Analysis of BLAKE2

Jian Guo[1], Pierre Karpman[1,2], Ivica Nikolić[1], Lei Wang[1], and Shuang Wu[1]

[1] Nanyang Technological University, Singapore
[2] École normale supérieure de Cachan, France
{ntu.guo,pierre.karpman}@gmail.com, {inikolic,Wang.Lei,WuShuang}@ntu.edu.sg

Abstract. We present a thorough security analysis of the hash function family BLAKE2, a recently proposed and already in use tweaked version of the SHA-3 finalist BLAKE. We study how existing attacks on BLAKE apply to BLAKE2 and to what extent the modifications impact the attacks. We design and run two improved searches for (impossible) differential attacks — the outcomes suggest higher number of attacked rounds in the case of impossible differentials (in fact we improve the best results for BLAKE as well), and slightly higher for the differential attacks on the hash/compression function (which gives an insight into the quality of the tweaks). We emphasize the importance of each of the modifications, in particular we show that an improper initialization could lead to collisions and near-collisions for the full-round compression function. We analyze the permutation of the new hash function and give rotational attacks and internal differentials for the whole design. We conclude that the tweaks in BLAKE2 were chosen properly and, despite having weaknesses in the theoretical attack frameworks of permutations and of fully-chosen state input compression functions, the hash function of BLAKE2 has only slightly lower security margin than BLAKE.

Key words: BLAKE2, BLAKE, hash function, rotational cryptanalysis, impossible differential cryptanalysis, differential cryptanalysis, internal differential, iterative differential.

1 Introduction

The BLAKE hash function [2] was one of the five finalists of the SHA-3 competition [12] that ended in November 2012, with Keccak [8] becoming the new SHA-3 standard. Along with the other finalists, BLAKE is assumed to be a very strong hash function [12]. Even though it was not selected as the winner, it enjoys a large security margin, very good performance in software, and has attracted a considerable amount of cryptanalysis. BLAKE uses addition, rotation, and XOR as building blocks for the compression function, has an iteration mode based on HAIFA [9] and thus it supports salt, uses an expanding to double-pipe internal state which makes meet-in-the-middle attacks unfeasible, and in the compression function applies only word permutations for the message schedule, thus making it very simple, elegant and more importantly efficient.

BLAKE2 [4, 3, 5] is a new family of hash functions based on BLAKE. Despite being a new design, BLAKE2 has already been adopted by several software packages — for instance, it is implemented in the CyaSSL library, and is supported in the RAR 5.0 archive format [5]. This surprisingly quick adoption of a new hash function is most likely due to the popularity and qualities of its predecessor BLAKE. The main objective of the new BLAKE2 is to provide a number of parameters for use in applications without the need of additional constructions and modes (*e.g.*,, it supports parallelism, tree-hashing and prefix-MAC), and also to speed-up even further the hash function to reach a level of compression rate close to MD5 [4]. The designers have achieved this goal by slightly altering the original BLAKE; in particular they have modified the initial setup of the compression function, changed the rotation constants to be optimal for software performance, excluded constants from the round functions, etc. To implement these tweaks only a small change in the code of BLAKE is required.

While the efficiency argument of the new BLAKE2 is undoubtedly correct and can be confirmed by a mere comparison of the speed of software implementations of BLAKE2 and BLAKE (or MD5), the security of the new function is unclear. The designers claim security levels similar to that of BLAKE, due to the similarity of the two designs. However they do not provide a strict analysis. Note that no universal method nor theory exists that can transitively prove the security of a symmetric primitive A obtained by modifying a primitive B, excluding of course trivial modifications such as increasing the number of rounds. Moreover, omitting constants in the round function of BLAKE2 is a major tweak and it might lead to exploits as now the rounds differ only in the order they process the message words.

Table 1. Summary of the analysis of BLAKE and BLAKE2.

Framework[a]	Type	# Rounds[b]	Complexity	Reference
BLAKE-256 perm.	impossible differential	5	—	[1]
		6.5	—	this paper, §5
	differential	6	2^{486}	[15]
	boomerang	8	2^{232}	[10]
BLAKE-512 perm.	impossible differential	5[c]	—	[1]
		6.5	—	this paper, §5
BLAKE-256 cf.	boomerang	7	2^{242}	[10]
	near collision	4	2^{56}	[1]
BLAKE-256	collision	2.5	2^{112}	[19]
	preimage	2.5	2^{241}	[19]
BLAKE-512	collision	2.5	2^{224}	[19]
	preimage	2.5	2^{481}	[19]
BLAKE2s perm.	impossible differential	6.5	—	this paper, §5
	rotational	7	2^{511}	this paper, §3
BLAKE2b perm.	impossible differential	6.5	—	this paper, §5
	rotational	**12**	2^{876}	this paper, §3
	differential	5.5	2^{928}	this paper, §6
BLAKE2s cf. chosen IV	collisions	**10**	2^{64}	this paper, §3
BLAKE2b cf. chosen IV	partial-collisions	**12**	2^{61}	this paper, §4
	2^{64} weak preimages	**12**	1	this paper, §4
BLAKE2b cf.	differential	4.5	2^{495}	this paper, §6
BLAKE2b	differential	3.5	2^{480}	this paper, §6

[a] The notations ‘perm.’ and ‘cf.’ stand for the permutation and compression function of the associated hash function.

[b] The total number of rounds in BLAKE-256, BLAKE-512, BLAKE2s, and BLAKE2b is 14,16,10, and 12 rounds, respectively.

[c] The initial analysis claimed a 6-rounds attack, but it was shown to be incorrect.

Our contribution. In this paper we give a thorough security analysis of this new hash function. Our main objective is to find out if the security level of BLAKE2 has dropped due to the tweaks. We try to exploit each tweak separately, as well as in combination with the others, in order to mount attacks on as many rounds as possible. The starting point of our analysis in the framework of permutations and compression function with chosen IV's are three promising techniques that can be highly successful against primitives that employ low usage of constants (*i.e.,* no adding constants to the message words): rotational cryptanalysis [18], internal differentials [20] (more precisely the squeeze attack [13, 14]) and iterative differentials based on rotational trails. We show that in these two frameworks, the attacker can penetrate through all 12 rounds of BLAKE2b. Further, we focus on the previous attacks on the original design, in particular the differential and impossible differential attacks [1]. We improve the previous results and approaches and along the way show the impact of the new initialization used in the compression function. We develop more advanced techniques to search for differentials — in particular, we implement a search for the best differential characteristics from a certain subspace which is much larger compared to all the previously analyzed ones. We show that due to the new rotations, the best result is now a 3.5-round differential distinguisher for the hash function of BLAKE2b, while a 4.5-round differential exists for the compression function. In the impossible differential analysis, we are able both to find and *confirm* theoretically probability-one characteristics. In the previously published analysis the search of characteristics was mostly experimental, and in the case of longer characteristics was actually incorrect. Our analysis is valid for BLAKE as well, *i.e.,* we improve the best known results for impossible differentials for the original design. We summarize the result of our analysis of BLAKE2 and the best existing attacks on BLAKE in Tbl. 1.

This paper is organised as follows. In section 2, we give a brief description of the BLAKE2 hash function family. In sections 3, 4, 5, 6, we describe our rotational, fixed points, impossible differential, and differential analyses of BLAKE2, respectively. We conclude in section 7.

2 Description of BLAKE2

As a successor of the BLAKE family, the BLAKE2 hash functions share many similarities with the original design. However differences occur at all levels of the design: internal permutation, compression function, and hash function construction. In this section we give a brief specification of BLAKE2 and highlight the differences with BLAKE. We use notations similar to [4], in particular:

- '$\leftarrow$' denotes variable assignment;
- '$+$' denotes addition in $\mathbb{Z}_{2^{32}}$ or in $\mathbb{Z}_{2^{64}}$ (modular addition);
- '$-$' denotes subtraction in $\mathbb{Z}_{2^{32}}$ or in $\mathbb{Z}_{2^{64}}$ (modular subtraction);
- '$\oplus$' denotes addition in $\mathbb{Z}_2^{32}$ or in $\mathbb{Z}_2^{64}$ (bitwise exclusive or);
- '$\lll r$' denotes rotation of r bits towards the most significant bit;
- '$\ggg r$' denotes rotation of r bits towards the least significant bit;
- if not specified otherwise, numbers written in typewriter font are in base 16, *e.g.,* f is the number 15.

The internal state of the BLAKE2 compression function is composed of 16 words of size 64 bits for BLAKE2b, and 32 bits for BLAKE2s. The compression function takes as an input an 8-words chaining value $h_0, \ldots, h_7$, 8 constant initialization vectors $IV_0, \ldots, IV_7$, a 2-words counter $t_0 t_1$ that counts the number of *bytes* hashed so far, and two finalization flags f_0, f_1. The flag f_0 is set to ff...ff when the the current message block is the last, and to 00...00 otherwise; the f_1 counter plays a similar role in tree-hashing (and is not detailed here). The input to the compression function is initialized as (we follow the notations of the design paper here):

$$\begin{pmatrix} v_0 & v_1 & v_2 & v_3 \\ v_4 & v_5 & v_6 & v_7 \\ v_8 & v_9 & v_{10} & v_{11} \\ v_{12} & v_{13} & v_{14} & v_{15} \end{pmatrix} \leftarrow \begin{pmatrix} h_0 & h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 & h_7 \\ IV_0 & IV_1 & IV_2 & IV_3 \\ t_0 \oplus IV_4 & t_1 \oplus IV_5 & f_0 \oplus IV_6 & f_1 \oplus IV_7 \end{pmatrix}.$$

The main differences between BLAKE2 and BLAKE at this stage are the removal of the optional salt value, the addition of the finalization flags instead of the repeated counter words, and the fact that the counter now counts the number of bytes rather than bits.

The initial state is then processed by 10 (resp. 12) rounds of a column and diagonal application of the defined below G function for BLAKE2s (resp. BLAKE2b). In comparison, BLAKE-256 and BLAKE-512 functions have 14 and 16 rounds. The G functions take four state words (a, b, c, d) and two message words m_i, m_j as input. The latter are defined by a position index i of the function: at round r, m_i is given by $\sigma_{r \bmod 10}(2i)$ and m_j by $\sigma_{r \bmod 10}(2i+1)$, where $\sigma_{r \bmod 10}$ is one of the 10 permutations given in the Appendix A.

The G function of BLAKE2b $\mathtt{G}(a, b, c, d)$ is defined as:

$$\begin{array}{ll} 1: a \leftarrow a + b + m_i & 5: a \leftarrow a + b + m_j \\ 2: d \leftarrow (d \oplus a) \ggg 32 & 6: d \leftarrow (d \oplus a) \ggg 16 \\ 3: c \leftarrow c + d & 7: c \leftarrow c + d \\ 4: b \leftarrow (b \oplus c) \ggg 24 & 8: b \leftarrow (b \oplus c) \ggg 63 \end{array}.$$

The G function of BLAKE2s $\mathtt{G}(a, b, c, d)$ is defined as:

$$\begin{array}{ll} 1: a \leftarrow a+b+m_i & 5: a \leftarrow a+b+m_j \\ 2: d \leftarrow (d \oplus a) \ggg 16 & 6: d \leftarrow (d \oplus a) \ggg 8 \\ 3: c \leftarrow c+d & 7: c \leftarrow c+d \\ 4: b \leftarrow (b \oplus c) \ggg 12 & 8: b \leftarrow (b \oplus c) \ggg 7 \end{array}.$$

The differences between the G functions of BLAKE2 and BLAKE are the omission in BLAKE2 of an '$\oplus$' addition between the message words and round constants in steps 1 and 5, and modified rotation constants for BLAKE2b. We also give the definition of the inverses $\mathtt{G}^{-1}$ of the G functions in the Appendix A.

A column step of BLAKE2 computes

$$\mathtt{G}_0(v_0, v_4, v_8, v_{12}) \quad \mathtt{G}_1(v_1, v_5, v_9, v_{13}) \quad \mathtt{G}_2(v_2, v_6, v_{10}, v_{14}) \quad \mathtt{G}_3(v_3, v_7, v_{11}, v_{15}),$$

and a diagonal step computes

$$\mathtt{G}_4(v_0, v_5, v_{10}, v_{15}) \quad \mathtt{G}_5(v_1, v_6, v_{11}, v_{12}) \quad \mathtt{G}_6(v_2, v_7, v_8, v_{13}) \quad \mathtt{G}_7(v_3, v_4, v_9, v_{14}).$$

Finally, the output of the compression function $h'_0, \ldots, h'_7$ combines the input chaining value and the final state $v_0, \ldots, v_{15}$ by computing

$$\begin{array}{ll} h'_0 \leftarrow h_0 \oplus v_0 \oplus v_8 & h'_4 \leftarrow h_4 \oplus v_4 \oplus v_{12} \\ h'_1 \leftarrow h_1 \oplus v_1 \oplus v_9 & h'_5 \leftarrow h_5 \oplus v_5 \oplus v_{13} \\ h'_2 \leftarrow h_2 \oplus v_2 \oplus v_{10} & h'_6 \leftarrow h_6 \oplus v_6 \oplus v_{14} \\ h'_3 \leftarrow h_3 \oplus v_3 \oplus v_{11} & h'_7 \leftarrow h_7 \oplus v_7 \oplus v_{15} \end{array}.$$

The only difference between BLAKE2 and BLAKE in this step is again the omission of the optional salt value.

The BLAKE2 hash function is defined in a straightforward way from the above compression function. We give a high-level overview of this process here, and refer to [4] for more details.

1. A 'parameter block' (described below) is added ($\oplus$) with the same initialization vectors used in the compression function. This makes the first input chaining value to the compression function.
2. The message is padded with null bytes if and only if necessary to make it a multiple of a block length (*i.e.,* 512 bits for BLAKE2s and 1024 bits for BLAKE2b).
3. The compression function is iterated on the padded message, and its (possibly truncated) final output is taken as the hash value.

The 'parameter block' mentioned above encodes various parameters that specify an instance of the BLAKE2 hash function. General parameters are the digest length, the optional key length, an optional salt, and a personalization string. Additional parameters are defined for tree hashing. Again, we refer to [4] for the full specifications.

The main differences with BLAKE in this respect is the simplified padding and the inclusion of a parameter block. Some of the optional functionalities of BLAKE (*e.g.,* the salt) have been moved from the compression function to the parameter block. We would like to make a remark at this point: as the padding of BLAKE2 is not separable, and the counter of the compression function keeps track of bytes, there are trivial collisions for the BLAKE2 functions when messages of arbitrary bit-length are allowed. The messages '0' (one zero bit) and '00' (two zero bits) are one such example. Therefore, BLAKE2 is only secure as a hash function when processing data at the byte granularity.

Current state of security of BLAKE2. In the submission document, the designers state that BLAKE2 inherits the security level of its predecessor BLAKE-256/512. In particular, they expect that the number of attacked rounds in BLAKE2 and BLAKE should be the same (possibly with slightly different complexities) with regards to the published analysis. For BLAKE the designers single out three attacks that penetrate the most number of rounds:

1. The 2.5-round preimage attack for the hash function by Ji and Liangyu [19].
2. The 6-round distinguisher for the permutation of BLAKE-256 proposed by Dunkelman and Khovratovich [15].

3. The 8-round boomerang distinguisher for the permutation of BLAKE-256, and the 7-round boomerang distinguisher for the compression function of BLAKE-256 by Biryukov *et al.* [10].

However, it seems the designers have overlooked the fact that the setup of the initial state of BLAKE2, *i.e.,* the initialization, gives less degrees of freedom to the attacker and more importantly fixes completely the values of six state words $v_8, v_9, v_{10}, v_{11}, v_{14}, v_{15}$. Hence the boomerangs for the compression function of BLAKE cannot be trivially extended to BLAKE2. In particular, as the 3-round trail used at the top of the 6-round boomerang of BLAKE has differences in the words of the third row, it cannot be applied to BLAKE2s. After a careful examination of all the trails given in [10], and under the assumption that trails with similar probabilities can be found in BLAKE2, boomerangs can be launched for 5 rounds $(2+3$ rounds) of BLAKE2s, and 5.5 rounds $(2+3.5)$ of BLAKE2b.

Our Attack Frameworks. The previous published analysis of BLAKE target the permutation, the compression function, and the hash function of BLAKE. In this paper we show attacks on round-reduced versions of all of these three primitives. We assume a standard generic security level for them, for example the cipher BLAKE2s is a 512-bit block cipher with 512-bit key, thus an exhaustive key recovery attack requires 2^{512} encryptions.

In the hash function framework we assume that the initial state is fixed, i.e. $v_0, v_1, \ldots, v_{15}$ are some predefined constants. The compression function framework is similar, but this time we allow freedom in $v_0, v_1, \ldots, v_7, v_{12}, v_{13}$, while $v_8, \ldots, v_{11}, v_{14}, v_{15}$ stay fixed (equal to IV_0, IV_1, IV_2, IV_3, IV_6, IV_7), *i.e.,* we assume the attacker can control the chaining value and the counters t_0, t_1. We also analyze the case when the attacker can control the IV's — so called chosen IV. Finally, in the framework of permutations, we assume we can fully control the plaintext, thus all v_i can be chosen, however the key (the message) is unknown. The reader should be aware that the importance of the attacks drops as one goes from the framework of hash functions to the one of permutations.

3 Rotational Analysis and Internal Differentials

BLAKE2 is an ARX primitive, *i.e.,* the only operations used are modular and bitwise addition, as well as rotations on various amounts. Moreover, due to the absence of constants in the G function (which were present in BLAKE), it is a good target for rotational attacks. Recall that in such attacks, one starts with rotational pairs of inputs $(x, x \lll r)$, and checks if the output of the primitive F is also rotational, *i.e.,* if $F(x) \lll r = F(x \lll r)$. In [18] it was shown that the probability of a rotational output for ARX primitive depends only on the number of modular additions used in F.

The function G in BLAKE2 has 6 additions. To maximize the probability we fix the rotation amount to 1, thus the rotational probability of modular addition becomes around $2^{-1.4}$. Hence, for the whole G function we obtain $2^{6\cdot(-1.4)} \approx 2^{-8.4}$. Experiments show that the actual probability is slightly lower, *i.e.,* around $2^{-9.1}$. As one round of BLAKE2 has eight G function, the rotational probability of a round is $2^{8\cdot(-9.1)} \approx 2^{-73}$.

The block cipher of BLAKE2b has 12 rounds, thus the rotational probability for the whole cipher is $2^{12\cdot(-73)} = 2^{-876}$. Hence in a related-key framework, where the second key is a rotation by 1 of the first key, we can distinguish the cipher. Similarly, for the 10-round cipher of BLAKE2s we can attack 7 rounds with a complexity slightly faster than an exhaustive search of 512-bit key. Converting the distinguisher into a key-recovery attacks is possible as well. We can use the knowledge of the plaintext and ciphertext, to recover $4 \cdot 1.4 \approx 6$ bits at the top and the same amount at the bottom, thus from a rotational pair of plaintexts/ciphertexts we can reduce the entropy of the key by 12 bits.

Let us try to apply to above distinguisher to the compression function of BLAKE2. Note, the constants $IV_0, \ldots, IV_3$ used in the initialization are non-rotational. To overcome this issue, we can try to obtain rotational pairs after the first half round of BLAKE2, and use the message freedom of the second half round to satisfy probabilistically the rest of the rotational trail on t rounds. The first half round is composed of four applications of the function G with independent inputs that can be rotational in three of the four coordinates. That is, for each input IV_i, we have to find a pair of triplets $(a_1, b_1, d_1), (a_2, b_2, d_2)$ such that

$$\mathtt{G}(a_1, b_1, IV_i, d_1, m_1, m_2) \lll 1 = \mathtt{G}(a_2, b_2, IV_i, d_2, m_1 \lll 1, m_2 \lll 1). \qquad (1)$$

In total, we have 8 words of freedom to satisfy a 4-words equation, thus it seems a solution should exist. Surprisingly, this is not the case for a randomly chosen values of IV_i. A simple analysis shows that the above problem (1) can be reduced to the problem of finding solution for the equation

$$(X + Y + IV_i) \lll 1 = X \lll 1 + Y \lll 1 + IV_i \tag{2}$$

Hence, IV_i needs to be highly structured, *i.e.,* has to be a sum of a fully rotational word and two rotational errors. Thus we obtain a rather strange fact, that *the simplicity of the function* G[3] *prevents straightforward application of rotational distinguishers.* We note that one can try to obtain rotational pairs after the first full round of BLAKE2, but then the problem becomes much more complex, while the message freedom drops.

The absence of constants in the function G can be used to launch a distinguisher on the permutation of BLAKE2 based on internal differentials introduced by Peyrin [20]. More precisely, we will use its variant called the squeeze attack used in the attack on Keccak by Dinur *et al.* [13]. We note that a similar distinguisher was already applied to the permutations of Salsa and ChaCha [6, 7] — two ciphers that inspired the design of BLAKE.

Let the four columns at the input of the permutation of BLAKE2 be equal, *i.e.,* $(v_0, v_4, v_8, v_{12}) = (v_1, v_5, v_8, v_{13}) = (v_2, v_6, v_{10}, v_{14}) = (v_3, v_7, v_{11}, v_{15})$, and let all the message words (the words of the key) be the same. Then after the column step, all columns remain equal. Moreover, in the diagonal step, the first input is always taken from the top row (with all elements the same), the second from the second, etc., thus after the diagonal step, all the columns are still identical. Hence, a round of BLAKE2 preserves this property of the state. We can use the above property to launch a distinguishing attack for the permutation of BLAKE2. We need only a single query to the permutation — for plaintext composed of four identical columns, we check if the ciphertext has four such columns as well. Thus for w-bit word version there are 2^w keys (one key word is arbitrarily chosen, the rest are equal) for the cipher of BLAKE2 that can be distinguished with only one chosen plaintext. If all the inputs to the compression function could be chosen then the above approach could be used to produce collisions using the squeeze attack: 1) fix all the message words to some arbitrary value; 2) compress 2^{2w} different inputs, with the first column arbitrarily chosen, and the remaining three columns equal to the first. If there is a collision in one of the columns at the output, then the rest of the columns have to collide. As a column has $4w$ bits, 2^{2w} trials should be sufficient to produce collisions for the compression function — this is equivalent to 2^{128} calls for BLAKE2b, and 2^{64} for BLAKE2s. Similarly, it is possible to speed-up the search for preimages of a weak class of digests which are produced from the symmetric states — the size of the class is 2^{2w}. Again, the freedom in the input state and the message words is sufficient for the attacker to target digests from this class by only considering symmetric preimages, in time 2^{2w}. We would like to emphasize that in BLAKE2 the initialization once again prohibits this type of trivial attacks as $IV_0 \neq IV_1 \neq IV_2 \neq IV_3$, thus the above squeeze attack is not applicable to the compression/hash function of BLAKE2.

4 Fixed Points and Iterative Rotational Differentials for Search of Collisions and Preimages

The approach of section 3 can be enhanced further with the use of fixed points and iterative one-round differential characteristics. Assume P is a fixed point for the round function of BLAKE2b when all the message words are equal. Then, as there are no constants in the function G, and the message permutation for each round produces the same set of message words, P is a fixed point for any number of rounds of BLAKE2b. Further, let $\Delta \to \Delta$ be a one-round iterative characteristic with a low hamming weight difference. If for the pair of states $(P, P \oplus \Delta)$ the one round differential holds, *i.e.,* $\texttt{BLAKE2b}_{1\text{ round}}(P) \oplus \texttt{BLAKE2b}_{1\text{ round}}(P \oplus \Delta) = \Delta$, then the differential would hold for any number of rounds. Hence at the output we will end up with a low hamming weight difference in the states and thus a partial-collision. To apply this technique to BLAKE2b we have to be able to find an iterative one-round characteristic with probability $2^{-p}, p < 256$, and 2^p fixed points. Note that as all the message words are identical, we have only 2^{64} different permutations and approximately the same number of fixed points, hence we must have $p < 64$.

[3] If G were a random function, the solution would exist for any IV_i.

Our first task is to find fixed points for one round of BLAKE2b. We can accomplish this by finding fixed points for the function G and repeating the same value in all columns of P. In fact, this leads to a fixed point after only one half of the round, which in return results in a fixed point for the whole round. Let (a, b, c, d, m_1, m_2) be the inputs of the function G. We are looking for values such that $\mathtt{G}(a, b, c, d, m, m) = (a, b, c, d)$ (note that the message words coincide). From the definition of G, this is equivalent to solving the following system of equations:

$$(-d) \oplus a = d \ggg 16 \tag{3}$$
$$a + b + m + (c \oplus b \lll 1) + m = a \tag{4}$$
$$b \oplus (c - d) = (c \oplus b \lll 1) \ggg 24 \tag{5}$$
$$d \oplus (a + b + m) = (-d) \ggg 32 \tag{6}$$

With basic algebraic transformations the system can be reduced to:

$$a = d \ggg 16 \oplus (-d) \tag{7}$$
$$b + 2m = -(c \oplus b \lll 1) \tag{8}$$
$$b \oplus (c - d) = (c \oplus b \lll 1) \ggg 24 \tag{9}$$
$$b + m = [(-d) \ggg 32 \oplus d] - a \tag{10}$$

Let $V = [(-d) \ggg 32 \oplus d] - a$. Then we get:

$$a = d \ggg 16 \oplus (-d) \tag{11}$$
$$c = (b - 2V) \oplus b \lll 1 \tag{12}$$
$$b \oplus (c - d) = (c \oplus b \lll 1) \ggg 24 \tag{13}$$
$$m = V - b \tag{14}$$

If in (13) we replace the value of c from (12), we obtain

$$b \oplus [((b - 2V) \oplus b \lll 1) - d] = (b - 2V) \ggg 24 \tag{15}$$

Lemma 1. *The solution for the equation*

$$X \oplus [((X + A) \oplus X \lll 1) + B] = (X + A) \ggg 24 \tag{16}$$

where X is unknown, and A, B are constant 64-bit words, can be found on average in 2^{25} time.

Proof. The proof is given in the Appendix B.

We can now present the algorithm for solving the system:

1. Fix a random value for d. Compute a from (11), and the value of V according to the above formula.
2. Compute the value of b from (15).
3. Compute the value of c from (12).
4. Compute the value of m from (14).

Thus we can find one fixed point with around 2^{25} computations. Note that the value of d can take any 64-bit values, thus the number of fixed points is around 2^{64}. For each of these inputs, the 12-round compression function of BLAKE2b (with modified IV's !) has the form:

$$\begin{pmatrix} a & a & a & a \\ b & b & b & b \\ c & c & c & c \\ d & d & d & d \end{pmatrix} \xrightarrow{\text{12 rounds}} \begin{pmatrix} a & a & a & a \\ b & b & b & b \\ c & c & c & c \\ d & d & d & d \end{pmatrix} \xrightarrow{\text{feedforward}} \begin{pmatrix} c & c & c & c \\ d & d & d & d \end{pmatrix}. \tag{17}$$

The problem of finding iterative one-round characteristics has already been discussed for BLAKE-256 in the work of Dunkelman and Khovratovich [15]. The new rotation constants in BLAKE2b allow to apply their analysis without any significant modifications. However, straightforward use of their one-round characteristic based on two trails (with probabilities $2^{-12}, 2^{-21}$) for the function G is impossible. The problem lies in the condition $p < 64$:

- If we use the two trails and take four different columns for P then the probability of the first half round would be 2^{-66}.
- If we take only two different columns, then the probability of the first half round is 2^{-33}, and the same for the second half round. One can reduce the probability of the second half only with a special type of fixed points — instead of independent fixed points for each column (each function G) in the first half, one needs to deal with values that somehow depend on each other, thus it is not clear if such values exist at all.
- If we take the same value for all four columns, then we get a contradiction from the trails. No value can satisfy both trails as in the first modular addition $(a + b + m)$, we want 4 and 8 to cancel in the first trail (thus 4 should produce carries), while we want to stay at 4 in the second (no carries).

Hence we need to find a high probability one-round differential characteristic that can be used in combination with fixed points. We have implemented our own search based on the analysis of the above authors, and found that none of these type of characteristics are compatible — there is no iterative trail $\varDelta \to \varDelta$ for G, and all two round trails $\varDelta_1 \to \varDelta_2, \varDelta_2 \to \varDelta_1$ are incompatible, *i.e.,* they do not hold for the same value of the input (the value is the same as we work with fixed points).

We can nonetheless produce iterative differentials but based on the rotational property of the function G. Assume (P_1, P_2) is a rotational input pair for G producing the rotational output pair (Q_1, Q_2), *i.e.,* $P_2 = P_1 \lll 1, Q_2 = Q_1 \lll 1$. If P_1 is a fixed point for G, then for the second pair of input-output we get: $P_2 = P_1 \lll 1, Q_2 = Q_1 \lll 1 = P_1 \lll 1 = P_2$, *i.e.,* the second input is also a fixed point. Therefore for these fixed points the iterative differential has the input (as well as the output) difference $P_1 \oplus P_2 = P_1 \oplus P_1 \lll 1$. Now recall that we want to minimize the hamming weight of this difference in order to produce partial-collisions on as many bits as possible. In fact from (17) it is clear that we want to minimize only the hamming weight of the difference in c and d. As we work with rotation on 1 to the left, it follows that if the value of c (or d) has zeroes in t most significant bits then $c \oplus c \lll 1$ has zeroes in at least $t - 1$ most significant bits. This gives a hint of how to choose the fixed point P_1 using the above algorithm for finding fixed points:

1. Choose an arbitrary value of d that has zeroes in 27 MSBs.
2. Compute the values of a, b, c, d, m using the algorithm.
3. Check if c has zeroes in 27 MSBs.
4. If not, go to step 1.
5. Check if the input $(a \lll 1, b \lll 1, c \lll 1, d \lll 1, m \lll 1)$ is a fixed points.
6. If not, go to step 1.

The correct value of c at step 3 will be found after around 2^{27} different trials of d. As the rotational probability of the G function is $2^{-9.1}$, after $2^{9.1}$ good values of c one can find the second fixed point. Step 1 will be repeated $2^{27+9.1} \approx 2^{36}$ times, hence we have enough degrees of freedom in d (there are $2^{64-27} = 2^{37}$ possible values). The total complexity of the algorithm is $2^{25} \cdot 2^{27} \cdot 2^{9.1} \approx 2^{61}$. The hamming weight of the differences in both c and d will be at most 26 bits, and hence we can produce partial-collisions[4] on $8 \cdot 26 = 208$ bits. However this is *with chosen IV's.* That is, we can produce the collisions only when the values of the IV correspond to our discovered values for fixed points. Note as the original IV's used in BLAKE2b do not coincide, *our approach cannot be applied to the compression function of* BLAKE2b . Nonetheless, we show that the choice of IV's is sensitive to certain attacks.

A similar strategy can be applied for search of preimages for a special type of digests with $h'_0 = h'_1 = h'_2 = h'_3 = H_1$ and $h_4 = h_5 = h_6 = h_7 = H_2$. Let us assume that (h_0, h_1, H_1, H_2) is a fixed point (along with some message word m) for the function G. Then the full 12-rounds compression function of BLAKE2b (with modified IV's) can be described as:

$$\begin{pmatrix} h_0 & h_0 & h_0 & h_0 \\ h_1 & h_1 & h_1 & h_1 \\ H_1 & H_1 & H_1 & H_1 \\ H_2 & H_2 & H_2 & H_2 \end{pmatrix} \xrightarrow{\text{12 rounds}} \begin{pmatrix} h_0 & h_0 & h_0 & h_0 \\ h_1 & h_1 & h_1 & h_1 \\ H_1 & H_1 & H_1 & H_1 \\ H_2 & H_2 & H_2 & H_2 \end{pmatrix} \xrightarrow{\text{feedforward}} \begin{pmatrix} H_1 & H_1 & H_1 & H_1 \\ H_2 & H_2 & H_2 & H_2 \end{pmatrix}.$$

[4] Lately, collisions on some particular bits have been called partial-collisions.

Hence, if we can find the corresponding h_0, h_1, m, we will be able to recover the preimage of the target digest. For this purpose, we use the system (7) - (10):

1. Set $c = H_1$ and $d = H_2$.
2. Compute the value of a from (7).
3. Compute the value of b from (9) — it is a system of linear equations.
4. Compute the value of m^a from (8), and m^b from (10).
5. If $m^a = m^b$ then $h_0 = a, h_1 = b, m = m^a$ is the preimage.

The condition $m^a = m^b$ holds with probability 2^{-64} and therefore among all the possible 2^{128} digests from the class (recall that $|H_1| = |H_2| = 64$), preimage based on a fixed point can be found for $2^{128-64} = 2^{64}$ of them with a negligible effort.

5 Impossible Differential Analysis

In this section we perform an impossible differential (ID) analysis for the internal block ciphers of the whole `BLAKE` and `BLAKE2` families. A similar analysis was done for the original `BLAKE` by Aumasson *et al.* at FSE 2010 [1], where the authors claimed a 5-round ID for `BLAKE-256` and a 6-round ID for `BLAKE-512`. However these IDs were mainly found experimentally, and some of the presented characteristics had probabilities less than 1. Hence the analysis from [1] does not seem to cover more than five rounds for both `BLAKE-256` and `BLAKE-512`.

We carry a similar analysis on the four internal ciphers of `BLAKE-256`, `BLAKE-512`, `BLAKE2s` and `BLAKE2b`. Note that in contrast to the rotational analysis, the bitwise addition of constants plays no role in these impossible differentials, while the value of the rotation amounts is of importance. Hence the analysis of `BLAKE-256` and `BLAKE2s` is identical as their respective internal ciphers only differ in constant addition. On the other hand, the analysis must be performed independently for `BLAKE-512` and `BLAKE2b`. Our result is a 6.5-round impossible differential for all the four internal block ciphers of `BLAKE` and `BLAKE2`. As we need to insert differences in the message words, which play the role of the key when the permutation are seen as block ciphers, the analysis is performed in the related-key framework. The ID is found by using the miss-in-the-middle technique that connects a forward and a backward characteristic with incompatible probability-one differences. The forward characteristic is on 2.5 rounds and it can be extended for an additional half round, while the backward characteristic is on 3.5 rounds. As in the original analysis, our approach heavily relies on the good (for us) properties of the different σ_r message words permutations, which allow to delay the propagation of differences for 1.5 rounds in both forward and backward directions.

The analysis in this paper is innovative in the way it uses additive differences to cancel a difference in the message word of $\mathtt{G}^{-1}$ with probability one. Besides being an interesting result on the $\mathtt{G}^{-1}$ function itself, this is an important part of extending the ID to more rounds. Moreover, *we also formally checked the validity of our probability-one characteristics and we were able to prove they are correct* — this was not fully done in [1] and was a cause of invalid IDs. This check was performed by manually propagating the probability-one differences through the whole differential paths, however due to space constraints we only give the final differences.

We now detail the probability-one differential characteristics used in the ID. Differences are expressed with a subset of the generalized constraints of De Cannière and Rechberger [11]. In particular we use:

- ‘`-`’ to denote that two bits are equal;
- ‘`0`’ to denote that two bits are identical and equal to zero; similarly, we can define ‘`1`’.
- ‘`x`’ to denote that two bits are different;
- ‘`n`’ to denote that two bits are different, and the first bit is zero;
- ‘`u`’ to denote that two bits are different, and the first bit is one;
- ‘`?`’ to denote that both bits can take an arbitrary value; we refer to this one as a ‘trivial’ difference.

5.1 Forward characteristic on 2.5 rounds

The forward characteristic starts at round 3[5] and is based on the fact that the message word m_{13} is used in the first half of a column-step call in round 3, and is not used again before the second half of a diagonal step in round 4. Consequently, we can introduce a difference in m_{13} and cancel it immediately with a difference in v_2; no difference will be introduced again for 1.5 rounds.

If we note `MSB` an 'x' difference in the most significant bit, the initial differences in this characteristic are then `MSB` for m_{13} and v_2, and no difference in any other state or message word.

In the diagonal step of round 4, a difference is introduced in the state by the difference in m_{13}. This difference quickly propagates to every state word, but some non-trivial differences occur with probability one. After the column step of round 5, *i.e.,* at round 5.5, the state words for which there are non-trivial probability-one differences are listed below along with their differences (in the following, the leftmost constraint is for the MSB).
For `BLAKE2b` we have:

v_0:	`????????????????????????????????????????????????????????x-------`
v_3:	`????????????????????????????????????????x-----------------------`
v_7:	`???????????????????????????????????????????????????????x-------?`
v_{11}:	`????????????????????????????????????????????????????????x-------`
v_{12}:	`????????x-------????????????????????????????????????????????????`
v_{15}:	`----------------????????????????????????????????????????x-------`

For `BLAKE-512` we have:

v_0:	`?????????????????????????????????????????????????????????x------`
v_3:	`?????????????????????????????????????????x----------------------`
v_7:	`????x------?????????????????????????????????????????????????????`
v_{11}:	`?????????????????????????????????????????????????????????x------`
v_{12}:	`?????????x------????????????????????????????????????????????????`
v_{15}:	`----------------?????????????????????????????????????????x------`

For `BLAKE2s` and `BLAKE-256` we have:

v_0:	`????????????????????????????x---`
v_3:	`????????????????????x-----------`
v_7:	`???x---?????????????????????????`
v_{11}:	`????????????????????????????x---`
v_{12}:	`????x---????????????????????????`
v_{15}:	`--------????????????????????x---`

5.2 Backward characteristic on 3.5 rounds

The backward differential characteristic starts in the diagonal step of round 8. As we want to use this characteristic to mount a miss-in-the-middle with the previous forward characteristic, we need to use differences in the message words consistent with the ones used in the latter. Hence we use a single difference in the MSB of m_{13}. This message word is used in the second half of a $\texttt{G}^{-1}$ call in the inverse of the diagonal step of round 8, and is not used again before the second half of a $\texttt{G}^{-1}$ call in the inverse of a column step in round 7.

In order to delay the propagation of differences as much as possible, we want to proceed as for the forward characteristic and cancel the difference introduced by the message at round 8 by specifying an appropriate state difference. It is again possible to do so with probability one; in this case however, the difference will be somewhat more complex.
For `BLAKE2b`, we have the following initial differences in the state at the beginning of the *inverse* of round 8:

v_4 (input a to $\texttt{G}^{-1}$):	`x-------------------------------0-----------------------n-------`
v_9 (input b to $\texttt{G}^{-1}$):	`-------x-------x---------------x---------------x-------n--------`
v_{14} (input c to $\texttt{G}^{-1}$):	`--------n-------n---------------n1--------------n-------0-------`
v_3 (input d to $\texttt{G}^{-1}$):	`--------n-------n---------------00--------------n---------------`

One should note two things about this input difference. The first one is that the signed differences '`n`' can all be replaced together with a signed difference '`u`' of opposite sign: only important fact is that all differences are signed similarly. Moreover, some '`0`' and '`1`'constraints in the difference for v_3 and v_{14} are here to avoid a carry propagation in the update of c in $\texttt{G}^{-1}$, which is $c \leftarrow c - d$. However, this is a sufficient condition only, and the same result can be achieved by specifying

[5] We start indexing the rounds from 0, so as to match the indexing of the σ permutations.

alternative differences. In other words, these differences only make a subset of the state difference we were looking for. We do not specify the whole set in here, as the existence of a subset already serves our purpose.
Similarly, for BLAKE-512, we have the following state difference:

v_4 (input a to $\texttt{G}^{-1}$): `x-------------------------------0------------------------n------`
v_9 (input b to $\texttt{G}^{-1}$): `----n---------------x------x---------------x---------------x----`
v_{14} (input c to $\texttt{G}^{-1}$): `---------n------n---------------n1--------------n--------0------`
v_3 (input d to $\texttt{G}^{-1}$): `---------n------n---------------00--------------n---------------`

Finally, for BLAKE2s and BLAKE-256 we have:

v_4 (input a to $\texttt{G}^{-1}$): `x---------------0-----------n---`
v_9 (input b to $\texttt{G}^{-1}$): `---n-------x---x-------x-------x`
v_{14} (input c to $\texttt{G}^{-1}$): `----n---n-------n1------n---0---`
v_3 (input d to $\texttt{G}^{-1}$): `----n---n-------00------n-------`

As for the forward characteristic, the difference in m_{13} again introduces a difference that propagates to the rest of the state. However, due to the slower diffusion of $\texttt{G}^{-1}$ with respect to G, it is possible to keep non-trivial differences of probability one for more rounds. We then get the following differences after the inverse of the diagonal step of round 5, *i.e.,* at round 5.5 (only the differences occurring on state words for which there were non-trivial differences after the forward characteristic are listed here, but note that there were additional ones which are omitted here).
For BLAKE2b we have:

v_0: `????????????????????????????????????????????????x---------------`
v_3: `----------------????????????????????????????????x---------------`
v_7: `????????????????????????????????????????????????x---------------`
v_{12}: `????????????????????????????????????????????????x---------------`
v_{15}: `----------------------------------------------------------------`

In this case, the differences for BLAKE-512 are actually identical.
Similarly, for BLAKE2s and BLAKE-256, we have:

v_0: `????????????????????????x-------`
v_3: `--------????????????????x-------`
v_7: `????????????????????????x-------`
v_{12}: `????????????????????????x-------`
v_{15}: `--------------------------------`

5.3 Mounting the miss-in-the-middle

Now that we have established probability-one differences obtained at round 5.5 from two different characteristics, we show that these characteristics are incompatible. The result is immediate, when noticing that the differences on state words v_0, v_3, and v_{15} are incompatible for all four internal ciphers of BLAKE2b, BLAKE-512, BLAKE2s, and BLAKE-256, and the differences on word v_7 are further incompatible for BLAKE2b.

As one characteristic goes in the forward direction and one in the backward, inverse direction, this incompatibility consists in effect in a miss-in-the-middle which gives a 6-rounds impossible differential. This family of ID goes from round 3 to round 8, and is specified by the differences in the message word m_{13} and in the state v_2 (at round 3) and v_3, v_4, v_9, and v_{14} (at round 8) from the two families of characteristics presented above.

5.4 Extending by one more half-round

The 2.5-rounds forward characteristic used in the above can easily be extended for one more half-round for all the block ciphers of BLAKE2 and BLAKE, thereby increasing the number of rounds reached by the ID to 6.5. The extension works as follow.

First note that the message word m_{13} is not used in the diagonal step of round 2. Thus no difference will be introduced by the message words in that step. Second, we use one of the probability-one differential characteristics for G mentioned in [1]. This characteristic has no differences in the message word, and simply maps through G the state input difference (MSB, 0, MSB, MSB $\oplus$ (MSB $\lll$ r), 0, 0) to the state output difference (MSB, 0, 0, 0). It is straightforward to check that this happens with probability one, where r is 32 for BLAKE2b and BLAKE-512, and 16 for BLAKE2s and BLAKE-256. As the output difference of this characteristic is precisely the input difference of the forward characteristic used in the ID, it is therefore possible to join the two characteristics together. The initial differences of this new forward characteristic starting at round 2.5 are then MSB for m_{13}, v_2, and v_8, and MSB $\oplus$ (MSB $\lll$ r) for v_{13}, with all other words having no differences.

6 Differential Analysis

In this section we show differential attacks on BLAKE2. The target of our attacks would be the compression function and the hash function BLAKE2b only — the analysis applies to BLAKE2s however the number of attacked rounds is much smaller. To build high probability differential characteristics we expand the analysis of Guo and Matusiewicz [16] (see also [1]) and Dunkelman and Khovratovich [15] of BLAKE-256. In both of these papers, the difference is of a special rotational type and is chosen to cancel the effects of the rotations on 16, 12, 8, and 7 bits in the function G of BLAKE-256. The first authors note that among the four rotations in G, only the last one (on 7 bits) is not divisible by 4. Thus they choose to work with the difference 88888888 and analyze only the trails where before the last rotation the difference in b is 0. They linearize G, assume each modular addition involving differences has a probability of 2^{-7} (the difference in MSB saves one 2^{-1}), and with a computer search find that the best characteristic is on 4 rounds. Although their characteristic has rather high probability of 2^{-56}, they could not go more as no trails exist on higher number of rounds due to the condition that no difference enters the rotation on 7. The authors argue that one can consider the special case of difference entering this rotation resulting in 2× difference at the output, and then canceling in the next G function, but state that their experiments show that in this case the probability of the characteristics drops significantly. Dunkelman and Khovratovich choose to work with the difference 04040404 (the probability of modular addition increases to around 2^{-4}) and consider trails where no difference enters the rotation on 12 bits[6]. Moreover, they consider two additional type of differences obtained by multiplying the initial difference by 2 and 3 – this way they can allow difference in rotation on 7. The authors run a full search of round-reduced characteristics with all possible configurations for the difference in the state (*i.e.,* in each of the 16 words, the difference can be 0, 04040404, 08080808, 0c0c0c0c), and *no difference in the message words*. The characteristics they find are on more rounds, but have lower probability.

The new rotation amounts of 32,24,16 and 63 bits in the function G of BLAKE2b are very similar to the rotations from BLAKE-256. Hence we can apply the technique of finding round-reduced characteristics from the previous two papers by considering the 64-bit differences 0404040404040404 and 0004000400040004. We also use the following improvements for the search methods:

1. In the first search we work with $\delta =$ 0404040404040404 as well as with two additional differences 0808080808080808, 0c0c0c0c0c0c0c0c, that is the difference in the words can be $0, \delta, 2 \times \delta$, and $3 \times \delta$. This helps us to overcome the rotation on 63, *i.e.,*, instead of the condition that no difference enters $\ggg 63$ now we can allow δ to be at the input of this rotation which results in $2 \times \delta$ at the output.
2. In the second search we work with $\nabla =$ 0004000400040004 and again with two additional differences $2 \times \nabla, 3 \times \nabla$. As in the analysis from Dunkelman and Khovratovich, we require no difference at the input of rotation on 24 bits, but improve their search by considering two possibilities for the difference in each of the message words (instead of one: no difference).

The choice of 4 differences (instead of only 2) leads to the situation where in the modular addition, for the same input there are possibly several outputs. For example, $\delta + const$ can give both δ and $3 \times \delta$. Hence after the linearization, for fixed input differences for G, there can be several output differences. Dunkelman and Khovratovich note[7] that they get 276 possible differentials for G when the differences in a, b, c, d are one of the four, and there is no difference in the message words. As we allow the differences $0, \delta$ (or $0, \nabla$) in the messages (see below), in the first search we end up with 4531 differentials for G, and with 1192 in the second. There are 1024 possible input

[6] This type of trails were mentioned by Guo-Matusiewicz, but no detailed analysis was provided in [16].

[7] Guo and Matusiewicz work with only 2 difference, 0 and δ, thus modular additions in G are uniquely determined and for each input they get a single output.

differences (each of a, b, c, d can take 4 different values, while the message words can take 2), hence we get that on average in the first search, we have 4 outputs per single input, while only one in the second. In theory (without taking into account the probabilities) this results in around $2^{2 \cdot 8} = 2^{16}$ outputs for the whole round that can be obtained from a single input in the first search, while in the second this number is 1. Thus to keep the first search practical we cannot have too many input differences. We note that the probabilities of the differentials[8] range from 2^{-8} to 2^{-75} in the first search, and 2^{-4} to 2^{-36} in the second.

In both of our searches we try to maximize the number of starting differences in the state and in the message words. We can do this up to a certain extend. For example, there are 16 message words, thus if we want to try all four possible starting differences, we will end up with $2^{16 \cdot 2} = 2^{32}$ starting points (without considering any difference in the state). To make the searches feasible, in certain cases we restrict the differences to only $0, \delta$ (or $0, \nabla$). Note that the initializations in BLAKE2 differs from BLAKE, and in particular no difference can be introduced in $v_8, v_9, v_{10}, v_{11}, v_{14}, v_{15}$. We follow strictly the definition of BLAKE2 and do not allow starting differences in any of these six words. As we will see further, this has a major impact on the maximal number of rounds the best characteristics can cover in the case of compression functions.

One final note on the message modification. In our searches we assume the attacker can always pass for free the modular additions that involve the message words in the function G, of the first round only. This is reasonable as he always controls the message and to pass these additions he needs to fix only a small amount of bits in the message words per active bit, and can use the remaining degrees of freedom in the message to go through the rest of the rounds probabilistically. Recall that in the first round all the message words are independent. More advanced message modification techniques might be available, however, as we do not know in advance the best characteristic, it is hard to predict which of the remaining modular additions in the first round can be passed for free. Using message modification anywhere but in the first round is very hard due to the condition on the fixed IV's, *i.e.,* once a state has been fixed in some middle round, the attacker should be sure that after going backwards the resulting initial state complies with the initialization, *i.e.,* has correct values for $v_8, v_9, v_{10}, v_{11}, v_{14}, v_{15}$.

We have run the second search (with the main difference $\nabla =$ 0004000400040004) and obtained the following results:

- For the hash function of BLAKE2b, when the difference in the message words can take any of the values $0, \nabla, 2 \times \nabla, 3 \times \nabla$ (in total $2^{16 \cdot 2} = 2^{32}$ starting differences), the best characteristic is only on 2 rounds and holds with probability 2^{-198}.
- For the compression function of BLAKE2b, when the difference in the chaining values and the counters can take $0, \nabla, 2 \times \nabla, 3 \times \nabla$, and the difference in the message words is 0 or ∇ (in total $2^{10 \cdot 2 + 16 \cdot 1} = 2^{36}$ starting differences), the best characteristics is on 3 rounds with probability 2^{-336}.

The first search (with the main difference $\delta =$ 0404040404040404) requires much more computational power as we are dealing with average forking on 4, *i.e.,* for each input of G there are 4 outputs. We had to optimize the code significantly in order to try all possible inputs. The outcome of this search is as follow:

- For the hash function of BLAKE2b, when the difference in the message words can take any of the values 0 or δ (in total 2^{16} starting differences), the best characteristic is on 3 rounds and holds with probability 2^{-344}.
- For the compression function of BLAKE2b, when the difference in the chaining values, the counters, and the message words can take $0, \delta$ (in total $2^{10 \cdot 1 + 16 \cdot 1} = 2^{26}$ starting differences), the best characteristics is on 4 rounds with probability $2^{-366.5}$.

Note that in both of the cases (hash and compression), the first search produced better characteristics. Moreover, note that although we have matched the number of attacked rounds in the case of compression function (both BLAKE2 and BLAKE have differentials on 4 rounds), the probability of the characteristic of BLAKE2 is only $2^{-366.5}$ whereas the best known characteristics for BLAKE hash function is of 2.5 rounds with probability 2^{-56}. Therefore, despite launching a search with much

[8] The probability of the trivial differential with zero input-output difference is 1.

higher number of starting differences, *the new initialization used in* BLAKE2 *significantly limits the freedom*[9] *of the attacker against this type of differentials attacks.* Thus the tweaked initalization seems to have much better security properties.

We are able to extend for one half round each of differentials for the compression and the hash function. In the case of former, we allow any difference in the last rotation on 63 bits (our search prohibits this, thus it was not able to find it). We end up with a differential characteristic on 4.5 rounds for the compression function of BLAKE2b that holds with probability $2^{-494.5}$ — see Appendix C for the details. Similarly, we can go for an additional half round for the hash of BLAKE2b. We get low probability characteristic, however by using neutral bits we can find a pair of messages that conform to the differential with a complexity of around 2^{480} hash function calls — in Appendix D we give the method to achieve this. Without the initalization limitations, we extend similar characteristics search to the permutation and obtain 5.5 round results with probability 2^{-928}, shown in Appendix E.

7 Conclusion

A comparison of the security of BLAKE2 and BLAKE against the attacks we have examined in this paper is given in Tbl. 2. Based on our findings we can deduce several important facts about the impact of the tweaks in BLAKE2:

Table 2. Comparison of the attacks on BLAKE2 and BLAKE.

Attack	BLAKE2			BLAKE		
	perm.	cf.	chosen IV	perm.	cf.	chosen IV
Rotational	12	-	7	-	-	-
Collisions with Internal Differentials	12	-	12	-	-	-
Near-Collisions	-	3	12	4	4	4
Weak-class of keys/preimages	12	-	12	-	-	-
Impossible differentials	6.5	-	-	6.5	-	-
Boomerangs	5.5	5.5	5.5	8	7	8
Differentials	5.5	4.5	12	4	4	4-6
Hash function differentials	3.5			2.5		

1. The absence of constants in the function G has a major impact on the basic building block, *i.e.,* the keyed permutation of BLAKE2, and this cipher can be fully attacked. We can launch a key recovery rotational attack on all 12 rounds of the permutation BLAKE2b with a high complexity, and a distinguisher based on internal differentials that holds for 2^{64} keys of BLAKE2b (2^{32} for BLAKE2s) based on a single query. Thus one should be careful when using this permutation in applications. Note that neither of these attacks is applicable to BLAKE.
2. The change of rotation amounts in BLAKE2b does matter against certain types of attacks. The differentials we have presented in section 6 are based in particular on the fact that all rotations are either divisible by 8 or are close to being divisible by 8 (*e.g.,* 63). In fact, the same search of differential characteristics applies to BLAKE2b and BLAKE-256, however the later is 256-bit while the former is 512-bit hash, and thus permits characteristics with lower probabilities.
3. In the initialization, omitting the double use of the counter, as well as *introducing constants* IV_i reduces the number of attacked rounds, *i.e.,* increases the security of the compression function. Note that in the differential attacks, we were able to match (and advance more) the number of rounds as in BLAKE only because we used a much more complex search of differential characteristics and we were dealing with 512-bit hash. For instance, if the initialization in BLAKE2 were the same as in BLAKE, most likely we could penetrate more rounds in the differential attack (we could not run the search for this version as it requires significant amount of

[9] No difference can be introduced in $v_8, v_9, v_{10}, v_{11}, v_{14}, v_{15}$.

computations). In fact, the new initialization is crucial as if one used the same as in BLAKE, then collisions (respectively partial-collisions) could be produced with only 2^{128} (respectively 2^{61}) compression function calls.

4. The complete absence of constants in G makes the security of the compression function highly dependent on the right choice of IV's (unsurprisingly, this is not the case of BLAKE). That is, even with the new initialization but different IV's, one can still launch attacks — see sections 3 and 4. The 'weak' IV's on the other hand are highly structured (either rotational, all equal, or some particular values). The random choice of IV's as in BLAKE2 makes these weaknesses impossible to exploit.

To summarize, *based on our results*, we have shown that the tweaks introduced by BLAKE2, if analyzed separately, may reduce the security of the version in certain theoretical attack frameworks. However, taken together the tweaks do not have a significant impact on the security of the hash/compression function, aside from the one round increase (resulting in a 3.5 round attack) against the hash function. Thus BLAKE2, similarly to its predecessor BLAKE, has a very high security margin against all known attacks.

Acknowledgements

The work in this paper is supported by the Singapore National Research Foundation Fellowship 2012 (NRF-NRFF2012-06).

References

1. Jean-Philippe Aumasson, Jian Guo, Simon Knellwolf, Krystian Matusiewicz, and Willi Meier. Differential and invertibility properties of BLAKE. In Hong and Iwata [17], pages 318–332.
2. Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and Raphael C-W Phan. SHA-3 proposal BLAKE, version 1.3, 2008. Available online at https://131002.net/blake/.
3. Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein. BLAKE2: simpler, smaller, fast as MD5. In *ACNS (to appear)*, Lecture Notes in Computer Science. Springer, 2013.
4. Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein. BLAKE2: simpler, smaller, fast as MD5 — version 2013.01.29, 2013. Available online at https://blake2.net.
5. Jean-Philippe Aumasson, Samuel Neves, Zooko Wilcox-O'Hearn, and Christian Winnerlein. The BLAKE2 website, May 2013. https://blake2.net.
6. Daniel J Bernstein. ChaCha, a variant of Salsa20. In *Workshop Record of SASC*, 2008.
7. Daniel J. Bernstein. The Salsa20 Family of Stream Ciphers. In Matthew J. B. Robshaw and Olivier Billet, editors, *The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 84–97. Springer, 2008.
8. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. The Keccak reference, January 2011. Available online at http://keccak.noekeon.org/.
9. Eli Biham and Orr Dunkelman. A framework for iterative hash functions - haifa. *IACR Cryptology ePrint Archive*, 2007:278, 2007.
10. Alex Biryukov, Ivica Nikolic, and Arnab Roy. Boomerang Attacks on BLAKE-32. In Antoine Joux, editor, *FSE*, volume 6733 of *Lecture Notes in Computer Science*, pages 218–237. Springer, 2011.
11. Christophe De Cannière and Christian Rechberger. Finding SHA-1 characteristics: General results and applications. In Xuejia Lai and Kefei Chen, editors, *ASIACRYPT*, volume 4284 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006.
12. Shu-jen Chang, Ray Perlner, William E. Burr, Meltem Sönmez Turan, John M. Kelsey, Souradyuti Paul, and Lawrence E. Bassham. Third-Round Report of the SHA-3 Cryptographic Hash Algorithm Competition. *NIST Interagency Report*, 7896, 2012.
13. Itai Dinur, Orr Dunkelman, and Adi Shamir. Self-differential cryptanalysis of up to 5 rounds of SHA-3. *IACR Cryptology ePrint Archive*, 2012:672, 2012.
14. Itai Dinur, Orr Dunkelman, and Adi Shamir. Collision attacks on up to 5 rounds of SHA-3 using generalized internal differentials. In *FSE*, 2013.
15. Orr Dunkelman and Dmitry Khovratovich. Iterative differentials, symmetries, and message modification in BLAKE-256. In *ECRYPT2 Hash Workshop*, 2011.
16. Jian Guo and Krystian Matusiewicz. Round-reduced near-collisions of BLAKE-32. In *WEWoRC*, 2009. Available via http://guo.crypto.sg/blake-col.pdf.

17. Seokhie Hong and Tetsu Iwata, editors. *Fast Software Encryption, 17th International Workshop, FSE 2010, Seoul, Korea, February 7-10, 2010, Revised Selected Papers*, volume 6147 of *Lecture Notes in Computer Science*. Springer, 2010.
18. Dmitry Khovratovich and Ivica Nikolic. Rotational cryptanalysis of ARX. In Hong and Iwata [17], pages 333–346.
19. Ji Li and Liangyu Xu. Attacks on Round-Reduced BLAKE. *IACR Cryptology ePrint Archive*, 2009:238, 2009. `https://eprint.iacr.org/2009/238`.
20. Thomas Peyrin. Improved differential attacks for ECHO and Grøstl. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 370–392. Springer, 2010.

A Additional Specification Parameters of BLAKE2

We give here the definition of the inverses $\texttt{G}^{-1}$ of the $\texttt{G}$ functions, as part of our analysis make use of $\texttt{G}^{-1}$. With notations similar as in section 2, the $\texttt{G}^{-1}$ function of BLAKE2b $\texttt{G}^{-1}(a, b, c, d)$ is defined as:

$$\begin{array}{ll} 1 : b \leftarrow (b \lll 63) \oplus c & 5 : b \leftarrow (b \lll 24) \oplus c \\ 2 : c \leftarrow c - d & 6 : c \leftarrow c - d \\ 3 : d \leftarrow (d \lll 16) \oplus a & 7 : d \leftarrow (d \lll 32) \oplus a \\ 4 : a \leftarrow (a - m_j) - b & 8 : a \leftarrow (a - m_i) - b \end{array} .$$

The $\texttt{G}^{-1}$ function of BLAKE2s $\texttt{G}^{-1}(a, b, c, d)$ is defined as:

$$\begin{array}{ll} 1 : b \leftarrow (b \lll 7) \oplus c & 5 : b \leftarrow (b \lll 12) \oplus c \\ 2 : c \leftarrow c - d & 6 : c \leftarrow c - d \\ 3 : d \leftarrow (d \lll 8) \oplus a & 7 : d \leftarrow (d \lll 16) \oplus a \\ 4 : a \leftarrow (a - m_j) - b & 8 : a \leftarrow (a - m_i) - b \end{array} .$$

Table 3. Permutations of {0,. . ., 15} used in the BLAKE and BLAKE2 families.

σ_0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
σ_1	14	10	4	8	9	15	13	6	1	12	0	2	11	7	5	3
σ_2	11	8	12	0	5	2	15	13	10	14	3	6	7	1	9	4
σ_3	7	9	3	1	13	12	11	14	2	6	5	10	4	0	15	8
σ_4	9	0	5	7	2	4	10	15	14	1	11	12	6	8	3	13
σ_5	2	12	6	10	0	11	8	3	4	13	7	5	15	14	1	9
σ_6	12	5	1	15	14	13	4	10	0	7	6	3	9	2	8	11
σ_7	13	11	7	14	12	1	3	9	5	0	15	4	8	6	2	10
σ_8	6	15	14	9	11	3	0	8	12	2	13	7	1	4	10	5
σ_9	10	2	8	4	7	6	1	5	15	11	9	14	3	12	13	0

B The Proof of Lemma 1

The lemma claims that we can find the solution of

$$X \oplus [((X + A) \oplus X \lll 1) + B] = (X + A) \ggg 24 \tag{18}$$

in 2^{25} time, for the 64-bit words X, A, B. We achieve this by eliminating the rotation at the right side.

Let $X = x_{63} \ldots x_0, A = a_{63} \ldots a_0, B = b_{63} \ldots b_0, (X + A) \ggg 24 = H = h_{63} \ldots h_0$, be the bit values of the words of the equation. We start by guessing the 24 bits $x_{47} \ldots x_{24}$ and the carry at position 24 from the addition $(X + A)$ (this process will be repeated for all possible values), hence we can determine the value of $h_{23} \dots h_0$. From (18) we can immediately find $x_{63} = a_0 \oplus b_0 \oplus h_0$. Let us try to solve (18) for the first 24 bits of X, *i.e.,* let us find the solution of the equation:

$$X' \oplus [((X' + A) \oplus X' \lll 1) + B] = H', \tag{19}$$

on 24-bit words $X' = x_{23} \dots x_0$, etc. This is a T-function and a solution can be found efficiently by solving it from the least to the most significant bit. Further we sketch the method. By analyzing the above equation for bit position 1 we can determine the value of x_0 as well as the values of the carries r_0, t_0 from the modular additions $(X' + A), ((X' + A) \oplus X' \lll 1) + B$. The condition on the bit i can be expressed as:

$$x_1 \oplus x_1 \oplus a_1 \oplus r_0 \oplus x_0 \oplus b_1 \oplus t_0 = h_i, \tag{20}$$

where $r_0 = x_0 a_0, t_0 = (x_0 \oplus a_0 \oplus x_{63}) b_0$. Thus (20) becomes:

$$x_0(1 \oplus a_0 \oplus b_0) = a_0 b_0 \oplus x_{63} b_0 \oplus h_0 \tag{21}$$

Depending on the values of a_0, b_0, h_0, x_{63} the above equation can have no solution, one or two solutions. If there is at least one solution, we store into a table the values for x_0 and the values of the carries r_0, t_0. Further we move to the next bit, i.e. position 2, and obtain an equation similar to (20). When determining r_1, t_1 we take all possible values for r_0, t_0 from the table of stored values, and determine the set of possible values of x_1 and r_1, t_1 (and store them). This procedure is repeated for each bit. Obviously, the solution is linear in the number of bits (i.e. 24), hence we can easily find all solutions. On average there would be one solution (recall that at some bit positions there will be no solutions for (21)).

Once we find the value of X' determine the rest of the bit of X is trivial. Note that solving the initial equation (18) for the bits 24-47 is easy as we have already guessed the values of $x_{47}, \dots, x_{24}$ at the beginning. Hence from (18) we can determine the remaining 16 bits of X (the left side is constant), *i.e.,* $x_{63}, \dots, x_{48}$. Finally, we can determine an additional 24-bits of X on the right side and use it as a filter for incorrect solutions (or as soon as we determine all bits of X we immediately check if it is a solution for the equation). As this is 25-bit filter, after repeating the whole procedure for all possible $x_{47}, \dots, x_{24}$ and the carry, we will find on average 1 solution. Thus the total complexity is as claimed.

C An Example of 4.5-round Differential Characteristics for the Compression Function of BLAKE2b

Here we give the best differential characteristic on 4 rounds found by the first search. With the message modification in the first round, the probability of the characteristic is $2^{-45.8 \cdot 8} \approx 2^{-366.5}$. We can extend it for one additional half round to obtain a 4.5-round differential characteristic for the compression function of BLAKE2b with probability $2^{-61.8 \cdot 8} \approx 2^{-494.5}$. We use $1, 2, 3$ to denote the differences 0404040404040404, $2 \cdot$ 0404040404040404, $3 \cdot$ 0404040404040404, while the difference at output of v_5 (i.e. $2 \ggg 63$) is defined as $(2 \cdot 0404040404040404) \ggg 63$. The difference in the message words is $1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0$. Note that the characteristic starts at the first round (i.e. round 0), and goes for 4.5 rounds. We give the values of the differences after each half round. The numbers above the arrows denote the probability, while below, the differences in the message words used in the four G functions of the half round.

$$\text{ROUND 0:} \begin{pmatrix} 0&0&1&1\\ 1&0&0&0\\ 0&0&0&0\\ 1&0&0&0 \end{pmatrix} \xrightarrow[10\ 00\ 10\ 10]{2^{-2\cdot 8}\text{(with MM)}} \begin{pmatrix} 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 1&0&0&0 \end{pmatrix} \xrightarrow[00\ 00\ 00\ 00]{2^{-2\cdot 8}\text{(with MM)}} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&0&0&1\\ 0&0&0&0 \end{pmatrix}$$

$$\text{ROUND 1:} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&0&0&1\\ 0&0&0&0 \end{pmatrix} \xrightarrow[00\ 10\ 00\ 01]{2^{-4\cdot 8}} \begin{pmatrix} 0&0&0&0\\ 0&0&0&0\\ 0&0&0&1\\ 0&0&0&0 \end{pmatrix} \xrightarrow[00\ 10\ 00\ 00]{2^{-3\cdot 8}} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0 \end{pmatrix}$$

$$\text{ROUND 2:} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[00\ 01\ 00\ 00]{2^{-4\cdot 8}} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&1&0&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[00\ 01\ 00\ 01]{2^{-7\cdot 8}} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&1&0&1\\ 0&0&0&0 \end{pmatrix}$$

$$\text{ROUND 3:} \begin{pmatrix} 0&1&0&0\\ 0&0&0&0\\ 0&1&0&1\\ 0&0&0&0 \end{pmatrix} \xrightarrow[00\ 00\ 00\ 00]{2^{-6\cdot 8}} \begin{pmatrix} 0&1&0&1\\ 0&0&0&2\\ 0&0&0&0\\ 0&0&0&1 \end{pmatrix} \xrightarrow[01\ 00\ 11\ 00]{2^{-17.8\cdot 8}} \begin{pmatrix} 0&0&3&0\\ 2&2&2&0\\ 3&0&0&0\\ 1&2&1&1 \end{pmatrix}$$

$$\text{ROUND 4:} \begin{pmatrix} 0&0&3&0\\ 2&2&2&0\\ 3&0&0&0\\ 1&2&1&1 \end{pmatrix} \xrightarrow[01\ 00\ 01\ 00]{2^{-(6+2+5+3)\cdot 8}} \begin{pmatrix} 1&0&3&1\\ 0\ 2 \ggg 63&0&0\\ 2&0&2&1\\ 2&0&2&0 \end{pmatrix}$$

D Differential Attack on 3.5 rounds for the Hash Function of BLAKE2b

Below we present the best differential characteristic on 3 rounds (from round 6 to 9) found as well with the first search, that has a probability of $2^{-43\cdot 8} \approx 2^{-344}$ (using the message modification to pass for free some of the additions in the first round). We can extend it for one additional half round to obtain a 3.5-round characteristic for the hash function of BLAKE2b with probability $2^{-67.8\cdot 8} \approx 2^{-542.5}$. Although the probability is below 2^{-512} by using neutral bits in the first round we can find a pair that follows the characteristic. Note that the last G function in round 1, has no input/output differences. Hence, if we find a pair for the first 7 G functions of this round, then we can use the freedom in the two message words of the last G function, to produce another 2^{128} pairs for free. As for the remaining 3 rounds we need around 2^{480} pairs, we have to repeat the first round around $2^{480-128} = 2^{352}$ times. Thus the total complexity for find a pair that follows the whole 3.5 characteristic is $2^{352+(2+6)\cdot 8} + 2^{480} \approx 2^{480}$.

$$\text{ROUND 6:} \begin{pmatrix} 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[11\ 00\ 00\ 00]{2^{-2\cdot 8}\text{(with MM)}} \begin{pmatrix} 1&0&0&0\\ 0&0&0&0\\ 1&0&0&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[01\ 11\ 01\ 00]{2^{-6\cdot 8}\text{(with MM)}} \begin{pmatrix} 1&1&0&0\\ 0&0&0&0\\ 1&0&1&1\\ 0&0&0&0 \end{pmatrix}$$

$$\text{ROUND 7:} \begin{pmatrix} 1&1&0&0\\ 0&0&0&0\\ 1&0&1&1\\ 0&0&0&0 \end{pmatrix} \xrightarrow[00\ 10\ 10\ 10]{2^{-10\cdot 8}} \begin{pmatrix} 1&0&1&1\\ 0&0&0&0\\ 0&0&0&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[10\ 00\ 01\ 10]{2^{-6\cdot 8}} \begin{pmatrix} 0&0&1&0\\ 0&0&0&0\\ 1&0&0&0\\ 0&0&0&0 \end{pmatrix}$$

$$\text{ROUND 8:} \begin{pmatrix} 0&0&1&0\\ 0&0&0&0\\ 1&0&0&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[10\ 00\ 01\ 00]{2^{-7\cdot 8}} \begin{pmatrix} 1&0&1&0\\ 0&0&0&0\\ 0&0&1&0\\ 0&0&0&0 \end{pmatrix} \xrightarrow[11\ 01\ 00\ 01]{2^{-12\cdot 8}} \begin{pmatrix} 0&1&0&1\\ 2&0&2&2\\ 0&1&1&1\\ 1&1&1&0 \end{pmatrix}$$

$$\text{ROUND 9:} \begin{pmatrix} 0&1&0&1\\ 2&0&2&2\\ 0&1&1&1\\ 1&1&1&0 \end{pmatrix} \xrightarrow[01\ 00\ 11\ 01]{2^{-(7+4+6.8+7)\cdot 8}} \begin{pmatrix} 0&0&3&0\\ 2&0&2&2\\ 2&1&2&3\\ 3&0&3&1 \end{pmatrix}$$

E Differential Attack on 5.5 rounds for the permutation of BLAKE2b

Below, we present the 5.5-round characteristic for the permutation with probability 2^{-928}, from round 0 to 5. Note this characteristic is no longer under compression function limitation, e.g., differences can appear in any input state and message words. It is meaningful in a much broader setting, i.e., the underlining primitive is a 1024-bit block cipher with 1024-bit key. Under such setting, one has immediate distinguishers for a set of 2^{64} weak keys with full rounds, while this characteristic is useful for key recovery with no restriction on the key set.

$$\text{ROUND 0:}\ \begin{pmatrix}1&1&0&0\\0&1&0&1\\0&0&0&0\\0&1&0&1\end{pmatrix} \xrightarrow[\text{10 11 00 10}]{2^{-(1+3+0+3)\cdot 8}} \begin{pmatrix}0&1&0&0\\0&0&0&0\\0&1&0&0\\0&1&0&1\end{pmatrix} \xrightarrow[\text{00 10 00 10}]{2^{-(3+1+3+3)\cdot 8}} \begin{pmatrix}1&0&1&1\\0&0&0&0\\1&0&1&0\\0&0&0&0\end{pmatrix}$$

$$\text{ROUND 1:}\ \begin{pmatrix}1&0&1&1\\0&0&0&0\\1&0&1&0\\0&0&0&0\end{pmatrix} \xrightarrow[\text{11 00 00 01}]{2^{-(4+0+3+4)\cdot 8}} \begin{pmatrix}0&0&1&1\\0&0&0&0\\1&0&0&1\\0&0&0&0\end{pmatrix} \xrightarrow[\text{00 11 00 01}]{2^{-(0+4+3+4)\cdot 8}} \begin{pmatrix}0&0&1&1\\0&0&2&0\\0&1&0&1\\1&0&0&0\end{pmatrix}$$

$$\text{ROUND 2:}\ \begin{pmatrix}0&0&1&1\\0&0&2&0\\0&1&0&1\\1&0&0&0\end{pmatrix} \xrightarrow[\text{00 01 01 00}]{2^{-(3+3+6+3)\cdot 8}} \begin{pmatrix}1&0&1&1\\0&0&0&0\\1&1&1&0\\0&0&0&0\end{pmatrix} \xrightarrow[\text{11 11 00 00}]{2^{-(4+4+3+3)\cdot 8}} \begin{pmatrix}0&1&1&1\\0&0&0&0\\0&0&1&1\\0&0&0&0\end{pmatrix}$$

$$\text{ROUND 3:}\ \begin{pmatrix}0&1&1&1\\0&0&0&0\\0&0&1&1\\0&0&0&0\end{pmatrix} \xrightarrow[\text{00 10 00 01}]{2^{-(0+1+3+4)\cdot 8}} \begin{pmatrix}0&0&1&0\\0&0&0&2\\0&0&0&1\\0&0&0&1\end{pmatrix} \xrightarrow[\text{11 01 01 00}]{2^{-(2+3+6+0)\cdot 8}} \begin{pmatrix}0&0&1&0\\0&0&0&0\\1&0&0&1\\0&0&0&0\end{pmatrix}$$

$$\text{ROUND 4:}\ \begin{pmatrix}0&0&1&0\\0&0&0&0\\1&0&0&1\\0&0&0&0\end{pmatrix} \xrightarrow[\text{01 00 10 10}]{2^{-(3+0+1+3)\cdot 8}} \begin{pmatrix}0&0&0&1\\0&0&0&0\\1&0&0&0\\0&0&0&0\end{pmatrix} \xrightarrow[\text{10 00 10 10}]{2^{-(4+0+3+1)\cdot 8}} \begin{pmatrix}0&0&1&0\\0&2&0&0\\0&0&0&0\\0&0&0&1\end{pmatrix}$$

$$\text{ROUND 5:}\ \begin{pmatrix}0&0&1&0\\0&2&0&0\\0&0&0&0\\0&0&0&1\end{pmatrix} \xrightarrow[\text{10 11 10 01}]{2^{-(4+6+1+3)\cdot 8}} \begin{pmatrix}0&1&0&0\\2&0&0&2\\0&1&0&0\\1&0&0&1\end{pmatrix}$$