IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 0, No. 0, pp. 0–0. DOI:10.46586/tosc.v0.i0.0-0

Mind Your Path: On (Key) Dependencies in Differential Characteristics

Thomas Peyrin[1] and Quan Quan Tan[1]

Nanyang Technological University, Singapore
thomas.peyrin@ntu.edu.sg,quanquan001@e.ntu.edu.sg

Abstract. Cryptanalysts have been looking for differential characteristics in ciphers for decades and it remains unclear how the subkey values and more generally the Markov assumption impacts exactly their probability estimation. There were theoretical efforts considering some simple linear relationships between differential characteristics and subkey values, but the community has not yet explored many possible nonlinear dependencies one can find in differential characteristics. Meanwhile, the overwhelming majority of cryptanalysis works still assume complete independence between the cipher rounds. We give here a practical framework and a corresponding tool to investigate all such linear or nonlinear effects and we show that they can have an important impact on the security analysis of many ciphers. Surprisingly, this invalidates many differential characteristics that appeared in the literature in the past years: we have checked differential characteristics from 8 articles (4 each for both SKINNY and GIFT) and most of these published paths are impossible or working only for a very small proportion of the key space. We applied our method to SKINNY and GIFT, but we expect more impossibilities for other ciphers. To showcase our advances in the dependencies analysis, in the case of SKINNY we are able to obtain a more accurate probability distribution of a differential characteristic with respect to the keys (with practical verification when it is computationally feasible). Our work indicates that newly proposed differential characteristics should now come with an analysis of how the key values and the Markov assumption might or might not affect/invalidate them. In this direction, more constructively, we include a proof of concept of how one can incorporate additional constraints into Constraint Programming so that the search for differential characteristics can avoid (to a large extent) differential characteristics that are actually impossible due to dependency issues our tool detected.

Keywords: differential cryptanalysis · key dependent characteristics · lightweight ciphers · block ciphers

1 Introduction

Differential cryptanalysis is one of the earliest types of cryptanalysis in modern cryptography. It was first introduced by Eli Biham and Adi Shamir to analyze DES-like cryptosystems [BS91]. To simplify the probabilistic analysis of their attack, they assumed that all the subkeys generated by the master key were independent. This is extremely useful especially as most of the block cipher designs are iterative; an analysis for a single round can help us estimate the strength of the cipher using the Markov cipher assumption [LMM91]. Cryptanalysts also use the (reduced) number of rounds that can be attacked as a baseline for the security of the cipher. Differential cryptanalysis has become so important that it is now a de-facto requirement for cryptographic primitive designers to provide strong arguments of the resistance of their design against such attacks.

In [DR07], Daemen and Rijmen introduced the concept of plateau characteristics: differential characteristics that either have a probability of $p \neq 0$ or 0 only when given

a fixed key. This effect is caused by simple linear dependencies between two consecutive rounds. The authors also proved that for two rounds of AES, as well as many other ciphers, all the differential characteristics are plateau characteristics. This is an indication that one should not completely ignore the key values when manipulating differential characteristics.

In the most recent research papers that focus on differential cryptanalysis, automated methods are used to find differential characteristics for the targeted cipher(s) [DDH+20, ZDY19]. These works again often employ the assumption that the subkeys are independent and that each round can be treated independently. Indeed, if the key schedule of a cipher is complex enough and also if the cipher has a good diffusion, it may help to transform any distribution to a uniform one, and the impact of the key values on the probability may be minimal. It is also fair to mention that it is complex to do a proper theoretical analysis when key values come into the picture. However, even if the analysis is difficult, this does not remove the fact that a differential characteristic may be dependent on the values of the key.

In the past decade, we have seen a shift in the cipher designs gearing towards lightweight applications. This usually implies a lighter round function, accompanied by a lighter key schedule for the cipher. This means that the validity of the key independence assumption might be even less true for this new design paradigm. This is further amplified by the fact that the diffusion of the round function also decreases, increasing the dependencies between the state and the key values. While using the techniques described using plateau characteristics may help to partially deduce what key values are not possible, we still do not really have a full picture of the situation. This is mainly because of the other effects that are in play that were not described yet. In this paper, we aim to shed some light on this open problem.

Our Contributions. In this article we propose several contributions. Firstly, we noticed that past works on finding right pairs for differential characteristics only focused on linear dependencies (we call them here *linear constraints*, they will induce plateau characteristics), which lead to necessary conditions on the round keys in order for the differential characteristic to be valid. Alongside these linear constraints, we identify a new and more complex type of dependency that we call *nonlinear constraints*. Including this new effect allows us to better model the behavior of the differential characteristic and we can also extract sufficient conditions on the key for the differential characteristic to work. These nonlinear constraints are responsible for most of the "non-plateau" differential characteristics. We consider all these linear and nonlinear constraints and provide a new framework that cryptanalysts can employ to look for potential key/round dependencies. We have also created a tool based on this framework that is able to detect, very efficiently, these linear and nonlinear constraints the differential characteristics induced on the key.

In addition, when these dependencies are not too complex (typically not stretching over a too large number of complex rounds, otherwise it would be computationally too intensive to handle), we can approximate the size of the valid key space and produce an estimation of the probability distribution of a differential characteristic according to the key values. We show for example that this is achievable on SKINNY cipher [BJK+16] and we have conducted experiments to verify our modeling when it was computationally feasible. For the GIFT cipher [BPP+17], we use the linear constraints to deduce an upper bound on the size of the key space (though we did use the nonlinear constraints to check for inconsistencies). Together with the nonlinear constraints, our tool is able to detect if these constraints are actually going to cause the differential characteristic to be impossible, either due to round or key dependencies. We applied our tools to existing SKINNY and GIFT differential characteristics [DDH+20, HBS21, QDW+21, AST+17, SWW21, LWZZ19, ZDY19, LLL+21] and we discovered that 21 out of the 43 SKINNY characteristics and 1 out of the 15 GIFT characteristics we tested are actually impossible. Most of the remaining characteristics

that are possible actually only work on a small subset of the entire key space available. In fact, only 1 out of the 43 SKINNY characteristics and 5 out of the 15 GIFT characteristics tested work for half or more of all possible keys. In the case of GIFT, the differential characteristic that we found to be impossible was actually due to a round dependency issue and this issue is present regardless of the key schedule used. The results are summarized in Table 1, Table 2 and Table 3 for SKINNY-64, SKINNY-128 and GIFT respectively.

Finally, we provide some workarounds as to what cryptanalysts could do to detect/reduce the number of key dependencies. We also include a proof of concept of how one can incorporate additional constraints into Constraint Programming so that the search for differential characteristics can avoid (to a large extent) differential characteristics that are actually impossible due to the dependency issues our tool detected.

Table 1: The analysis of SKINNY-64 differential characteristics. "Stated prob." refers to the probability given by their respective authors. LC (resp. NLC) refers to the number of (resp. non) linear constraints we identified on the key values. The ones in brackets for LC are the number of higher-order linear constraints detected. "Key space" refers to our estimation of the proportion of the total number of keys for which the differential characteristic can work. "Indep. prob" refers to our estimation of the probability of the differential characteristic, given that a key in the possible key space is used. The "Poss.?" column indicates more visually if a differential characteristic is actually possible and an entry with "Part." means partially which indicates that the differential characteristic validity depends on the key used. (E) indicates that the value has not been deduced theoretically by our tool, but estimated experimentally.

SKINNY	Rds	Stated prob.	No. of LC	No. of NLC	Key space	Indep. prob	Poss.?	Source
64-64	7	2^{-52}	3	0	2^{-6}	2^{-46}	Part.	Table 5 [DDH+20]
	10	2^{-46}	2	0	0	—	No	Table 6 [DDH+20]
64-128	13	2^{-55}	2	0	2^{-4}	2^{-51}	Part.	Table 7 [DDH+20]
	11	2^{-58}	7(+1)	1	$2^{-14.42}$	$2^{-43} - 2^{-44}$(E)	Part.	17-L-I [HBS21]
	12	2^{-69}	9	3	$2^{-11.415}$	$2^{-50.415} - 2^{-53}$	Part.	17-U-I [HBS21]
	12	2^{-64}	7(+1)	1	$2^{-13.95}$	$2^{-49} - 2^{-51}$(E)	Part.	18-L-I [HBS21]
	13	2^{-77}	9	3	$2^{-12.415}$	$2^{-60} - 2^{-61}$	Part.	19-U-I [HBS21]
	11	2^{-61}	8	3	0	—	No	17-L-II [HBS21]
	12	2^{-79}	9	3	0	—	No	17-U-II [HBS21]
	12	2^{-67}	8	3	0	—	No	18-L-II [HBS21]
	13	2^{-87}	10	3	0	—	No	19-U-II [HBS21]
	8	$2^{-17}/2^{-19}$*	1	0	$2^{-2}/2^{-3}$	$2^{-15}/2^{-16}$	Part.	Table 16 [QDW+21]
64-196	15	2^{-54}	2	1	$2^{-6.193}$	$2^{-47} - 2^{-48}$	Part.	Table 8 [DDH+20]
	14	2^{-68}	7(+1)	1	$2^{-15.33}$	$2^{-52} - 2^{-53}$(E)	Part.	22-L-BDM1 [HBS21]
	14	2^{-70}	10	2	0	—	No	22-U-BDM1 [HBS21]
	14	2^{-65}	7(+1)	1	2^{-13}	$2^{-51.415} - 2^{-53}$(E)	Part.	22-L-BDM2 [HBS21]
	14	2^{-70}	9	3	0	—	No	22-U-BDM2 [HBS21]
	14	2^{-69}	9	3	0	—	No	22-U-BDM3 [HBS21]
	15	2^{-79}	11	2	0	—	No	23-U-BDM1 [HBS21]
	15	2^{-79}	10	3	0	—	No	23-U-BDM2 [HBS21]
	10	2^{-20}**	1	0	2^{-1}	2^{-19}	Part.	Table 17 [QDW+21]

* Rounds 0 - 7 of the boomerang distinguisher (** Rounds 0 - 9 of the boomerang distinguisher), where we tested multiple trails following the same valid active Sbox patterns given in the paper. ** Note that the differential characteristics from [HBS21] are the output of their automated tools, and this does not affect the validity of their boomerang distinguishers. The differential characteristics can be found here.

Table 2: The analysis of SKINNY-128 differential characteristics. "Stated prob." refers to the probability given by their respective authors. LC (resp. NLC) refers to the number of (resp. non) linear constraints we identified on the key values. The ones in brackets for LC are the number of higher-order linear constraints detected. "Key space" refers to our estimation of the proportion of the total number of keys for which the differential characteristic can work. "Indep. prob" refers to our estimation of the probability of the differential characteristic, given that a key in the possible key space is used. The "Poss.?" column indicates more visually if a differential characteristic is actually possible and an entry with "Part." means partially (which indicates that the differential characteristic validity depends on the key used). (E) indicates that the value has not been deduced theoretically by our tool, but estimated experimentally.

SKINNY	Rds	Stated prob.	No. of LC	No. of NLC	Key space	Indep. prob	Poss.?	Source
128-128	14	2^{-120}	13	3	$2^{-6.578}$	$2^{-102} - 2^{-97.608}$	Part.	Table 9 [DDH+20]
	13	2^{-123}	21	7	0	—	No	Table 11 [AST+17]
128-256	16	$2^{-127.66}$	13	4	$2^{-5.86}$	$2^{-130.12} - 2^{-117.58}$(E)	Part.	Table 10 [DDH+20]
	12	2^{-97}	10(+1)	3	$2^{-11.11}$	$2^{-83.03} - 2^{-89.81}$(E)	Part.	18-L-I [HBS21]
	12	$2^{-222.075}$	34	0	0	—	No	18-U-I [HBS21]
	13	$2^{-276.415}$	27	1	0	—	No	19-U-I [HBS21]
	12	$2^{-211.075}$	34	0	0	—	No	20-L-I [HBS21]
	14	$2^{-240.905}$	32	2	0	—	No	20-U-I [HBS21]
	13	$2^{-102.415}$	11(+1)	1	$2^{-8.67}$	$2^{-98.91} - 2^{-89.52}$(E)	Part.	21-L-I [HBS21]
	12	2^{-73}	7(+1)	4	$2^{-4.43}$	$2^{-78.67} - 2^{-64.34}$(E)	Part.	18-L-II [HBS21]
	12	$2^{-172.508}$	28	0	0	—	No	18-U-II [HBS21]
	12	2^{-73}	7(+1)	4	$2^{-4.43}$	$2^{-78.67} - 2^{-64.34}$(E)	Part.	19-L-II [HBS21]
	13	$2^{-208.66}$	29	0	0	—	No	19-U-II [HBS21]
	14	$2^{-209.608}$	26	4	0	—	No	20-U-II [HBS21]
	9	2^{-44} - 2^{-53}*	1	0	2^{-5}-2^{-6}	$2^{-48} - 2^{-39}$	Part.	Table 18 [QDW+21]
128-384	14	$2^{-77.415}$	7(+1)	1	$2^{-5.81}$	$2^{-74.83} - 2^{-69.35}$(E)	Part.	22-L-I [HBS21]
	14	2^{-187}	33	1	0	—	No	22-U-I [HBS21]
	9	2^{-10}	0	0	—	—	Yes	23-L-I [HBS21]
	15	$2^{-202.338}$	27	1	0	—	No	23-U-I [HBS21]
	16	$2^{-197.245}$	35	0	0	—	No	24-U-I [HBS21]
	15	$2^{-95.415}$	11(+1)	1	$2^{-7.86}$	$2^{-93.84} - 2^{-85.82}$(E)	Part.	25-L-I [HBS21]
	10	2^{-38} - 2^{-42}**	1	0	2^{-5}-2^{-6}	2^{-36}-2^{-33}	Part.	Table 19 [QDW+21]

* Rounds 0 - 8 of the boomerang distinguisher (** Rounds 0 - 9 of the boomerang distinguisher), where we tested multiple trails following the same valid active Sbox patterns given in the paper. ** Note that the differential characteristics from [HBS21] are the output of their automated tools, and this does not affect the validity of their boomerang distinguishers. The differential characteristics can be found here.

2 Preliminaries

2.1 Basic definitions and notations

Differential Cryptanalysis. Differential cryptanalysis is the study of how differences propagate through a cryptographic function F [BS91]. When used to study modern ciphers, we are bound to expect a complex combination of operations that ensures proper security and it is often difficult to study the differential propagation for the full cipher directly. Fortunately, due to performance reasons, most modern block ciphers are based on the repetition of a single round function (iterated block ciphers), which consequently makes the analysis on a single round translatable. However, to extend the analysis from a round function to multiple rounds, we would require some underlying assumptions that we will discuss below. We denote n the block size of a block cipher.

Definition 1 (1-round differential characteristic [BS91])**.** A 1-round differential characteristic is a pair $(\Delta_{in}, \Delta_{out})$ where Δ_{in} and Δ_{out} represent n-bit differences.

The probability of a 1-round differential characteristic of a vectorial Boolean function,

Table 3: The analysis of GIFT differential characteristics. "Stated prob" refers to the probability given by their respective authors. "Key space" refers to our estimation of the proportion of the total number of keys for which the differential characteristic can work. "Indep. prob" refers to the probability of the differential characteristic, given that a key in the possible key space is used. The "Poss.?" column indicates if a differential characteristic is actually possible and an entry with "Part." means partially which indicates that the differential characteristic depends on the key values.

Cipher	Rds	Stated prob.	No. of Constr.	Key space	Indep. prob	Poss.?	Source
GIFT-64	13	2^{-64}	1	2^{-1}	2^{-63}	Part.	1DT Table 8 [SWW21]
	13	2^{-64}	5	2^{-5}	2^{-59}	Part.	2DT Table 8 [SWW21]
	13	2^{-64}	3	2^{-3}	2^{-61}	Part.	3DT Table 8 [SWW21]
	9	2^{-42}	4	2^{-4}	2^{-38}	Part.	Table 2 [LWZZ19]
	12	2^{-58}	5	2^{-4}	2^{-53}	Part.	Table 7 [LWZZ19]
	13	2^{-62}	6	2^{-4}	2^{-56}	Part.	Table 8 [LWZZ19]
	12	2^{-59}	3	2^{-3}	2^{-56}	Part.	Table 4 [ZDY19]
	12	2^{-60}	0	2^{-0}	2^{-60}	Yes	Table 6 [ZDY19]
GIFT-128	12	2^{-60}	1	2^{-1}	2^{-59}	Part.	Table 5 [LLL+21]
	13	2^{-67}	1	2^{-1}	2^{-66}	Part.	Table 6 [LLL+21]
	21	2^{-126}	7	2^{-5}	2^{-118}	Part.	Table 7 [LLL+21]
	21	2^{-126}	7	2^{-5}	2^{-118}	Part.	Table 4 [LWZZ19]
	18	2^{-109}	—	0	—	No	Table 10 [ZDY19]
	12	2^{-62}	1	2^{-1}	2^{-61}	Part.	Table 15 [ZDY19]
	14	2^{-85}	2	2^{-2}	2^{-83}	Part.	Table 16 [ZDY19]

$F : \mathbb{F}_2^n \to \mathbb{F}_2^n$, can be computed as such:

$$\mathbb{P}(\Delta_{in} \to \Delta_{out}) = \frac{\#\{F(x) \oplus F(x \oplus \Delta_{in}) = \Delta_{out}\}}{2^n}$$

Definition 2 (r-round differential characteristic [BS91])**.** An r-round differential characteristic is a tuple $\Delta = (\Delta_{in} = \Delta_0, \Delta_1, \Delta_2, ..., \Delta_r = \Delta_{out})$ where Δ_i are n-bit differences after the i^{th} round of the function $\forall\ i \in \{0 \ ... \ r\}$.

The probability of an r-round differential characteristic is then computed as the product of all the probabilities of the r 1-round differential characteristics:

$$\mathbb{P}(\Delta_{in} \to \Delta_1 \to ... \to \Delta_{out}) = \mathbb{P}(\Delta_0, \Delta_1) \cdot \mathbb{P}(\Delta_1, \Delta_2) \cdot ... \cdot \mathbb{P}(\Delta_{r-1}, \Delta_r) \tag{1}$$

Note that this probability computation assumes that the differential propagation throughout the rounds can be evaluated independently. In the case where the assumption is valid, we can say that the underlying cipher is a Markov cipher [LMM91] (see Definition 3 and Theorem 1). In modern day cryptanalysis, researchers usually assumed analyzed primitives to be Markov ciphers. Again, this allows for the decomposition of the cipher into multiple independent components, which facilitates the analysis.

Definition 3 (Markov cipher [LMM91])**.** An iterated cipher with round function $Y = f(X, k)$ is a Markov cipher if there is a group operation $\otimes$ for defining differences such that, for all choices of $\Delta_{in} \neq 0$ and $\Delta_{out} \neq 0$,

$$\mathbb{P}(\Delta Y = \Delta_{out} | \Delta X = \Delta_{in}, X = \gamma)$$

is independent of γ when the subkey k is uniformly random.

Theorem 1 ([LMM91]). *If an r-round iterated cipher is a Markov cipher and the r round keys are independent and uniformly random, then the sequence of differences $\Delta X = \Delta Y(0), \Delta Y(1), ..., \Delta Y(r)$ is a homogeneous Markov chain. Moreover, this Markov chain is stationary if ΔX is uniformly distributed over the non-neutral elements of the group.*

Hypothesis of stochastic equivalence [LMM91]. For an r-round differential characteristic $(\Delta_0 = \alpha_0 \to \Delta_1 = \alpha_1 \to ... \to \Delta_r = \alpha_r)$,

$$\mathbb{P}(\Delta_r = \alpha_r \mid \Delta_0 = \alpha_0) \approx \mathbb{P}(\Delta_r = \alpha_r \mid \Delta_0 = \alpha_0, k^1 = \omega_1, ..., k^r = \omega_r)$$

for almost all subkey values $(\omega_1, ..., \omega_r)$ where k^i refers to the subkey on round i.

The hypothesis of stochastic equivalence is another commonly cited assumption when analyzing a cipher. Without it, cryptanalysts would have to analyze the cipher for each possible key. However, the hypothesis can only give us a rough estimation of what are the differential properties when we average over all keys.

Differential Distribution Table. The Difference Distribution Table (DDT) is a tool to represent the distribution of every input difference/output difference pair for a function. For example, the DDT of the 4-bit S-box used in SKINNY-64 is shown in Table 8 in the Appendix A.

$\mathcal{X}_{DDT}$ and $\mathcal{Y}_{DDT}$. While the DDT contains the number of right pairs for each difference transition of a function F, the actual values of the pair that allow the difference to pass is not captured and shown. Thus, to overcome this lack of information, we define the following sets:

$$\mathcal{X}_{DDT}(\Delta_{in}, \Delta_{out}) \coloneqq \{a : a \oplus b = \Delta_{in}, F(a) \oplus F(b) = \Delta_{out}, b \in \mathbb{F}_2^n\}$$
$$\mathcal{Y}_{DDT}(\Delta_{in}, \Delta_{out}) \coloneqq \{F(a) : a \oplus b = \Delta_{in}, F(a) \oplus F(b) = \Delta_{out}, b \in \mathbb{F}_2^n\}$$

$\mathcal{X}_{DDT}$ is basically the set of all the input values that allow the required differential transition $\Delta_{in} \to \Delta_{out}$ to pass, while $\mathcal{Y}_{DDT}$ does the same for the output values.

Notations. We will overload the operator $\oplus$: other than the usual notation in cryptography to represent the XOR of two elements in $\mathbb{F}_2^n$, we will use it on sets as well. Let $A, B \subseteq \mathbb{F}_2^n$, then $A \oplus B \coloneqq \{a \oplus b \,|\, a \in A, b \in B\}$. Similarly, for $c \in \mathbb{F}_2^n$ and $D \subseteq \mathbb{F}_2^n$, we have $c \oplus D \coloneqq \{c \oplus d \,|\, d \in D\}$. Note that $c \oplus D$ can also be viewed as a coset of D and $|c \oplus D| = |D| \; \forall \; c \in \mathbb{F}_2^n$.

2.2 The SKINNY family of tweakable block ciphers [BJK+16]

The SKINNY family of tweakable block ciphers was first proposed in [BJK+16]. It follows the TWEAKEY framework [JNP14] which takes in a tweakey input instead of just a key. There are two proposed block sizes, $n = 64, 128$. The state can be visualized as a 4×4 array. When $n = 64$, each cell is a nibble and when $n = 128$, each cell is a byte. The tweakey has three different sizes, $t = n, 2n, 3n$ (TK1, TK2, TK3) for both values of n. Note that each TKi for $i = 1, 2, 3$, can also be viewed as a 4×4 array. The round key is computed by XOR-ing cell-wise all the TKi arrays. In this paper, we will refer the state/key cell at r^{th} row and c^{th} column as the (r, c) state/key cell or simply the $(4r + c)$ cell when only a single number (instead of a tuple) is used. For a more generic discussion in the framework in Section 4, we will use the term component instead of cell.

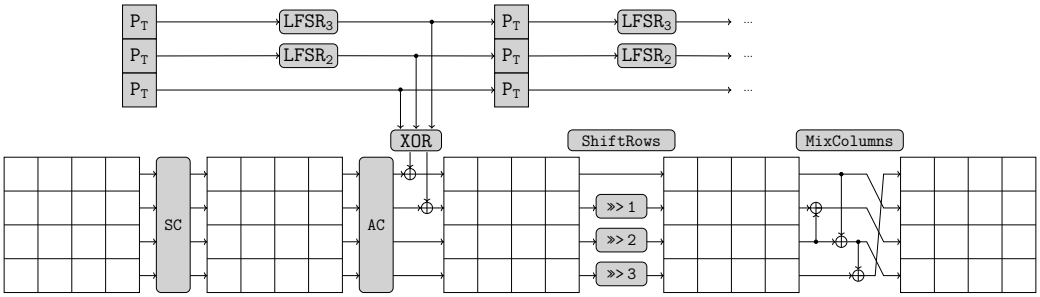


Figure 1: A high level schematic of SKINNY-n-$3n$ round function with the key schedule

The round function of SKINNY has five sub-functions: SubCells (SC), AddConstants (AC), AddRoundTweakey (ART), ShiftRows (SR) and MixColumns (MC). Unlike AES which uses a Maximum Distance Separable (MDS) matrix in the MixColumns function, the SKINNY designers decided to take a more lightweight approach: with only a total number of 12 XOR operations on the cell level for the diffusion. Despite only having eight unique key cells involved in an ART function, at the end of a single round of SKINNY (after the MC function), every state cell will be XORed with one and only one cell of the round key.

The tweakey schedule (which we will refer to as key schedule for simplicity) of SKINNY is a rather simple one. For each TK array, a permutation P_T is used to swap the positions of each cell. Note that all TKi arrays share the same permutation, so the cells always remain aligned. The following illustrates P_T:

$$(0, ..., 15) \rightarrow (9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7)$$

Next, every cell of the first two rows of TK2 and TK3 is also updated with their respective LFSRs (LFSR_2 and LFSR_3). Note that the entire key schedule is linear. The round transformation and key schedule are also illustrated in Figure 1. We refer to [BJK+16] for more information about the specifications of SKINNY.

3 Related Works

As mentioned in [LMM91], the hypothesis of stochastic equivalence has been implicitly assumed in differential cryptanalysis. When searching for differential characteristics, most cryptanalysts tend to acknowledge that they are working under the assumption that the probabilities can be evaluated independently. However, to what extent does this hypothesis actually hold? [KM04] had tested a cipher with several simple and complicated key schedules and they found that probabilities of the best differentials on ciphers with simple key schedules tend to not converge towards the uniform distribution as the number of rounds increases. Also, as pointed out by the designers of Rijndael in [DR20], block ciphers may contain weak keys which cause the probabilities of some differential characteristics to be very high and the rest to be very low. In [DR07], they introduced the concept of plateau characteristics where the probability depends on the key values and could only hold two values: either 0 and a fixed probability $p \neq 0$. They showed that for AES, as well as for many other block ciphers, all two-round differential characteristics are plateau characteristics. Many similar efforts proceeded from this work to get a more accurate differential probability. For example, the LED cipher [GPPR11] has a structure very similar to that of AES, but only applies a subkey every four AES-like rounds (equivalent to one `STEP` function). In this case, consecutive AES-like rounds within a `STEP` can certainly not be treated as independent. Thus, the differential cryptanalysis of the LED cipher had

to focus on the STEP function instead [MRTV12, NWW15]. From the perspective of a 4-round AES-like round function (1-round STEP function), fixed keys (round constants) are taken into account in forming the characteristic. [SWW18] used SAT to run through all possible right pairs for LED-64: by taking the key schedule into consideration, they gave an estimation on the upper bound for the weak-key ratio. A similar technique can be seen in [LZS+20] and [CLN+17] where the authors have also warned on the validity of some differential characteristics (with regards to the valid keys and exact probabilities).

We emphasize that in all these works the authors rely on the required output values of the nonlinear functions in round r (i.e. $\mathcal{Y}_{DDT}$) and the required input values of the nonlinear functions in round $r+1$ (i.e. $\mathcal{X}_{DDT}$) to form a linear subset of values required by the round key bits that are involved in between. In our work, this is known as the linear constraints as they do not exactly involve an Sbox (more details are given in Section 4.3). In our work, we explore a more general framework to include a larger set of dependencies: these include not only linear constraints, but also nonlinear ones. With all these constraints, we can obtain a better estimation of the probability distribution of a given differential characteristic.

This problem is not unique to just SPN ciphers. In [SRB21], the authors have identified issues on previously reported differential characteristics for SPECK and SIMECK [LWRA17]. In [Leu12], Leurent introduced his toolkit, `arxtools`, which uses multi-bit constraints to describe differential characteristics in the context of ARX ciphers and which helps to detect the validity of differential characteristics. Recently, [XLJ+22] remarked that many differential characteristics for ARX-ciphers may actually have to undergo modular additions more than once without any key input and this might lead to impossibilities. They provided an example showing that the XOR-ing of a branch with a (round) constant does not necessarily guarantee the output distribution of the signed differences to be uniformly distributed. While hash functions do not have keys, messages can be crafted in such a way that they fulfill the conditions of a chosen differential characteristic, so that the probability of finding a valid pair increases [WLF+05]. In the case of SHA-2, [MNS11] searches for conforming pairs alongside the differential characteristic.

Despite these results in the literature, papers that focused on automated methods to search for differential characteristics usually do not provide any experimental verification (like finding a right pair for the characteristic) to argue about their validity. Cryptanalysts tend to assume that the differential characteristic will work as expected, and thus usually do not perform any form of verification. Of course, to a large extent, another obvious reason is that most of the differential characteristics have very low probabilities and thus are impractical to verify experimentally. At FSE 2021, the authors of [DDV20], when searching for differential characteristics for boomerang attacks, noted that the rounds may not be independent of each other, but still followed the convention and computed the probability using Equation 1.

4 A Framework for Detecting and Analyzing Constraints

4.1 The issue with current differential characteristics search methods

Consider a generic round function that we can further decompose into a nonlinear part S and a linear part L (note that L also consists of the key addition layer too). A block cipher can then be built by concatenating round functions. When searching for differential characteristics, cryptanalysts often consider only the difference value in the internal state and not the actual value. While this will not pose a problem for the linear part (since the composition of linear functions remains linear), nonlinear functions may be a little trickier. When dealing with an Sbox-based nonlinear layer, cryptanalysts usually turn to the DDT to estimate the probability of a propagation from a particular input difference

to a particular output difference through the Sbox. When we have an active Sbox where the input and output differences are fixed to some specific non-zero differences, the set of input and output values that satisfy the differential transition is also reduced to a subset of $\mathbb{F}_2^n$ (see Figure 2).

Key distribution. While the selection of a key for an instance of a cipher may be uniformly random, we should not forget that in the setting of an actual attack, the key remains fixed, although at some unknown value. Thus, it does not necessarily contribute as a source of randomness to the state. What it does is merely transform the state values from one coset to another coset (not necessarily different).

Diffusion layer. The strength of the diffusion varies a lot from cipher to cipher. For SPN primitives, on the one hand, we have ciphers like AES which use an MDS matrix to obtain a strong diffusion at each round. On the other hand, we have ciphers like GIFT or SKINNY which opted to have a weaker diffusion layer, in exchange for a larger number of rounds. For example, in SKINNY ciphers, the diffusion in each round requires only 12 XORs (working on either 4 or 8 bits), with four cells of the state not being updated by any XOR at all. This actually brings the two nonlinear functions (the Sboxes) from two consecutive rounds closer together. If insufficient randomness is added to the state before the second Sbox, then it is highly likely that the key will dictate if the expected differential transition through the second Sbox is going to succeed or not.

Inactive bits in active Sboxes. In ciphers where the linear layer permutes at bit-level, such as PRESENT and GIFT, differential characteristics are traced only based on where the active bits are traveling, instead of where the active Sboxes are. When we require the Sboxes to fulfill certain differential transitions, there may be some constraints on the "inactive bits" of the active Sboxes that are ignored by the cryptanalyst. We will discuss this in Section 5.2 when we will review in more detail the case of GIFT cipher.

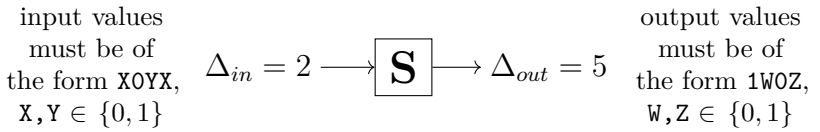


Figure 2: An illustration of how the input and output values (in binary representation) are restricted through a nonlinear layer, using SKINNY-64 4-bit Sbox as an example.

4.2 General framework for detecting incompatibilities

Detecting constraints. As mentioned previously, when an Sbox is active, the sets of input and output values are restricted to some particular subsets. As a result, restrictions can be found on the inputs and the outputs of the nonlinear functions. We call these *half constraints* as they require another half constraint to form a full constraint on the key. A full constraint on a subkey at round r must consist of two half constraints, one from just before r^{th} round key addition and another from just after the key addition (see Figure 3). Suppose we have half constraints from just before round r. We are interested to know if they can form a full constraint on the round key at round r. Let x_i^r, y_i^r be the i^{th} component of the state at round r before and after the function S respectively. Since each component x_i^{r+1} is a linear function of all the components from the output of the S function in round r and the round keys, i.e. $x_i^{r+1} = L(y_{i_0}^r, y_{i_1}^r, ..., y_{i_m}^r, k_{i_0}^r, ..., k_{i_m}^r)$, there are exactly two possible outcomes:

1. There exists a $y_{i_j}^r$ that is uniformly distributed (i.e. a component without half constraints) that is XORed to the rest of the components i.e.,

$$x_i^{r+1} = L(y_{i_0}^r, ..., y_{i_{j-1}}^r, y_{i_{j+1}}^r, ..., y_{i_m}^r, k_{i_0}^r, ..., k_{i_m}^r) \oplus c \cdot y_{i_j}^r$$

   where $c \neq 0$ is an arbitrary constant. In this case, x_i^{r+1} will not have any half constraints from round r.

2. All the input components of L have half constraints. In this case, a full constraint will be formed.

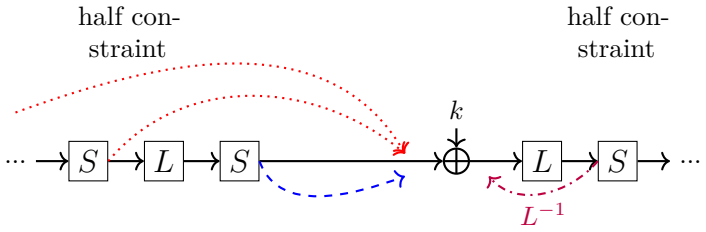


Figure 3: An illustration of how a constraint is formed from using two half constraints. A combination of a red and purple half-constraint forms a nonlinear constraint and a combination of the blue and the purple half-constraint forms a linear constraint.

The first case is not really interesting, as we can just assume that x_i^{r+1} is uniformly distributed with respect to the differential characteristic since it can take up any possible value. Thus, we will focus on the second case. By having all the components restricted to a smaller subset, there is a chance that x_i^{r+1} is being restricted after the linear layer. This will depend on the strength of the diffusion. After propagating through L, if the set of values that x_i^{r+1} can take up is the entire space despite having half constraints on $y_{i_0}^r, ..., y_{i_m}^r$, then we will have the same conclusion as the first case. Now, suppose that x_i^{r+1} is also active (i.e. $x_i^{r+1}, y_{i_0}^r, ..., y_{i_m}^r$ are all active), then we will obtain a full constraint on $k_{i_0}^r, ..., k_{i_m}^r$. On the contrary, suppose that x_i^{r+1} is not active, we will have to propagate the constraints further down the rounds, until one of the scenarios occurs in a latter round: it propagates to an active Sbox and forms a constraint, goes into the first case, or we have reached the end of the differential characteristic.

Combining constraints to evaluate the size of the valid key space. After we have obtained the constraints on each round key, we can then use the key schedule to propagate the round key constraints to the same subkey and check the dimension of the master key space that allows the differential characteristic to pass.

Impact on the probability. Other than affecting the number of keys that would allow a differential characteristic to pass, this analysis has more generally an impact on the probability of the differential characteristic. We exhibit the two extreme cases below. Suppose we have two constraints C_0, C_1 acting on the same key component k (i.e. $C_0 \implies k \in A \subseteq \mathbb{F}_2^n$ and $C_1 \implies k \in B \subseteq \mathbb{F}_2^n$) then,

1. if $A \cap B = \emptyset$, then there is no key value that allows the differential characteristic to pass (i.e. the differential characteristic is impossible)

2. if $A \cap B = A = B$, then C_0 is satisfied if and only if C_1 is satisfied. This means that either C_0 or C_1 is redundant.

When there is only a subset of the keys that are possible for the differential characteristics, the best way to present the probability of a characteristic would be to break it down into two parts: the dimension of the right key space and the remaining probability of the characteristic that is independent of the key (i.e. the probability of characteristic given that a key in the right key space is used). Note that the dimension only takes into account linear constraints and not nonlinear constraints (as we will explain in Section 4.3), but it represents a better approximation than what is currently conventionally presented.

4.3 Example: SKINNY ciphers

In this section, we give an example of how we can apply the framework onto SKINNY. As an illustration, we choose a differential characteristic from [HBS21] as it is rather dense and it covers various types of constraints within a short number of rounds. In particular, we are looking at the first four rounds of the lower differential characteristic in "Boomerang Distinguisher II for SKINNY-64-128". The shortened differential characteristic can be found in Figure 4. Note that we only illustrate some of the constraints found in this example. Also, we have to clarify that this does not invalidate their work (the differential characteristics are obtained via automated tools which is an intermediate product) and they do not just rely on differential characteristics in their work.

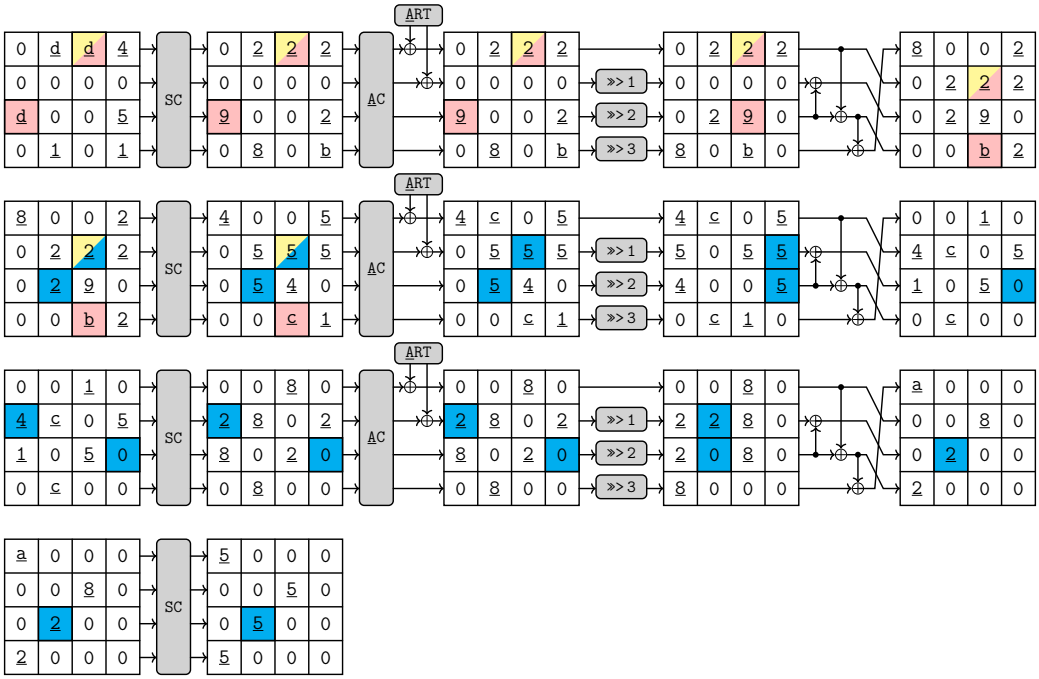


Figure 4: The first four rounds of the differential characteristic in "Boomerang Distinguisher II for SKINNY-64-128 from [HBS21]. Each line represents a single round of the SKINNY round function. The cells highlighted in yellow and pink form linear constraints on the round key and the cells highlighted in blue form a nonlinear constraint. All the active cells are underlined.

Some constraints are more complicated than others, as Sbox applications are required to express the constraint, in other words such constraints are nonlinear. We will therefore split them into two different groups and describe them separately. We define a constraint to be nonlinear if its expression includes at least an (inactive) Sbox. A constraint is therefore linear if the expression does not include any Sboxes.

Linear constraints. Linear constraints usually involve the input values of active Sboxes in a particular round and the input values of Sboxes in the preceding round, but not the Sboxes themselves. We differentiate them into two different subcategories of constraints. The first type is a simple linear constraint where only one Sbox is involved. The second type is a higher-order linear constraint where more than one output Sbox is involved. Simple linear constraints are well analyzed in the literature [DR07,MRTV12,NWW15,SWW18,CLN+17]. In Figure 4, we traced one of the simple linear constraints in pink. The constraints are formed because of the output and input restrictions of the Sboxes from the first and second rounds respectively. We can represent the constraint as:

$$u \oplus v \oplus \texttt{0x2} \oplus k^0_{0,2} = w$$

with $u \in \mathcal{Y}_{DDT}(\texttt{0xd}, \texttt{0x2}), v \in \mathcal{Y}_{DDT}(\texttt{0xd}, \texttt{0x9}), w \in \mathcal{X}_{DDT}(\texttt{0xb}, \texttt{0xc})$ and $k^0_{0,2}$ being the key nibble at the $(0, 2)$ position of the first round. Note that `0x2` is the round constant involved (in hexadecimal display). Solving the equation for the range of possible values of $k^0_{0,2}$, we obtain the following constraint:

$$k^0_{0,2} \in \{\texttt{0x0}, \texttt{0x1}, \texttt{0x2}, \texttt{0x3}, \texttt{0x8}, \texttt{0x9}, \texttt{0xa}, \texttt{0xb}\}$$

which shows that this differential characteristic depends on the value of $k^0_{0,2}$. When there is more than one constraint on the same key nibble, it may affect the probability of the differential characteristic. Following the constraint discussed above, we have another constraint also acting on the same key nibble and we highlighted it in yellow in Figure 4. This constraint requires

$$k^0_{0,2} \in \{\texttt{0x4}, \texttt{0x5}, \texttt{0x6}, \texttt{0x7}, \texttt{0xc}, \texttt{0xd}, \texttt{0xe}, \texttt{0xf}\}$$

Since the two constraints are mutually exclusive, the entire differential characteristic is actually impossible.

Higher-order linear constraints. Higher-order linear constraints combine more than one Sboxes at the output level. Depending on how we combine them, they may involve no key input, one key input, or two key inputs. These constraints are dependent on the XORs from the MixColumns function. To illustrate it, we restate the MC function as cells equations here (x_i and y_i represent the input and output cells respectively for one column of the AddRoundTweakey/MixColumns combined function, with k_0 and k_1 representing the two key cells involved):

$$\begin{aligned} y_0 &= x_0 \oplus x_2 \oplus x_3 \oplus k_0 \\ y_1 &= x_0 \oplus k_0 \\ y_2 &= x_1 \oplus x_2 \oplus k_1 \\ y_3 &= x_0 \oplus x_2 \oplus k_0 \end{aligned}$$

There are multiple equivalent constraints we can derive from here. For example, we can obtain

$$\begin{aligned} y_1 \oplus y_3 &= (x_0 \oplus k_0) \oplus (x_0 \oplus x_2 \oplus k_0) \\ &= x_2 \end{aligned} \tag{2}$$

In total, by combining, we can obtain a total of 11 equations, with 3 equations being independent of any keys, 4 equations depending on a single key input, and 4 equations depending on both key inputs. As an example, we can refer once again to column 2 of the round 0 given in Figure 4. Using Equation (2), we can obtain a constraint shown in Figure 5. With that, we can then compute the possible values for each of the cells involved:

$$\left\{ s^0_{2,0} = s^1_{1,2} \oplus s^1_{3,2} \;\middle|\; \begin{array}{l} s^0_{2,0} \in \{\texttt{1,3,8,a}\} \\ s^1_{1,2} \in \{\texttt{0,2,9,b}\} \\ s^1_{3,2} \in \{\texttt{6,d}\} \end{array} \right\}$$

By simply enumerating, we observe that this equation does not hold. Thus, this is the second impossibility constraint we have detected.

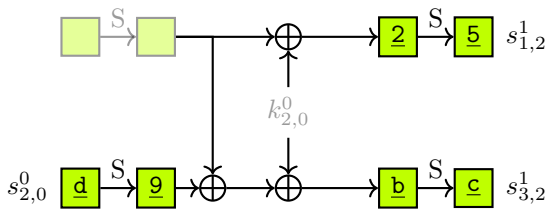


Figure 5: An illustration of a higher-order linear constraint for SKINNY-64-128. $k^n_{r,c}$ represents the (r, c) key cell at round n.

Nonlinear constraints. Generally, we classify nonlinear constraints as constraints that involve keys from multiple rounds, which means that Sboxes are required to represent the constraint. One example is highlighted in blue in Figure 4. The intermediate part of these constraints involves inactive Sboxes. This is an example of how inactive Sboxes can be part of a constraint as we are concerned about the set of values it can take up, instead of whether there is a difference or not. A more succinct illustration of the constraint is shown in Figure 6.

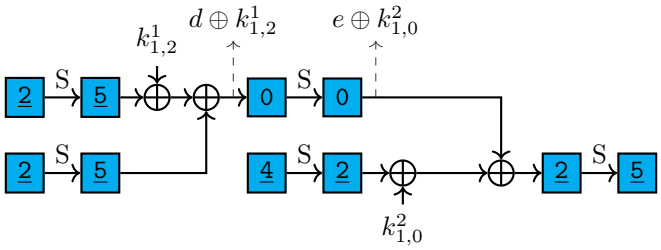


Figure 6: An illustration of a nonlinear constraint for SKINNY-64-128. $k^n_{r,c}$ represents the (r, c) key cell at round n. $d \oplus k^1_{1,2}$ and $e \oplus k^2_{1,0}$ represent the possible input and output values of the inactive Sbox respectively. The active cells are underlined.

Simplifying the constraint, we get:

$$S(k^1_{1,2} \oplus d) = k^2_{1,0} \oplus e$$

where $d \in \mathcal{Y}_{DDT}(2, 5) \oplus \mathcal{Y}_{DDT}(2, 5) = \{\texttt{0x0}, \texttt{0x1}, \texttt{0x4}, \texttt{0x5}\}$ and
$e \in \mathcal{Y}_{DDT}(4, 2) \oplus \mathcal{X}_{DDT}(2, 5) = \{\texttt{0x4}, \texttt{0x5}, \texttt{0x6}, \texttt{0x7}, \texttt{0xc}, \texttt{0xd}, \texttt{0xe}, \texttt{0xf}\}$.

The impact of nonlinear constraints. Linear constraints are easy to analyze and have been explored multiple times by previous works. However, nonlinear constraints are harder to detect, as they rely very much on how strong the diffusion layer in a round function of a cipher is. Linear constraints have also an almost-uniform distribution across all valid keys (plateau characteristics). In this case, all linear constraints of SKINNY-64 will lead to plateau characteristics, but not SKINNY-128. For nonlinear constraints, this may vary:

they may or may not lead to a plateau characteristic. For instance, in the example given in Figure 6, there are three possible probabilities: 0, 2^{-1} and 2^{-2} depending on the values of $k^1_{1,2}$ and $k^2_{1,0}$.

To be more concrete, below we describe a differential characteristic that is not a plateau characteristic and with a more complex linear and nonlinear constraints setting. The differential characteristic can be found in Table 4 with two nonlinear constraints and four linear constraints. We computed the theoretical differential distribution and it is summarized in Table 5. To verify this, we have also experimented with 2048 different valid keys, and the results are shown in Figure 7. We can observe that the probability distribution is as expected a graph that has a Gaussian distribution around the points given in Table 5.

Table 4: A 5-round differential characteristic of SKINNY-64-128 that is not a plateau characteristic. The first, second, third, and fourth row in each cell of the table represent the difference before SB, the difference after SB and the round key (TK1 and TK2) involved respectively in hexadecimal notation.

Round	Differential transition
0	2,0,0,2,0,2,0,0,0,0,0,0,2,2,8,0 6,0,0,1,0,1,0,0,0,0,0,0,1,1,4,0 8,0,0,0,f,0,4,0 e,0,0,0,f,0,1,0
1	1,4,0,0,0,0,0,1,0,0,1,5,0,0,0,1 b,2,0,0,0,0,0,b,0,0,b,2,0,0,0,8 0,0,0,0,0,0,0,0 0,0,0,0,0,0,0,0
2	0,0,8,0,b,2,0,0,0,2,0,0,0,0,0,0 0,0,4,0,1,6,0,0,0,1,0,0,0,0,0,0 0,0,8,0,0,4,f,0 0,0,c,0,0,2,e,0
3	0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1 0,0,0,8,0,0,0,0,0,8,0,0,0,0,0,b 0,0,0,0,0,0,0,0 0,0,0,0,0,0,0,0
4	0,0,b,0,0,0,0,8,0,0,0,8,0,0,0,0 0,0,1,0,0,0,0,4,0,0,0,4,0,0,0,0 0,0,0,4,8,f,0,0 8,0,0,4,8,c,4,0

Table 5: Predicted probability distribution across all valid keys for the differential characteristic in Table 4.

Prob. in $-\log_2$	35.415	36.415	37	37.415	38	39
Percentage	5.56%	22.2%	5.56%	22.2%	22.2%	22.2%

The algorithm that searches for constraints on the round keys can be found in Algorithm 1. We summarize below its main steps:

1. We initialize a 4×4 zero matrix M^r_{res} for each round r, where $M^r_{res}[i][j] = 1$ indicates that the possible values in the corresponding state cell are restricted to a subset, and 0 otherwise.
2. For the Sbox layer, if the (i, j) state cell is active, set $M^r_{res}[i][j] = 1$
3. Apply SR to M^r_{res}
4. For MC, for each output cell, if all of the relevant input cells have $M^r_{res}[i][j] = 1$, assign $M^{r+1}_{res}[i][j] = 1$, otherwise, set $M^{r+1}_{res}[i][j] = 0$.
5. If $M^{r+1}_{res}[i][j] = 1$ and the corresponding state cell is active, we have obtained a constraint.

The idea of Algorithm 1 is rather intuitive: it evaluates round by round, in a top-bottom strategy. It keeps track of which state cells have some limitations on the values they can hold if they were to follow a differential characteristic. This usually occurs right after an

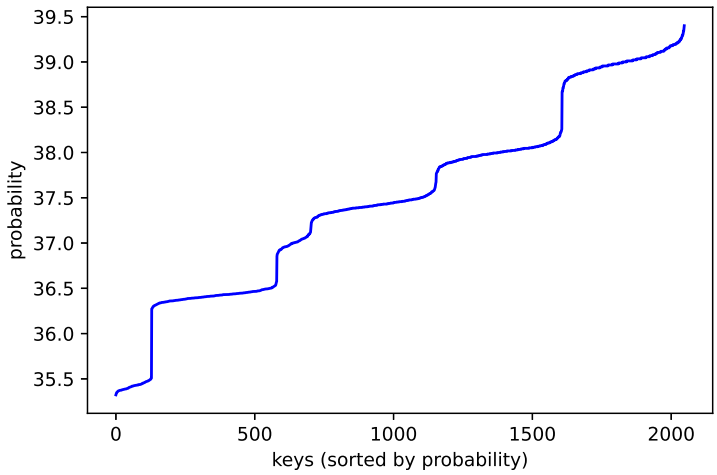


Figure 7: The measured probability distribution (in $-\log_2$ notation) of the differential characteristic across 2048 random but valid keys in Table 4.

Sbox application. If the diffusion did not aid it to become uniform again (by means of XOR-ing with other uniformly distributed cells), then this restriction will be passed to the next round. In the next round, if once again there are some requirements on the cell to be limited to a certain range of values, we will then be looking at some potential constraints.

Combining constraints from different rounds. We have identified the round key constraints using the SKINNY round function, with the constraints involving their respective subkeys. Yet, how do we know if the constraints at round r_1 are going to influence those in round r_2? In the case of SKINNY, the key schedule is rather simple. We can apply its permutation to see if the constraints at different rounds are acting on the same key cell. Suppose now that we have two constraints acting on the same key cell, the value we have restricted is the XOR of all the TKs (if any). Thus, we have to take care of the various TK versions differently. In the case of SK/TK1, we do not have to apply the LFSR at all and we can directly compare the values in a straightforward manner. On the other hand, for TK2 and TK3 cases, we have to consider that the key values we have restricted above are only the XOR of two or three key cells. We can also make use of the fact that the LFSRs used in the key schedule of SKINNY have a cycle of 15 (since only half of the state is XORed with key material in SKINNY, the cycle will appear after 30 rounds). Hypothetically, consider a key nibble, k, that can only take up the value `0x3` as restricted by some constraints. In a TK2 context, we actually have 16 possibilities of (TK1,TK2) combinations: $(x, x \oplus \texttt{0x3})\ \forall x \in \mathbb{F}_2^4$.

Since the LFSR is a linear permutation, we can guarantee that exactly one of every non-zero difference will appear when we apply the LFSR, as long as we do not exceed its period (as identity mapping will map back to the same difference). As we are dealing with characteristics with a number of rounds lesser than 30, this is sufficient for our discussion. This means that actually we can compute the number of possible keys easily: let V_0 and V_1 be the sets of possible values that the XORed key can take up for k_0 and k_1 respectively, where k_0 and k_1 are from different rounds, but acting on the same key cell when we apply the key schedule. Then, the number of possible keys is $|V_0| \times |V_1|$ for the case of TK2, and $|V_0| \times |V_1| \times \mathbb{F}_2^n$ for the case of TK3. In the case that the number of constraints acting on the cell exceeds the TK size, then the existence of at least a valid key for that cell cannot be guaranteed (which means that there is a chance for the differential characteristic to be impossible).

Impact on differential characteristics. By considering both linear and nonlinear constraints, we can measure the dependencies quite accurately. Since differential characteristics are usually sparse in nature, these constraints can usually be split into several subgraphs, which sometimes makes the measurement of the distribution computationally feasible when the entire characteristic has a very low probability. For most of the differential characteristics in SKINNY-64, we are able to compute them purely by counting. In the case of SKINNY-128, we can rely on the subgraphs and compute the probability distribution experimentally.

Impact on differentials. While differential characteristics might only work for a very small subset of keys, providing an analysis on merely the dominant characteristic of a differential may not necessarily give a good estimation of the impact on a key recovery attack. Consider a differential D consisting of z differential characteristics, $d_0, d_1, ..., d_{z-1}$, each with their respective probability distribution $P_0, P_1, ..., P_{z-1}$. Then, given a particular key k the differential D usually only activates a subset of $\{d_0, d_i, ..., d_{z-1}\}$ and thus the probability will be a sum of a (potentially empty) subset of these probabilities under the key k, instead of merely the sum of all the probabilities of all the differential characteristics. As much as possible, we would like to verify the impacts our constraints have on differentials. For this, we experimented with two SKINNY-64 differential characteristics that were not possible (due to key constraints). The first one is a related key differential characteristic (TK1) from [DDH+20]. There are two linear constraints in round (4-5) and round (8-9) which have constraints on the same key cell **after taking the key schedule into consideration**. These constraints caused the differential characteristic to fail under all keys. On the other hand, we would like to test the capabilities of the corresponding differential. However, due to the high complexity of the characteristic, we cannot test it entirely, thus, we tested from round 4 to round 9. We ran an experiment to obtain the probability of the differential with 1040 random keys, with 2^{34} plaintext pairs per key. Out of the 1040 keys, we are still unable to find any right pairs for 782 of the keys. For the remaining keys, the probability averaged at approximately $2^{-23.65}$ for a right pair. The other differential characteristic is from the upper differential characteristic from "Boomerang Distinguisher II for SKINNY-64-128". There are a couple of constraints, but most importantly, there were two linear constraints on round 10-11. In this case, **we do not need to take the key schedule into consideration**. We ran the experiment to find the probability for the differential from round 7 to round 10 (along with the two active Sboxes in round 11 that are involved in the constraints). Again, we ran the experiment for 1040 random keys, with 2^{32} plaintext pairs per key. In this case, none of the keys are possible. From the examples above, we can see that an invalid differential characteristic potentially limits the pool of valid keys for the differential.

5 Applications to SKINNY and GIFT

5.1 Application to SKINNY [BJK+16]

There have been quite a lot of works searching for differential characteristics of SKINNY. While some focused on finding differential characteristics with the best probability for a certain number of rounds, others look for some differential characteristics to maximize the probability for boomerang and rectangle distinguishers. However, based on what we have searched, none of them really address the problem of key dependencies. To check if the differential characteristic is possible (i.e. there exists at least a key that allows this differential characteristic to pass), we can use a quick check using a SAT solver such as cryptominisat [SNC09]. To find the distribution according to the keys, we can split the constraints (linear and nonlinear) into mutually exclusive sets: while the linear

constraints involve just a single key cell, nonlinear constraints involve more than just one, thus complicating this process. Then, as these sets of constraints are independent, the probability distribution would be fairly easy to compute. We acknowledge that for a really dense differential characteristic, it is really hard to split all of the constraints into independent spheres. In these cases, we have to make some compromises in terms of independency analysis. We also note that it might happen for SKINNY-128 that some of the nonlinear constraints require a high memory and time complexity to compute exactly for every key, in which case we proceeded to conduct a small experiment to compute the probability distribution. As these nonlinear constraints represent just a small part compared to the entire characteristics, they can be computed within a very reasonable time.

In summary, out of 21 tested differential characteristics for SKINNY-64, 10 differential characteristics are found to be impossible and among the remaining ones, 10 differential characteristics only work for $\leq 25\%$ of the keys. For SKINNY-128, we have tested 22 characteristics: we found that 11 of them are impossible and among the remaining ones 10 of them work for $\leq 25\%$ of the keys. A breakdown is given in Table 1 and Table 2. For those that are possible, we have obtained the estimated theoretical probability distribution according to different keys, while some were experimentally determined (labeled with (E) in Table 2). This is shown in Table 6 and Table 7 for SKINNY-64 and SKINNY-128 respectively.

Table 6: The probability distribution estimation for each differential characteristic from Table 1. "Stated prob." refers to the $-\log_2$ probability given by their respective authors. The first row in each probability distribution refers to the probability (in $-\log_2$) with the percentage of keys having this probability shown below them.

SKINNY	Source	Stated prob.	Probability Distribution				
64-64	Table 5 [DDH+20]	52	46 100%				
64-128	Table 7 [DDH+20]	55	51 100%				
	17-L-I [HBS21](E)	58	43 31.0%	44 69.0%			
	17-U-I [HBS21]	69	60.415 8.33%	61 16.67%	61.415 16.67%	62 41.67%	63 16.67%
	18-L-I [HBS21](E)	64	51 13.8%	51.415 30.2%	52 28.9%	53 27.2%	
	19-U-I [HBS21]	77	70 33.3%	71 66.67%			
	Table 16 [QDW+21]	17-19	16 100%				
64-192	Table 8 [DDH+20]	54	47 14.29%	48 85.71%			
	22-L-BDM1 [HBS21](E)	68	52 13.8%	53 86.2%			
	22-L-BDM2 [HBS21](E)	65	51.415 47.8%	53 52.9%			
	Table 17 [QDW+21]	20	19 100%				

5.2 Application to GIFT [BPP+17]

Brief description of GIFT. The GIFT family of block ciphers was first proposed in CHES 2017 by [BPP+17] and inspired by PRESENT [BKL+07]. It is a Substitution-bitPermutation Network (SbPN): instead of the permutation layer working at the word level, it comprises of a bit-permutation (see Definition 1 from [BPP+17]). There are two variants in this family: GIFT-64 and GIFT-128, having state sizes of 64 and 128 bits respectively. There

Table 7: The probability distribution estimation for each differential characteristic from Table 2. "Stated prob." refers to the $-\log_2$ probability given by their respective authors. The first row in each probability distribution refers to the probability (in $-\log_2$) with the percentage of keys having this probability shown below them. 'X' indicates a range from 0 to 9.

SKINNY	Source	Stated prob.	Probability Distribution			
128-128	Table 9 [DDH+20]	120	97.83 18.75%	99.415 37.5%	101 18.75%	Others 25%
128-256	Table 10 [DDH+20](E)	127.66	125.9X 5.21%	124.6X 4.21%	122.6X 4.14%	Others 86.44%
	18-L-I [HBS21](E)	97	86.7X 11.2%	89.8X 10.6%	86.1X 9.45%	Others 68.75%
	21-L-I [HBS21](E)	102.415	94.3X 2.70%	95.0X 2.68%	94.9X 2.52%	Others 92.1%
	18-L-II [HBS21](E)	73	71.9X 3.0%	70.9X 2.68%	69.1X 2.55%	Others 91.77%
	19-L-II [HBS21](E)	73	71.9X 3.0%	70.9X 2.68%	69.1X 2.55%	Others 91.77%
	Table 18 [QDW+21]	44-53	39-48 100%			
128-384	22-L-I [HBS21](E)	77.415	72.9X 9.30%	72.7X 6.39%	73.6X 5.43%	Others 78.88%
	23-L-I [HBS21]	10	10 100%			
	25-L-I [HBS21](E)	95.415	88.1X 4.17%	90.9X 4.07%	26.4X 3.16%	Others 88.6%
	Table 19 [QDW+21]	38-42	33-36 100%			

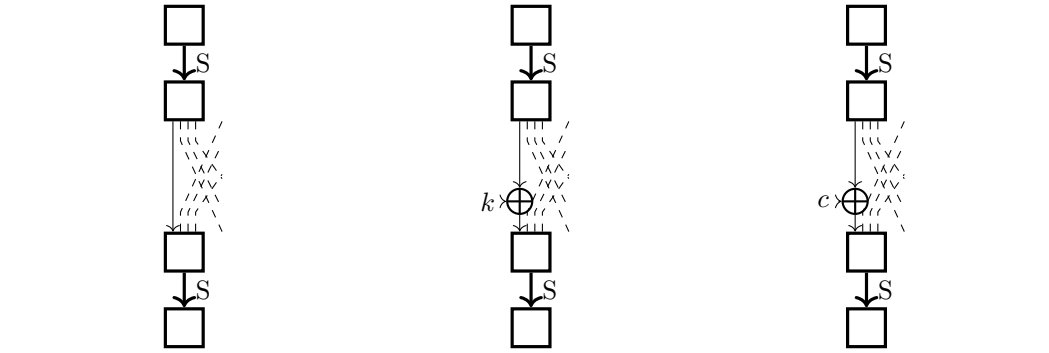
are three main layers. The first layer is the substitution layer which is built using 16 (resp. 32) 4-bit parallel Sboxes for GIFT-64 (resp. GIFT-128). The second layer is the bit-permutation layer and the last layer is the AddRoundKey and AddConstants functions. The key schedule is just a simple bit-permutation. Once again, we will refer to [BPP+17] for more details regarding the specifications of GIFT.

Bit permutation layer. As the permutation layer is described at the bit level, the addition of round keys and constants have to be described at bit-level as well. If we are tracing each bit through one round of the round function, there are three different types. These three types are shown in Figure 8. Note that some parts do not have any key bits. This is in contrast to ciphers like SKINNY where the MixColumns ensures that every cell has a key component XORed to it.

Linear constraints. Unlike SKINNY, the constraints in GIFT involve individual key bits or constant bits, or in some cases it simply causes an incompatibility. To have an efficient way to compare the linear constraints from two consecutive rounds, we first have to recall the symmetry in the bit permutation layer. We will use the permutation of GIFT-64 as an illustration. The permutation for GIFT-64 is given in Table 9.

If we look at it at the Sbox level, we can rearrange them to form 4 super-Sboxes as shown in Figure 9. When we are searching for linear constraints, we can just look at all the constraints that arise within their respective groups independently. This reduces the number of combinations of half constraints we have to compare to get full constraints.

Consider the transition pictured below, where the MSB of the first Sbox output is permuted to another arbitrary MSB of the Sbox output in the next round without any



(a) without key or constants (b) with single key bit (c) with a single constant bit

Figure 8: The three types of permutation in GIFT round function.

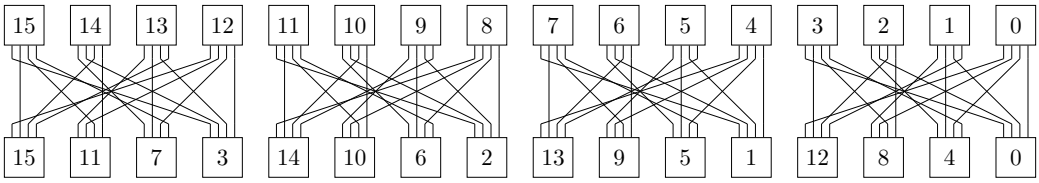


Figure 9: The bit permutation of GIFT-64, with a rearrangement of the positions of the Sboxes.

key bits or constant bits involved.

In the first Sbox, we have $\mathcal{Y}_{DDT}(2,6) = \{\texttt{0x9}, \texttt{0xa}, \texttt{0xc}, \texttt{0xf}\}$, i.e. the MSB of the output must be 1. In the second Sbox, we have $\mathcal{X}_{DDT}(2,6) = \{\texttt{0x1}, \texttt{0x3}, \texttt{0x5}, \texttt{0x7}\}$, i.e. the MSB of the input must be 0. In this case, where no key and/or constant bits are involved, this transition is impossible. Note that, the problem here is not about key dependency, but more globally that the Markov assumption is not valid regardless of the key value. If a constant bit was involved, then it would depend on the rounds in which the differential characteristic is being used. If a key bit was involved, then it would lead to a constraint that is dependent on the key.

Propagation. As the GIFT key schedule is just a bit permutation, the propagation can be performed by a reassignment/rewiring. In the end, we can just perform a Gaussian Elimination to find out the key space that allows the differential characteristic to pass (if possible).

Results. We have analyzed several differential characteristics. Out of 15 differential characteristics, we found 1 that is impossible. Almost all of them have constraints on the key bits and thus only work for a very small proportion of the keys. The results can be found in Table 3.

Nonlinear constraints. The nonlinear constraints will be more complicated as we have to take into account the inactive Sboxes. Once again, the idea is to look for inactive bits that belong to active Sboxes. However, this time around, we focus on those inactive outputs (resp. inputs) bits from round $r-1$ (resp. $r+1$) that are connected to inactive Sboxes

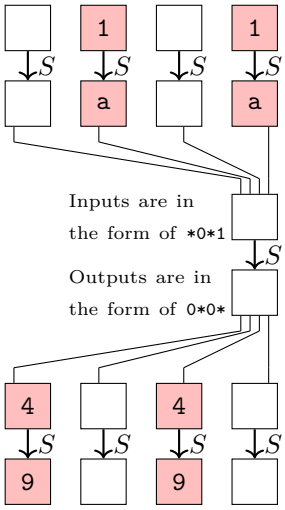


Figure 10: An example of a nonlinear constraint in GIFT. There are three rounds in the figure, focusing only on the connections of the inputs and the outputs of a single inactive Sbox in the middle round. The pink cells are the active Sboxes.

at round r, limiting the number of input (resp. output) values that inactive Sboxes in round r can have. If there does not exist a pair of input-output values that satisfies the input and output restrictions, then we can deduce that the differential characteristic is impossible. To illustrate how these constraints may change the probability, we provided an example in Figure 10. Notice that the Sboxes in pink are active. For the inactive Sbox in the middle, the input and output values have to be of the form `*0*1` and `0*0*` respectively where `*` can take up either `0` or `1`. In this example, according to the GIFT Sbox, none of the input-output values satisfies them. Note that in a not-so-extreme case, the input (resp. output) restrictions will induce new half constraints on the output (resp. input) values using the Sbox values too. Theoretically, the nonlinear constraints can stretch over more than just a single intermediate round and even multiple Sboxes in the same round. However, taking all of them into account requires a huge computational cost that renders it practically impossible. For GIFT-64, we decided to look for nonlinear constraints that span over (three consecutive) rounds sharing the same pairs of round keys: every four rounds, the key schedule reuses the same 64-bit master key as the subkey. Thus, we divided the rounds into four batches and processed them independently. For instance, the first batch would be the ones involving the last 64 bits of the master key, and the values involved are:

- The output values from the Sbox of round $4r$
- The inactive Sboxes from round $4r + 1$
- The input values from the Sbox of round $4r + 2$

Note that there will be 32 bits of keys that are overlapping from different batches, but due to computational difficulties, we ignored them. In GIFT-128, we cannot perform the same as what we did for GIFT-64 as each batch will involve all 128 bits of the key which will be too computationally expensive. Instead, we decided to search if the nonlinear constraints are consistent for all the possible three consecutive rounds independently (i.e. we can check if the differential characteristic is consistent w.r.t. both the linear and nonlinear constraints). To achieve this search, we used cryptominisat [SNC09] as a sub-routine.

6 Solutions

Ideally, when a differential characteristic has a reasonably high probability, one expects some form of experimental verification to ensure that the differential characteristic is actually possible. However, when the differential characteristic has a low probability, there is no convention as to how we can verify them yet. Furthermore, the problem of the key space is not addressed too. In this section, we propose some workarounds in an attempt to validate the differential characteristic or at least increase the cryptanalyst's confidence in its validity. We emphasize that these workarounds we are proposing are necessary but not sufficient to ensure the differential characteristics are functioning properly.

Our tools. We have uploaded the tools we used to analyze the differential characteristics shown in this paper in

https://github.com/QuanQuanTan/key-dependency-tool

They can directly help to check if a differential characteristic is possible under all keys or only some keys or for no key at all. This could be of interest to cryptanalysts who work on SKINNY and GIFT.

Experimental verification of shorter rounds. A majority of the constraints are actually linear, which only involve values from two consecutive rounds. From there, we can already check to a large extent what keys are compatible/incompatible with the differential characteristics. Thus, we propose that cryptanalysts should, to the best of their ability, test their differential characteristics with the involved subkeys to check if there exists a subspace of subkeys that will not work. The next thing is then to use the key schedule to check if the constraints are consistent with each other.

Incorporating into automated methods. In [SRB21], the authors proposed an MILP-based method to experimentally verify differential characteristics of Rotation-Xor ciphers. They have a two-step process for this:

1. Use a MILP model to search for a (related-key) differential characteristic
2. Check if there exists a right pair of messages and keys that satisfy the characteristic

In our case, we would like to have a similar program for SKINNY, with the additional requirement: given an impossible differential characteristic, find one that is similar, but works for at least a key. Thus, instead of separating them into a two-step process, we combined them into one instead. We provide a proof of concept which we have tested out using Constraint Programming (CP) on SKINNY-64. The pseudocode can be found in Algorithm 1. The program can also be found in our Github repository. In our paper, we have shown three types of constraints: linear constraints, nonlinear constraints and same-round constraints. While nonlinear constraints are rather complicated to incorporate into search tools, linear and same-round constraints are easier to encode into CP. Thus, we show that linear and same-round constraints can be encoded into CP in order to avoid differential characteristics that are impossible because of these constraints. We acknowledge that our CP program is not optimized and may have widened the search space by a significant margin. However, we believe that it is still simpler as compared to searching for an actual conforming pair. For our CP program, we used the `MiniZinc` [NSB+07] interface with the `Chuffed` [CSS+] CP solver. For the sake of this discussion, we will just focus on the description of the additional simple linear constraints on our CP program. We defined two new table constraints encoding the $\mathcal{X}_{DDT}$ and $\mathcal{Y}_{DDT}$. Each table is three-dimensional, and the $(i, j, k)^{th}$ entry of $\mathcal{X}_{DDT}$ is 1 if the value k is in the set $\mathcal{X}_{DDT}(i, j)$ and 0 otherwise. We

use the linear constraints as the conditional statement, and whenever they are detected, we restrict the involved round key to be a linear function of the relevant entries of $\mathcal{X}_{DDT}$ and $\mathcal{Y}_{DDT}$ entries. For instance, the row that passes through a SKINNY round without any XOR in the MixColumns function can be represented as:

```
constraint forall (n in 0..NR-2, c in 0..3) (
   if ((state0[n,c] > 0) /\(state0[n+1,4+c] > 0)) then
      forall(i in 0..15) (
         roundKeys1[n,c,i] = max (j in 0..15) (
               yddt[state0[n,c],state1[n,c],j] *
               xddt[state0[n+1,4+c],state1[n+1,4+c],XOR(i,j)]
      )
   )
   else true
   endif
);
```

where `state0[n,c]` (resp. `state1[n,c]`) refers to the difference of the c^{th} cell in the n^{th} round of the state before (resp. after) the SubCells function in round n. `roundKeys1[n][c][i]` contains the required round key imposed by this particular set of constraints. Next, we simply combine all these constraints together and compare them using the key schedule and master key. Note that we only present the case of TK1.

```
constraint forall (c in 0..15, i in 0..15) (
   masterKeys[c,i] = product([combinedRoundKeys[n,c,i] | n in 0..NR-1])
);
```

As a proof of concept, we have applied it to the 10-round differential characteristic D in Table 6 of [DDH+20] where when using our tool, we have detected that it is actually impossible due to two linear constraints (in round 4 and round 8 respectively) not being compatible with each other, as well as a "same-round" constraint in round 0. The stated probability for D is 2^{-46}. When we fix D in our CP program, our solver will return "unsatisfiable", showing that it actually detects the incompatibility. Next, to minimize the search space to find a differential characteristic that is actually possible, but with a similar probability, we fixed the difference at the start of round 1 to round 9 (except the very last output difference and the first input difference). The reason is simple: these are the places where constraints are detected. Our CP program managed to find one with a slightly different differential characteristic (see Table 10) with the same (stated) probability, but being actually possible. Note that this differential characteristic still has two linear constraints. The independent probability is 2^{-42} working within 2^{-3} of the key space. As the search space is largely limited to just the neighborhood of the original differential characteristic, the search time is negligible in this case.

In the case of GIFT, additional constraints on the actual values can be created whenever a Sbox transition is fixed. For example, suppose the k^{th} Sbox transition is ($\mathtt{0x2} \rightarrow \mathtt{0x6}$) and $\mathcal{X}_{DDT} = \{\mathtt{0x1}, \mathtt{0x3}, \mathtt{0x5}, \mathtt{0x7}\}$, the half constraints to be added are:

$$s_{4k+3} = 0 \qquad s_{4k} = 1$$

7 Conclusion

As it has long been cautioned by theoretical works/discussions in the past, the assumption of key independence and more generally the Markov assumption cannot be neglected when we are using differential characteristics, especially for lightweight ciphers where the diffusion

in each round is not strong and where the key scheduling function is rather simple. In this paper, we have discussed a generic framework that cryptanalysts can use to analyze existing differential characteristics and provided a clear picture of the probability distribution of the SKINNY cipher as an example. We have also provided a tool and its implementation that cryptanalysts can use to analyze differential characteristics of SKINNY and GIFT. We believe that our tool can be applied quite naturally to other AES-like ciphers. Our results indeed validate the concerns those theoretical works had, as we showed that many differential characteristics in multiple published works are plain impossible, and almost all of them are at best only working for a very small proportion of the keys (potentially impacting the cryptanalysis results built on top of them). We hope that this article would change how cryptanalysts present their newly-found differential characteristics in the future. We believe it is important when possible to show the subspace of the possible keys for which their differential characteristics can work, as well as the probability of the differential characteristic that is independent of the key. However, undoubtedly, our algorithm is still relying on detecting patterns that we have identified. While we have experimentally verified some differential characteristics, it is likely that if a differential characteristic is dense enough, even more complex constraints may emerge. As future work, one could look at an algorithm that work similarly to automated tools such as SAT, but as a more readable version so that cryptanalysts can identify the problematic parts of the trails and make the changes accordingly. One can also look into how our framework can apply to other ciphers as well. Moreover, as the key dependencies of differential characteristics seem a bit clearer now, a very interesting future research topic would be to explore how cryptanalysts can actually exploit such an effect to their advantage, perhaps in an impossible differential attack setting.

Acknowledgements

The authors are grateful to the anonymous reviewers for their insightful comments that improved the quality of the paper. The authors are supported by the Temasek Laboratories.

References

[AST+17] Ahmed Abdelkhalek, Yu Sasaki, Yosuke Todo, Mohamed Tolba, and Amr M. Youssef. MILP Modeling for (Large) S-boxes to Optimize Probability of Differential Characteristics. *IACR Trans. Symmetric Cryptol.*, 2017(4):99–129, 2017.

[BJK+16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. In *CRYPTO 2016*, volume 9815 of *LNCS*, pages 123–153. Springer, 2016.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *CHES 2007*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.

[BPP+17] Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A Small Present - Towards Reaching the Limit of Lightweight Encryption. In *CHES 2017*, volume 10529 of *LNCS*, pages 321–345. Springer, 2017.

[BS91] Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. *J. Cryptol.*, 4(1):3–72, 1991.

[CLN+17] Anne Canteaut, Eran Lambooij, Samuel Neves, Shahram Rasoolzadeh, Yu Sasaki, and Marc Stevens. Refined Probability of Differential Characteristics Including Dependency Between Multiple Rounds. *IACR Trans. Symmetric Cryptol.*, 2017(2):203–227, 2017.

[CSS+] Geoffrey Chu, Peter J. Stuckey, Andreas Schutt, Thorsten Ehlers, Graeme Gange, and Kathryn Francis. Chuffed, a lazy clause generation solver. https://github.com/chuffed/chuffed.

[DDH+20] Stéphanie Delaune, Patrick Derbez, Paul Huynh, Marine Minier, Victor Mollimard, and Charles Prud'homme. SKINNY with Scalpel - Comparing Tools for Differential Analysis. *IACR Cryptol. ePrint Arch.*, page 1402, 2020.

[DDV20] Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. Catching the Fastest Boomerangs Application to SKINNY. *IACR Trans. Symmetric Cryptol.*, 2020(4):104–129, 2020.

[DR07] Joan Daemen and Vincent Rijmen. Plateau characteristics. *IET Inf. Secur.*, 1(1):11–17, 2007.

[DR20] Joan Daemen and Vincent Rijmen. *The Design of Rijndael - The Advanced Encryption Standard (AES), Second Edition.* Information Security and Cryptography. Springer, 2020.

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. In *CHES 2011*, volume 6917 of *LNCS*, pages 326–341. Springer, 2011.

[HBS21] Hosein Hadipour, Nasour Bagheri, and Ling Song. Improved Rectangle Attacks on SKINNY and CRAFT. *IACR Trans. Symmetric Cryptol.*, 2021(2):140–198, 2021.

[JNP14] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In *ASIACRYPT 2014*, volume 8874 of *LNCS*, pages 274–288. Springer, 2014.

[KM04] Lars R. Knudsen and John Erik Mathiassen. On the Role of Key Schedules in Attacks on Iterated Ciphers. In *ESORICS 2004*, volume 3193 of *LNCS*, pages 322–334. Springer, 2004.

[Leu12] Gaëtan Leurent. Analysis of Differential Attacks in ARX Constructions. In *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 226–243. Springer, 2012.

[LLL+21] Yu Liu, Huicong Liang, Muzhou Li, Luning Huang, Kai Hu, Chenhe Yang, and Meiqin Wang. STP models of optimal differential and linear trail for S-box based ciphers. *Sci. China Inf. Sci.*, 64(5), 2021.

[LMM91] Xuejia Lai, James L. Massey, and Sean Murphy. Markov Ciphers and Differential Cryptanalysis. In *EUROCRYPT '91*, volume 547 of *LNCS*, pages 17–38. Springer, 1991.

[LWRA17] Yunwen Liu, Glenn De Witte, Adrián Ranea, and Tomer Ashur. Rotational-XOR Cryptanalysis of Reduced-round SPECK. *IACR Trans. Symmetric Cryptol.*, 2017(3):24–36, 2017.

[LWZZ19] Lingchen Li, Wenling Wu, Yafei Zheng, and Lei Zhang. The Relationship between the Construction and Solution of the MILP Models and Applications. *IACR Cryptol. ePrint Arch.*, page 49, 2019.

[LZS+20] Yunwen Liu, Wenying Zhang, Bing Sun, Vincent Rijmen, Guoqiang Liu, Chao Li, Shaojing Fu, and Meichun Cao. The phantom of differential characteristics. *Des. Codes Cryptogr.*, 88(11):2289–2311, 2020.

[MNS11] Florian Mendel, Tomislav Nad, and Martin Schläffer. Finding SHA-2 Characteristics: Searching through a Minefield of Contradictions. In *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 288–307. Springer, 2011.

[MRTV12] Florian Mendel, Vincent Rijmen, Deniz Toz, and Kerem Varici. Differential Analysis of the LED Block Cipher. In *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 190–207. Springer, 2012.

[NSB+07] Nicholas Nethercote, Peter J. Stuckey, Ralph Becket, Sebastian Brand, Gregory J. Duck, and Guido Tack. MiniZinc: Towards a Standard CP Modelling Language. In *CP 2007*, volume 4741 of *LNCS*, pages 529–543. Springer, 2007.

[NWW15] Ivica Nikolic, Lei Wang, and Shuang Wu. Cryptanalysis of Round-Reduced LED. *IACR Cryptol. ePrint Arch.*, page 429, 2015.

[QDW+21] Lingyue Qin, Xiaoyang Dong, Xiaoyun Wang, Keting Jia, and Yunwen Liu. Automated Search Oriented to Key Recovery on Ciphers with Linear Key Schedule Applications to Boomerangs in SKINNY and ForkSkinny. *IACR Trans. Symmetric Cryptol.*, 2021(2):249–291, 2021.

[SNC09] Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending SAT Solvers to Cryptographic Problems. In *SAT 2009*, volume 5584 of *LNCS*, pages 244–257. Springer, 2009.

[SRB21] Sadegh Sadeghi, Vincent Rijmen, and Nasour Bagheri. Proposing an MILP-based method for the experimental verification of difference-based trails: application to SPECK, SIMECK. *Des. Codes Cryptogr.*, 89(9):2113–2155, 2021.

[SWW18] Ling Sun, Wei Wang, and Meiqin Wang. More Accurate Differential Properties of LED64 and Midori64. *IACR Trans. Symmetric Cryptol.*, 2018(3):93–123, 2018.

[SWW21] Ling Sun, Wei Wang, and Meiqin Wang. Improved Attacks on GIFT-64. *IACR Cryptol. ePrint Arch.*, page 1179, 2021.

[WLF+05] Xiaoyun Wang, Xuejia Lai, Dengguo Feng, Hui Chen, and Xiuyuan Yu. Cryptanalysis of the Hash Functions MD4 and RIPEMD. In *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 1–18. Springer, 2005.

[XLJ+22] Zheng Xu, Yongqiang Li, Lin Jiao, Mingsheng Wang, and Willi Meier. Do NOT Misuse the Markov Cipher Assumption - Automatic Search for Differential and Impossible Differential Characteristics in ARX Ciphers. Cryptology ePrint Archive, Report 2022/135, 2022. https://ia.cr/2022/135.

[ZDY19] Baoyu Zhu, Xiaoyang Dong, and Hongbo Yu. MILP-Based Differential Attack on Round-Reduced GIFT. In *CT-RSA 2019*, volume 11405 of *LNCS*, pages 372–390. Springer, 2019.

Appendix

A DDT of the SKINNY-64 Sbox

Table 8: The DDT of the SKINNY-64 Sbox, where Δ_{in} (resp. Δ_{out}) represents the input (resp. output) difference in hexadecimal notation. Each entry provides the number of input values valid with regards to the differential transition $\Delta_{in} \rightarrow \Delta_{out}$.

		Δ_{out}															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
Δ_{in}	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	0	0	0	0	0	0	4	4	4	4	0	0	0	0
	2	0	4	0	4	0	4	4	0	0	0	0	0	0	0	0	0
	3	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
	4	0	0	4	0	0	0	2	2	0	0	0	4	2	2	0	0
	5	0	0	4	0	0	0	2	2	0	0	4	0	2	2	0	0
	6	0	2	0	2	2	0	0	2	2	0	2	0	0	2	2	0
	7	0	2	0	2	2	0	0	2	0	2	0	2	2	0	0	2
	8	0	0	0	0	4	4	0	0	0	0	0	0	2	2	2	2
	9	0	0	0	0	4	4	0	0	0	0	0	0	2	2	2	2
	a	0	0	0	0	0	4	4	0	2	2	2	2	0	0	0	0
	b	0	4	0	4	0	0	0	0	0	0	0	0	2	2	2	2
	c	0	0	4	0	0	0	2	2	4	0	0	0	0	0	2	2
	d	0	0	4	0	0	0	2	2	0	4	0	0	0	0	2	2
	e	0	2	0	2	2	0	0	2	0	2	0	2	0	2	2	0
	f	0	2	0	2	2	0	0	2	2	0	2	0	2	0	0	2

B Algorithm to search for key dependencies in a differential characteristic for SKINNY

Algorithm 1 Algorithm to search for key constraints in SKINNY

Require: A N round differential characteristic of SKINNY, T[0..N-1][0..3][0..3]
Ensure: A list of tuples that give the key constraints

```
Initialize FS[0..3][0..3] = ∅
for n = 0..N-1 do
    for c = 0..4 do
        if T[n][0][c] > 0 then
            FS[1][c].append((n,0,c))
        else
            FS[1][c] = ∅
        end if
        if T[n][1][(c+1)%4] > 0 and T[n][2][(c+2)%4] > 0 then
            FS[2][c].append((n,1,(c+1)%4))
            FS[2][c].append((n,2,c))
        else
            FS[2][c] = ∅
        end if
        if T[n][0][c] > 0 and T[n][2][(c+2)%4] > 0 then
            FS[3][c].append((n,0,c))
            FS[3][c].append((n,2,(c+2)%4))
        else
            FS[3][c] = ∅
        end if
        if T[n][0][c] > 0 and T[n][2][(c+2)%4)] > 0 and T[n][3][(c+3)%4)] > 0 then
            FS[0][c].append((n,1,c))
            FS[0][c].append((n,2,(c+2)%4))
            FS[0][c].append((n,3,(c+3)%4))
        else
            FS[0][c] = ∅
        end if
    end for
    for r = 0..3 and c = 0..3 do
        if FS[r][c].length() > 0 and T[n+1][r][c] > 0 then
            print FS[r][c] and (n+1,r,c)                ▷ A constraint is detected
            FS[r][c] = ∅
        end if
    end for
end for
```

C GIFT-64 bit permutation layer

Table 9: GIFT-64's bit permutation layer.

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P_{64}(i)$	0	17	34	51	48	1	18	35	32	49	2	19	16	33	50	3
i	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
$P_{64}(i)$	4	21	38	55	52	5	22	39	36	53	6	23	20	37	54	7
i	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
$P_{64}(i)$	8	25	42	59	56	9	26	43	40	57	10	27	24	41	58	11
i	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
$P_{64}(i)$	12	29	46	63	60	13	30	47	44	61	14	31	28	45	62	15

D A corrected best 10-round TK1 differential characteristic of SKINNY-64

Table 10: A corrected best 10-round TK1 differential characteristic of SKINNY-64, originally from Table 6 of [DDH+20]. The first, second and third row in each cell of the table represent the difference before SB, the difference after SB and the round key involved respectively in hexadecimal notation. The values highlighted in red are the entries that differ from the original differential characteristic. The independent probability is 2^{-42} working with 2^{-3} of the key space.

Round	Differential transition
0	0,0,0,0,0,0,0,2,0,0,2,0,0,2,0,0 0,0,0,0,0,0,0,1,0,0,1,0,0,1,0,0 1,0,0,0,0,0,0,0
1	1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0 b,0,0,0,a,0,0,0,0,0,0,0,0,0,0,0 b,0,0,0,a,0,0,0
2	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0,0,1,0,0,0,0,0
3	0,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0 0,0,b,0,0,0,8,0,0,0,0,0,0,0,b,0 0,0,b,0,0,0,a,0
4	0,b,0,0,0,0,0,0,0,0,0,2,0,0,0,0 0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0 0,0,0,0,1,0,0,0
5	0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0 0,0,0,0,0,0,a,0,0,0,0,0,0,0,0,0 0,0,0,0,b,a,0,0
6	0,0,0,0,0,0,0,0,0,b,0,0,0,0,0,0 0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0 0,0,0,0,0,0,1,0
7	0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1 0,0,0,a,0,0,0,0,0,0,0,0,0,0,0,8 0,0,0,a,0,0,b,0
8	0,0,8,0,0,0,0,0,0,0,b,0,0,0,0,0 0,0,4,0,0,0,0,0,0,0,1,0,0,0,0,0 0,0,0,0,0,1,0,0
9	0,1,4,0,0,0,4,0,0,1,1,0,0,1,4,0 0,a,2,0,0,0,2,0,0,8,8,0,0,8,2,0 0,0,0,0,0,b,0,a