Two Sides of The Same Coin: Weak Tweak-Keys and More Efficient Variant of CRAFT *

Gregor Leander[1] and Shahram Rasoolzadeh[2]

[1] Ruhr University Bochum, Bochum, Germany, firstname.lastname@rub.de
[2] Radboud University, Nijmegen, The Netherlands, firstname.lastname@ru.nl

Abstract. CRAFT is a lightweight tweakable Substitution-Permutation-Network (SPN) block cipher optimized for efficient protection of its implementations against Differential Fault Analysis (DFA) attacks. In this paper, we present an equivalent description of CRAFT up to a simple mapping on the plaintext, ciphertext and round tweakeys. We show that the new representation, for a sub-class of keys, leads to a new structure which is a Feistel network, i.e., with half state non-linear operation and half state key addition. This has two interesting consequences: First, it reveals a class of weak keys for which CRAFT is less resistant against differential and linear cryptanalyses. Second, on the constructive side, it suggests how a new version of the cipher with more efficient implementation can be built.

Keywords: CRAFT · partial key addition · partial non-linear layer

1 Introduction

CRAFT is a tweakable block cipher presented at FSE 2019 and designed by Beierle, Leander, Moradi, and Rasoolzadeh [BLMR19]. The cipher follows the SPN design with 32 rounds and iterates 4 round tweakeys as of the tweakey schedule. The main goal of CRAFT was to efficiently provide protection of its implementations against DFA attacks [BS97] while to provide decryption on top of the encryption with minimum overhead was considered as a side goal in the design criteria. The encryption-only implementation of the cipher needs 949 GE (using the IBM 130nm ASIC library), which is less than that of the any reported round-based implementation of a lightweight block cipher. Besides, considering the protected against DFA implementation of the cipher, under the same settings with respect to the employed error-detection code, its area overhead (even with decryption and tweak support) is smaller than all block ciphers considered in [BLMR19] with compatible state and key size.

The designers of CRAFT provided a detailed security analysis of the cipher in their proposal paper which covers differential, linear, impossible differential, zero-correlation linear hull, meet-in-the-middle, time-data-memory trade-offs, integral (and division property), and invariant attacks. Overall, they claimed 124 bit security in the related-tweak attacker model. After the publication of the design, some other follow-up cryptanalysis has been published [HSN+19, MA19, EY19, GSS+20] and in the following, we briefly explain results of these analyses.

*The work described in this paper has been partially supported by the German Research Foundation (DFG) within the project LE 3372/5-1 and also by the Netherlands Organisation for Scientific Research (NWO) under TOP grant TOP1.18.002 SCALAR. Besides, we want to thank Christof Beierle for his valuable comments while preparing this work.

1.1 Known Results on CRAFT

Hadipour et al. [HSN+19] presented a detailed security analysis of CRAFT. In particular, they presented 14-round related-tweak zero-correlation linear hull distinguishers. Using the same distinguishers and following the connection between zero-correlation and integral distinguisher, they also presented a 14-round related-tweak integral distinguisher. Furthermore, using the automated search model based on an MILP tool, they found the mistake reported on the differential probability and on the maximum number of rounds for single-tweak differential distinguishers.

Moghaddam and Ahmadian [MA19] used a MILP-based tool to find truncated differentials distinguishers for CRAFT, MIDORI, and SKINNY block ciphers. In the case of CRAFT, they reported a 12-round distinguisher. ElSheikh and Youssef [EY19] presented a related-key differential attack that recovers the whole 128-bit key in a full-round CRAFT with querying corresponding ciphertext for about 2^{36} chosen plaintexts and time complexity of about 2^{36} encryptions using a negligible memory. Note that the designers did not claim any security in the related-key model.

More recently, and most relevant for our work, Guo et al. [GSS+20] studied the combination of the involutory S-box and the simple tweakey schedule used in the CRAFT block cipher. They found that some input difference at a particular position can be preserved through any number of rounds if the input pair follows certain truncated differential trails. They used this property to construct weak-key truncated differential distinguishers of round-reduced CRAFT. As a result, they found some 16-round and one 18-round truncated differential distinguishers of CRAFT that can be extended to a 20-round distinguisher with probability 2^{-63}. Moreover, they presented a key recovery attack on the 19-round CRAFT with 2^{61} data, 2^{68} memory, $2^{94.6}$ time complexity and success probability of about 80%.

1.2 Our Contribution

In this paper, we first study the round operations used in CRAFT in detail. Using properties of these operations, we redefine the round function of the cipher which leads to an equivalent description of CRAFT up to a simple mapping on the plaintext, ciphertext and round tweakeys. In a weak tweak-key scenario, mainly thanks to the involutory S-box and the special choice of MixColumns used in CRAFT, the equivalent representation of the cipher leads to a Feistel network where the non-linear operation (S-box layer) only is applied on half of the state. In this weak tweak-key class of the encryption, the 128-bit key must be one of 2^{88} weak keys, and for each weak-key there are exactly 2^8 64-bit tweaks those are included in the set of 2^{32} weak tweaks.

We analyze the security of the new weak tweak-key structure of the cipher and show that compared to the original structure of CRAFT, the new structure is less resistant against differential and linear cryptanalyses. This part of our results in particular gives another explanation of the results in [GSS+20] and explains the weak tweak-keys identified there.

As a result, we find several 18-round single-tweak differential and several 21-round related-tweak differentials with higher EDP than 2^{-64}. We apply one of those 21-round related-tweak differentials to do a differential key recovery attack on 25-round CRAFT under the weak key model. The weak key space covers 2^{108} and the complexity of the attack is 2^{94} related tweak data and 25-round encryptions together with 2^{38} blocks of memory.

Finally, we focus on an interesting constructive side of the new structure. We show that when decryption is added on top of the encryption, the area overhead of the implementation is almost zero. Besides, to protect against fault analysis attacks, the area overhead for implementing the countermeasure is about half of the case for CRAFT original structure. Therefore, by increasing number of rounds and applying an appropriate tweakey schedule, it is possible to use the new structure as a new block cipher with more efficiency in hardware.

1.3 Outline

First in Section 2, we specify the design of CRAFT. Then in Section 3, we present an equivalent definition for CRAFT round function and using the new representation, we introduce a weak-key structure for the cipher. In Section 4, we use a MILP tool to find all the activity patterns with the minimum number of active S-boxes in differential and linear trails of the weak tweak-key CRAFT structure. We estimate the expected differential probability (EDP) for the differential effect within these differential activity patterns and we show that the actual weak-key space in the differential activity patterns can be larger than the weak tweak-key space for the weak tweak-key structure. Later, in Section 5, we describe the details of our attack on 25-round CRAFT. Finally, in Section 6, we show that the new structure allows an even better implementation of the cipher specially when the encryption is combined with decryption and also when it is combined with the countermeasures against fault analysis.

2 CRAFT Specification

CRAFT is a lightweight tweakable block cipher consisting of a 64-bit block, a 128-bit key, and a 64-bit tweak. The state is viewed as a 4×4 array of nibbles. The notation $X[i, j]$ denotes the nibble located at row i and column j of the state. By concatenating the rows of the state, one can denote the state as a vector of nibbles that $X[i]$ denotes the nibble in i-th position of this vector, i.e., $X[i, j] = X[4i + j]$.

The 128-bit key is split into two 64-bit keys K_0 and K_1. Together with the 64-bit tweak input T, four 64-bit round-tweakeys TK_0, TK_1, TK_2 and TK_3 are derived. The cipher uses 31 identical round functions $\mathcal{R}_i$ with $0 \leq i \leq 30$ together with a linear round $\mathcal{R}_{31}$. CRAFT makes use of the following five operations:

- SB: The 4-bit involutory S-box S is applied to each nibble of the state. This S-box is the same as the S-box used in the block cipher MIDORI [BBI+15].

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	c	a	d	3	e	b	f	7	8	9	1	5	0	2	4	6

- MC: Each column of the state is multiplied with the following involutory binary matrix:

$$M = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} .$$

- PN: Using an involutory permutation P, the position of the nibbles in the state changes. Particularly, $X[i]$ is replaced with $X[P(i)]$, where

$$P = [15, 12, 13, 14,\ 10, 9, 8, 11,\ 6, 5, 4, 7,\ 1, 2, 3, 0] .$$

- $\mathtt{A}_{RC_i}$: One 4-bit and one 3-bit round-constant value is XORed with the forth and the fifth state nibble, respectively.

- $\mathtt{A}_{TK_i}$: The cipher uses four 64-bit tweakeys TK_0, TK_1, TK_2 and TK_3 from the tweak T and the key $(K_0 \,\|\, K_1)$ as

$$TK_0 = K_0 \oplus T\ ,\ TK_1 = K_1 \oplus T\ ,\ TK_2 = K_0 \oplus \mathtt{QN}(T)\ ,\ TK_3 = K_1 \oplus \mathtt{QN}(T)\ ,$$

where $\mathtt{QN}(T)$ applies the permutation

$$Q = [12, 10, 15, 5,\ 14, 8, 9, 2,\ 11, 3, 7, 4,\ 6, 0, 1, 13]$$

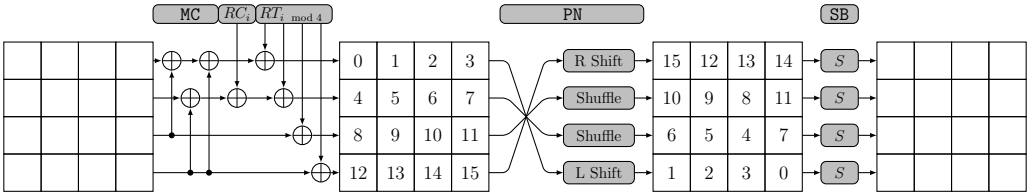

Figure 1: One full round of CRAFT.

on the position of tweak nibbles which $T[i]$ is replaced by $T[Q(i)]$. Then in each round i, without any key update, the tweakey $TK_{i \bmod 4}$ is XORed to the state. For simplicity, we will use TK_i.

The round functions $\mathcal{R}_i$, with $0 \leq i < 31$, are defined as

$$\mathcal{R}_i = \mathtt{SB} \circ \mathtt{PN} \circ \mathtt{A}_{TK_i} \circ \mathtt{A}_{RC_i} \circ \mathtt{MC}$$

and the last round $\mathcal{R}_{31}$ is defined as

$$\mathcal{R}_{31} = \mathtt{A}_{TK_3} \circ \mathtt{A}_{RC_{31}} \circ \mathtt{MC}\,.$$

The full one-round function of CRAFT is depicted in Figure 1.

3 CRAFT Weak Tweak-Key Structure

In the following, based on the given properties, first, we present an equivalent definition for the CRAFT round function. Then, using the new representation, we introduce a weak tweak-key class for the cipher. In this weak tweak-key class of the encryption, the 128-bit key must be one of 2^{88} weak keys, and for each weak-key there are exactly 2^8 64-bit tweaks those are included in the set of 2^{32} weak tweaks.

We use X' and X'' to denote left and right halves of the state X, i.e., $X' = (X[0], \ldots, X[7])$ and $X'' = (X[8], \ldots, X[15])$. We use the same notation to denote each halves of the key, tweak, and tweakey, e.g., we use TK_i' and TK_i'' for the latter case.

Property 1. In MC operation, for each column index $j \in \{0, \ldots, 3\}$, we have

$$M\left(\begin{bmatrix} X[0,j] \\ X[1,j] \\ X[2,j] \\ X[3,j] \end{bmatrix}\right) = \begin{bmatrix} X[0,j] \\ X[1,j] \\ X[2,j] \\ X[3,j] \end{bmatrix} \oplus \begin{bmatrix} X[2,j] \oplus X[3,j] \\ X[3,j] \\ 0 \\ 0 \end{bmatrix}.$$

That is, a linear combination of the right half is XORed with the left half; i.e.,

$$\mathtt{MC}(X' \,\|\, X'') = \left(X' \oplus \mathtt{MC}'(X'') \,\|\, X''\right),$$

where $\mathtt{MC}'$ is the corresponding linear operation with binary matrix M' to each column of the right half:

$$M' = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} X[2,j] \\ X[3,j] \end{bmatrix} \mapsto \begin{bmatrix} X[2,j] \oplus X[3,j] \\ X[3,j] \end{bmatrix}.$$

Property 2. PN operation replaces the left half of the state with a nibble permutation of the right half and vice versa; i.e.,

$$\mathtt{PN}(X' \,\|\, X'') = \left(\mathtt{PN}''(X'') \,\|\, \mathtt{PN}'(X')\right),$$

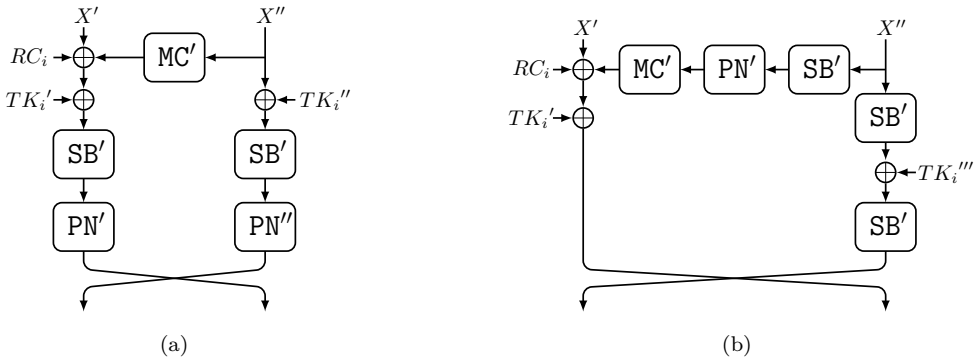

Figure 2: Representation and equivalent round functions of CRAFT.

with $\mathtt{PN}'$ using the following P' permutation to replace $X[i]$ by $X[P'(i)]$:

$$P' = [6, 5, 4, 7, 1, 2, 3, 0] \ .$$

Moreover, since PN is an involutive operation, we have $\mathtt{PN}'' = \mathtt{PN}'^{-1}$.

Using Property 1 and Property 2, it is possible to represent the round function of CRAFT as the function shown in Figure 2(a) where we use $\mathtt{SB}'$ operation to denote the application of the S-box S to each of eight nibbles. Besides, we use $\mathtt{A}'$ to denote the tweakey or round constant addition in each half of the state.

Proposition 1. *CRAFT encryption is equivalent (up-to a nibble-permutation and an S-box operation on the right half of the plaintext/ciphertext and a nibble-permutation on the right half of the round tweakeys) to the encryption with the round function*

$$\mathcal{R}_i'(X' \,\|\, X'') = \left(\mathtt{SB}' \circ \mathtt{A}'_{TK_i'''} \circ \mathtt{SB}'(X'') \,\|\, \mathtt{A}'_{TK_i'} \circ \mathtt{A}'_{RC_i} \circ \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'') \oplus X'\right),$$

that $TK_i''' = \mathtt{PN}''(TK_i'')$, except in the last round, similar to the Feistel network, the final swapping between the right and left halves are omitted. The equivalent round function is shown in Figure 2(b).

Proof. For CRAFT round function, we have:

$$\begin{aligned}
\mathcal{R}_i(X' \,\|\, X'') &= \mathtt{SB} \circ \mathtt{PN} \circ \mathtt{A}_{TK_i} \circ \mathtt{A}_{RC_i} \circ \mathtt{MC}(X' \,\|\, X'') \\
&= \mathtt{PN} \circ \mathtt{SB} \circ \mathtt{A}_{TK_i} \circ \mathtt{A}_{RC_i} \circ \mathtt{MC}(X' \,\|\, X'') \\
&= \mathtt{PN} \circ \mathtt{SB} \circ \mathtt{A}_{TK_i} \circ \mathtt{A}_{RC_i}(X' \oplus \mathtt{MC}'(X'') \,\|\, X'') \\
&= \mathtt{PN} \circ \mathtt{SB} \circ \mathtt{A}_{TK_i}\Big(\mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}'(X'')\big) \,\|\, X''\Big) \\
&= \mathtt{PN} \circ \mathtt{SB}\Big(\mathtt{A}'_{TK_i'} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}'(X'')\big) \,\|\, \mathtt{A}'_{TK_i''}(X'')\Big) \\
&= \mathtt{PN}\Big(\mathtt{SB}' \circ \mathtt{A}'_{TK_i'} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}'(X'')\big) \,\|\, \mathtt{SB}' \circ \mathtt{A}'_{TK_i''}(X'')\Big) \\
&= \Big(\mathtt{PN}'' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK_i''}(X'') \,\|\, \mathtt{PN}' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK_i'} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}'(X'')\big)\Big) \ .
\end{aligned}$$

This is the same representation of CRAFT round function in Figure 2(a). Similarly for the last linear round, we have:

$$\mathcal{R}_{31}(X' \,\|\, X'') = \Big(\mathtt{A}'_{TK_3'}\big(X' \oplus \mathtt{MC}'(X'') \oplus RC'_{31}\big) \,\|\, \mathtt{A}'_{TK_3''}(X'')\Big) \ .$$

Consider now a bijective function $\mathcal{G}$. By iterating $\mathcal{R}'_i = \mathcal{G} \circ \mathcal{R}_i \circ \mathcal{G}^{-1}$ instead of $\mathcal{R}_i$ round functions, we reach to an encryption equivalent to the CRAFT encryption.

$$\begin{aligned}\mathcal{R}'_{31} \circ \ldots \circ \mathcal{R}'_1 \circ \mathcal{R}'_0 &= \mathcal{G} \circ \mathcal{R}_{31} \circ \mathcal{G}^{-1} \circ \ldots \circ \mathcal{G} \circ \mathcal{R}_1 \circ \mathcal{G}^{-1} \circ \mathcal{G} \circ \mathcal{R}_0 \circ \mathcal{G}^{-1} \\ &= \mathcal{G} \circ \mathcal{R}_{31} \circ \ldots \circ \mathcal{R}_1 \circ \mathcal{R}_0 \circ \mathcal{G}^{-1}\,.\end{aligned}$$

Precisely, for the plaintext X and the corresponding ciphertext Y in the CRAFT encryption, the plaintext $\mathcal{G}(X)$ will be encrypted to the ciphertext $\mathcal{G}(Y)$ in the equivalent cipher with $\mathcal{R}'_i$ round functions. By choosing $\mathcal{G}$ as

$$\mathcal{G}(X' \,\|\, X'') = \big(X' \,\|\, \mathtt{SB}' \circ \mathtt{PN}''(X'')\big) \;\Rightarrow\; \mathcal{G}^{-1}(X' \,\|\, X'') = \big(X' \,\|\, \mathtt{PN}' \circ \mathtt{SB}'(X'')\big),$$

it is possible to simplify the equivalent round functions:

$$\begin{aligned}&\mathcal{R}'_i(X' \,\|\, X'') = \mathcal{G} \circ \mathcal{R}_i \circ \mathcal{G}^{-1}(X' \,\|\, X'') = \mathcal{G} \circ \mathcal{R}_i\big(X' \,\|\, \mathtt{PN}' \circ \mathtt{SB}'(X'')\big) \\ &= \mathcal{G}\Big(\mathtt{PN}'' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK''_i} \circ \mathtt{PN}' \circ \mathtt{SB}'(X'') \,\|\, \mathtt{PN}' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK'_i} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big)\Big) \\ &= \mathcal{G}\Big(\mathtt{SB}' \circ \mathtt{A}'_{TK'''_i} \circ \mathtt{PN}'' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'') \,\|\, \mathtt{PN}' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK'_i} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big)\Big) \\ &= \mathcal{G}\Big(\mathtt{SB}' \circ \mathtt{A}'_{TK'''_i} \circ \mathtt{SB}'(X'') \,\|\, \mathtt{PN}' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK'_i} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big)\Big) \\ &= \Big(\mathtt{SB}' \circ \mathtt{A}'_{TK'''_i} \circ \mathtt{SB}'(X'') \,\|\, \mathtt{SB}' \circ \mathtt{PN}'' \circ \mathtt{PN}' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK'_i} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big)\Big) \\ &= \Big(\mathtt{SB}' \circ \mathtt{A}'_{TK'''_i} \circ \mathtt{SB}'(X'') \,\|\, \mathtt{SB}' \circ \mathtt{SB}' \circ \mathtt{A}'_{TK'_i} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big)\Big) \\ &= \Big(\mathtt{SB}' \circ \mathtt{A}'_{TK'''_i} \circ \mathtt{SB}'(X'') \,\|\, \mathtt{A}'_{TK'_i} \circ \mathtt{A}'_{RC_i}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big)\Big)\,,\end{aligned}$$

where $TK'''_i = \mathtt{PN}''(TK''_i)$. Note that this is the same round function as in Figure 2(b). Similarly for the last round, we have:

$$\mathcal{R}'_{31}(X' \,\|\, X'') = \Big(\mathtt{A}'_{TK'_3} \circ \mathtt{A}'_{RC_{31}}\big(X' \oplus \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'(X'')\big) \,\|\, \mathtt{SB}' \circ \mathtt{A}'_{TK'''_3} \circ \mathtt{SB}'(X'')\Big)\,,$$

which is same as the other round functions $\mathcal{R}'_i$ without the final swapping between the left and the right halves of the state. □

Hereafter, we will simply call *the equivalent CRAFT* to this equivalent representation of the cipher. As depicted in Figure 2(b), the equivalent CRAFT encryption is quite similar to the Feistel network. The only difference is in the transition of the right half of the state which is through $\mathtt{SB}' \circ \mathtt{A}_{TK'''_i} \circ \mathtt{SB}'$ function while in the case of a Feistel network, this functions is the identity function.

On the other hand, the equivalent CRAFT provides a weak tweak-key structure for CRAFT. If the function $\mathtt{SB}' \circ \mathtt{A}_{TK'''_i} \circ \mathtt{SB}'$ is equal to the identity function, the equivalent CRAFT will be a Feistel network which includes partial nonlinear round and partial key-addition. Therefore, for such weak tweak-key classes, the equivalent CRAFT might shows weaker resistance against some analysis, e.g., differential and linear attacks. Hereafter, we will simply call *the weak tweak-key CRAFT* to this weak tweak-key encryption of the equivalent CRAFT. In the following, we discuss the necessary requirements for the weak tweak-key CRAFT.

Lemma 1. $\mathtt{SB}' \circ \mathtt{A}'_{TK'''} \circ \mathtt{SB}'$ *is equal to the identity function if and only if* $TK''' = 0$.

Proof. $\mathtt{SB}' \circ \mathtt{A}'_{TK'''} \circ \mathtt{SB}'$ is the identity function if and only if for any $X' \in \mathbb{F}_2^{32}$,

$$\mathtt{SB}' \circ \mathtt{A}'_{TK'''_i} \circ \mathtt{SB}'(X') = X'\,.$$

Since $\mathtt{SB}'$ is involution, this equally means for any X', we must have

$$\mathtt{A}'_{TK_i'''} \circ \mathtt{SB}'(X') = \mathtt{SB}'(X') \;\Leftrightarrow\; TK_i''' \oplus \mathtt{SB}'(X') = \mathtt{SB}'(X') \;\Leftrightarrow\; TK_i''' = 0\,.$$

□

Lemma 2. *In the weak tweak-key CRAFT, the 128-bit key must be one of 2^{88} weak keys, and for each weak-key there are exactly 2^8 64-bit tweaks those are included in the set of 2^{32} weak tweaks.*

Proof. As of a corollary from 1, to achieve the weak tweak-key CRAFT, it is necessary to have $TK_i''' = 0$ for all rounds. This equally means $TK_0'' = TK_1'' = TK_2'' = TK_3'' = 0$. Simplifying these equation to the key and tweak variables, we reach to

$$K_0'' = K_1'' = T'' = \mathtt{QN}(T)[8, \ldots, 15]\,.$$

For the given Q permutation in the CRAFT specification, the above equation is same as

$$\begin{array}{llll} K_0[8] = K_1[8] = T[8] = T[11]\,, & & K_0[9] = K_1[9] = T[9] = T[3]\,, \\ K_0[10] = K_1[10] = T[10] = T[7]\,, & & K_0[11] = K_1[11] = T[11] = T[4]\,, \\ K_0[12] = K_1[12] = T[12] = T[6]\,, & & K_0[13] = K_1[13] = T[13] = T[0]\,, \\ K_0[14] = K_1[14] = T[14] = T[1]\,, & & K_0[15] = K_1[15] = T[15] = T[13]\,. \end{array}$$

This means that in the weak tweak-key CRAFT, the 128-bit key must be one of 2^{88} weak keys satisfying above equations, i.e. both $K_0'' = K_1''$ must be in the following form:

$$K_0'' = K_1'' = \big(x_0, x_1, x_2, x_0, x_3, x_4, x_5, x_4\big)\,.$$

Besides, the 64-bit tweak must be one of 2^{32} weak tweaks in the form of

$$T = \big(t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_4, t_3, t_7, t_4, t_6, t_0, t_1, t_0\big)\,.$$

Moreover, the weak key and the weak tweak must satisfy the followings:

$$x_0 = t_4\,, \quad x_1 = t_3\,, \quad x_2 = t_7\,, \quad x_3 = t_6\,, \quad x_4 = t_0\,, \quad x_5 = t_1\,.$$

Therefore, for each weak key, there is exactly 2^8 weak tweaks with freedom on the t_2 and t_5 nibbles, and in total there are $2^{88+32-24} = 2^{96}$ weak tweak-key pairs. □

It is important to mention that depending on the choice of Q permutation, the size of the weak key set can be in the form of $2^{64+4\cdot\ell}$ with $0 \le \ell \le 8$. Note that in the design rationale for CRAFT, the only criterion for QN operation was that the permutation Q to be a circulant one. Therefore, one (as a designer) can reduce the size of the weak key set by choosing another Q permutation while (s)he must consider resistance of the cipher against related-tweak attacks.

The round function of the weak tweak-key CRAFT is shown in Figure 3(a) which by using an equivalent tweakey schedule changes to the Feistel round function shown in Figure 3(b).

Lemma 3. *If all the right halves of the tweakeys in CRAFT encryption are equal to zero, i.e. all $TK_i'' = 0$ with $0 \le i < 4$, then the encryption is equivalent to the Feistel network with the following round function*

$$\mathcal{R}_i(X', X'') = \big(X'', X' \oplus \mathcal{F}_i(X'' \oplus ETK_i)\big) \text{ with } \mathcal{F}_i := \mathtt{A}_{RC_i'} \circ \mathtt{MC}' \circ \mathtt{PN}' \circ \mathtt{SB}'\,,$$

where ETK_i is an equivalent tweakey. Moreover, the equivalent round tweakeys are

$$\begin{array}{llll} ETK_0 = 0\,, & ETK_1 = K_0' \oplus T'\,, & ETK_2 = K_1' \oplus T'\,, & ETK_3 = T' \oplus T''', \\ ETK_4 = T' \oplus T''', & ETK_5 = K_0' \oplus T''', & ETK_6 = K_1' \oplus T''', & ETK_7 = 0 \end{array}$$

while for $i \ge 8$, we have $ETK_i = ETK_{i \bmod 8}$. Besides, T''' denotes the left half of $\mathtt{QN}(T)$, i.e., $T''' = \big(\mathtt{QN}(T)[0], \ldots, \mathtt{QN}(T)[7]\big)$.

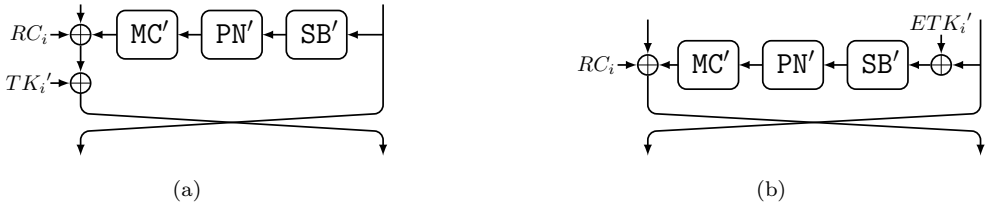

Figure 3: Feistel round function of weak-key CRAFT.

Proof. The behavior of the key addition in the Feistel network is already studied well and it is known in the literature that the Feistel cipher with round functions defined by $\mathcal{R}_i(X', X'') = \big(X'', RK_i \oplus X' \oplus \mathcal{F}_i(X'')\big)$ is equal to the Feistel cipher with round functions of $\mathcal{R}'_i(X', X'') = \big(X'', X' \oplus \mathcal{F}_i(X'' \oplus ERK_i)\big)$ where for $i > 1$,

$$ERK_i = RK_{i-1} \oplus ERK_{i-2} \text{ with } ERK_0 = 0, ERK_1 = RK_0\,,$$

together with a whitening key in the ciphertext. And the whitening key is ERK_{r-1} on the right half and ERK_r on the left half.

The weak tweak-key CRAFT tweakey schedule uses four 32-bit round tweakeys, TK'_0, TK'_1, TK'_2 and TK'_3 repeatedly. For the equivalent round tweakeys, we would have the following eight round tweakeys:

$$\begin{array}{lllll}
ETK_0 = & & = & = 0 & ,\\
ETK_1 = & & = TK'_0 & = K'_0 \oplus T' & ,\\
ETK_2 = ETK_0 \oplus TK'_1 & = TK'_1 & & = K'_1 \oplus T' & ,\\
ETK_3 = ETK_1 \oplus TK'_2 & = TK'_2 \oplus TK'_0 & & = T' \oplus T''' & ,\\
ETK_4 = ETK_2 \oplus TK'_3 & = TK'_3 \oplus TK'_1 & & = T' \oplus T''' & ,\\
ETK_5 = ETK_3 \oplus TK'_0 & = TK'_2 & & = K'_0 \oplus T''' & ,\\
ETK_6 = ETK_4 \oplus TK'_1 & = TK'_3 & & = K'_1 \oplus T''' & ,\\
ETK_7 = ETK_5 \oplus TK'_2 & = & & = 0 & ,
\end{array}$$

that are used repeatedly. For the whitening keys we have $ERK_{31} = ERK_{32} = 0$. □

As you see, half of the round tweakeys in the equivalent tweakey schedule are independent of the key value. More important, $ETK_0 = ETK_7 = 0$ makes the first and the last rounds of the weak tweak-key structure of CRAFT to be key-less rounds. This means, the security of the 32-round weak tweak-key CRAFT cipher with Feistel network structure is based on the middle 30 rounds and the other two rounds are actually useless.

4 Differential and Linear Analysis

In this section, first we use an MILP tool to find all the activity patterns with the minimum number of active S-boxes in reduced-round differential and linear trails of the weak tweak-key CRAFT structure. We apply the methods introduced in [ELR20] to estimate the EDP for the differential effect within these differential activity patterns. Then, we discuss the conditions for applying the same differential trails of the weak tweak-key CRAFT in the equivalent CRAFT structure. There, we show that the weak tweak-key space in the differential activity patterns of the equivalent CRAFT can be larger than the one for the weak tweak-key CRAFT. Later, we investigate possibility of applying related-tweak differentials in the weak tweak-key or in the equivalent CRAFT structures.

As a result, we present 18-round single-tweak differential activity patterns which provide multiple differentials with EDP of $2^{-58.11}$, and several 21-round related-tweak differential activity patterns that include multiple differentials with EDP of higher than 2^{-64}.

Table 1: The minimum number of active S-boxes in differential and linear activity patterns in up to 31 rounds of CRAFT. n_1 and n_2 denote the numbers for the original and the weak tweak-key structures, respectively.

r	1	2	3	4	5	6	7	8	$9 \leq r$
n_1	1	2	4	6	10	14	20	26	$4 \cdot (r-1)$
n_2	0	1	2	3	4	7	10	13	$2 \cdot (r-1)$

4.1 Minimum Number of Active S-boxes

To compare the resistance of CRAFT cipher in its original structure and in the weak tweak-key structure against the differential and linear attacks, we compute the minimum number of active S-boxes in the single-tweak model.[1] In order to compute these bounds, similar to the one in the CRAFT proposal paper [BLMR19], we use the MILP as explained in [MWGP11]. It is noteworthy to mention that this approach is independent of the specification of the S-box and it takes only the properties of the linear layer into account.

While for computing the minimum number of active S-boxes in the original structure of the cipher, in the MILP codes, in each round, we need to consider all the 16 corresponding variables to the S-box layer as objective variables, in the case for the weak tweak-key structure, we must consider only the 8 corresponding variables to the right half variables in the S-box layer. Beyond this difference in two structures of CRAFT, by taking benefit of the Feistel-like design of the cipher, we slightly improved the MILP codes used in [BLMR19] to reduce the number of variables.

As it is already mentioned in [MWGP11], to find the the minimum number of active S-boxes for linear activity patterns, it is enough to replace the matrix of MC layer, M, by the corresponding inverse of transpose matrix, $(M^{-1})^T$ (which equals to M^T). The current choice for M causes that solving the equation to find the minimum number of active S-boxes with matrix M, to be the same as solving with M^T. This means that for a given number of rounds, the minimum number of active S-boxes in differential activity patterns is the same as the minimum number of active S-boxes in linear activity patterns.

Table 1 shows the minimum number of active S-boxes in the single-tweak model for 1 up to 31 rounds in both differential and linear activity patterns. We use n_1 and n_2 to denote the numbers for the original and the weak tweak-key structures, respectively. One interesting observation from Table 1 is that for most of the number of rounds, n_2 is exactly half of n_1. Intuitively saying, this makes sense because in the weak tweak-key structure, each left half of the state is considered once, while in the original structure, each left half-state is considered twice: first in the left half-state of the current round and second in the right half-state of the next round.

While for the original structure, after 9 rounds, all the numbers of active S-boxes are higher than 32, for the weak tweak-key structure, this happens after 17 rounds. Note that having at least 32 active S-boxes is important because the maximum differential probability (resp. absolute linear correlation) for an active S-box is 2^{-2} (resp. 2^{-1}) and this makes the probability (resp. absolute correlation) of a differential (resp. linear) characteristic to be less than or equal to 2^{-64} (resp. 2^{-32}). Therefore, such a characteristic cannot distinguish the (reduced-round) cipher from a random permutation.

4.2 Differential Effects

Finding the minimum number of active S-boxes considers only a single characteristic in the analysis. Therefore, the differential or linear distinguisher might actually be stronger due to

[1] We recall that in a differential (resp. linear) activity pattern, an S-box is called *active*, if the input difference (resp. mask) is non-zero.

Table 2: The maximum EDP for the differentials within the activity patterns with minimum number of active S-boxes. Note we use $p = -\log_2 \text{EDP}$ instead of showing values for the EDP.

r	9	10	11	12	13	14	15	16	17	18	19
p	25.79	29.79	35.54	41.42	41.19	45.19	50.42	56.42	54.00	58.11	63.25

differential or linear hull effects, respectively. To have a better estimation about the strength of the differential distinguishers, we compute the EDP of the differentials. To this point, we use the MILP technique introduced in [SHW+14] to find all the differential activity patterns with the minimum possible active S-boxes. Then, for each given differential activity pattern, we use the methods in [ELR20] to compute the EDP of the differentials within the activity pattern. That is by fixing the input and output differences in the differential, we consider all different the single characteristics which follow the same activity pattern with the minimum number of active S-boxes. Then by summing all these probabilities of each single characteristics, we find a lower bound for the probability of corresponding differential. We repeat this computation for all the different values for the input and output differences in the differential to find the one with the maximum EDP.

It is noteworthy to mention again that the computed values for the EDPs are lower bounds, because for a fixed input and output difference, there might be some other single characteristics that are not following the S-box activity pattern. However, as for such characteristics the number of active S-boxes will be higher, we assume their affect on the probability of differential to be negligible.

Table 2 summarizes the maximum EDP for the differentials within the activity patterns with minimum number of active S-boxes up to 19 rounds. For 19 rounds and more, there is no differentials within the activity patterns of minimum number of active S-boxes that has EDP of significantly higher than 2^{-64}. Note that in this table, instead of showing value of the EDP, we use $p = -\log_2 \text{EDP}$.

The differentials of 18-round with the highest EDP ($= 2^{-58.11}$) within the differentials of the activity patterns with the minimum number of active S-boxes (34 S-boxes) are listed in Table 3 which are from four different activity patterns. Note that in these activity patterns, the active and the inactive nibbles of states are denoted by `1` and `0`, respectively; besides, the values of the input difference (ΔP) and the output difference (ΔC) are shown in the hexadecimal.

4.3 Enlarging Weak Tweak-Key Space in a Differential Activity Pattern

To achieve the Feistel round function of CRAFT (shown in Figure 3(b)), it is necessary to have $TK_i''' = 0$ for all the i values which leads to 2^{88} weak keys (out of the 2^{128}) and 2^{32} weak tweaks (out of the 2^{64}) together with 24 bit conditions between them; i.e., 2^{96} weak tweak-keys out of the 2^{196}. But for the differentials within an activity pattern, considering $TK_i''' = 0$ is a generous condition. Considering the Figure 2(b), to assure that the differential probability of a differential transition over the right branch (over the $\mathtt{SB}' \circ \mathtt{A}_{TK_i'''} \circ \mathtt{SB}'$ operation) is equal 1, it is enough that only the nibbles of TK_i''' be zero which are the corresponding nibbles to the active nibbles in the difference. This is because of the property of a bijective S-box which a zero difference in the input leads to zero difference in the output and vice versa.

Therefore, it is possible to use the same activity patterns found for the weak tweak-key structure with Feistel round functions shown in Figure 3, also for the structure with the round functions shown in Figure 2(b). To do this, we only need to consider weak tweak-keys which make the active nibbles of all TK_i'''s to be zero.

Table 3: 18-round differentials with the highest EDP ($= 2^{-58.11}$) within the differentials of the activity patterns with the minimum number of active S-boxes ($= 34$).

	trail 1	*trail 2*	*trail 3*	*trail 4*
	01010101 10100000	10101010 01010000	01010101 10100000	10101010 01010000
	10100000 00010000	01010000 10000000	10100000 01000000	01010000 00100000
	00010000 10000010	10000000 01000001	01000000 00101000	00100000 00010100
	10000010 10000001	01000001 00011000	00101000 00100100	00010100 01000010
	10000001 10000011	00011000 01000011	00100100 00101100	01000010 00011100
	10000011 00000000	01000011 00000000	00101100 00000000	00011100 00000000
	00000000 10000011	00000000 01000011	00000000 00101100	00000000 00011100
	10000011 10000001	01000011 00011000	00101100 00100100	00011100 01000010
	10000001 10000011	00011000 01000011	00100100 00101100	01000010 00011100
	10000011 00000000	01000011 00000000	00101100 00000000	00011100 00000000
	00000000 10000011	00000000 01000011	00000000 00101100	00000000 00011100
	10000011 10000001	01000011 00011000	00101100 00100100	00011100 01000010
	10000001 10000011	00011000 01000011	00100100 00101100	01000010 00011100
	10000011 00000000	01000011 00000000	00101100 00000000	00011100 00000000
	00000000 10000011	00000000 01000011	00000000 00101100	00000000 00011100
	10000011 10000001	01000011 00011000	00101100 00100100	00011100 01000010
	10000001 10000010	00011000 01000001	00100100 00101000	01000010 00010100
	10000010 00010000	01000001 10000000	00101000 01000000	00010100 00100000
	00010000 10100010	10000000 01010001	01000000 10101000	00100000 01010100
ΔP	0x0a0x0a a0y00000	a0x0a0x0 0a0y0000	0a0x0a0x a0y00000	x0a0x0a0 0a0y0000
ΔC	000y0000 a0z000t0	y0000000 0a0z000t	0a000000 z0y0t000	00a00000 0z0y0t00
	$t = y \oplus z$ and $(xyz) \in \{(5,a,0), (7,5,f), (7,d,7), (a,5,f), (a,a,0), (a,d,7), (a,f,5), (d,a,0), (f,a,0), (f,f,5)\}$		$t = z \oplus a$, $x \in \{5,a,d,f\}$ and $(y, z) \in \{(5,f), (a,0), (d,7), (f,5)\}$	

Example 1. Consider a differential distinguisher corresponding to the trail 1 from Table 3. This distinguisher works for the weak tweak-key space which leads to

$$
\begin{aligned}
TK_0'''[0] = TK_0'''[2] &= 0, & TK_{10}'''[0] = TK_{10}'''[6] = TK_{10}'''[7] &= 0,\\
TK_1'''[3] &= 0, & TK_{11}'''[0] = TK_{11}'''[7] &= 0,\\
TK_2'''[0] = TK_2'''[6] &= 0, & TK_{12}'''[0] = TK_{12}'''[6] = TK_{12}'''[7] &= 0,\\
TK_3'''[0] = TK_3'''[7] &= 0, & TK_{14}'''[0] = TK_{14}'''[6] = TK_{14}'''[7] &= 0,\\
TK_4'''[0] = TK_4'''[6] = TK_4'''[7] &= 0, & TK_{15}'''[0] = TK_{15}'''[7] &= 0,\\
TK_6'''[0] = TK_6'''[6] = TK_6'''[7] &= 0, & TK_{16}'''[0] = TK_{16}'''[6] &= 0,\\
TK_7'''[0] = TK_7'''[7] &= 0, & TK_{17}'''[3] &= 0,\\
TK_8'''[0] = TK_8'''[6] = TK_8'''[7] &= 0, & TK_{18}'''[0] = TK_{18}'''[2] &= 0.
\end{aligned}
$$

Moreover, considering that $TK_{i+4} = TK_i$, all the above conditions can be combined in below conditions:

$$
\begin{aligned}
TK_0'''[0] = TK_0'''[2] = TK_0'''[6] = TK_0'''[7] &= 0, & TK_1'''[3] &= 0,\\
TK_2'''[0] = TK_2'''[2] = TK_2'''[6] = TK_2'''[7] &= 0, & TK_3'''[0] = TK_3'''[7] &= 0.
\end{aligned}
$$

These conditions lead to 2^{48} weak tweaks (with four conditions of $T[0] = T[13] = T[15]$ and $T[4] = T[8] = T[11]$), and 2^{120} weak-keys (with two conditions of $K_0[8] = K_0[11]$ and $K_0[13] = K_0[15]$), with five extra conditions between the tweak and key nibbles. All together, the distinguisher works for a weak tweak-key space of size $2^{120+48-20} = 2^{148}$.

It is important to emphasize that the size of the weak tweak-key space by changing the index of starting round can change. Besides, the size of the weak tweak-key space for

other distinguishers from Table 3 is the same. Moreover, it is noteworthy to mention that there are some other distinguishers with a larger size for the weak tweak-key space, but with a lower value for the EDP.

By appending some rounds before and some after, one can use these distinguishers to do a key recovery attack on the reduced-round CRAFT. For all the distinguishers in Table 3, appending 3 (resp. 4) rounds before (resp. after) the distinguisher activates all the nibbles in the plaintext (resp. ciphertext) difference. Therefore, by appending 2 (resp. 3) rounds before (resp. after) the distinguisher, it is possible to do a key recovery attack on 23 rounds of the CRAFT cipher.

4.4 Related-Tweak Differentials in the Weak Tweak-Key Structure

While the differentials discussed in the previous subsections were based on the single-tweaks, in this subsection, we investigate the possibility for existence of related-tweak differential in the weak tweak-key structure of CRAFT.

As explained previously in Section 3 and Subsection 4.3, it is possible that any difference α in the input of $S\big(S(\cdot) \oplus t\big)$ transits to the same difference in the output of the function, if α is equal to zero or if the round tweakey nibble t is equal to zero. Therefore, it is still possible to consider existence of related-tweak differentials, if we keep these conditions. Precisely, if the difference in the right half of the round tweakeys, TK_i'', (consequently, in the right half of the round tweaks T_i'') is equal to zero. Therefore, by considering $\Delta T_0'' = \Delta T_1'' = \Delta T_2'' = \Delta T_3'' = 0$, we can check if there is any related-tweak differential in the weak tweak-key structure of CRAFT.

$$\begin{cases} \Delta T_0'' = \Delta T_1'' = 0 \;\Rightarrow\; \Delta T[8,9,10,11,12,13,14,15] = 0 \\ \Delta T_2'' = \Delta T_3'' = 0 \;\Rightarrow\; \Delta T[0,1,3,4,6,7,11,13] = 0\,. \end{cases}$$

This means that there is still freedom in choosing the difference in two nibbles of tweak, namely $T[2]$ and $T[5]$. Considering that $\Delta T[2] = x$ and $\Delta T[5] = y$, the difference in the left half of the round tweaks would be equal to $\Delta T_0' = \Delta T_1' = (0, 0, x, 0, 0, y, 0, 0)$ and $\Delta T_2' = \Delta T_3' = (0, 0, 0, y, 0, 0, 0, x)$.

Similar to the single-tweak differential case, we use an MILP tool to find all the activity patterns with minimum and close to minimum number of active S-boxes. We show the minimum number of active S-boxes in the related-tweak model of the weak tweak-key structure for 18 up to 22 round in Table 4. Note that in the related-tweak cases, the differentials are dependent on the starting round. For this reason, we use the index of RT to show the starting round.

One interesting observation in searching the activity patterns is that for most of them, there is no differential characteristics that can follow the activity pattern. It is important to mention that this is independent of the choice of S-box, and is only because of the

Table 4: The minimum number of active S-boxes and the maximum EDP for the related-tweak differentials within the activity patterns with minimum (or close to minimum) number of active S-boxes. Note that RT_i refers to the characteristic starting with round round $4 \cdot j + i$, and also, we use $p = -\log_2 \mathrm{EDP}$ instead of showing values for the EDP.

	minimum number of active S-boxes					highest expected differential probability				
r	18	19	20	21	22	18	19	20	21	22
RT_0	31	33	36	37	38	46.64	49.54	57.09	60.83	66.00
RT_1	32	35	36	37	40	47.54	55.54	56.00	62.25	64.96
RT_2	34	35	36	39	42	54.45	54.45	56.45	60.30	64.30
RT_3	32	33	36	39	40	47.54	49.54	55.54	63.54	66.25

set of linear equations has to be satisfied between the variables of difference in the input and output of the active S-boxes and also the active nibble(s) in the tweak difference. Therefore, before using the activity pattern to find the differentials with high EDP value, we used an algorithm to sieve the activity patterns which lead to an invalid set of linear equations. Briefly explaining, in this algorithm for a given activity pattern, we consider a variable for input difference and one another for output difference of each active S-box, and also one variable for each active nibble in the tweak. Then, based on the relations in the linear layer of the structure, we build a set of linear equations that must be satisfied for the given pattern. Each of these equations includes some variables which their xor sum must be equal to zero. After applying Gaussian elimination in this set of equations, there must be no equation with a single variable. Otherwise, it means that this single variable is equal to zero and this contradicts the activity of this variable. Hence, this algorithm can efficiently determine if the given activity pattern is a valid one.

After sieving the useful activity patterns, we again use the method of [ELR20] to compute the EDP of differentials within the activity pattern and find the differentials with highest EDP. It worth to recall that this method by fixing the input and output differences in the differential together with the tweak difference, we consider all different the single characteristics which follow the same activity pattern with the minimum number of active S-boxes. Table 4 summarizes the maximum EDP for the differentials within the activity patterns with minimum (or close to minimum) number of active S-boxes for 18 up to 22 rounds and for different index of starting round. For 22 rounds and more, we did not find any differentials within the activity patterns of minimum (or close to minimum) number of active S-boxes that has EDP of significantly higher than 2^{-64}. Note that in this table, instead of showing value of the EDP, we use $p = -\log_2 \mathrm{EDP}$.

As a result, we find several 21-round differentials with EDP higher than 2^{-64} for different index of RT; precisely, for each RT_i, we find only one activity pattern that includes several differentials with the highest possible EDP. These highest EDP values are $2^{-60.83}$, $2^{-62.25}$, $2^{-60.30}$ and $2^{-63.54}$ for RT_0, RT_0, RT_2 and RT_3, respectively, that are listed in Table 5. Note that in these activity patterns, the active and the inactive nibbles of states are denoted by `1` and `0`, respectively; besides, the values of the input difference (ΔP) and the output difference (ΔC) and tweak difference (ΔT) are shown in the hexadecimal. Moreover, since there are many choices for the differential with highest possible EDP, we simply denote the variables by $p, q, \ldots, y, z$ to show how the active nibbles in $\Delta P, \Delta C$ and ΔT are related. It might be interesting to mention that the number of differentials with highest EDP are 2688, 16, 24 and 250000, for trail 1, 2, 3 and 4, respectively.

About the weak tweak-keys for given related-tweak differentials in Table 5, we use the same approach as in Example 1 to find the conditions between the key and tweak nibbles. For both trail 1 and trail 2, we need to have

$$\begin{aligned} TK_0'''[0] = TK_0'''[3] = TK_0'''[6] = TK_0'''[7] = \mathtt{0}, \qquad & TK_2'''[0] = TK_2'''[6] = TK_2'''[7] = \mathtt{0}, \\ TK_1'''[0] = TK_1'''[3] = TK_1'''[6] = TK_1'''[7] = \mathtt{0}, \qquad & TK_3'''[0] = TK_3'''[6] = TK_3'''[7] = \mathtt{0}, \end{aligned}$$

which leads to

$$\begin{aligned} K_0[8] \;&= K_0[11] = K_1[8] = K_1[11] = T[4] \;= T[8] \;= T[11] \\ K_0[15] &= K_1[15] = \qquad\qquad T[13] = T[15] \\ K_0[14] &= K_1[14] = \qquad\qquad T[14]\ . \end{aligned}$$

For trail 3, we need to have

$$\begin{aligned} TK_1'''[0] = TK_1'''[3] = TK_1'''[6] = TK_1'''[7] = \mathtt{0}, \qquad & TK_0'''[3] = TK_0'''[7] = \mathtt{0}, \\ TK_2'''[0] = TK_2'''[2] = TK_2'''[6] = TK_2'''[7] = \mathtt{0}, \qquad & TK_3'''[0] = TK_3'''[6] = TK_3'''[7] = \mathtt{0}, \end{aligned}$$

that is same as

$$\begin{array}{llll} K_0[8] & = K_0[11] = K_1[8] = K_1[11] = T[4] & = T[8] & = T[11] \\ K_0[15] = K_1[15] = & & = T[13] = T[15] & \\ K_0[14] = K_1[14] = & & = T[14] & \\ K_0[13] = & & = T[0]\,. & \end{array}$$

And for trail 4, we need to have

$$\begin{array}{r} TK_0'''[1] = TK_0'''[2] = TK_0'''[3] = TK_0'''[6] = TK_0'''[7] = 0, \\ TK_1'''[2] = TK_0'''[3] = TK_0'''[5] = TK_0'''[6] = TK_0'''[7] = 0, \\ TK_2'''[0] = TK_2'''[1] = TK_2'''[2] = TK_2'''[3] = TK_2'''[5] = TK_2'''[6] = TK_2'''[7] = 0, \\ TK_3'''[0] = TK_3'''[1] = TK_3'''[2] = TK_3'''[3] = TK_3'''[5] = TK_3'''[6] = TK_3'''[7] = 0, \end{array}$$

which is equal to

$$\begin{array}{llll} K_0[8] & = K_0[11] = K_1[8] & = K_1[11] = T[4] & = T[8] & = T[11] \\ K_0[13] = K_0[15] = K_1[13] = K_1[15] = & & T[0] & = T[13] & \\ K_0[12] = K_1[12] = & & = T[6] & = T[12] & \\ K_0[14] = K_1[14] = & & = T[1] & = T[14] & \\ K_0[9] & = K_1[9] & = & = T[3] & = T[9] \\ K_1[10] = & & & = T[7]\,. & \end{array}$$

All together, the size of the space for weak keys, weak tweaks, and weak tweak-keys are 2^{108}, 2^{52} and 2^{148} for both trail 1 and 2, 2^{108}, 2^{52} and 2^{144} for trail 3, 2^{92}, 2^{40} and 2^{108} for trail 4.

Table 5: 21-round related-tweak differentials with the highest EDP within the differentials of the activity patterns with the minimum (or close to minimum) number of active S-boxes.

trail 1 with ST_0	*trail 2 with* ST_1	*trail 3 with* ST_2	*trail 4 with* ST_3
00110001 10000000	10100011 00010010	00010011 10000000	11101101 01100100
10000000 10000000	00010010 00000001	10000000 00000011	01100100 00100000
10000000 10000010	00000001 00000011	00000011 00010001	00100000 00000000
10000010 10000000	00000011 10010000	00010001 00010001	00000000 00000000
10000000 10010010	10010000 00010001	00010001 00000011	00000000 00000001
10010010 00010011	00010001 10000010	00000011 10000000	00000001 00010001
00010011 10000000	10000010 10000000	10000000 00010011	00010001 00010011
10000000 00000011	10000000 10010010	00010011 10010010	00010011 10000011
00000011 00010001	10010010 00010011	10010010 10000000	10000011 10000011
00010001 00010001	00010011 10000000	10000000 10000010	10000011 00010011
00010001 00000011	10000000 00000011	10000010 00010001	00010011 00010001
00000011 10000000	00000011 00010001	00010001 10010000	00010001 00000001
10000000 00010011	00010001 00010001	10010000 00000011	00000001 00000000
00010011 10010010	00010001 00000011	00000011 10000001	00000000 00000000
10010010 10000000	00000011 10000000	10000001 00010011	00000000 00100000
10000000 10000010	10000000 00010011	00010011 00010011	00100000 01100100
10000010 00010001	00010011 10010010	00010011 10000001	01100100 10101101
00010001 10010000	10010010 10000000	10000001 00000011	10101101 01000000
10010000 00000011	10000000 10000010	00000011 00010000	01000000 00000101
00000011 00000001	10000010 00010000	00010000 00000001	00000101 00110000
00000001 00010010	00010000 10000000	00000001 00100000	00110000 01100010
00010010 10100011	10000000 00110001	00100000 01000100	01100010 11111101
ΔT 00x00000 00000000	00x00000 00000000	00a00000 00000000	00a00000 00000000
ΔP 00xy000t a0000000	y0z000tx 000a00x0	000x00ax y0000000	xya0xa0a 0za00z00
ΔC 000w00x0 t0u000vx	v0000000 00xa000u	00a00000 0a000a00	0au000v0 pqastq0a

In order to do a key recovery attack, the attacker can append several rounds before and after the above 21-round related tweak differentials. But, it should be take in account that the number of appended rounds before the differential starting with round i, RT_i, is either i or $i+4$. For the above differentials, appending more than 3 rounds actives all the nibbles of the stater; hence, for the differentials with RT_i we can append i rounds in before the differential. However, there is no such a restriction on the number of rounds to append after the differentials and it is possible to extend those differentials by adding at most 3, 2, 3 and 3 rounds for key recovery. Therefore, it is possible that an attacker have a successful related-tweak differential attack on 24, 24, 26 and 26 rounds, respectively. It is noteworthy that here, we did not consider the time complexity for the key recovery to check the feasibility of the attack. Hence, the number of rounds that can be analyzed by the attacker is an upper bound.

Note that even though, we only analyzed security of the weak tweak-key structure against differential and linear cryptanalyses, but since we believe that it can be applied to improve the result against other attacks (such as impossible differential, zero-correlation linear hull, meet-in-the-middle, and integral), in contrary to [GSS+20], we keep the general weakness as it is and do not specialize the weakness only for differential attack.

4.5 Differential Properties of $S_c^* := S\Big(S(\cdot) \oplus c\Big)$

We already mentioned that since S is an involution, S_0^* is the same as identity function. This makes it possible for any input difference of $\alpha \in \mathbb{F}_2^4$ to transit to the same difference in the output of S-box S_0^* with probability 1. Here, we study the probability for transition of an input difference α to the same difference in the output of S_c^* for non-zero values of $c \in \mathbb{F}_2^4$. Table 6 shows the number of $x \in \mathbb{F}_2^4$ such that for each given c and α, $S_c^*(x \oplus \alpha) \oplus S_c^*(x) = \alpha$.

Interestingly, there are some high values in this table; specially, there are two non-zero values for c and α pair which the input difference α stays the same in the output with probability 1; namely for $c = \alpha = \mathtt{2}$ and $c = \alpha = \mathtt{a}$. For our application, this means that even if the corresponding nibble of the round tweakey for an active S-box of S^* is not zero (i.e. the tweakey value is not included in the weak tweak-key space), it may lead to a high EDP. But this EDP is always smaller than the one we computed for the weak tweak-keys

Table 6: Number of entries x for $S_c^*(x \oplus \alpha) \oplus S_c^*(x) = \alpha$.

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

(which this nibble of the round tweakey is necessarily zero) and this is because of the restriction in the values of the input/output difference of S^* S-box. For instance, in case of $c = \mathtt{a}$, while only the input difference value of $\mathtt{a}$ can transit with probability 1, input difference with a value in $\{\mathtt{5}, \mathtt{7}, \mathtt{d}, \mathtt{f}\}$ can also transit to the same difference in the output with probability 2^{-1}. Therefore, the differentials discussed in the previous sections are not only useful for the tweakeys within the weak tweak-key spaces, but it might be possible to be applied for the tweakeys out of the weak tweak-key spaces by using a smaller EDP.

It is noteworthy to mention that the transition probability for $c = \alpha = \mathtt{a}$, previously was observed and applied in [GSS+20] to enlarge the weak tweak-key space. While their technique makes it possible to enlarge the weak tweak-key space, it fixes the difference value in some intermediate nibbles. Therefore, the EDP of the differential gets smaller in favor of making the weak tweak-key space larger. This can be used as a trade-off between the EDP and the size of weak tweak-key space, which in case of a key recovery attack, both of these parameters affect number of needed plaintext differential pairs. Hence, the attacker can take advantage of this to reduce the data or time complexity of the attack.

5 Differential Key Recovery Attack

In this section, we describe a related-tweak differential key recovery attack on 25-round CRAFT block cipher that is based on the previously described weak tweak-key structure of the cipher. To do this, we use the 21-round activity pattern entitled trail 3 at Table 5 which is an RT_2 trail and starts from the second round. We extend the differentials with highest EDP in this activity pattern by appending two rounds in the beginning (in the plaintext side) and another two round at the end (in the ciphertext side) of the trail.

In the following, we describe the form of weak tweak-key space, the attack procedure, and complexity of the attack in detail. As a result, we recover the key (out of 2^{108} weak keys) using 2^{95} related-tweak data, in about 2^{95} time of 25-round CRAFT encryptions and with 2^{38} blocks of memory usage.

5.1 Overview of the Attack

The highest EDP for differentials within the trail 3 of Table 5, is $2^{-60.30}$ and it happens for 24 differentials. The difference in the tweak value (ΔT) and in the output after 21 rounds (ΔX_{23}), for all of these differentials are the same, while they all have different differnce value in the input (ΔX_2). The value of these differences are shown below.

$$\Delta T_0 = \Delta T_1 = (\mathtt{00a0\ 0000\ 0000\ 0000}), \quad \Delta T_2 = \Delta T_3 = (\mathtt{0000\ 000a\ 0000\ 0000}),$$
$$\Delta X_2 = (\mathtt{000}x\ \mathtt{00a}x\ y\mathtt{000\ 0000}), \quad \Delta X_{23} = (\mathtt{00a0\ 0000\ 0a00\ 0a00}),$$

$$(x, y) \in \{(\mathtt{1,2}),(\mathtt{2,1}),(\mathtt{2,4}),(\mathtt{2,9}),(\mathtt{2,c}),(\mathtt{3,6}),(\mathtt{4,2}),(\mathtt{5,7}),(\mathtt{5,a}),(\mathtt{6,3}),(\mathtt{7,5}),(\mathtt{7,d}),$$
$$(\mathtt{9,2}),(\mathtt{a,5}),(\mathtt{a,a}),(\mathtt{a,d}),(\mathtt{a,f}),(\mathtt{b,b}),(\mathtt{c,2}),(\mathtt{d,7}),(\mathtt{d,a}),(\mathtt{e,e}),(\mathtt{f,a}),(\mathtt{f,f})\}.$$

We recall that to achieve these 21-round related-tweak differentials we need to have

$$TK_1'''[0] = TK_1'''[3] = TK_1'''[6] = TK_1'''[7] = \mathtt{0}, \qquad TK_0'''[3] = TK_0'''[7] = \mathtt{0},$$
$$TK_2'''[0] = TK_2'''[2] = TK_2'''[6] = TK_2'''[7] = \mathtt{0}, \qquad TK_3'''[0] = TK_3'''[6] = TK_3'''[7] = \mathtt{0},$$

which makes the weak tweak and weak key in the following forms:

$$K_0 = (k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}, k_8, k_{12}, k_{13}, k_{14}, k_{15}),$$
$$K_1 = (k'_0, k'_1, k'_2, k'_3, k'_4, k'_5, k'_6, k'_7, k_8, k'_9, k'_{10}, k_8, k'_{12}, k'_{13}, k_{14}, k_{15}),$$
$$T = (t_0, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_4, t_9, t_{10}, t_4, t_{12}, t_{13}, t_{14}, t_{13}),$$

with four conditions between key and tweak: $k_8 = t_4,\ k_{13} = t_0,\ k_{14} = t_{14},\ k_{15} = t_{13}$. Therefore, there are 2^{108} weak keys, 2^{52} weak tweaks, and 2^{144} weak tweak-keys.

Since, the values for key nibbles k_8, k_{13}, k_{14} and k_{15} are unknown to the attacker, it is not possible to check their equality with t_4, t_0, t_{14} and t_{13}, respectively. However, the attacker can repeat the key recovery attack for all 2^{16} different values of t_4, t_0, t_{14} and t_{13} and see that for which value of these 4 nibbles, the differential distinguisher occurs as it is expected. In other words, each of these differential will occur with probability of $2^{-60.30}$, if $k_8 = t_4,\ k_{13} = t_0,\ k_{14} = t_{14},\ k_{15} = t_{13}$; otherwise, it occurs with probability of 2^{-64}.

Extending the differentials by two rounds in the plaintext side, activates 11 nibbles in the plaintext difference, namely nibbles 0, 1, 2, 3, 5, 6, 7, 8, 13, 14 and 15. On the other side, extending the differentials by two rounds in the ciphertext side, activates 14 nibbles in the ciphertext difference, namely nibbles 0, 1, 2, 3, 4, 5, 7, 10, 11, 12, 13 and 15. We depict the details of both extensions in Figure 4 in the Appendix.

5.2 Attack Procedure

As mentioned before, to recover the whole key, we need to repeat the attack for each 2^{16} values of t_0, t_4, t_{13} and t_{14}. Therefore, we first fix the other 9 nibbles of tweak $t_1, t_2, t_3, t_5, t_6, t_7, t_9, t_{10}, t_{12}$, and then choose the value for t_0, t_4, t_{13}, t_{14}.

Collecting Data Structures We choose a random plaintext $PT = (p_0, \ldots, p_{15})$ and ask for its encryption $CT = (c_0, \ldots, c_{15})$ using the aforementioned built weak tweak. Then, we ask for encryption of 2^{20} plaintexts in the form of $PT' = (p'_0, \ldots, p'_{15})$ with the following 11 conditions using the corresponding related-tweak.

$$p'_4 = p_4\,, \quad p'_9 = p_4\,, \quad p'_{10} = p_{10}\,, \quad p'_{11} = p_{11}\,, \quad p'_{12} = p_{12}\,,$$

$$p'_1 \oplus p'_5 = p_1 \oplus p_5\,, \quad p'_1 \oplus p'_{13} = p_1 \oplus p_{13}\,, \quad p'_2 \oplus p'_6 = p_2 \oplus p_6\,,$$

$$p'_3 \oplus p'_7 = p_3 \oplus p_7\,, \quad p'_4 \oplus p'_8 = p_4 \oplus p_8\,, \quad p'_2 \oplus p'_{14} = p_2 \oplus p_{14} \oplus \mathtt{a}\,.$$

Note that while the first five conditions are necessary for in-active nibbles in the plaintext difference ΔP, the other six conditions ensures that the right half of ΔX_1 is in the right form. Besides, to find the value for those 2^{20} plaintexts, it is enough to go through all the choices for p'_0, p'_1, p'_2, p'_3 and p'_{15}; that is by knowing the exact value for PT, p'_0, p'_1, p'_2, p'_3 and p'_{15}, the exact value for PT' will be determined.

These data structure include one data using the weak tweak and 2^{20} data using the related tweak. However, it builds 2^{20} differential pairs that already insures the difference in the right half of ΔX_1. The probability that one of these differentials lead to those 24 differentials in ΔX_2 is $24 \cdot 2^{-20}$. That is, in each data structure, averagely there are 24 differential pairs with the correct difference in ΔX_2. Therefore, for each t_0, t_4, t_{13}, t_{14} value, the attacker needs to use $a \cdot 2^{60.30}/24 \approx a \cdot 2^{56}$ data structures ($a \cdot 2^{76}$ differential pairs) to insure there are a of right differential pairs with the corresponding ΔX_{23}.

Filtering Wrong Differential Pairs On the other side of the differential pairs, the attacker can use the ciphertext differentials to sieve the potantially right differential pairs. These conditions are listed below.

$$c'_6 = c_6\,, \quad c'_8 = c_8\,, \quad c'_9 = c_9\,, \quad c'_{14} = c_{14}\,, \quad c'_0 \oplus c'_4 = c'_0 \oplus c'_4\,, \quad c'_0 \oplus c'_{12} = c'_0 \oplus c'_{12}\,,$$

$$c'_2 \oplus c'_{10} = c'_2 \oplus c'_{10}\,, \quad c'_3 \oplus c'_{11} = c'_3 \oplus c'_{11}\,, \quad c'_7 \oplus c'_{15} = c'_7 \oplus c'_{15}\,, \quad S(c'_{11}) = S(c_{11}) \oplus \mathtt{a}\,.$$

It is noteworthy to mention that the first four conditions are necessary for in-active nibbles in the ciphertext difference ΔC, the other five conditions ensures that the left half of ΔX_{24} is in the correct form and the last one is to ensure the difference in $\Delta X_{25}[15]$. All together, for each t_0, t_4, t_{13}, t_{14} value, after this filtering, from all $a \cdot 2^{76}$ differential pairs, there will be only $a \cdot 2^{76-40} = a \cdot 2^{36}$ of them left. We save all these remaining differentials pairs in a memory to use them in the key recovery step.

Recovering Key Nibbles The attacker can recover 13 nibbles of the key by using the following twelve equations which are based on the partial encryption or decryption of the differential pairs. These 13 nibbles of the key are included in $TK_0'''[0,1,2,4,6]$, $TK_0'[1,3,5,6,7]$, $TK_3'''[1,5]$ and $TK_1'[0]$.

$$S(c_{13} \oplus TK_0'''[2]) = S(c'_{13} \oplus TK_0'''[2]) \oplus \mathtt{a}$$

$$S(c_{10} \oplus TK_0'''[4]) \oplus S(c_{15} \oplus TK_0'''[0]) = S(c'_{10} \oplus TK_0'''[4]) \oplus S(c'_{15} \oplus TK_0'''[0])$$

$$S(c_{12} \oplus TK_0'''[1]) \oplus S(c_5 \oplus c_{13} \oplus TK_0'[5]) = S(c'_{12} \oplus TK_0'''[1]) \oplus S(c'_5 \oplus c'_{13} \oplus TK_0'[5])$$

$$S(c_{15} \oplus TK_0'''[0]) \oplus S(c_1 \oplus c_9 \oplus c_{13} \oplus TK_0'[1]) = S(c'_{15} \oplus TK_0'''[0]) \oplus S(c'_1 \oplus c'_9 \oplus c'_{13} \oplus TK_0'[1])$$

$$S\big(S(c_5 \oplus c_{13} \oplus TK_0'[5]) \oplus TK_3'''[5]\big) = S\big(S(c'_5 \oplus c'_{13} \oplus TK_0'[5]) \oplus TK_3'''[5]\big) \oplus \mathtt{a}$$

$$S\big(S(c_1 \oplus c_9 \oplus c_{13} \oplus TK_0'[1]) \oplus TK_3'''[1]\big) = S\big(S(c'_1 \oplus c'_9 \oplus c'_{13} \oplus TK_0'[1]) \oplus TK_3'''[1]\big) \oplus \mathtt{a}$$

$$S(m_7 \oplus m_{15} \oplus TK_0'[7]) \oplus S(m_{14}) = S(m'_7 \oplus m'_{15} \oplus TK_0'[7]) \oplus S(m'_{14})$$

$$S(m_3 \oplus m_{11} \oplus m_{15} \oplus TK_0'[3]) \oplus S(m_8 \oplus TK_0'''[2]) = S(m'_3 \oplus m'_{11} \oplus m'_{15} \oplus TK_0'[3]) \oplus S(m'_8 \oplus TK_0'''[2])$$

$$S(m_8 \oplus TK_0'''[2]) \oplus S(m_{13} \oplus TK_0'''[2]) = S(m'_8 \oplus TK_0'''[2]) \oplus S(m'_{13} \oplus TK_0'''[2]) \oplus \mathtt{a}$$

$$S(m_6 \oplus m_{14} \oplus TK_0'[6]) \oplus S(m_{15} \oplus TK_0'''[0]) = S(m'_6 \oplus m'_{14} \oplus TK_0'[6]) \oplus S(m'_{15} \oplus TK_0'''[0]) \oplus y$$

$$S\big(S(m_{15} \oplus TK_0'''[0]) \oplus S(m_6 \oplus m_{14} \oplus TK_0'''[6]) \oplus S(m_1 \oplus m_9 \oplus m_{13} \oplus TK_0'''[1]) \oplus TK_1'[0]\big) \oplus$$

$$S\big(S(m'_{15} \oplus TK_0'''[0]) \oplus S(m'_6 \oplus m'_{14} \oplus TK_0'''[6]) \oplus S(m'_1 \oplus m'_9 \oplus m'_{13} \oplus TK_0'''[1]) \oplus TK_1'[0]\big) = x$$

Note that the average probability for satisfying each of above equations are 2^{-4}, except for the second last one which for each value of $x = m_7 \oplus m_{15} \oplus m'_7 \oplus m'_{15}$, in average there are 1.5 possible candidates for y. Hence, for each remaining differential pair, there are in average $2^{52} \cdot 3 \cdot 2^{-49} = 24$ key candidates that satisfy all the equations. Moreover, this means that in average, each of these keys will be counted about $a \cdot 3 \cdot 2^{36-49} = 3 \cdot a \cdot 2^{-13}$ times, while for the right value of the key it is expected to be a times. In other meaning, signal to noise ratio of the differential key recovery attack is about 2^{11}.

To not use a memory for 2^{52} key counters, instead, the attacker can guess value of these 13 key nibbles and then find the differentials which this key value is a candidate to satisfy those 12 equations. Note that if the attacker guesses all 52 key bits at once, then the computation time of this step will be dominant. To avoid this problem, he can guess the key nibbles one-by-one and at each level he checks for the corresponding equation. For instance, he can start with the corresponding key nibble of the equations which only need one key nibble to be guessed. In this way, the computation complexity of this step will be reduced significantly.

Beyond these 13 key nibbles, the value of t_0, t_4, t_{13}, t_{14} which the right differential pairs happen for determines value of the other 4 key nibbles. This leaves $27 - 13 - 4 = 10$ key nibbles that can be found by doing an exhaustive search.

Attack Complexity Since the signal to noise ratio of the attack is quite high, $a = 4$ would be enough to recover the whole 108 bit of the weak key. Overall, we need to ask for $2^{16} \cdot 2^{20} \cdot 4 \cdot 2^{56} = 2^{94}$ data encryption which determines the data and time complexity of the attack. Besides, to keep the possible right differential pairs for each value of t_0, t_4, t_{13}, t_{14}, we need 2^{38} blocks of memory.

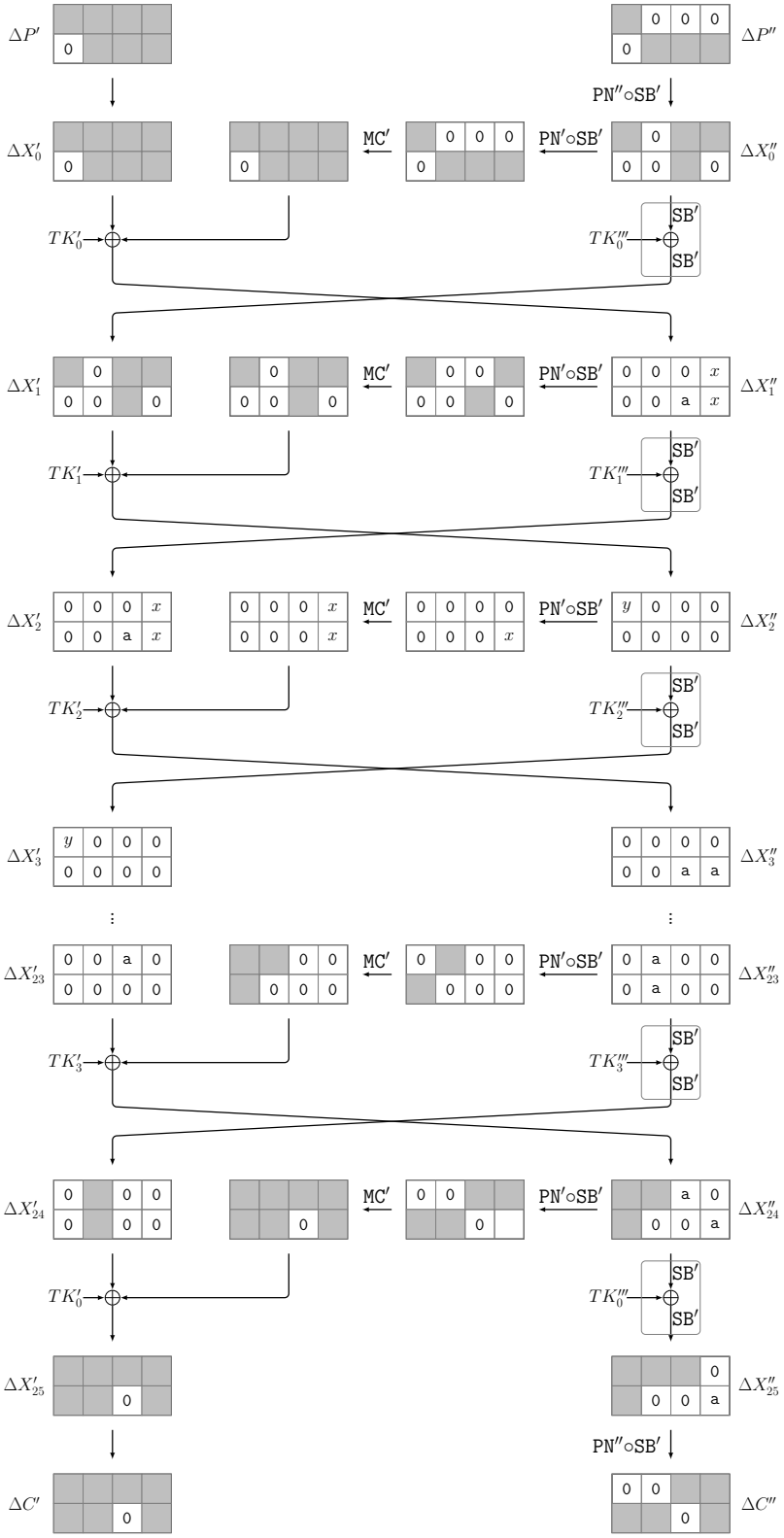

Figure 4: Extending the 21-round activity pattern shown as trail 3 in Table 5 by two rounds in the beginning and two rounds at the end. Note that the third round is shown due to the key recovery matter. Besides, the numbers or the variables shown in each nibble determines the difference of the nibble and for the nibbles which their difference is not fixed we show them by gray color.

6 Hardware Implementation of the New Structure

Even though with the current twekey schedule, the new structure of the CRAFT block cipher with Fiestel network provides less cryptographic security than the original design, in this section, we show that the new structure makes it possible to have a better implementation of the cipher specially when the encryption is combined with decryption and also when it is combined with the countermeasures against fault analysis. Therefore, by designing another tweakey schedule for the Fiestel network structure of the cipher, it is possible to introduce a new tweakable block cipher which provides 128-bit key and 64-bit tweak effectively. We emphasize that the new cipher msut use another tweakey schedule than the one in Lemma 3 (with effective 64-bit key and effective 32-bit tweak).

To this point, we start with recalling the specification of the new structure. The new structure is a Feistel-network block cipher with the following round function that we denote the round tweakeys by tk_is.

$$\mathcal{R}_i(x_{i-1}, x_i) = (x_i, x_{i+1}) \text{ with } x_{i+1} = x_{i-1} \oplus \mathcal{F}_i(x_i) \text{ and } \mathcal{F}_i := \mathtt{A}_{RC'_i} \circ \mathtt{MC'} \circ \mathtt{PN'} \circ \mathtt{SB'} \circ \mathtt{A}_{tk_i} .$$

Encryption Only: Since with an equal number of rounds, the new structure is less cryptographically secure than the original CRAFT structure, it is necessary to use more number of rounds to provide a secure cipher using the new structure. In the round-based design architecture (which was the target implementation of the CRAFT cipher), the overhead of increasing number of rounds is on the increasing number of clock cycles used for an encryption and does not affect the area cost of implementation.

Figure 5 illustrates the round-based implementation of the new structure excluding the circuit for the tweakey structure. Compared to the round-based implementation of CRAFT shown in [BLMR19, Figure 8], the new design decreases the area of implementation and it is because of using 8 S-boxes per round (instead of 16) and using 32 XOR gates for the key addition (instead of 64).

Encryption & Decryption Thanks to the being a Feistel network, in the new structure, the decryption is the same as the encryption with the reverse order of the round constants and round tweakeys. Therefore, to provide the decryption on top of the encryption, we only need to add extra circuits for reversing the order of the round constants and the round tweakeys. For reversing order of the round constants, we only need to implement the inverse of the updating function for the LFSRs. Besides, if the tweakey schedule iterates the round-tweakeys (similar to the one in the CRAFT cipher), it is possible to reverse the order of the round tweakeys by updating the circuits for the selector bits of the multiplexer choosing the round tweakey. Note that both of these techniques are already already applied in the CRAFT design with a very small area overhead.

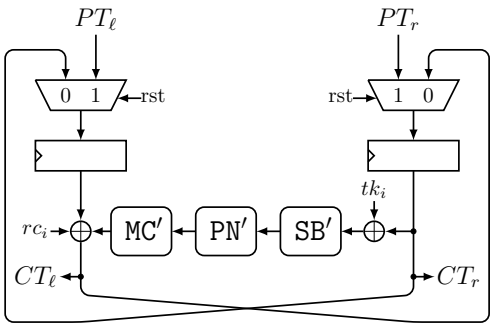

Figure 5: Round-based design architecture of the new structure.

The prominent difference of combining encryption and decryption for the new structure and the CRAFT cipher is that in the case of the latter one, we need to modify the round tweakeys from TK_i to $\texttt{MC}(TK_i)$. This modification needs extra 48 XOR (to implement MC circuit) and 32 MUX gates (to choose between these two round tweakeys). All together, the encryption+decryption circuit of the new structure is smaller than the one for CRAFT cipher with difference of 8 Sboxes, 80 XOR and 32 MUX gates which in the IBM 130nm ASIC library equals to 328 GE.

It is noteworthy to mention that here, we do not consider the difference in the implementation of the new tweakey schedule. Just as an example, consider the 128-bit master key as four 32-bit keys $(k_0\|k_1\|k_2\|k_3)$, and the 64-bit tweak as two 32-bit tweakeys $(t_0\|t_1)$. The similar tweakey schedule to the CRAFT tweakey schedule would be iterating following 12 round tweakeys:

$$\begin{array}{llllll} tk_0 = k_0 \oplus t_0, & tk_1 = k_1, & tk_2 = k_2 \oplus t_1, & tk_3 = k_3 \oplus t_0, & tk_4 = k_0, & tk_5 = k_1 \oplus t_1, \\ tk_6 = k_2 \oplus t_0, & tk_7 = k_3, & tk_8 = k_0 \oplus t_1, & tk_9 = k_1 \oplus t_0, & tk_{10} = k_2, & tk_{11} = k_3 \oplus t_1. \end{array}$$

Please note that the new tweakey schedule not only has smaller footprint in the implementation, it also makes the Time-Data-Memory Trade-off attack mentioned in [BLMR19, Section 5.3] impossible.

Applying Fault Analysis Countermeasures In [AMR+20, SRM20], it is shown that applying concurrent error detection-based countermeasures against fault analysis have overhead of more than 100% (even if there is one bit redundancy per nibble) and this gets worse in the case of error correction-based countermeasures. The main reason for this is that the *check points* or the *correction points* in these countermeasures are several times larger than the circuit for the round operation.

While in the fault analysis countermeasures for the CRAFT cipher, it is necessary to have 16 check points (or correction points), for the new structure having only 8 of these points suffices to protect against a univariate adversary model. We recall that the univariate adversary model considers an adversary that can inject up-to a *known* limited number of faults in the entire of encryption. Therefore, the new structure makes it possible to have even more efficient design for protection against fault analysis attacks.

7 Conclusion

Most of the security analysis for new block cipher designs are based on the assumption of independent round keys. While this is not an issue in most cases, it may result in overestimating the resistant of the design, in particular for in the weak-key scenario.

In this work, we showed how the SPN structure of CRAFT block cipher (with full-state non-linear layer) changes to a Feistel-network structure (with half-state non-linear layer) in the weak tweak-key scenario. Consequently, in the same number of number of rounds, the weak tweak-key structure of the cipher is less resistant against differential and linear cryptanalyses. As an application of this observation, we present a weak-key related-tweak differential attack on 25-rounds of the cipher with time and data complexity of 2^{94} encryptions that works for 2^{108} weak keys.

On the constrictive side, the new structure suggests some possible improvements in the hardware implementation of the cipher. It is possible to use the new structure as a new block cipher with improved hardware efficiency. All that is left here is to develop an appropriate tweakey schedule. The benefits are providing decryption on top of encryption with almost zero overhead, and reducing the overhead of protecting against FA attacks by factor of about two.

References

[AMR+20] Anita Aghaie, Amir Moradi, Shahram Rasoolzadeh, Aein Rezaei Shahmirzadi, Falk Schellenberg, and Tobias Schneider. Impeccable circuits. *IEEE Trans. Computers*, 69(3):361–376, 2020.

[BBI+15] Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A block cipher for low energy. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 411–436. Springer, Heidelberg, November / December 2015.

[BLMR19] Christof Beierle, Gregor Leander, Amir Moradi, and Shahram Rasoolzadeh. CRAFT: Lightweight tweakable block cipher with efficient protection against DFA attacks. *IACR Trans. Symm. Cryptol.*, 2019(1):5–45, 2019.

[BS97] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 513–525. Springer, Heidelberg, August 1997.

[ELR20] Maria Eichlseder, Gregor Leander, and Shahram Rasoolzadeh. Computing expected differential probability of (truncated) differentials and expected linear potential of (multidimensional) linear hulls in SPN block ciphers. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *INDOCRYPT 2020*, volume 12578 of *LNCS*, pages 345–369. Springer, Heidelberg, December 2020.

[EY19] Muhammad ElSheikh and Amr M. Youssef. Related-key differential cryptanalysis of full round CRAFT. Cryptology ePrint Archive, Report 2019/932, 2019. https://eprint.iacr.org/2019/932.

[GSS+20] Hao Guo, Siwei Sun, Danping Shi, Ling Sun, Yao Sun, Lei Hu, and Meiqin Wang. Differential attacks on CRAFT exploiting the involutory s-boxes and tweak additions. *IACR Trans. Symmetric Cryptol.*, 2020(3):119–151, 2020.

[HSN+19] Hosein Hadipour, Sadegh Sadeghi, Majid M. Niknam, Ling Song, and Nasour Bagheri. Comprehensive security analysis of CRAFT. *IACR Trans. Symm. Cryptol.*, 2019(4):290–317, 2019.

[MA19] AmirHossein E. Moghaddam and Zahra Ahmadian. New automatic search method for truncated-differential characteristics: Application to midori, SKINNY and CRAFT. Cryptology ePrint Archive, Report 2019/126, 2019. https://eprint.iacr.org/2019/126.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Inscrypt 2011*, volume 7537 of *LNCS*, pages 57–76. Springer, December 2011.

[SHW+14] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 158–178. Springer, Heidelberg, December 2014.

[SRM20] Aein Rezaei Shahmirzadi, Shahram Rasoolzadeh, and Amir Moradi. Impeccable circuits II. In *DAC 2020*, pages 1–6. IEEE, 2020.