rk}_j^{(r)} \in \{0, 8\}$ then (according to Lemma 1) we have the following important consequence: $x_3^{(r)} = x_3^{(r-1)} \pmod 2$. Therefore, instead of imposing the 4 bit conditions $\text{rk}_i^{(r)} = \text{rk}_j^{(r)} \pmod{16}$ for each random round, we only need to impose 3 bit conditions by exploiting the negacyclic property of the random S-boxes. However, the condition for the 8 fixed rounds remain the same. In this way, there are still $\binom{31}{2}$ different sets of weak keys, and each set is of size

$$2^{256 - 32 \cdot 3 - 8 \cdot 4} = 2^{128}. \tag{8}$$

If the intersection of these sets is small enough, the weak key space size is estimated as

$$\binom{31}{2} \cdot 2^{128} > 2^{128}.$$

3.2 Generating Weak Random S-boxes with Chosen Nonce

As is clear in our weak key attack, for any r-th round $\mathcal{R}^{\text{alt}}$ and for any three-branch structure $\mathcal{B}_{(i,j)}$, Equation (7) holds. It implies that $x_3^{(r)} = x_3^{(r-1)} \pmod 2$ provided that $\text{rk}_i^{(r)} = \text{rk}_j^{(r)} \pmod{16}$. If $S_{\text{erf}}^{(r)}(x)$ is a negacyclic function, the condition can be improved to $\text{rk}_i^{(r)} = \text{rk}_j^{(r)} \pmod 8$. Both cases have been discussed above. Are there other methods that can further relax the condition? In the following, we describe the second method to enlarge the weak key space.

During each random round, each negacyclic function $S_{\text{erf}}^{(r)}(x)$ over $\mathbb{Z}_{16}$ is randomly generated by an XOF seeded with NC||CTR. Moreover, due to the negacyclic property, only the values of $S_{\text{erf}}^{(r)}(0), S_{\text{erf}}^{(r)}(1), \ldots, S_{\text{erf}}^{(r)}(7)$ are randomly sampled, and $S_{\text{erf}}^{(r)}(8), S_{\text{erf}}^{(r)}(9), \ldots, S_{\text{erf}}^{(r)}(15)$ are accordingly determined. Hence, 32 random bits are required to specify each random 4-bit S-box $S_{\text{erf}}^{(r)}(x)$. With this in mind, we define a *weak* random S-box as follows:

Definition 1 (Weak S-box)**.** We say that the r-th round random S-box $S_{\text{erf}}^{(r)}$ is *weak* if it satisfies the following condition:

$$S_{\text{erf}}^{(r)}(0) = S_{\text{erf}}^{(r)}(1) = \ldots = S_{\text{erf}}^{(r)}(7) \pmod 2, \tag{9}$$

The negacyclic property imposes that if Equation (9) holds, then we have that

$$\forall x \in \mathbb{Z}_{16}: \; S_{\text{erf}}^{(r)}(x) = \boxminus S_{\text{erf}}^{(r)}(x \boxplus 8) \implies S_{\text{erf}}^{(r)}(x) = S_{\text{erf}}^{(r)}(x \boxplus 8) \pmod 2.$$

and thus:

$$S_{\text{erf}}^{(r)}(0) = S_{\text{erf}}^{(r)}(1) = \ldots = S_{\text{erf}}^{(r)}(7) = S_{\text{erf}}^{(r)}(8) = S_{\text{erf}}^{(r)}(9) = \ldots = S_{\text{erf}}^{(r)}(15) \pmod 2.$$

A random negacyclic S-box is *weak* with probability $2^{-8+1} = 2^{-7}$, meaning each random S-box is a weak S-box in our weak key attack with probability 2^{-7}. If a weak random S-box $S_{\text{erf}}^{(r)}(x)$ is used at the r-th round then we have

$$\forall x_1^{(r-1)}, \text{rk}_i^{(r)}, \text{rk}_j^{(r)} \in \mathbb{Z}_{16}: \; S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_i^{(r)}) + S_{\text{erf}}^{(r)}(x_1^{(r-1)} \boxplus \text{rk}_j^{(r)}) = 0 \pmod 2.$$

Hence, there is no need to add conditions on the round key at this round, and the following deterministic iterative relation is still preserved:

$$x_3^{(r)} = x_3^{(r-1)} \pmod 2.$$

This motivates the following definition of weak key, taking into account the possibility for an attacker to find weak S-boxes for w random rounds.

Definition 2 (Weak key)**.** Let $2 \leq i < j \leq 32$, and $I \subset \{r \in \mathbb{N} \mid 1 \leq r \leq 40, r \neq 0 \pmod 5\}, |I| = w$. We say that $\mathbf{k}$ is a *weak key* for branches i, j and weak S-boxes at rounds r for $r \in I$ if:

$$\forall 1 \leq r \leq 40, \; r = 0 \pmod 5, \; \text{rk}_i^{(r)} = \text{rk}_j^{(r)}, \text{ and}$$

$$\forall 1 \leq r \leq 40, \; r \notin I, \; r \neq 0 \pmod 5, \; \text{rk}_i^{(r)} = \text{rk}_j^{(r)} \pmod 8 .$$

In particular, when $\mathbf{k}$ is a weak key for branches i, j and a subset I, and the nonce and counter are chosen such that $S_{\text{erf}}^{(r)}$ is weak whenever $r \in I$, we have that

$$x_3^{(0)} = x_3^{(40)} \pmod 2 .$$

Counting the Number of Weak Keys and Possible Combinations. In the following, we denote the number of rounds featuring weak S-boxes by w where $w \leq 32$. Thus, the weak key condition must apply on $32 - w$ random rounds, and must still apply on 8 full rounds. For each of the $\binom{31}{2}$ choices of branches i, j, there is a set of weak keys of size

$$2^{256-(32-w)\cdot 3-8\cdot 4} = 2^{128+3w}. \tag{10}$$

This indicates that even if there is only one random S-box satisfying Equation (9), i.e., $w = 1$, then a weak key occurs with probability higher than 2^{-128}.

Complexity to Generate Each Combination of w Weak S-boxes. The generation of weak S-boxes can be done offline, as it is only related to NC||CTR which is input to the XOF. For each combination, w S-boxes are required to be weak and their positions are fixed. As a weak S-box occurs with probability 2^{-7}, the offline complexity to generate w weak S-boxes for w pre-determined random rounds is 2^{7w}. Furthermore, to reduce the false-positive probability of our distinguisher, we check whether the 1-bit relation holds for ϵ independent keystreams. For a random function, the probability that the 1-bit relation holds for ϵ independent keystreams is $2^{-\epsilon}$. We set $\epsilon = 128$ in order to obtain a negligible false-positive probability. Thus, for each set of weak keys, the offline complexity of the attack is $\epsilon \cdot 2^{7w}$ and the online complexity is $\epsilon \cdot \binom{31}{2}$.

3.3 Add Key-recovery Rounds

The weak key space can be further increased by a factor of 2^7 at the cost of increasing the complexity of the online phase. In the previous analysis, we have stated that there are several sets of weak keys and the sets are divided according to the choice of w rounds instantiated with weak S-boxes. In this basic attack, to detect whether a weak key is used, we simply test whether there exists a three-branch structure with $\mathcal{B}_{(i,j)}$ such that $x_3^{(40)} = x_3^{(0)} \pmod 2$ always holds for ϵ tests. This test can be improved by involving a key-recovery strategy, which is our third method. In other words, in our third method, we not only improve the weak key space, but also recover partial key bits.

Specifically, in the key-recovery attack on full FRAST, we assume that $\mathrm{rk}_i^{(1)} \neq \mathrm{rk}_j^{(1)} \pmod 8$ at the first round and $\mathrm{rk}_i^{(40)} \neq \mathrm{rk}_j^{(40)}$ at the last round[2]. Note that the first round is instantiated with random S-boxes, while the last round is instantiated with the fixed S-box. Assume that we have found ϵ different NC that can cause w weak S-boxes at the same pre-determined w random rounds where the first round is not included. This is done offline and the complexity is still $\epsilon \cdot 2^{7w}$.

At the online phase, for each NC||CTR, we get an output of the form $\mathbf{y}^{(40)}$. At first we rewrite it using an alternative basis $\mathbf{x}^{(40)} = \mathcal{L}(\mathbf{y}^{(40)})$. Thus, we obtain: $(x_1^{(40)}, x_2^{(40)}, x_3^{(40)})$.

Then, we guess $(\mathrm{rk}_i^{(1)}, \mathrm{rk}_j^{(1)}, \mathrm{rk}_i^{(40)}, \mathrm{rk}_j^{(40)}, \mathrm{rk}_1^{(40)})$ where

$$\mathrm{rk}_i^{(1)} \neq \mathrm{rk}_j^{(1)} \pmod 8, \ \mathrm{rk}_i^{(40)} \neq \mathrm{rk}_j^{(40)}.$$

Therefore, there are about 2^{20} candidates to guess (in fact, slightly less than 2^{20} candidates). For each guess, we want to compute $x_i^{(1)}$ and $x_i^{(39)}$) so that we can check if $x_i^{(1)} = x_i^{(39)} \pmod 2$. We compute $x_i^{(1)}$ and $x_i^{(39)}$ as follows:

$$\begin{aligned} x_3^{(1)} &= x_3^{(0)} \boxplus \left(S_{\mathrm{erf}}^{(1)}(x_1^{(0)} \boxplus \mathrm{rk}_i^{(1)}) \boxminus S_{\mathrm{erf}}^{(1)}(x_1^{(0)} \boxplus \mathrm{rk}_j^{(1)}) \right) && (11) \\ x_1^{(39)} &= x_1^{(40)} \boxminus \left(S_{\mathrm{fixed}}(x_2^{(40)} \boxplus \mathrm{rk}_1^{(40)}) \right) \\ x_3^{(39)} &= x_3^{(40)} \boxminus \left(S_{\mathrm{erf}}^{(1)}(x_1^{(39)} \boxplus \mathrm{rk}_i^{(40)}) \boxminus S_{\mathrm{erf}}^{(40)}(x_1^{(39)} \boxplus \mathrm{rk}_j^{(40)}) \right). && (12) \end{aligned}$$

If weak round keys are used in the middle $38 - w$ rounds, for the correct guess of the 20 key bits, the following relation

$$x_3^{(1)} = x_3^{(39)} \pmod 2 \tag{13}$$

always holds for any NC||CTR. However, due to the special form of Equation (13), there are more than one correct keys left. We call *equivalently correct keys*. For other key guesses, Equation (13) will hold with probability 2^{-1}, and hence it holds for ϵ tests with probability $2^{-\epsilon}$. Again, in our setting, $\epsilon = 128$.

Equivalently Correct Keys. From Equation (11), it is clear that guessing $(\mathrm{rk}_i^{(1)}, \mathrm{rk}_j^{(1)}) = (v_0, v_1) \in \mathbb{Z}_{16}^2$ and $(\mathrm{rk}_i^{(1)}, \mathrm{rk}_j^{(1)}) = (v_1, v_0) \in \mathbb{Z}_{16}^2$ are equivalent and will result in the same value for $x_3^{(1)} \pmod 2$. This similarly applies to the guess of $(\mathrm{rk}_i^{(40)}, \mathrm{rk}_j^{(40)})$ for computing $x_3^{(39)}$. In other words, if the guess $(\mathrm{rk}_i^{(1)}, \mathrm{rk}_j^{(1)}, \mathrm{rk}_i^{(40)}, \mathrm{rk}_j^{(40)}) = (v_0, v_1, v_2, v_3)$ always leads to (13) for ϵ chosen nonces, it is redundant to test the following 3 candidates

$$(v_0, v_1, v_3, v_2), \ (v_1, v_0, v_2, v_3), \ (v_1, v_0, v_3, v_2)$$

[2]If assuming $\mathrm{rk}_i^{(1)} = \mathrm{rk}_j^{(1)} \pmod 8$ and $\mathrm{rk}_i^{(40)} \neq \mathrm{rk}_j^{(40)}$, this will be a similar key-recovery attack where the last round is used for key recovery. If assuming $\mathrm{rk}_i^{(1)} \neq \mathrm{rk}_j^{(1)} \pmod 8$ and $\mathrm{rk}_i^{(40)} = \mathrm{rk}_j^{(40)}$, only the first round is used for key recovery. If both hold, no need to add key-recovery rounds.

since the 4 candidates cannot be distinguished by (13). This reduces the key space to guess by a factor of 4, but also increases the candidates left for the correct key by a factor of 4.

Moreover, as $S^{(1)}_{\mathrm{erf}}(x)$ is a negacyclic function satisfying $\forall x \in \mathbb{Z}_{16}:\ S^{(1)}_{\mathrm{erf}}(x) = \boxminus S^{(1)}_{\mathrm{erf}}(x \boxplus 8)$, we have $\forall x \in \mathbb{Z}_{16}:\ S^{(1)}_{\mathrm{erf}}(x) = S^{(1)}_{\mathrm{erf}}(x \boxplus 8) \pmod 2$. Therefore, we only need to guess $(\mathrm{rk}^{(1)}_i, \mathrm{rk}^{(1)}_j) = (v_0, v_1)$ satisfying $0 \le v_0, v_1 < 8$. This is because after guessing such a (v_0, v_1), we also get

$$\begin{aligned} & \left(S^{(1)}_{\mathrm{erf}}(x^{(0)}_1 \boxplus \mathrm{rk}^{(1)}_i) + S^{(1)}_{\mathrm{erf}}(x^{(0)}_1 \boxplus \mathrm{rk}^{(1)}_j)\right) \\ = \ & \left(S^{(1)}_{\mathrm{erf}}(x^{(0)}_1 \boxplus \mathrm{rk}^{(1)}_i \boxplus 8) + S^{(1)}_{\mathrm{erf}}(x^{(0)}_1 \boxplus \mathrm{rk}^{(1)}_j \boxplus 8)\right) \pmod 2 . \end{aligned}$$

In other words, the following 4 candidates for $(\mathrm{rk}^{(1)}_i, \mathrm{rk}^{(1)}_j)$ cannot be distinguished by (13):

$$(v_0, v_1),\ (v_0 \boxplus 8, v_1),\ (v_0, v_1 \boxplus 8),\ (v_0 \boxplus 8, v_1 \boxplus 8).$$

As a consequence, this further reduces the key space to guess by a factor of 4, but also further increases the number of candidates for the correct key by a factor of 4. In summary, due to the equivalently correct key for Equation (13), we do not need to exhaust about 2^{20} candidates. Instead, only fewer than 2^{16} candidates need to be tested.

Complexity Evaluation. The complexity of the offline phase is $\epsilon \cdot 2^{7w} = 2^{7w+7}$. The online complexity is $2^{16} \cdot \epsilon \cdot \binom{31}{2} = 2^{23} \cdot \binom{31}{2}$. There are still $\binom{31}{2}$ sets of weak keys, and each set is of size

$$2^{256-(31-w)\cdot 3-7\cdot 4} = 2^{135+3w}.$$

This increases the number of weak keys by a factor of 2^7 compared with the second method under the same value of w. The offline complexity remains the same as for the second method, but the online complexity of the third method increases by a factor of 2^{16}.

Experimental Validation on Reduced-Round FRAST. To provide a proof of concept for our theoretical framework, we conducted an experimental verification of the key-recovery attack on a 10-round version of FRAST, (8 random and 2 fixed rounds). In our experiments, we set $w = 2$ and applied the techniques for enlarging the weak key space via the negacyclic properties discussed above. The implementation successfully recovered a total of 20 bits of the round key: 8 bits from the initial round and 12 bits from the final rounds. This result shows the validity of our proposed distinguisher. A complete Rust implementation of our experiment is available at https://github.com/ShibamCrS/Cryptanalysis_FRAST.git, with detailed instructions in the accompanying `README` file.

3.4 The Number of Weak Keys

In the above analysis, we assume that each condition

$$\mathrm{rk}^{(r)}_i = \mathrm{rk}^{(r)}_j \pmod{16} \text{ or } \mathrm{rk}^{(r)}_i = \mathrm{rk}^{(r)}_j \pmod 8$$

reduces the weak key space by a factor of 2^4 and 2^3, respectively. Is this a reasonable estimate? In this section, we will address this issue.

First, according to the key schedule of FRAST, each round key nibble $\mathrm{rk}^{(r)}_i$ is represented by a linear polynomial over $\mathbb{Z}_{16}$ in the master key $\boldsymbol{k} = (k_1, k_2, \ldots, k_{64}) \in \mathbb{Z}^{64}_{16}$. Moreover, $\mathrm{rk}^{(r)}_i = \mathrm{rk}^{(r)}_j \pmod 8$ is equivalent to

$$\mathrm{rk}^{(r)}_i \boxminus \mathrm{rk}^{(r)}_j = 0 \text{ or } \mathrm{rk}^{(r)}_i \boxminus \mathrm{rk}^{(r)}_j = 8.$$

Hence, all conditions can be written as a set of linear equations in $(k_1, k_2, \ldots, k_{64})$ over $\mathbb{Z}_{16}$.

For the three-branch structure with $\mathcal{B}_{(i,j)}$, we consider the case where there are 40 conditions specified in (14), i.e., note that this corresponds to increasing weak keys with the negacyclic property of the random S-boxes.

$$\mathrm{rk}_i^{(r)} \boxminus \mathrm{rk}_j^{(r)} = \delta_r, \quad \delta_r = \begin{cases} 0, & \text{if } r \equiv 0 \pmod 5, \\ 0 \text{ or } 8, & \text{otherwise.} \end{cases} \tag{14}$$

The linear equation system over $\mathbb{Z}_{16}$ corresponding to these 40 conditions is denoted by

$$\boldsymbol{M}_{i,j} \cdot \boldsymbol{k} = \boldsymbol{b} \pmod{16}, \tag{15}$$

where $\boldsymbol{M}_{i,j} \in \mathbb{Z}_{16}^{40\times 64}$ is the coefficient matrix, and $\boldsymbol{b} \in \mathbb{Z}_{16}^{40}$ is parameterized by $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_{40}) \in \mathbb{Z}_{16}^{40}$. Note that $\boldsymbol{b}$ can take 2^{32} possible values since there are 32 random rounds.

Studying the weak key set for the three-branch structure with $\mathcal{B}_{(i,j)}$ is equivalent to studying the solution space of the linear equation system (15). If the equation system is defined over a finite field, computing the rank of $\boldsymbol{M}_{i,j}$ is enough. However, (15) is over the integer ring $\mathbb{Z}_{16}$. Even if $\boldsymbol{M}_{i,j}$ is of full rank, i.e., all rows are linearly independent over $\mathbb{Z}_{16}$, it does not necessarily mean that there are $16^{64-40} = 2^{96}$ solutions for any $\boldsymbol{b}$.

The Number of Solutions to Equation (15). For each $\boldsymbol{M}_{i,j}$, we use Gaussian elimination to find 40 columns that can form an invertible square matrix over $\mathbb{Z}_{16}$, and denote the matrix formed by these 40 columns of $\boldsymbol{M}_{i,j}$ by $\boldsymbol{M}_{i,j}^{\text{left}} \in \mathbb{Z}_{16}^{40\times 40}$. The matrix formed by the remaining $64-24$ columns of $\boldsymbol{M}_{i,j}$ is denoted by $\boldsymbol{M}_{i,j}^{\text{right}} \in \mathbb{Z}_{16}^{40\times 24}$. Specifically, after applying Gaussian elimination to $\boldsymbol{M}_{i,j}^{\text{left}}$ to obtain its reduced row echelon form, it can become the identity matrix. It is found that for each possible (i,j), it is always possible to find such an invertible square matrix $\boldsymbol{M}_{i,j}^{\text{left}}$. This implies that the number of solutions to Equation (15) is exactly $2^{96+32} = 2^{128}$. Specifically, the equation can always be rewritten as

$$\boldsymbol{M}_{i,j}^{\text{left}} \cdot \boldsymbol{k}_L = \boldsymbol{b}' - \boldsymbol{M}_{i,j}^{\text{right}} \cdot \boldsymbol{k}_R \pmod{16}, \tag{16}$$

where $\boldsymbol{k}_L \in \mathbb{Z}_{16}^{40}, \boldsymbol{k}_R \in \mathbb{Z}_{16}^{24}$. Since the matrix

$$\left[\boldsymbol{M}_{i,j}^{\text{left}} \;\middle|\; \boldsymbol{M}_{i,j}^{\text{right}}\right]$$

is obtained by permuting the columns of $\boldsymbol{M}_{i,j}$, $\begin{pmatrix}\boldsymbol{k}_L \\ \boldsymbol{k}_R\end{pmatrix}$ and $\boldsymbol{b}'$ are obtained by permuting the rows of $\boldsymbol{k}$ and $\boldsymbol{b}$, respectively. Note that $\boldsymbol{b}'$ can take 2^{32} possible values, while $\boldsymbol{k}_R$ can take $16^{24} = 2^{96}$ possible values. As $\boldsymbol{M}_{i,j}^{\text{left}}$ is invertible over $\mathbb{Z}_{16}$, there are in total 2^{128} solutions to $\boldsymbol{k}$.

The above analysis also indicates that the number of solutions is 2^{128+3w} if removing w equations from (16) with the second method, and that the number of solutions is 2^{135+3w} if removing $w+2$ equations from (16) with the third method. Hence, the claim for the size of each set of weak keys in the above analysis is correct.

Intersection Between Different Sets of Weak Keys. If the intersection between different sets of weak keys is small enough compared with each set, after applying the third method, it is reasonable to claim that the total number of weak keys is about

$$\binom{31}{2} \cdot 2^{135+3w} = 2^{143.9+3w}.$$

Indeed, we can interpret the intersection between two different sets of weak keys from the perspective of the solutions to a larger linear equation system. For two sets corresponding to the coefficient matrices $\boldsymbol{M}_{i,j}$ and $\boldsymbol{M}_{i^\star,j^\star}$ where $(i,j) \neq (i^\star,j^\star)$, the intersection between the two sets corresponds to the solutions to the following linear equation system:

$$\begin{pmatrix} \boldsymbol{M}_{i,j} \\ \boldsymbol{M}_{i^\star,j^\star} \end{pmatrix} \cdot \boldsymbol{k} = \begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{b}^\star \end{pmatrix} \pmod{16},$$

where $\begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{b}^\star \end{pmatrix}$ can take 2^{64} possible values. However, as can be observed, compared with the case where one set of weak keys is studied, the number of equations to consider here is doubled. Hence, even if there are solutions to this linear equation system, the number is too small compared with 2^{128}. To partially verify this, for all $\binom{\binom{31}{2}}{2} = 107880$ possible combinations, we have checked that the number of solutions is smaller than 100 when the right-hand side is set to zero. Hence, we believe that it is reasonable to estimate the number of weak keys as $2^{143.9+3w}$. Or we can simply use a conservative estimate, that is, we could consider only the number of weak keys in each set, which is a lower bound on the number of weak keys, i.e., the lower bound is 2^{135+3w}.

3.5 Distinguishing Attack on FRAST Keystream Generator

We now present our distinguishing attack on the FRAST keystream generator, where it is treated as a Pseudo-Random Function (PRF). The foundation of our attack is the deterministic linear relationship from Equation (5), which, as established in Section 2.3, holds for any key belonging to the weak key class. Based on this property, we construct a deterministic distinguisher, $\mathcal{D}$, which interacts with a challenger oracle. The Challenger and the full algorithm for $\mathcal{D}$ are provided in Algorithm 1 and Algorithm 2, respectively. We note that the master key $\mathbf{k}$ used by the challenger is randomly chosen by the challenger and is fixed throughout the duration of the distinguishing game.

Algorithm 1 $\mathsf{Challenger}^{\mathbf{k}}$

Require: $\mathbf{x} = \text{NC}||\text{CTR}$
Ensure: A keystream in $\mathbb{Z}_{16}^{32}$
1: Samples $b \leftarrow \{0,1\}$ uniformly at random;
2: **if** $b = 1$ **then**
3: **return** $\mathsf{FRAST}[\mathbf{k}, \mathbf{x}]$
4: **end if**
5: **return** a random value in $\mathbb{Z}_{16}^{32}$

Complexity Analysis. Here we summarize the complexity of the attack. The attack consists of an offline pre-computation phase and an online phase. The pre-computation phase involves key-independent computation, with its time complexity denoted by T_{precomp}. The online phase includes computations performed after the first query, measured in bit operations and table lookups, and the time complexity of this phase is denoted by T_{online}.

- **Pre-computation Phase:** The goal of this phase is to construct a set X_{weak} containing ϵ inputs $\mathbf{x} = \text{NC}||\text{CTR}$ such that each of these inputs induce a *weak* S-box (as defined in Section 3.2) at w predetermined random rounds. The repetition parameter ϵ is chosen to filter the false positive cases in the online phase. The generation of weak S-boxes can be done offline, as it is only related to NC||CTR which

Algorithm 2 Deterministic Distinguisher $\mathcal{D}$

Ensure: A binary value χ
Precompute a set X_{weak} of weak NC||CTR of size ϵ as discussed in Section 3.2;
Define a binary array γ of size $\binom{32}{2}$ indexed by (i,j) for $2 \leq i < j \leq 32$, initialized to 1
for $\mathbf{x} = \text{NC}||\text{CTR} \in X_{\text{weak}}$ **do**
    query $\mathbf{x}$ to get $\mathbf{y}_x = \mathsf{Challenger}^{\mathbf{k}}(\mathbf{x}) \in \mathbb{Z}_{16}^{32}$;
    for all $2 \leq i < j \leq 32$ **do**
        if $y_{x,i} \boxminus y_{x,j} \neq (\text{IC}_i \boxminus \text{IC}_j) \pmod 2$ **then** ▷ Checking Equation (5) for $\mathcal{B}_{(i,j)}$
            $\gamma[(i,j)] = 0$
        end if
    end for
end for
for all $2 \leq i < j \leq 32$ **do**
    if $\gamma[(i,j)] = 1$ **then**
        return 1
    end if
end for
return 0

is input to the XOF. Given that a weak S-box occurs with probability 2^{-7}, finding a single NC||CTR that gives weak S-boxes for w rounds requires approximately 2^{7w} evaluations of the XOF. To construct the full set X_{weak} of ϵ different NC||CTR such that each leads to w weak S-boxes at the same w random rounds has a complexity of $T_{\text{precomp}} = \epsilon \cdot 2^{7w}$ evaluations of the XOF.

- **Online Phase:** During the online phase, the distinguisher queries each of the ϵ inputs from X_{weak} to the challenger. For each response, it checks the linear relation across all $\binom{31}{2}$ possible three-branch structures $\mathcal{B}_{(i,j)}$. The total online time complexity is therefore $T_{\text{online}} = \epsilon \cdot \binom{31}{2}$. Furthermore, if the key recovery is added (as discussed in Section 3.3), the online time complexity increases by a factor of 2^{16}, i.e., $T_{\text{online}} = \epsilon \cdot 2^{16} \cdot \binom{31}{2}$.

The memory complexity is to store ϵ different NC||CTR which is negligible. A summary of these complexities for various choices of w, considering scenarios both with and without the final round key recovery, is provided in Table 2.

Distinguishing Advantage. The advantage of our distinguisher $\mathcal{D}$ is its ability to distinguish the *real world* (where the challenger uses FRAST) from the *ideal world* (where the challenger returns random outputs). In the real world, if the challenger's key $\mathbf{k}$ is a weak key corresponding to some structure $\mathcal{B}_{(i,j)}$, the following linear relation holds for all ϵ tests

$$x_3^{(40)} = x_3^{(0)} \pmod 2 \implies y_i^{(40)} + y_j^{(40)} = y_i^{(0)} + y_j^{(0)} = \text{IC}_i + \text{IC}_j \pmod 2 ,$$

and $\mathcal{D}$ outputs 1. If $\mathbf{k}$ is not a weak key, the relation behaves randomly, and the probability of it holding for all ϵ queries for any single structure is $2^{-\epsilon}$. On the other hand, in the ideal world, the challenger's outputs are random, so the probability of the linear relation holding for all ϵ queries for any given structure is also $2^{-\epsilon}$.

Let p_{weak} be the probability that a randomly chosen master key belongs to a weak key

Table 2: Complexity of the distinguishing attack for various numbers of weak S-boxes w.

w	T_{precomp}	T_{online}	$\text{Adv}(\mathcal{D})$	**Last Round Key Recovery**
13	2^{98}	$2^{15.86}$	$2^{-80.14}$	no
17	2^{126}		$2^{-68.14}$	
14	2^{105}	$2^{31.86}$	$2^{-70.14}$	yes
17	2^{126}		$2^{-61.14}$	

space. Thus, the total distinguishing advantage of $\mathcal{D}$ is given by

$$\begin{aligned}\text{Adv}(\mathcal{D}) &:= \Pr_{\mathbf{k}}[\mathcal{D} \text{ outputs } 1 \mid b = 1] - \Pr_{\mathbf{k}}[\mathcal{D} \text{ outputs } 1 \mid b = 0] \\ &= \binom{31}{2}\left(p_{\text{weak}} + (1 - p_{\text{weak}}) \cdot \frac{1}{2^{\epsilon}} - \frac{1}{2^{\epsilon}}\right) \\ &= \binom{31}{2} p_{\text{weak}}\left(1 - \frac{1}{2^{\epsilon}}\right) \approx \binom{31}{2} p_{\text{weak}},\end{aligned}$$

for a sufficiently large ϵ (e.g., $\epsilon = 128$). The calculated complexities and corresponding advantages are detailed in Table 2. We can also consider $\epsilon = 256$ with slight increases in the complexities.

Discussion and Implications. Since the FRAST specification does not define a formal security model against distinguishing attacks, we establish the following metric for success: our attack is considered meaningful if its advantage surpasses that of a generic attack with the same online complexity on a 128-bit key cipher, i.e., $\text{Adv}(\mathcal{D}) > T_{\text{online}}/2^{128}$. The advantage of generic attack is 2^{-112} or 2^{-96} depending on whether we consider the last round key recovery. Our distinguisher achieves advantages over the generic attack for a wide range of values of w, some of them are shown in Table 2.

Furthermore, we can strategically select the parameter w to balance the attack's trade-offs. To equalize the offline pre-computation cost T_{precomp} with $T_{\text{online}}/\text{Adv}(\mathcal{D})$, we can set $2^{7+7w} \approx 2^{7}\binom{31}{2} \cdot 2^{128-3w}$ (with no key recovery step), which yields an optimal value of $w = 13$, and $2^{7+7w} \approx 2^{7+16}\binom{31}{2} \cdot 2^{128-3w}$ (with key recovery step), which yields an optimal value of $w = 14$. Also, if we aim to maximize the probability that a weak key occurs, we can choose $w = 17$. In this case, the pre-computation complexity is 2^{126}, and a weak key occurs in each set with probability $2^{-68.14}$ or $2^{-61.14}$ depending on whether we consider the last round key recovery.

3.6 Distinguishing Attack on FRAST Keystream Generator In a Multi-User Setting

We now extend our analysis on FRAST to the multi-user setting, a standard model for evaluating the security of cryptographic primitives when deployed at scale [BBM00]. In this scenario, an adversary's goal is to distinguish FRAST keystreams generated from a PRF for at least one of many users, each of whom possesses an independently and randomly generated key. This model is particularly relevant for weak key attacks, as the adversary's probability of encountering at least one weak key increases with the number of users.

Our multi-user attack is a natural generalization of the distinguisher presented in Section 3.5. In that setting, we interact with a challenger, $\text{Challenger}_{\text{MU}}$, and construct a new distinguisher against it, $\mathcal{D}_{\text{MU}}$, as detailed in Algorithm 3 and Algorithm 4, respectively. The set of keys for all users, $\mathbb{K} = \{\mathbf{k}_1, \ldots, \mathbf{k}_{N_{\text{user}}}\}$, is sampled once at the beginning of the

game and remains fixed throughout. The challenger, upon receiving a query for a user u, will use the corresponding key $\mathbf{k}_u$ in the real world to generate the keystream.

Algorithm 3 $\mathsf{Challenger}^{\mathbb{K}}_{\mathsf{MU}}$

Require: $(u, \mathbf{x})$ where $\mathbf{x} = \text{NC}||\text{CTR}$
Ensure: A keystream in $\mathbb{Z}_{16}^{32}$
Samples $b \leftarrow \{0, 1\}$ uniformly at random;
if $b = 1$ **then**
  return $\mathsf{FRAST}[\mathbf{k}_u, \mathbf{x}]$
end if
return a random value in $\mathbb{Z}_{16}^{32}$

Algorithm 4 Deterministic Distinguisher $\mathcal{D}_{\mathsf{MU}}$

Ensure: A binary value χ
Precompute a set X_{weak} of weak NC||CTR of size ϵ as discussed in Section 3.2;
for $u = 1, \ldots, N_{\text{user}}$ **do**
  Define a binary array γ of size $\binom{32}{2}$ indexed by (i, j) for $2 \leq i < j \leq 32$, initialized to 1
  for $\mathbf{x} = \text{NC}||\text{CTR} \in X_{\text{weak}}$ **do**
    query $\mathbf{x}$ to get $\mathbf{y}_x = \mathsf{Challenger}^{\mathbb{K}}_{\mathsf{MU}}(u, \mathbf{x}) \in \mathbb{Z}_{16}^{32}$;
    for all $2 \leq i < j \leq 32$ **do**
      if $y_{x,i} \boxminus y_{x,j} \neq (\text{IC}_i \boxminus \text{IC}_j) \pmod 2$ **then** ▷ Checking Equation (5) for $\mathcal{B}_{(i,j)}$
        $\gamma[(i, j)] = 0$
      end if
    end for
  end for
  for all $2 \leq i < j \leq 32$ **do**
    if $\gamma[(i, j)] = 1$ **then**
      return 1
    end if
  end for
end for
return 0

The complexity of the offline pre-computation phase, which involves finding the set X_{weak}, is identical to the previous attack: $T_{\text{offline}} = \epsilon \cdot 2^{7w}$ evaluations of the XOF. Note that we can amortize the cost of finding X_{weak} and use it for multiple users (like in Hellman's time memory trade-off attacks [Hel80]). However, the online complexity increases, scaling linearly with the number of users to $T_{\text{online}} = N_{\text{user}} \cdot \epsilon \cdot \binom{31}{2}$ (or $T_{\text{online}} = N_{\text{user}} \cdot \epsilon \cdot 2^{16} \cdot \binom{31}{2}$ when key recovery is included).

Finally, as the user keys are sampled randomly, the distinguishing advantage of $\mathcal{D}_{\mathsf{MU}}$ is also increased by the number of users, i.e., the probability that one user out of N_{user} has a weak key follows a geometric probability distribution with parameter $\binom{31}{2} \cdot p_{\text{weak}}$. This yields $\text{Adv}(\mathcal{D}_{\mathsf{MU}}) = 1 - (1 - \binom{31}{2} \cdot p_{\text{weak}})^{N_{\text{user}}}$. Thus, taking $N_{\text{user}} \simeq 2/\binom{31}{2} \cdot p_{\text{weak}}$, we get distinguishing advantage close to $1 - e^{-2} \simeq 0.85$. The complexities and number of users for several tradeoffs are given in Table 3. We observe that we can achieve an advantage close to 1 for a number of users as low as $2^{62.14}$, and we still have $\text{Adv}(\mathcal{D}_{\mathsf{MU}}) > T_{\text{online}}/2^{128}$.

Table 3: Complexity of the distinguishing attack for various numbers of weak S-boxes w and numbers of users N_{user}.

w	N_{user}	T_{precomp}	T_{online}	$\text{Adv}(\mathcal{D}_{\mathsf{MU}})$	**Last Round Key Recovery**
13	$2^{81.14}$	2^{98}	2^{97}	0.85	no
17	$2^{69.14}$	2^{126}	2^{85}		
14	$2^{71.14}$	2^{105}	2^{103}	0.85	yes
17	$2^{62.14}$	2^{126}	2^{94}		

4 An Attempt at Linear Cryptanalysis

In this section, we present a general framework for the linear cryptanalysis of ciphers that employ randomized S-boxes, with a specific application to FRAST. Our primary objective was to mount a key-recovery attack in the *average key* setting. However, our investigation revealed that several common assumptions underpinning statistical cryptanalysis do not hold for our attack, a phenomenon we detail in Section 4.4. Although the attack was ultimately unsuccessful in the average key setting, we believe our findings are valuable and important to report for two main reasons. First, the formalism we develop for analyzing linear properties over randomized components (Section 4.2) is interesting for future research into similar cryptographic designs. Second, the failure of the attack itself is insightful, as it shows the limits of commonly made statistical assumptions.

4.1 Linear Cryptanalysis Over Non-binary Rings

Linear cryptanalysis, first introduced by Matsui to attack the FEAL block cipher [MY92], is an important known-plaintext attack. The core principle of this statistical method is to exploit an approximate linear relationship between plaintext, ciphertext, and key bits. Once such an equation is found, it can be used to make a guess at some of the key bits.

The original framework for linear cryptanalysis was developed over the binary field $\mathbb{F}_2$. We assume the reader is familiar with the basic principles of linear cryptanalysis over the binary field $\mathbb{F}_2$. Here, we begin by generalizing this concept to non-binary rings and recall some background. For simplicity, we restrict ourselves to the rings $\mathbb{Z}_t$ of integers modulo t. Following the approach in [BSV07], we use a primitive t-th root of unity, $\zeta_t = e^{i2\pi/t}$, to define the correlation of a function $F\colon \mathbb{Z}_t^m \to \mathbb{Z}_t^n$ for given input and output masks $b \in \mathbb{Z}_t^m$ and $a \in \mathbb{Z}_t^n$:

$$C^F(a,b) = \frac{1}{t^m} \sum_{\mathbf{x}\in\mathbb{Z}_t^m} \zeta_t^{a\cdot F(\mathbf{x})-b\cdot\mathbf{x}} \ .$$

Note that in the case where $t = 2$, we get $\zeta_2 = -1$, and $C^F(a,b)$ is simply the normalized Walsh transform of F, as in binary linear cryptanalysis.

In general, we study key-alternating ciphers $E_{\mathbf{k}}\colon \mathbb{Z}_t^n \to \mathbb{Z}_t^n$ that are of the form

$$E_{\mathbf{k}} = k_R + \mathcal{R}^{(R)} \circ \cdots \circ \mathcal{R}^{(1)} \ , \text{ where } \mathcal{R}^{(r)}(\mathbf{x}) = \mathcal{R}(\mathbf{x} + k_{r-1}) \ .$$

In that setting, the linear correlation of $E_{\mathbf{k}}$ with masks a, b is as follows:

$$C^{E_{\mathbf{k}}}(a,b) = \zeta_t^{a_R \cdot k_R} \cdot \sum_{\substack{a_r \in \mathbb{Z}_t^n,\ 1 \le r \le R-1 \\ a_0 = b,\ a_R = a}} \prod_{r=1}^{R} C^{\mathcal{R}^{(r)}}(a_r, a_{r-1})$$

$$= \zeta_t^{a_R \cdot k_R} \cdot \sum_{\substack{a_r \in \mathbb{Z}_t^n,\ 1 \le r \le R-1 \\ a_0 = b,\ a_R = a}} \prod_{r=1}^{R} C^{\mathcal{R}}(a_r, a_{r-1}) \cdot \zeta_t^{a_{r-1} \cdot k_{r-1}} \ .$$

We call a sequence of masks $b = a_0 \rightsquigarrow a_1 \rightsquigarrow \cdots \rightsquigarrow a_R = a$ a *linear trail*, and define the correlation of the trail as

$$\zeta_t^{\sum_{r=0}^{R} a_r \cdot k_r} \cdot \prod_{r=1}^{R} C^{\mathcal{R}}(a_r, a_{r-1}) \ .$$

The objective of linear cryptanalysis is to find masks a, b such that $|C^{E_{\mathbf{k}}}(a,b)|$ is sufficiently large over random $\mathbf{k}$. Since the exact computation of this value is generally infeasible, cryptanalysts often rely on heuristics, the most common of which is the *dominant trail* assumption, which states that there exists a linear trail $b = a_0 \rightsquigarrow a_1 \rightsquigarrow \cdots \rightsquigarrow a_R = a$ such that:

$$\left|C^{E_{\mathbf{k}}}(a,b)\right| \simeq \prod_{r=1}^{R} \left|C^{\mathcal{R}}(a_r, a_{r-1})\right| \ ,$$

i.e., the cipher's correlation is well-approximated by the correlation of a single, dominant linear trail. We will use this assumption throughout the next subsections.

While computing the correlation $C^{\mathcal{R}}(a,b)$ for a generic function is hard, for SPN or Feistel-based designs it decomposes into a more manageable form. The propagation of linear masks through a round function is based on its layers and there are known trail propagation rules for various cases (detailed for the non-binary case in [BSV07, Section3.2]), with the S-box layer being the most critical. A key implication of these propagation rules is that the correlation of one round function can be expressed as the product of the correlations of its individual S-boxes:

$$C^{\mathcal{R}}(a,b) = \prod_{\text{S-box } j} C^{S_j}(a_j, b_j)$$

for some masks a_j, b_j. We call an S-box active if its correlation is not 1. This means the correlation of round function is effectively the product of the correlations of all its active S-boxes:

$$C^{\mathcal{R}}(a,b) = \prod_{\text{active S-box } j} C^{S_j}(a_j, b_j)$$

and consequently the correlation of a trail is of the form:

$$\prod_{r=1}^{R} \prod_{\substack{\text{active S-box } j \\ \text{at round } r}} |C^{S_{r,j}}(a_{r,j}, b_{r,j})| \ .$$

Up to reindexing the active S-boxes, we can also write it as a product over the active S-boxes through all rounds:

$$\prod_{\text{active S-box } i} |C^{S_i}(a_i, b_i)| \ .$$

4.2 Extending linear attacks on ciphers with random S-boxes

In this section, we present a general framework for doing linear attacks on ciphers with randomized components, and in particular, public randomized S-boxes. The setting is as follows: we have a family of Feistel (or SPN) block ciphers $(E_{\mathbf{k},\mathtt{IV}})_{\mathbf{k}\in\mathcal{K},\mathtt{IV}\in\{0,1\}^n}$. Those ciphers all share the same structure (i.e., the same key schedule, linear layer for an SPN and same branch rotations for a Feistel), but the S-boxes $\mathbf{S}$ depend on the initialization vector IV. The adversary can choose the IV for each query but not the plaintext itself, which may be fixed to a constant (e.g., IC).

To understand the challenges this model presents, let us first recall the standard setting with fixed S-boxes. A linear trail whose active S-boxes S_i have masks a_i, b_i and corresponding key k_i as input will have correlation of the form:

$$c = \prod_{\text{active S-box } i} C^{S_i}(a_i, b_i) \cdot \zeta_t^{b_i \cdot k_i} \ .$$

An adversary can detect this correlation by computing an empirical linear correlation over $N_{\text{data}} \approx 1/|c|^2$ known plaintexts, which is defined as:

$$\hat{c} = \frac{1}{N_{\text{data}}} \sum_{u=1}^{\text{N}_{\text{data}}} \zeta_t^{a \cdot E_{\mathbf{k}}(x_u) - b \cdot x_u} \ .$$

In this case, with non-negligible probability we have that $|\hat{c}| \geq |c|/2$, allowing us to distinguish $E_{\mathbf{k}}$ from a random permutation.

This approach fails when the S-boxes are randomized and change at each call to the ciphe. Here, the trail's correlation c becomes a function of the IV used for each query:

$$c(\mathtt{IV}) = \prod_{\text{active S-box } i} C^{S_i(\mathtt{IV})}(a_i, b_i) \cdot \zeta_t^{b_i \cdot k_i} \ .$$

This dependence on IV introduces two critical problems. First, a trail that is dominant for one IV may not be for another. Second, and more fundamentally, the correlations $c(\mathtt{IV}_u)$ are complex numbers whose argument or phase may vary depending on the IV. To overcome these challenges, we introduce a two-part strategy. The first problem, ensuring trail dominance, can be tackled by selecting a trail that activates a minimal number of S-boxes, so that other trails activate much more S-boxes and thus have lower correlation. To this technique, we add a pre-computation step of a set of "good" IVs, for which this trail's correlation magnitude is maximized. On the other hand, the second challenge requires an additional term in the definition of $\hat{c}$. Indeed, taking the expectation of $\hat{c}$ over random choices of x_u would give:

$$\mathbb{E}_{\mathbf{x}}(\hat{c}) \ = \ \frac{1}{N_{\text{data}}} \sum_{u=1}^{N_{\text{data}}} \mathbb{E}_{x_u}\left(\zeta_t^{a \cdot E_{\mathbf{k},\mathtt{IV}_u}(x_u) - b \cdot x_u}\right) \ \simeq \ \frac{1}{N_{\text{data}}} \sum_{u=1}^{N_{\text{data}}} c(\mathtt{IV}_u) \ .$$

When there is no dependence on IV, this simply gives $\mathbb{E}_{\mathbf{x}}(\hat{c}) = c$. However, in the case where the complex arguments of the correlation $c(\mathtt{IV})$ differ from each other, they may simply cancel each other out based on destructive interference of the arguments. This motivates the introduction of the following *phase correction* function:

$$g(\mathtt{IV}) \ = \ \prod_{\text{active S-box } i} \text{sgn}\left(\overline{C^{S_i(\mathtt{IV})}(a_i, b_i)}\right) \ = \ \text{sgn}(\overline{c(\mathtt{IV})}) \cdot \prod_{\text{active S-box } i} \zeta_t^{b_i \cdot k_i} \ ,$$

where $\text{sgn}(\rho e^{i\theta}) = e^{i\theta}$ for $\rho > 0, \theta \in \mathbb{R}$ and $\text{sgn}(0) = 0$. Then, we define the corrected empirical correlation $\hat{c}$ as:

$$\hat{c} \ = \ \frac{1}{N_{\text{data}}} \sum_{u=1}^{\text{N}_{\text{data}}} \zeta_t^{a \cdot E_{\mathbf{k},\mathtt{IV}_u}(x_u) - b \cdot x_u} \cdot g(\mathtt{IV}_u) \ .$$

This way, we have that:

$$\begin{aligned}\mathbb{E}_{\mathbf{x}}(\hat{c}) &= \frac{1}{N_{\text{data}}} \sum_{u=1}^{N_{\text{data}}} \mathbb{E}_{x_u}\left(\zeta_t^{a\cdot E_{\mathbf{k},\mathtt{IV}_u}(x_u)-b\cdot x_u}\right)\cdot g(\mathtt{IV}_u) \\ &\simeq \frac{1}{N_{\text{data}}} \sum_{u=1}^{N_{\text{data}}} c(\mathtt{IV}_u)\cdot \operatorname{sgn}(\overline{c(\mathtt{IV}_u)})\cdot \prod_{\text{active S-box } i} \zeta_t^{b_i\cdot k_i} \\ &= \frac{1}{N_{\text{data}}} \sum_{u=1}^{N_{\text{data}}} |c(\mathtt{IV}_u)|\cdot \prod_{\text{active S-box } i} \zeta_t^{b_i\cdot k_i}\ .\end{aligned}$$

The phase correction function g then allows the trails for the different $\mathtt{IV}$ to all go in the same direction instead of canceling each other.

4.3 The Case of FRAST

We now apply the general framework discussed in the previous sections to FRAST. For our analysis, we use the alternative representation of the cipher defined in Section 2.2), with branches i, j. The linear trail we will exploit is the iterated trail:

$$(0,0,a) \rightsquigarrow (0,0,a) \rightsquigarrow \cdots \rightsquigarrow (0,0,a)\ .$$

The trail is illustrated in Figure 3. Setting $\Delta S_{ij}^{(r)}(x) = S_{\text{erf}}^{(r)}(x \boxplus \Delta k_{ij}^{(r)}) \boxminus S_{\text{erf}}^{(r)}(x)$ and $\Delta k_{ij}^{(r)} = \text{rk}_i^{(r)} \boxminus \text{rk}_j^{(r)}$, we get that this trail has correlation:

$$\prod_{r=1}^{R} C^{\Delta S_{ij}^{(r)}}(a,0)\ .$$

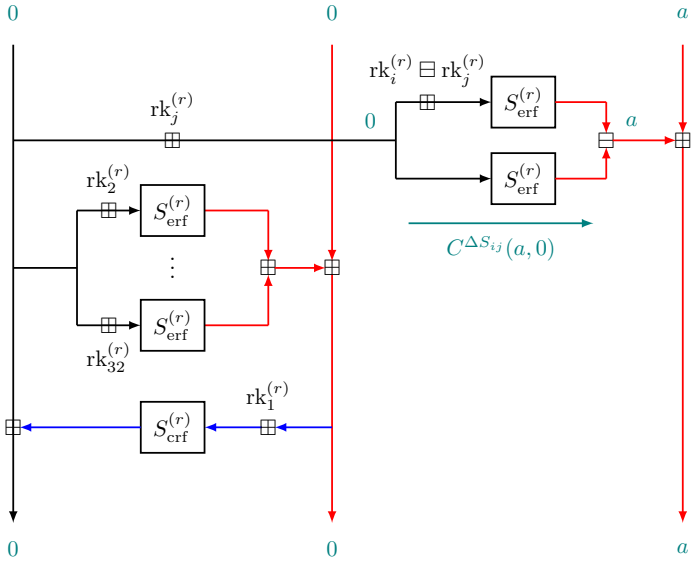

Figure 3: Iterated linear trail for FRAST.

Note that in that case, the trail's correlation has a non-trivial dependency on the secret key. This dependency arises because each function $\Delta S_{ij}^{(r)}$ is itself a function of the round

key difference $\Delta k_{ij}^{(r)} = \mathrm{rk}_i^{(r)} \boxminus \mathrm{rk}_j^{(r)}$. As a consequence, the function g is redefined as a function of both the $\mathtt{IV}$ and the key $\mathbf{k}$:

$$g(\mathtt{IV}, \mathbf{k}) = \mathrm{sgn}\left(\prod_{r=1}^{R} \overline{C^{\Delta S_{ij}^{(r)}}(a,0)}\right) ,$$

and will depend on the key materials $\Delta k_{ij}^{(r)} = \mathrm{rk}_i^{(r)} \boxminus \mathrm{rk}_j^{(r)}$. Thus, we will have to guess them while computing the empirical correlation $\hat{c}$. The outline of the attack is as follows:

1. Take T_{precomp} nonces and compute the corresponding random S-boxes. Keep N_{data} nonces according to some criteria specified later.

2. Query these N_{data} nonces to get $(\boldsymbol{y}_i^{(R)}, \boldsymbol{y}_j^{(R)})$ where

$$\boldsymbol{y}_i^{(R)} = (y_{i,1}^{(R)}, \ldots, y_{i,N_{\mathrm{data}}}^{(R)}),\ \boldsymbol{y}_j^{(R)} = (y_{j,1}^{(R)}, \ldots, y_{j,N_{\mathrm{data}}}^{(R)}).$$

   For the u-th nonce, $(y_{i,u}^{(R)}, y_{j,u}^{(R)})$ denote the output keystream words at the i-th and j-th branches. In addition, $y_{i,u}^{(1)}, y_{j,u}^{(1)}, y_{i,u}^{(R-1)}, y_{j,u}^{(R-1)}$ denote the outputs at the i-th and j-th branches after 1 round and $R-1$ rounds, respectively.

3. Guess some key material in the external rounds to obtain

$$(y_{i,u}^{(1)}, y_{j,u}^{(1)}, y_{i,u}^{(R-1)}, y_{j,u}^{(R-1)}),$$

   i.e., as in our key-recovery attack in the weak key setting. In addition, guess $t_r = \mathrm{rk}_i^{(r)} \boxminus \mathrm{rk}_j^{(r)}$ for the internal rounds where $2 \le r \le R-1$.

4. For each guess, compute a "corrected" empirical correlation defined as

$$\hat{c} = \frac{1}{N_{\mathrm{data}}} \sum_{u=1}^{N_{\mathrm{data}}} g\left(\boldsymbol{S}_u, t_2, \ldots, t_{R-1}\right) \cdot \zeta_{16}^{a \cdot \tilde{x}_u} .$$

   where $\zeta_{16} = e^{2i\pi/16}$ is a primitive 16-th root of unity, $a \in \mathbb{Z}_{16}$ is a mask, g is a "phase correction" function, and $\tilde{x}_u$ is the value of the difference

$$\tilde{x}_u = (y_{i,u}^{(R-1)} \boxminus y_{j,u}^{(R-1)}) \boxminus (y_{i,u}^{(1)} \boxminus y_{j,u}^{(1)})$$

   after having removed the first and last round according to the key guess at the first and last rounds. If the key guesses at the first and last rounds are correct, then we have

$$\tilde{x}_u = \sum_{r=2}^{R-1} S_u^{(r)}(x_{1,u}^{(r)} \boxplus \mathrm{rk}_i^{(r)}) \boxminus S_u^{(r)}(x_{1,u}^{(r)} \boxplus \mathrm{rk}_j^{(r)}) .$$

5. Keep the key guesses that give correlation greater than some threshold.

Precomputation Step. In our attack, we have selected our N_{data} nonces in order to maximize, for each nonce, the *expected linear potential* (ELP) over the $R-2$ middle rounds, i.e. the quantity

$$\prod_{r=2}^{R-1} \mathbb{E}_{t_r \in \mathbb{Z}_{16}} \left(\left| C^{S^{(r)}(x \boxplus t_r) \boxminus S^{(r)}(x)}(a,0) \right|^2 \right) .$$

The precomputation time T_{precomp} can be evaluated based on the probability, taken over a random choice of S-boxes $S^{(r)}$, that this quantity is larger than a threshold τ^2, i.e.

$$\prod_{r=2}^{R-1} \mathbb{E}_{t_r \in \mathbb{Z}_{16}} \left(\left| C^{S^{(r)}(x \boxplus t_r) \boxminus S^{(r)}(x)}(a,0) \right|^2 \right) \ge \tau^2 .$$

Let $p_{\ge \tau^2}$ be that probability, so that the precomputation time is $T_{\mathrm{precomp}} = N_{\mathrm{data}}/p_{\ge \tau^2}$.

On The Function g. For the phase correction function g, we let, under a guess of t_r for $\mathrm{rk}_i^{(r)} \boxminus \mathrm{rk}_j^{(r)}$:

$$g\left(\boldsymbol{S}_u, t_2, \ldots, t_{R-1}\right) \;=\; \mathrm{sgn}\left(\prod_{r=2}^{R-1} \overline{C^{S_u^{(r)}(x \boxplus t_r) \boxminus S_u^{(r)}(x)}(a,0)}\right) .$$

This function has been chosen so that, under the right key guess (which we denote $\boldsymbol{t}$)[3], the expectation of the empirical correlation becomes:

$$c \;=\; \mathbb{E}_{\substack{(x_{1,u}^{(r)})_{1\le u\le N_{\mathrm{data}} \\ 2\le r\le R-1}}}(\hat{c}) \;=\; \frac{1}{N_{\mathrm{data}}} \sum_{u=1}^{N_{\mathrm{data}}} \prod_{r=2}^{R-1} \left|C^{S_u^{(r)}(x \boxplus t_r) \boxminus S_u^{(r)}(x)}(a,0)\right| . \tag{17}$$

Finally, we let a key candidate pass the statistical test when $|\hat{c}|^2 \geq (1-\varepsilon)\tau^2$, where τ^2 is the threshold we used in the precomputation step.

4.4 Limits of the Assumptions Made

We tried to implement our attack on a version of FRAST reduced to 5 rounds. We made the tradeoffs between the precomputation and online phase so that both take approximately the same time. By guessing the first and last round keys in the key recovery, the attack's success relies on the statistical behavior of the three internal rounds, which use randomized S-boxes. We made the following standard hypotheses from statistical cryptanalysis:

The right key hypothesis: For the correct key guess, the empirical correlation of the trail will converge to its theoretical expected value, $|c|$, as defined in Equation (17).

The wrong key hypothesis: For any incorrect key guess, the empirical correlation will be negligible, effectively behaving as noise centered around zero.

Following established methodologies [BSV07], we set the decision threshold and data complexity to ensure that, with a probability of at least 90%, the correct key would be identified while nearly all incorrect keys would be discarded. However, our experiments produced an unexpected outcome: while the right key consistently passed the filter, a non-negligible fraction of wrong keys did so as well, contradicting the theoretical predictions. This discrepancy prompted a systematic investigation into the potential failure points of our underlying assumptions. We examined four primary areas:

The precomputation step: We verified that this step successfully generated S-boxes with the expected correlation distributions. It is therefore an unlikely source of the anomaly.

The right key hypothesis: In all our experiments, the correct key guess produced a correlation that surpassed the threshold. This shows that our right key assumption is realistic.

The independence of the round keys: This assumption (which is common in statistical attacks) seems reasonable in our case. Indeed, our precomputation step poses no conditions on the values of the random S_{crf}. Thus, the XOF randomizes the inputs given to the S_{erf} differences that appear after the random rounds. The only remaining places where round independence could not hold is after fixed round. We however do not think it is sufficient to explain our phenomena.

[3] It means that the guessed key material for the first/last rounds and the key difference at the middle rounds are correct.

The wrong key hypothesis: The fact that a significant number of incorrect keys survived the statistical filter strongly suggests that this hypothesis is the point of failure. Thus, we tried to simulate the correlations we get in the case of wrong key guesses. To simplify, we focus on the case where we guessed correctly the keys of round $1, 3, 4$ and 5, but guessed a wrong key for round 2. In that case, we get the right state after round 1 and before round 5, so that we can compute the quantities:

$$\zeta_{16}^{a\cdot\tilde{x}_u} = \prod_{r=2}^{4} \zeta_{16}^{a\cdot\left(S_u^{(r)}(x_{1,u}^{(r)} \boxplus \mathrm{rk}_i^{(r)}) \boxminus S_u^{(r)}(x_{1,u}^{(r)} \boxplus \mathrm{rk}_j^{(r)})\right)} .$$

We then multiply it with $g(\mathtt{IV}_u, t_2, t_3, t_4) = \prod_{r=2}^{4} \mathrm{sgn}(\overline{C^{S_u^{(r)}(x \boxplus t_r) \boxminus S_u^{(r)}(x)}(a, 0)})$, where $t_r = \mathrm{rk}_i^{(r)} \boxminus \mathrm{rk}_j^{(r)}$ for $r = 3, 4$ (right key guess for rounds 3 and 4), t_2 our (wrongly guessed) second round key, and take the expectation over random $x_{1,u}^{(r)}, 2 \leq r \leq 4$. This gives

$$\mathrm{sgn}\left(\overline{C^{S_u^{(2)}(x \boxplus t_2) \boxminus S_u^{(2)}(x)}(a, 0)}\right) \cdot C^{S_u^{(2)}(x \boxplus t_2') \boxminus S_u^{(2)}(x)}(a, 0) \cdot \prod_{r=3}^{4} \left| C^{S_u^{(r)}(x \boxplus t_r) \boxminus S_u^{(r)}(x)}(a, 0) \right|$$

where $t_2' = \mathrm{rk}_i^{(2)} \boxminus \mathrm{rk}_j^{(2)}$. Finally, computing the expectation over random $t_r, r = 3, 4$ and $1 \leq u \leq N_{\mathrm{data}}$ allows to estimate the empirical correlation $\hat{c}$ we would get after having correctly guessed the keys at all rounds except round 2, where the actual key difference is t_2' and we guessed t_2. In that setting, our wrong key assumption implies that we should have negligible biases for all pairs t_2, t_2' except when $t_2 = t_2'$. When running actual experiments on our precomputed S-boxes, we observed that those quantities have nontrivial biases for some values of $t_2 \neq t_2'$. This is the most plausible explanation for our failure in performing linear cryptanalysis.

We leave as an open problem a proper explanation of this phenomenon, which, so far, has remained outside of our grasp.

5 Conclusion

The inner tweakable block cipher within the FRAST stream cipher has a very strong structural flaw, making it feasible to find a simpler and weaker alternative representation. However, the mode in which this block cipher is used prevents a trivial use of this structural property for cryptanalysis. Still, we have used these alternative representations to find a large weak key space on the full primitive, which is made even larger by the negacyclic property of the random S-boxes.

In the average key setting, we made a first attempt at linear cryptanalysis by describing a general framework to work around the fact that new random S-boxes are generated at each call to the cipher. However, when applying it to a reduced version of the primitive, we observed that some wrong key assumptions did not hold. We think that an interesting open problem would be to have a deeper understanding of the phenomena we observed, and to mount a working linear cryptanalysis. This would provide a better understanding of the security margin of FRAST in this setting. Perhaps even more importantly, the gap between the theoretical and observed behavior in FRAST highlights the limits of the state-of-the-art on the linear cryptanalysis of non-binary ciphers, which could have consequences beyond the specifics of FRAST.

Acknowledgments

We thank the reviewers of ToSC 2026 (Issue 1) for insightful comments to improve the quality of this paper and the shepherd for their careful proofreading. Fukang Liu has been supported by JSPS KAKENHI Grant Number JP24K20733. The work of Antoine Bak, Shibam Ghosh and Léo Perrin is supported by the European Research Council (ERC, grant agreement no. 101041545 "ReSCALE"). The research of Antoine Bak is supported by the French DGA.

References

[AMT22] Tomer Ashur, Mohammad Mahzoun, and Dilara Toprakhisar. Chaghri - A fhe-friendly block cipher. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, pages 139–150. ACM, 2022. doi:10.1145/3548606.3559364.

[ARS+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 430–454. Springer, 2015. doi:10.1007/978-3-662-46800-5_17.

[BBB+25] Jules Baudrin, Sonia Belaïd, Nicolas Bon, Christina Boura, Anne Canteaut, Gaëtan Leurent, Pascal Paillier, Léo Perrin, Matthieu Rivain, Yann Rotella, and Samuel Tap. Transistor: a tfhe-friendly stream cipher. In Yael Tauman Kalai and Seny F. Kamara, editors, *Advances in Cryptology - CRYPTO 2025 - 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2025, Proceedings, Part V*, volume 16004 of *Lecture Notes in Computer Science*, pages 530–565. Springer, 2025. doi:10.1007/978-3-032-01901-1_17.

[BBM00] Mihir Bellare, Alexandra Boldyreva, and Silvio Micali. Public-key encryption in a multi-user setting: Security proofs and improvements. In Bart Preneel, editor, *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, volume 1807 of *Lecture Notes in Computer Science*, pages 259–274. Springer, 2000. doi:10.1007/3-540-45539-6_18.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. *ACM Trans. Comput. Theory*, 6(3):13:1–13:36, 2014. doi:10.1145/2633600.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, 2012. doi:10.1007/978-3-642-32009-5_50.

[BSV07] Thomas Baignères, Jacques Stern, and Serge Vaudenay. Linear cryptanalysis of non binary ciphers. In Carlisle M. Adams, Ali Miri, and Michael J. Wiener,

editors, *Selected Areas in Cryptography, 14th International Workshop, SAC 2007, Ottawa, Canada, August 16-17, 2007, Revised Selected Papers*, volume 4876 of *Lecture Notes in Computer Science*, pages 184–211. Springer, 2007. doi:10.1007/978-3-540-77360-3_13.

[CCF+18] Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. *J. Cryptol.*, 31(3):885–916, 2018. doi:10.1007/s00145-017-9273-9.

[CCH+24] Mingyu Cho, Woohyuk Chung, Jincheol Ha, Jooyoung Lee, Eun-Gyeol Oh, and Mincheol Son. FRAST: TFHE-Friendly Cipher Based on Random S-Boxes. *IACR Trans. Symmetric Cryptol.*, 2024(3):1–43, 2024.

[CGGI20] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: fast fully homomorphic encryption over the torus. *J. Cryptol.*, 33(1):34–91, 2020. doi:10.1007/s00145-019-09319-x.

[CHK+21] Jihoon Cho, Jincheol Ha, Seongkwang Kim, ByeongHak Lee, Joohee Lee, Jooyoung Lee, Dukjae Moon, and Hyojin Yoon. Transciphering framework for approximate homomorphic encryption. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part III*, volume 13092 of *Lecture Notes in Computer Science*, pages 640–669. Springer, 2021. doi:10.1007/978-3-030-92078-4_22.

[CHMS22] Orel Cosseron, Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. Towards case-optimized hybrid homomorphic encryption - featuring the elisabeth stream cipher. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part III*, volume 13793 of *Lecture Notes in Computer Science*, pages 32–67. Springer, 2022. doi:10.1007/978-3-031-22969-5_2.

[CIR22] Carlos Cid, John Petter Indrøy, and Håvard Raddum. FASTA - A stream cipher for fast FHE evaluation. In Steven D. Galbraith, editor, *Topics in Cryptology - CT-RSA 2022 - Cryptographers' Track at the RSA Conference 2022, Virtual Event, March 1-2, 2022, Proceedings*, volume 13161 of *Lecture Notes in Computer Science*, pages 451–483. Springer, 2022. doi:10.1007/978-3-030-95312-6_19.

[CKKS17] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yong Soo Song. Homomorphic encryption for arithmetic of approximate numbers. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 409–437. Springer, 2017. doi:10.1007/978-3-319-70694-8_15.

[DEG+18] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. Rasta: A cipher with low anddepth and few ands per bit. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology - CRYPTO 2018 - 38th*

Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I, volume 10991 of *Lecture Notes in Computer Science*, pages 662–692. Springer, 2018. doi:10.1007/978-3-319-96884-1_22.

[DGH+23] Christoph Dobraunig, Lorenzo Grassi, Lukas Helminger, Christian Rechberger, Markus Schofnegger, and Roman Walch. Pasta: A case for hybrid homomorphic encryption. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(3):30–73, 2023. doi:10.46586/tches.v2023.i3.30-73.

[DLR16] Sébastien Duval, Virginie Lallemand, and Yann Rotella. Cryptanalysis of the FLIP family of stream ciphers. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 457–475. Springer, 2016. doi:10.1007/978-3-662-53018-4_17.

[DM15] Léo Ducas and Daniele Micciancio. FHEW: bootstrapping homomorphic encryption in less than a second. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 617–640. Springer, 2015. doi:10.1007/978-3-662-46800-5_24.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat Practical Fully Homomorphic Encryption. *IACR Cryptol. ePrint Arch.*, page 144, 2012.

[GAH+23] Lorenzo Grassi, Irati Manterola Ayala, Martha Norberg Hovd, Morten Øygarden, Håvard Raddum, and Qingju Wang. Cryptanalysis of symmetric primitives over rings and a key recovery attack on rubato. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part III*, volume 14083 of *Lecture Notes in Computer Science*, pages 305–339. Springer, 2023. doi:10.1007/978-3-031-38548-3_11.

[GBJR23] Henri Gilbert, Rachelle Heim Boissier, Jérémy Jean, and Jean-René Reinhard. Cryptanalysis of elisabeth-4. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology - ASIACRYPT 2023 - 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4-8, 2023, Proceedings, Part III*, volume 14440 of *Lecture Notes in Computer Science*, pages 256–284. Springer, 2023. doi:10.1007/978-981-99-8727-6_9.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 169–178. ACM, 2009. doi:10.1145/1536414.1536440.

[GLR+24] Lorenzo Grassi, Fukang Liu, Christian Rechberger, Fabian Schmid, Roman Walch, and Qingju Wang. Minimize the randomness in rasta-like designs: How far can we go? - application to pasta. In Maria Eichlseder and Sébastien Gambs, editors, *Selected Areas in Cryptography - SAC 2024 - 31st International Conference, Montreal, QC, Canada, August 28-30, 2024, Revised Selected*

Papers, Part II, volume 15517 of *Lecture Notes in Computer Science*, pages 207–238. Springer, 2024. doi:10.1007/978-3-031-82841-6_9.

[Hel80] Martin E. Hellman. A cryptanalytic time-memory trade-off. *IEEE Trans. Inf. Theory*, 26(4):401–406, 1980. doi:10.1109/TIT.1980.1056220.

[HKL+22] Jincheol Ha, Seongkwang Kim, ByeongHak Lee, Jooyoung Lee, and Mincheol Son. Rubato: Noisy ciphers for approximate homomorphic encryption. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part I*, volume 13275 of *Lecture Notes in Computer Science*, pages 581–610. Springer, 2022. doi:10.1007/978-3-031-06944-4_20.

[HL20] Phil Hebborn and Gregor Leander. Dasta - Alternative Linear Layer for Rasta. *IACR Trans. Symmetric Cryptol.*, 2020(3):46–86, 2020.

[LAW+23] Fukang Liu, Ravi Anand, Libo Wang, Willi Meier, and Takanori Isobe. Coefficient grouping: Breaking chaghri and more. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 287–317. Springer, 2023. doi:10.1007/978-3-031-30634-1_10.

[LKSM24] Fukang Liu, Abul Kalam, Santanu Sarkar, and Willi Meier. Algebraic attack on fhe-friendly cipher HERA using multiple collisions. *IACR Trans. Symmetric Cryptol.*, 2024(1):214–233, 2024. doi:10.46586/tosc.v2024.i1.214-233.

[LSMI21] Fukang Liu, Santanu Sarkar, Willi Meier, and Takanori Isobe. Algebraic attacks on rasta and dasta using low-degree equations. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part I*, volume 13090 of *Lecture Notes in Computer Science*, pages 214–240. Springer, 2021. doi:10.1007/978-3-030-92062-3_8.

[LSMI22] Fukang Liu, Santanu Sarkar, Willi Meier, and Takanori Isobe. The inverse of χ and its applications to rasta-like ciphers. *J. Cryptol.*, 35(4):28, 2022. doi:10.1007/s00145-022-09439-x.

[LSW+22] Fukang Liu, Santanu Sarkar, Gaoli Wang, Willi Meier, and Takanori Isobe. Algebraic meet-in-the-middle attack on lowmc. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part I*, volume 13791 of *Lecture Notes in Computer Science*, pages 225–255. Springer, 2022. doi:10.1007/978-3-031-22963-3_8.

[MJSC16] Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, volume 9665 of *Lecture Notes in Computer Science*, pages 311–343. Springer, 2016. doi:10.1007/978-3-662-49890-3_13.

[MY92] Mitsuru Matsui and Atsuhiro Yamagishi. A new method for known plaintext attack of FEAL cipher. In Rainer A. Rueppel, editor, *Advances in Cryptology - EUROCRYPT '92, Workshop on the Theory and Application of of Cryptographic Techniques, Balatonfüred, Hungary, May 24-28, 1992, Proceedings*, volume 658 of *Lecture Notes in Computer Science*, pages 81–91. Springer, 1992. doi:10.1007/3-540-47555-9_7.

[NLV11] Michael Naehrig, Kristin E. Lauter, and Vinod Vaikuntanathan. Can homomorphic encryption be practical? In Christian Cachin and Thomas Ristenpart, editors, *Proceedings of the 3rd ACM Cloud Computing Security Workshop, CCSW 2011, Chicago, IL, USA, October 21, 2011*, pages 113–124. ACM, 2011. URL: https://dl.acm.org/citation.cfm?id=2046682.