# Rotational Rebound Attacks on Reduced Skein

Dmitry Khovratovich[1,2], Ivica Nikolić[1], and Christian Rechberger[3]

[1]: University of Luxembourg; [2]: Microsoft Research Redmond, USA;
[3]: Katholieke Universiteit Leuven, ESAT/COSIC, and IBBT, Belgium
dkhovrat@microsoft.com, ivica.nikolic@uni.lu,
christian.rechberger@esat.kuleuven.be

**Abstract.** In this paper we combine a recent rotational cryptanalysis with the rebound attack, which results in the best cryptanalysis of Skein, a candidate for the SHA-3 competition. The rebound attack approach was so far only applied to AES-like constructions. For the first time, we show that this approach can also be applied to very different constructions. In more detail, we develop a number of techniques that extend the reach of both the inbound and the outbound phase, leading to cryptanalytic results on an estimated 53/57 out of the 72 rounds of the Skein-256/512 compression function and the Threefish cipher.
The new techniques include an analytical search for optimal input values in the rotational cryptanalysis, which allows to extend the outbound phase of the attack with a precomputation phase, an approach never used in any rebound-style attack before. Further we show how to combine multiple inside-out computations and neutral bits in the inbound phase of the rebound attack, and give well-defined rotational distinguishers as certificates of weaknesses for the compression functions and block ciphers.

**Keywords:** Skein, hash function, rotational cryptanalysis, rebound attack, distinguisher.

## 1 Introduction

Rotational cryptanalysis and the rebound attack proved to be very effective in the analysis of SHA-3 candidates and related primitives. Rotational cryptanalysis succeeded in the analysis of Addition-Rotation-XOR primitives (ARX), particularly in reduced variants of Threefish [10], Shabal [1], BMW [18]. Rebound attack, first presented in [16], is mostly aimed at byte-oriented primitives with a SPN structure. It gives the best attacks so far on reduced variants of the SHA-3 candidates Grøstl and ECHO [16,15], LANE [14], Cheetah [22] and the hash function Whirlpool [12], among others.

In this paper we introduce the combination of these two attacks with the application to the Skein compression function. We start with a number of preliminaries in Section 2. Our attacks will be based on methods to show non-random properties. For this we need definitions and bounds for distinguishers, which we give in Section 3. There we introduce the rotational collision set property for $n$-bit compression functions and ideal ciphers, and demonstrate a lower bound about $q \cdot 2^n$ for the complexity of finding such set of size $q$ in the black-box approach.

Then we proceed to the analysis of Skein and Threefish. We provide a much more careful and precise estimation of rotational probabilities compared to [10]. We represent the propagation of the rotational property analytically, and derive necessary conditions on the key bits to enlarge the rotational probability. We also correct [10] in terms of the independence assumptions, and find the best values of key bits with optimized search. Although we attack the tweaked version of Threefish [8], we stress that our attack is well applicable to the first version, and even benefits from more from the better diffusion the tweaked rotation constants provide.

This analysis gives us a simple rotational distinguisher for Threefish on up to 44 rounds. We advance even further and show how to put the rotational property into the outbound phase of the recent powerful *rebound attack*. The inner part of the rebound attack, the inbound phase, is accelerated with the method of the auxiliary path [9] and neutral bits [3]. In contrast to the first attacks on Skein, where auxiliary paths were used in the differential attacks, we show how to involve them into the rotational attack. As a result, we get a rotational distinguisher for the reduced Skein compression function. We attack 53 rounds of Skein-256 and 57 rounds of Skein-512 (Section 4).

Our results demonstrate substantial weaknesses both in the reduced Threefish cipher and the Skein compression function. The designers of Skein do not directly address the security of these primitives in the model that we consider, although the security of Threefish against all “standard attacks” is claimed. Also, our attacks show that the reduced Threefish does not behave as an ideal cipher, which is essential for the Skein security proofs. Had Skein have the reduced Threefish inside, the indifferentiability from the random oracle property of the Skein hash would be violated.

## 2 Preliminaries

### 2.1 Description of Skein

Skein is a family of hash functions, based on the block cipher Threefish of which the following versions are relevant for the SHA-3 proposal: Threefish-256 — 256-bit block cipher with 256-bit key and Threefish-512 — 512-bit block and key. Both the internal state $I$ and the key $K$ consist of $N_w$ ($N_w = 4, 8$ for Threefish-256,-512, respectively) 64-bit words. The $N_w$ words of the $s$-th subkey $K^s$ are

| Rounds | Attack | Method | Reference |
|---|---|---|---|
| Skein/Threefish-256 (72 rounds) | | | |
| $24^*$ | Key recovery | Related-key differential | [7] |
| 39 | Key recovery | Related-key rotational | [10] |
| 53 | Distinguisher | Rotational rebound | Section 4 |
| Skein/Threefish-512 (72 rounds) | | | |
| $25^*$ | Key recovery | Related-key differential | [7] |
| $33^*$ | Key recovery | Related-key boomerang | [5] |
| $35^*$ | Key recovery | Known-related-key distinguisher | [2] |
| 42 | Distinguisher | Related-key rotational | [10] |
| 57 | Distinguisher | Rotational rebound | Section 4 |

**Table 1.** Summary of the attacks on Skein and Threefish.

$^*$ — the attack was designed for the untweaked version.

defined as follows:

$$\begin{aligned} K_j^s &= K_{(s+j) \bmod (N_w+1)}, \quad 0 \le j \le N_w - 4; \\ K_{N_w-3}^s &= K_{(s+N_w-3) \bmod (N_w+1)} + t_{s \bmod 3}; \\ K_{N_w-2}^s &= K_{(s+N_w-2) \bmod (N_w+1)} + t_{(s+1) \bmod 3}; \\ K_{N_w-1}^s &= K_{(s+N_w-1) \bmod (N_w+1)} + s, \end{aligned}$$

where $s$ is a round counter, $t_0$ and $t_1$ are tweak words, and

$$t_2 = t_0 + t_1, \quad K_{N_w} = \lfloor 2^{64}/3 \rfloor \oplus \bigoplus_{j=0}^{N_w-1} K_j.$$

The formal description of internal rounds is as follows. Let $N_r$ be the number of rounds ($N_r = 72$ for Threefish-256,-512). Then for every $1 \le d \le N_r$

- If $d \bmod 4 = 1$ add a subkey by setting $I_j \leftarrow I_j + K_j^{d/4}$;
- For $0 \le j < N_w/2$ set $(I_{2j}, I_{2j+1}) \leftarrow \mathrm{MIX}((I_{2j}, I_{2j+1}))$;
- Apply the permutation $\pi$ on the state words.

At the end, a subkey $K^{N_r/4}$ is added. The operation MIX has two inputs $x_0, x_1$ and produces two outputs $y_0, y_1$ with the following transformation:

$$\begin{aligned} y_0 &= x_0 + x_1 \\ y_1 &= (x_1 \lll_{R_{(d \bmod 8)+1,j}}) \oplus y_0 \end{aligned}$$

The exact values of the rotation constants $R_{i,j}$ as well the permutations $\pi$ (which are different for each version of Threefish) can be found in [7]. We note that the rotation constants were changed in the Skein tweak [8], and we attack the new version although a similar analysis is applicable to the old version as well.

The compression function $F(H_{i-1}, M_i)$ of Skein is defined as:

$$F(H_{i-1}, M_i) = E_{H_{i-1},T_i}(M_i) \oplus M_i,$$

where $E_{K,T}(P)$ is the Threefish cipher, $H_{i-1}$ is the previous chaining value, $T_i$ is the tweak, and $M_i$ is the message block.

The best known analysis of Skein is rotational distinguishers on the underlying Threefish cipher [10], which attack 39 rounds of Skein-256 and 42 rounds of Skein-512 (see Table 1).

### 2.2 Rotational cryptanalysis

The main idea of the rotational cryptanalysis is to consider a pair of words where one is a rotation of the other. The $(X, \overleftarrow{X})$ is called a *rotational pair* [with a rotation amount $r$], where $\overleftarrow{X}$ the rotation of $X$ by $r$ bits to the left. A rotational pair is preserved by any bitwise transformation, particularly by the bitwise XOR and by any rotation. The probability that the rotational pair comes out of the addition is given by the following formula[6]

$$\mathbf{P}(\overleftarrow{x+y} = \overleftarrow{x} + \overleftarrow{y}) = \frac{1}{4}(1 + 2^{r-n} + 2^{-r} + 2^{-n}).$$

For large $n$ and small $r$ we get the following table:

| $r$ | $p_r$ | $\log_2(p_r)$ |
|---|---|---|
| 1 | 0.375 | $-1.415$ |
| 2 | 0.313 | $-1.676$ |
| 3 | 0.281 | $-1.831$ |

For $r = n/2$ the probability is close to $1/4$. The same holds for rotations to the right. When an addition of rotational inputs does not produce rotational outputs then we say that the addition produced a rotational error.

The use of constants can violate the rotational property. Yet, if the constants are rotational as well, then the property is preserved, i.e. if $C = \overleftarrow{C}$ then $\overleftarrow{X} \oplus C = \overleftarrow{X \oplus C}$.

Rotational analysis deals with constants by introducing rotational corrections in pairs of inputs:

$$(X, \overleftarrow{X}_{\text{modified}}).$$

Then the rotational path is constructed so that the pre-fixed corrections and the errors from the failed modular addition compensate the errors from the use of constants.

We stress that in order to apply the rotational attack for the full scheme, all its inputs must be rotational pairs [with corrections].

### 2.3 Rebound attack

The rebound attack [13,16] was described as a variant of differential cryptanalysis optimized to the cryptanalysis of hash functions, and at the same time can be seen as a high-level model for hash function cryptanalysis. So far it was mainly applied to AES-like constructions because of the simple way useful truncated differential characteristics can be found in them for a number of rounds.

The rebound attack is aimed to construct solutions for the most expensive part of a truncated differential trail. In the *inbound phase*, which covers only a few rounds, we construct solutions that connect low-weight input and output differences. In the *outbound phase* these solutions are propagated through the other rounds in both directions.

## 3 Rotational distinguishers

In order to convincingly argue that a particular attack algorithm indeed shows non-random behavior of a hash function or a compression function, we need to argue that an attacker with only a black-box access to an ideal primitive of the same domain and range is not able to produce the same behavior with the same or better effort and probability.

Next in this section, we define a basic rotational distinguisher with corrections and give bounds on complexity of the resulting problems. Any shortcut algorithm will have to beat those bounds in order to make a convincing case for an attack. To do this, we adapt two known distinguisher concepts. The q-multicollision distinguisher of [4] will be the basis for a rotational distinguisher with corrections fixed by the attacker.

### 3.1 Rotational distinguishers with fixed corrections

Due to the presence of counters, the rotational input pairs in Skein never convert to rotational output pairs. However, low-weight corrections applied to the input pairs, admit such a conversion:

$$\mathrm{Skein}(\overleftarrow{X} \oplus e) \overset{\mathbb{P}}{=} \overleftarrow{\mathrm{Skein}(X)}$$

, where Skein is the compression function $F$, with reasonably high probability. We say that $X$ is a *rotational collision* for function $f$, if

$$f(\overleftarrow{X}) = \overleftarrow{f}(X \oplus e).$$

When the rotational correction is not fixed, the rotational collision search complexity is given by an equivalent of the birthday paradox and is about $2^{n/2}$.

However, we provide a stronger distinguisher for the Skein compression function $F$, which asks for a set of rotational collisions with the same correction

$e$:

$$\begin{cases} \overleftarrow{F(X_1)} = F(\overleftarrow{X_1} \oplus e); \\ \overleftarrow{F(X_2)} = F(\overleftarrow{X_2} \oplus e); \\ \ldots \\ \overleftarrow{F(X_q)} = F(\overleftarrow{X_q} \oplus e). \end{cases}$$

Since the value of $e$ is defined from the first equation, each new rotational collision costs about $2^n$ for a random function, and less for the Skein compression function as we show in the further text.

However, we prove the advantage of our distinguisher in a more strong setting by taking into account the fact that the Skein compression function is built on a block cipher $E_K(P)$:

$$F(IV, M) = E_{IV}(M) \oplus M.$$

We admit corrections only in the IV, so a rotational collision is formulated as

$$\overleftarrow{F(IV, M)} = F(\overleftarrow{IV} \oplus e, \overleftarrow{M}) \iff$$
$$\iff \overleftarrow{E_{IV}(M)} \oplus \overleftarrow{M} = E_{\overleftarrow{IV} \oplus e}(\overleftarrow{M}) \oplus \overleftarrow{M} \Leftrightarrow \overleftarrow{E_{IV}(M)} = E_{\overleftarrow{IV} \oplus e}(\overleftarrow{M}).$$

Thus the appropriate definition is as follows.

**Definition 1.** *A set*

$$\{e;\ (P_1, K_1),\ (P_2, K_2), \ldots, (P_q, K_q))\}$$

*is called* a rotational $q$-collision set *for a cipher* $E_K(\cdot)$ *if*

$$\begin{cases} \overleftarrow{E_{K_1}(P_1)} = E_{\overleftarrow{K_1} \oplus e}(\overleftarrow{P_1}); \\ \overleftarrow{E_{K_2}(P_2)} = E_{\overleftarrow{K_2} \oplus e}(\overleftarrow{P_2}); \\ \ldots \\ \overleftarrow{E_{K_q}(P_q)} = E_{\overleftarrow{K_q} \oplus e}(\overleftarrow{P_q}). \end{cases}$$

We follow the line of the first attack on the full AES [4] and compare the problem of finding a rotational collision set for an ideal cipher with that for reduced Threefish. Our results demonstrate that the versions of Threefish that we consider do not behave like an ideal cipher, and, thus, does not provide required security level for the Skein mode of operation (i.e., violate the random oracle property).

The complexity of the generic attack in measured in the number of queries to the encryption and decryption oracles of an ideal cipher.

**Lemma 1.** *To construct a rotational $q$-collision set for an ideal cipher with an $n$-bit block an adversary needs at least $O(q \cdot 2^{\frac{q-2}{q+2}n})$ queries on the average.*

*Proof.* The proof is similar to the proof of the multicollision lemma in [4]. We provide only a sketch of it.

First, we show that a rotational collision set is uniquely determined by $q+1$ query parameters. Then for any such set we compute the probability that it gives a collision set. The exact formula depends on the total number $L$ of queries and their configuration, but the lower bound is

$$L \geq O(q \cdot 2^{\frac{q-2}{q+2}n})$$

# 4 Rotational rebound attack on Skein

## 4.1 Overview

Our attack consists of three parts: an inbound phase, an acceleration phase, and an outbound phase. In the inbound phase we prepare enough rotational pairs of states for the outbound phase. The acceleration phase speeds up the outbound phase. An illustration of the attack proposal is given Fig. 1), while also given in Table 2.

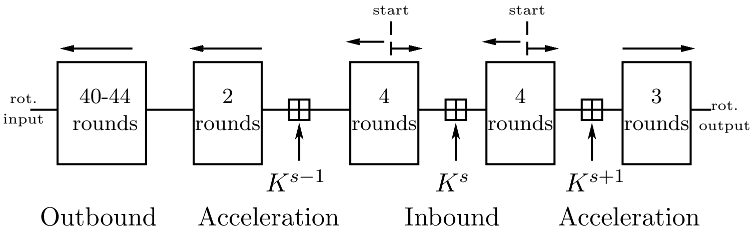


**Fig. 1.** The complete rotational rebound attack on Skein-256, -512.

**Table 2.** Structure of the rebound attack on Skein.

| Outbound | | Acceleration I | Inbound | Acceleration II |
|---|---|---|---|---|
| Rounds | Probability | Rounds | Rounds | Rounds |
| Skein-256 (53 rounds) | | | | |
| 3-42 | $2^{-244}$ | 43-44 | 45-52 | 53-55 |
| Skein-512 (57 rounds) | | | | |
| 3-46 | $2^{-495}$ | 47-48 | 49-56 | 57-59 |

The probability of the outbound phase depends on the values of particular key bits (see details in Section 4.4). As a result, we put global conditions on the keys, which are given in Tables 3 and 4.

**Table 3.** Pre-fixed values of key bits in Skein-256. The middle 58 bits of $k_i$ coincide (with regard to the rotation) in related keys.

|  | $K_0$ | $K_1$ | $K_2$ | $K_3$ | $K_4$ |
|---|---|---|---|---|---|
| $K$ | 0111..10 | 0100..11 | 0011..10 | 0000..11 | 0101..01 |
| $\overleftarrow{K} \oplus e$ | 11..0011 | 00..1010 | 11..0110 | 00..1001 | 01..0011 |

**Table 4.** Pre-fixed values of key bits in Skein-512

|  | $K_0$ | $K_1$ | $K_2$ | $K_3$ | $K_4$ | $K_5$ | $K_6$ | $K_7$ | $K_8$ |
|---|---|---|---|---|---|---|---|---|---|
| $K$ | 0111..01 | 0100..01 | 0011..01 | 0000..01 | 0111..10 | 0000..01 | 0011..01 | 0000..01 | 0001..10 |
| $\overleftarrow{K} \oplus e$ | 11..0011 | 00..0010 | 11..0010 | 00..0001 | 11..0011 | 00..0010 | 11..0010 | 00..0001 | 01..0101 |

For the distinguisher, we produce many $M$ and $K$, such that

$$E_{\overleftarrow{K}\oplus e}(\overleftarrow{M}) = \overleftarrow{E_K(M)},$$

where $E$ is the Threefish-256 reduced to rounds 2-54 (2-58 for the 512-bit version). For the Skein compression function, we produce many $M$, $IV$, and $T$ such that

$$F(\overleftarrow{IV||T} \oplus e, \overleftarrow{M}) = \overleftarrow{F(IV||T, M)}$$

for the same $e$. The total complexity is about $2^{244}$ per pair in Skein-256, and $2^{495}$ per pair in Skein-512. Therefore, we are able to construct a set of rotational collisions for the Skein compression function with complexity lower than for a random function. Also, we can construct a rotational $q$-collision set for the cipher Threefish with complexity lower than for an ideal cipher. This proves the relevance of our attack.

### 4.2 Inbound phase

The inbound phase can be seen as the inner loop of the attack algorithm. The goal is to use all degrees of freedom available to efficiently provide enough starting points for the outbound part. The details depend on the variant of Skein considered, the choice of round key additions that are covered by the inbound phase, etc. In the following we describe the technique in a way that is independent of such details.

Let us consider 8 consecutive rounds. The addition of the round key $K^s$ in the middle will be our matching point. We enumerate a large set of internal states both before and after the round key addition such that (1) the expected rotational trail is followed in the 8 rounds, and (2) it is possible to compute a subkey $K^s$ that matches the global constraints set up for later phases of the attack, and connects those two internal states. In experiments we found that by simply forcing a part (less than a quarter of the bits) of the state to a particular value can lead to pairs following a rotational trail with probability 1 for 3-5 rounds in forward direction. For the inbound phase we actually need less. Two rounds in forward direction and backwards direction is enough for both chunks of 4 rounds we operate on independently. In addition, for those two rounds, many differentials exist that allow for manipulation of the outputs of those 4-round chunks in a way that resembles message modification techniques in MD5 or SHA-1 [21,20]. To connect those chunks of 4-round computations, we use the degrees of freedom in the choice of the subkey $K_s$. The global conditions

On the other hand, note that this does not fully determine the key yet, as the compression function also has a tweak input which serves as another source for degrees of freedom. This leaves some control over subkeys $K^{k+1}$ and $K^{k-1}$.

### 4.3 Acceleration phase

The acceleration phase of the attack may be seen as part of the inbound phase or part of the outbound phase. Technically, starting from here computations are done in an inside-out manner, yet remaining degrees of freedom are used to accelerate the search for right pairs in the outbound phase.

As soon as we get a right pair of computations for the inbound phase, we produce many more of them from the given one as follows. We follow the simple idea of neutral bits as e.g. applied in the analysis of SHA-0 and SHA-1 [3]. We view them as auxiliary path [9] (also formalized as tunnels or submarines in [11,19,17]) and apply the differences specified by the path to the key and the tweak.

The configuration of the auxiliary path for Skein-256 is given in Table 5. We apply the original path difference to the first execution of the pair, and the rotated path difference to the second execution.

We consider $\oplus$-differences here, so we have to take into account the fact that the tweak and the key are added by the modular addition. Therefore, we choose the difference so that the probability of the carry is low. However, since adjacent bits are often neutral as well, a carry bit may still preserve the rotational pair.

In Skein-256 we take various $\delta$ and apply the resulting auxiliary path $\mathcal{P}_\delta$ to the right pair. We choose $\delta$ so that the differences in the subkey $K^{12}$ compensate each other. Then we check whether the modular additions in rounds 43-44 and 53-55 are not affected by the modification. If so, we get another rotational pair for rounds 43-55.

**Table 5.** Configuration of the auxiliary path for Skein-256. $K_i$ is the $i$-th word of the first subkey $K^0$.

| Round | Subkey | Subkey words | | | |
|---|---|---|---|---|---|
| 45 | $K^{11}$ | $K_1$<br>0 | $K_2$<br>0 | $K_3$<br>$\delta$ | $K_4$<br>$\delta$ |
| | Tweak | Tweak words | | | |
| | $T^{11}$ | | $T_\oplus$<br>0 | $T_0$<br>$\delta$ | |
| 49 | $K^{12}$ | $K_2$<br>0 | $K_3$<br>$\delta$ | $K_4$<br>$\delta$ | $K_0$<br>0 |
| | $T^{12}$ | | $T_0$<br>$\delta$ | $T_1$<br>$\delta$ | |
| 53 | $K^{13}$ | $K_3$<br>$\delta$ | $K_4$<br>$\delta$ | $K_0$<br>0 | $K_1$<br>0 |
| | $T^{13}$ | | $T_1$<br>$\delta$ | $T_\oplus$<br>0 | |

In experiments, we found that 44 of the 64 possible individual bits that result in a local collision of the latter type behave neutral with probability larger than 0.75 for three rounds in forward direction and simultaneously two rounds in backwards direction, 37 consecutive bits of those have a probability very close to 1[1]. Details for this phase will be found in Appendix in Table 6. Overall, the results mean that every time those five rounds in the outbound phase are computed, and the effort of those is less than $2^{37}$, the amortized effort for those computations will be negligible. If the effort for those five rounds is more, the effect of this acceleration phase, the speed-up, still grows roughly exponential with the number of neutral bits used.

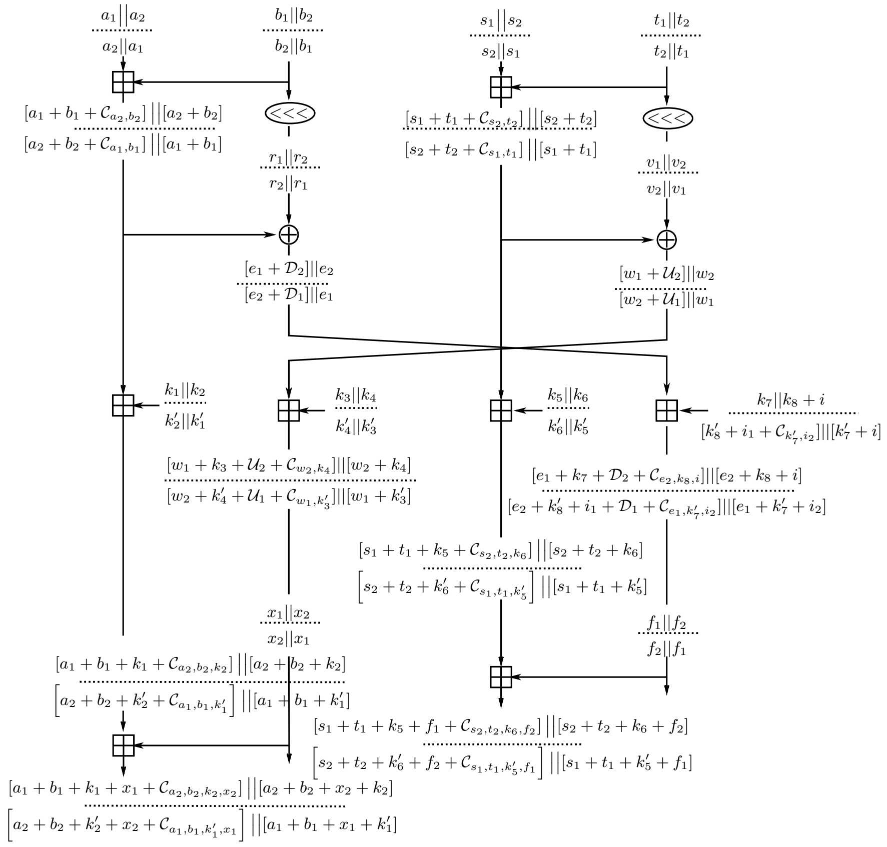
### 4.4 Outbound phase

We follow the idea of [10], and introduce corrections in the Threefish keys. But unlike [10], we consider modular corrections, i.e. we define the related-key pair by $(K, \overleftarrow{K} + e)$, where $e$ is a low-weight correction, "+" is modular addition, and the rotation amount is fixed to 2 to bypass the key schedule constant. Each 64-bit word $w$ in Skein can be seen as a concatenation of two words $w_1, w_2$, i.e.

[1] The fact that carries have to behave equivalently for round key additions in both forward and backward direction puts constraints on the inbound phase which are ignored here to keep the exposition simple. This either results in less degrees of freedom available to perform the exhaustive-search part of the attack, or reduces the number of possible combinations of neutral bits, and has to be taken into account in the overall estimate of the time complexity.

$w = w_1||w_2$ where $w_1$ represent the two most significant bits of $w$ and $w_2$ the rest 62 bits.

To obtain a high number of rounds in the outbound phase, we carefully choose optimal corrections and fix some of the key bits. More specifically, we found the best values of key bits with the optimized exhaustive search. Now we explain how to optimize the search in Skein-256 (Figure 2).



$a_1||a_2$
$a_2||a_1$
$b_1||b_2$
$b_2||b_1$
$s_1||s_2$
$s_2||s_1$
$t_1||t_2$
$t_2||t_1$
$<<<$
$<<<$
$[a_1 + b_1 + \mathcal{C}_{a_2,b_2}]\,||[a_2 + b_2]$
$[a_2 + b_2 + \mathcal{C}_{a_1,b_1}]\,||[a_1 + b_1]$
$[s_1 + t_1 + \mathcal{C}_{s_2,t_2}]\,||[s_2 + t_2]$
$[s_2 + t_2 + \mathcal{C}_{s_1,t_1}]\,||[s_1 + t_1]$
$r_1||r_2$
$r_2||r_1$
$v_1||v_2$
$v_2||v_1$
$[e_1 + \mathcal{D}_2]||e_2$
$[e_2 + \mathcal{D}_1]||e_1$
$[w_1 + \mathcal{U}_2]||w_2$
$[w_2 + \mathcal{U}_1]||w_1$
$k_1||k_2$
$k_2'||k_1'$
$k_3||k_4$
$k_4'||k_3'$
$k_5||k_6$
$k_6'||k_5'$
$k_7||k_8 + i$
$[k_8' + i_1 + \mathcal{C}_{k_7',i_2}]||[k_7' + i]$
$[w_1 + k_3 + \mathcal{U}_2 + \mathcal{C}_{w_2,k_4}]||[w_2 + k_4]$
$[w_2 + k_4' + \mathcal{U}_1 + \mathcal{C}_{w_1,k_3'}]||[w_1 + k_3']$
$[e_1 + k_7 + \mathcal{D}_2 + \mathcal{C}_{e_2,k_8,i}]||[e_2 + k_8 + i]$
$[e_2 + k_8' + i_1 + \mathcal{D}_1 + \mathcal{C}_{e_1,k_7',i_2}]||[e_1 + k_7' + i_2]$
$[s_1 + t_1 + k_5 + \mathcal{C}_{s_2,t_2,k_6}]\,||[s_2 + t_2 + k_6]$
$\left[s_2 + t_2 + k_6' + \mathcal{C}_{s_1,t_1,k_5'}\right]\,||[s_1 + t_1 + k_5']$
$x_1||x_2$
$x_2||x_1$
$f_1||f_2$
$f_2||f_1$
$[a_1 + b_1 + k_1 + \mathcal{C}_{a_2,b_2,k_2}]\,||[a_2 + b_2 + k_2]$
$\left[a_2 + b_2 + k_2' + \mathcal{C}_{a_1,b_1,k_1'}\right]\,||[a_1 + b_1 + k_1']$
$[s_1 + t_1 + k_5 + f_1 + \mathcal{C}_{s_2,t_2,k_6,f_2}]\,||[s_2 + t_2 + k_6 + f_2]$
$\left[s_2 + t_2 + k_6' + f_2 + \mathcal{C}_{s_1,t_1,k_5',f_1}\right]\,||[s_1 + t_1 + k_5' + f_1]$
$[a_1 + b_1 + k_1 + x_1 + \mathcal{C}_{a_2,b_2,k_2,x_2}]\,||[a_2 + b_2 + x_2 + k_2]$
$\left[a_2 + b_2 + k_2' + x_2 + \mathcal{C}_{a_1,b_1,k_1',x_1}\right]\,||[a_1 + b_1 + x_1 + k_1']$

**Fig. 2.** Rotational pair through two rounds with key addition of Skein-256.

We consider two rounds of Skein-256 with a subkey addition in between (rounds 4-5, 8-9, etc.). Note that the outer double rounds (6-7, 10-11, etc) simply keep the rotational pairs, so the probability does not depend on the number of round. The outer rounds probability is $2^{-8.5}$ for Skein-256 and $2^{-17}$ for Skein-512.

We denote the four words of the internal state before the double rounds by $(A, B, C, D)$. Therefore, we have

$$(A, B, C, D) = (a_1||a_2,\ b_1||b_2,\ s_1||s_2,\ t_1||t_2);$$
$$(\overleftarrow{A}, \overleftarrow{B}, \overleftarrow{C}, \overleftarrow{D}) = (a_2||a_1,\ b_2||b_1,\ s_2||s_1,\ t_2||t_1).$$

Similarly, we denote by

$$K = [k_1||k_2,\ k_3||k_4,\ k_5||k_6,\ k_7||k_8]; \quad \overleftarrow{K} \oplus e = [k'_2||k'_1,\ k'_4||k'_3,\ k'_6||k'_5,\ k'_8||k'_7].$$

the rotational pair of subkeys. Then the corrections $e_i$ can be defined as

$$e_i = k'_{2i+1}||k'_{2i+2} - k_{2i+1}||k_{2i+2}.$$

In Figure 2 the pairs are presented one a top of another with the symbol "- - - - - - -" between them. By $C_{z_1,\dots,z_k}$ we denote the carry from the sum $z_1+\dots+z_k$, i.e. when $z_i < 2^r$, then $C_{z_1,\dots,z_k} = (z_1+\dots+z_k) \ggg_r$. The variables $r, v, \mathcal{D}, \mathcal{U}, x, f$ are introduced to maintain the $2+62$ bit representation of the words. With $i = i_1||i_2$ we denote the round counter. Since the rotation preserves the rotational property, we can omit the rotations in the second round of the double subkey rounds, and only require rotational output pairs after the additions in this round. To obtain such pairs for the first output, the following conditions have to hold:

$$\begin{aligned} a_1 + b_1 + k_1 + x_1 + C_{a_2,b_2,k_2,x_2} &= a_1 + b_1 + x_1 + k_1^{'} \\ a_2 + b_2 + x_2 + k_2 &= a_2 + b_2 + k'_2 + x_2 + C_{a_1,b_1,k'_1,x_1} \end{aligned}$$

Similarly, for the rest 3 outputs, we get the following conditions:

$$\begin{aligned} w_1 + k_3 + \mathcal{U}_2 + C_{w_2,k_4} &= w_1 + k'_3 \\ w_2 + k_4 &= w_2 + k'_4 + \mathcal{U}_1 + C_{w_1,k'_3} \\ s_1 + t_1 + k_5 + f_1 + C_{s_2,t_2,k_6,f_2} &= s_1 + t_1 + k'_5 + f_1 \\ s_2 + t_2 + k_6 + f_2 &= s_2 + t_2 + k'_6 + f_2 + C_{s_1,t_1,k'_5,f_1} \\ e_1 + k_7 + \mathcal{D}_2 + C_{e_2,k_8,i} &= e_1 + k'_7 + i_2 \\ e_2 + k_8 + i &= e_2 + k'_8 + i_1 + \mathcal{D}_1 + C_{e_1,k'_7,i_2} \end{aligned}$$

The above 8 equations, can be reduced to:

$$k'_1 - k_1 = C_{a_2,b_2,k_2,x_2} \tag{1}$$
$$k'_2 - k_2 = -C_{a_1,b_1,k'_1,x_1} \tag{2}$$
$$k'_3 - k_3 = C_{w_2,k_4} + \mathcal{U}_2 \tag{3}$$
$$k'_4 - k_4 = -(C_{w_1,k'_3} + \mathcal{U}_1) \tag{4}$$
$$k'_5 - k_5 = C_{s_2,t_2,k_6,f_2} \tag{5}$$
$$k'_6 - k_6 = -C_{s_1,t_1,k'_5,f_1} \tag{6}$$
$$k'_7 - k_7 = C_{e_2,k_8,i} + \mathcal{D}_2 - i_2 \tag{7}$$
$$k'_8 - k_8 = i - i_1 - (C_{e_1,k'_7,i_2} + \mathcal{D}_1) \tag{8}$$

This system gives as a hint how to choose the corrections $e_i$ and the values of some of the subkey bits. For each carry $C_{z_1,\ldots,z_k}$ it holds $0 \leq C_{z_1,\ldots,z_k} < k$. Yet the probability that a carry will take a specific value in this range, when $z_i$ are randomly chosen, is not uniformly distributed. When the carries come from sums with 4 terms, the probability is highest for the values 1 and 2. Therefore, for our brute force, we limit the differences $k'_1 - k_1, k_2 - k'_2, k'_5 - k_5, k'_6 - k_6$, only to these two values.

The variables $\mathcal{U}_1, \mathcal{U}_2, \mathcal{D}_1, \mathcal{D}_2$, are determined as follows:

$$\begin{aligned}
\mathcal{U}_1 &= ((s_2 + t_2 + C_{s_1,t_1}) \oplus v_2) - ((s_2 + t_2) \oplus v_2)\\
\mathcal{U}_2 &= ((s_1 + t_1 + C_{s_2,t_2}) \oplus v_2) - ((s_2 + t_2) \oplus v_2)\\
\mathcal{D}_1 &= ((a_2 + b_2 + C_{a_1,b_1}) \oplus r_2) - ((a_2 + b_2) \oplus r_2)\\
\mathcal{D}_2 &= ((a_1 + b_1 + C_{a_2,b_2}) \oplus r_1) - ((a_1 + b_1) \oplus r_1)
\end{aligned}$$

These variables can take only odd values and a zero. Since $C_{w_2,k_4}$ can take $0, 1$ and $\mathcal{U}_2$ can take $0, 1$ it means that $k'_3 - k_3$ (see (3)) can also take 1 and 2 (the same values as the one for the subkeys discussed above). A similar reasoning is applicable to the difference $k_4 - k'_4$. The differences $k'_7 - k_7$, $k_8 - k'_8$ that are left, are the only one that actually depend on the round counter. Yet, since $C_{e_2,k_8,i}$ can take the values $0, 1$[2], i.e. it is not fixed but rather flexible, the whole expression $C_{e_2,k_8,i} + \mathcal{D}_2 - i_2$, for any $i_2$ can take the values $1, 2$ (recall that $\mathcal{D}_2$ can be any odd value). Therefore the difference $k'_7 - k_7$ can be 1 or 2 (with probability that depends on the round counter $i_2$). Finally, let us focus on the difference $k'_8 - k_8$ which is determined by the expression $i - i_1 - C_{e_1,k'_7,i_2} - \mathcal{D}_1$. For a specific counter $i$, when $k'_7 + e_2 = 0$, the carry $C_{e_1,k'_7,i_2}$ is fixed. Hence in this case, the whole expression can take only one value, 1 or 2, but not the both. This limits $k'_8 - k_8$ to only a single value.

Now recall that $k_i, k'_i$ are the values of the particular subkey words, and not the key words. Once we fix all of the differences in the subkey words of some round, then in the next round, practically the same differences will appear shifted by one index. Also, since the value of the difference in the last key word $K_4$ is determined from the other words, we would have to fix the values of $k_1, k_3, k_5, k_7$ and the two least significant bits of $k_2, k_4, k_6, k_8$ so that the difference in $K_4$ will be as expected. We fix only two bits because we choose the initial difference to be 1 or 2.

In our brute force search, first we find good values for the differences and the two most significant key bits of each key word. We try all possible differences 1 or 2, and then we fix the key bits values, such that the difference in the two most significant bits of $K_4$ will also be 1 or 2, and we take into account the limitation on $k'_8 - k_8$ for each counter. Then, we try all possible differences 1 and 2 in the least 62 bits of the each key word. We choose the differences that pass with highest probability through the double subkey rounds. Also, we fix the 2 least significant bits in each key word, so that the difference in the least 62 bits

[2] It can take the value 2 as well, but the probability is really low because the counter $i$ is only 4-5 bits.

of $K_4$ will also be 1 or 2. Finally, to increase the probability we fix the values of the bits 60,61 (the next two bits after the 2 most significant bits). This results in fixing the two most significant bits of $k_2, k_4, k_6, k_8$ which in return increases the probability that the carries take the expected values.

Rather than finding the above values through a theoretically small brute force, we have tested our approach on a real double subkey rounds Skein-256. That is, most of the values, were found and confirmed to be good by taking rotational input pairs of states and rotational input pair of key words with corrections and testing the probabilities on double subkey rounds. In some cases the theoretical probabilities did not coincide with the empirical. This is because there are some hidden dependencies. For example, both $\mathcal{U}_1$ and $k'_5 - k_5$ depend on $s_2, t_2$. Once we had the optimal corrections (and some bit values) of the keys for the double subkey rounds, we found the probability for 4 consecutive rounds. We start with a random rotational input pair of states and go through three rounds. Then we add the subkeys (with the particular counters) and then we go for an additional round.

We fix 6 bits in $K$: 4 MSBs and 2 LSBs, and 6 bits in $K_B$: 2 MSBs and 4 LSBs. The values of these bits are given at Table 3. In Skein-256 the probability to pass rounds 3–42 (i.e. 10 key additions) is $2^{-244}$. A detailed table with round-by-round probabilities is given at Table 7 of the Appendix.

Optimal values for the differences and some key bits can be obtained for Skein-512 as well. A property of the double subkey rounds Skein-512 that helps to run the brute force search is that these two double subkey rounds can be split into two non-intercepting halves (see Fig.3 in the Appendix). Then, for each half, the optimal differences can be found independently. Note that this simply speeds up the brute force for optimal differences and values, but has no impact on the actual probability of the inbound phase. Unlike Skein-256, in Skein-512 we could not find empirically the probabilities for 4 consecutive rounds because they were too low. Hence, we considered each 4 rounds as double round + double subkey round and simply multiplied the probabilities of these two. The values for the optimal 6 bits of each key word in Skein-512 are given in Table 4. In Skein-512 the probability to pass rounds 3–46 is about $2^{-494}$ (details in Table 8).

### 4.5 Probabilities in the Khovratovich-Nikolić analysis

The paper [10] provided the rotational analysis of Threefish on up to 42 rounds. The probability estimates were based on several independence assumptions, which must be corrected as follows:

– The probability of the rotational pair propagation through double rounds without key addition (2-3, 6-7, etc.) is not a multiplication of probabilities for a single round. The problem is that two consecutive modular additions $((a \boxplus b) \boxplus c)$ have lower rotational probability than expected. For example, the rotational probability of one round in Skein-256 is $2^{-3.35}$ for the rotation by 2, but the probability of two rounds is $2^{-8.52}$ instead of $2^{2\cdot(-3.35)} = 2^{-6.7}$.

- The rotational inputs to the round before the key addition (4, 8, etc.) are not uniformly distributed, and this partly compensates the negative effect of the dependency (see above). We note that the non-uniformity of inputs is best approximated with restricting the two most significant bits from the value {00}.
- The propagation of the rotational inputs through the double round with the key addition in Threefish-256, with the appearance and the correction of errors, can not be considered as two independent events (i.e., as getting rotational pairs in the further MIX operations independently). As a result, the probability of this event can not be computed as a multiplication of other probabilities, and must be computed as a single value.

### 4.6 Degrees of freedom analysis

Now we discuss the following question: How often can this inbound phase be repeated? After fixing the differences and the corrections, for Skein-256 we have

$$256 + 256 + 128 = 640$$

degrees of freedom available to perform the attack. The outbound phase fixes 24 of the 256 bits of the key, (also 12 bits of the 128-bit tweak), and in addition may need up to 256 bits to follow the longest possible trail with high probability. What remains is

$$640 - 36 - 256 = 348$$

degrees of freedom to be spent by the inbound and the acceleration phase. If variants with less rounds are targeted, this number is higher, as less repetitions are needed for the shorter outbound phase. Overall, this is enough for our purposes.

### 4.7 Summary and complexity estimates

We experimentally verified the probabilities of the outbound phase, and took various dependencies into account, and also experimentally verified parts of the acceleration and inbound phase.

Using the Skein-256 compression function as an example, we describe the resulting attack. As illustrated already in Fig. 1, the 8-round inbound part is performed close to the output of the cipher/compression function, the 5 round acceleration area (3 rounds in forward direction and 2 rounds in backward direction) surrounding it. The majority of the inside-out computation is then done in backwards direction, covering about 40 rounds. In total this gives about 53 rounds. Additionally, early stopping techniques will only require the computation of a small number of rounds in the outbound part before another trial is made, saving a factor of the computational complexity that is in the order of the number of rounds.

We estimate the amortized cost for the rounds covered by inbound and acceleration phase for both Skein-256 and Skein-512 by 1, as there are plenty of long ranging neutral bits that cover up costs in solving the right pairs in those inner rounds. In Skein-256, we will spend $2^{244}$ in the outbound+acceleration phases to find $2^{244}$ starting pairs for the outbound phase. One such pair will pass this phase with probability close to one. Therefore with an effort that is roughly equivalent to $2^{244}$ calls to the compression function of Skein-256 we can find one rotational pair of messages and chaining values (with corrections) that produces a rotational pair of updated chaining values. To produce $2^7$ such pairs, i.e. to find $2^7$-rotational collisions in Skein-256, we only need $2^{7+244} = 2^{251}$ calls. On the other hand, in a random function one has to make at least $2^7 \cdot 2^{\frac{128-2}{128+2}256} \approx 2^{255}$ calls (see Lemma 1).

Similarly, for the compression function of Skein-512, we can create $2^8$ rotational collisions with $2^{8+495} = 2^{503}$ compression function calls, while a random function would require $2^8 \cdot 2^{\frac{256-2}{256+2}512} \approx 2^{512}$ calls.

## 5 Conclusion and future work

Our results do not threaten the practical use of full-round Skein or Threefish. However, we show that these constructions behave non-random in settings where all or most inputs can be chosen, and this for more rounds than initially thought. We do not assume any other modifications. We argue that variants of Threefish reduced from 72 to about 53/57 rounds is not an ideal cipher in a similar way as AES-256 was shown not to be an ideal cipher in the first attack on AES [4]. For the Skein compression function a similar argument is made. Since Skein has a very light-weight output transformation, our non-randomness results can also carry over to the actual hash function. There, less degrees of freedom limit, but not prohibit, the applicability of some of our new techniques. To summarize, the following ideas and approaches lead to the improved results:

- The rebound approach as a high-level model for the attack.
- Considering rotational corrections with respect to integer addition instead of XOR
- Based on analytic reasoning, we find an efficient search method for fixing a subset of input bits before other phases of attacks.
- Using the degrees of freedom in the internal state to efficiently solve for the inner 8-rounds.
- Using the 8-round local collision as long-range neutral bits in an inside-out manner to speed up the outbound phase.

It will be interesting to study how rotational properties found in other constructions, some of which have been reported recently, can also be amplified in a way similar to what we demonstrated in this paper for Skein. The inbound and acceleration techniques we use in our analysis are to a large extent independent of the statistical property that is meant to be produced at the inputs and outputs of Skein. Hence, in addition to the rotational attacks described in this paper, also more traditional differential attacks aiming for collision or near-collision attacks will be able to take advantage of those techniques.

**Acknowledgements.** This work was sponsored in part by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and by the European Commission under contract ICT-2007-216646 (ECRYPT II). Ivica Nikolić is supported by the Fonds National de la Recherche Luxembourg grant TR-PHD-BFR07-031.

## References

1. G. V. Assche. A rotational distinguisher on Shabal's keyed permutation and its impact on the security proofs. Available online at `http://gva.noekeon.org/papers/ShabalRotation.pdf`, 2010.
2. J.-P. Aumasson, Çagdas Çalik, W. Meier, O. Özen, R. C.-W. Phan, and K. Varici. Improved cryptanalysis of Skein. In *ASIACRYPT'09*, volume 5912 of *Lecture Notes in Computer Science*, pages 542–559. Springer.
3. E. Biham and R. Chen. Near-collisions of SHA-0. In *CRYPTO'04*, volume 3152 of *Lecture Notes in Computer Science*, pages 290–305. Springer, 2004.
4. A. Biryukov, D. Khovratovich, and I. Nikolic. Distinguisher and related-key attack on the full AES-256. In *CRYPTO'09*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, 2009.
5. J. Chen and K. Jia. Improved related-key boomerang attacks on round-reduced threefish-512. Cryptology ePrint Archive, Report 2009/526, 2009.
6. M. Daum. *Cryptanalysis of Hash Functions of the MD4-Family.* PhD thesis, Ruhr-Universität Bochum, May 2005.
7. N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker. The Skein hash function family (2008). In *Submitted to SHA-3 Competition.*
8. N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker. The Skein hash function family - version 2. Submission to NIST (Round 2), 2009.
9. A. Joux and T. Peyrin. Hash functions and the (amplified) boomerang attack. In *CRYPTO'07*, volume 4622 of *Lecture Notes in Computer Science*, pages 244–263. Springer, 2007.
10. D. Khovratovich and I. Nikolić. Rotational cryptanalysis of ARX. In *FSE'10, to appear*, volume 6147 of *Lecture Notes in Computer Science*, pages 333–348. Springer, 2010.
11. V. Klima. Tunnels in hash functions: MD5 collisions within a minute. available at `http://eprint.iacr.org/2006/105.pdf`, 2006.
12. M. Lamberger, F. Mendel, C. Rechberger, V. Rijmen, and M. Schläffer. Rebound distinguishers: Results on the full Whirlpool compression function. In *ASIACRYPT'09*, volume 5912 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2009.
13. M. Lamberger, F. Mendel, C. Rechberger, V. Rijmen, and M. Schläffer. The Rebound Attack and Subspace Distinguishers: Application to Whirlpool. Cryptology ePrint Archive, Report 2010/198, 2010. `http://eprint.iacr.org/`.
14. K. Matusiewicz, M. Naya-Plasencia, I. Nikolic, Y. Sasaki, and M. Schläffer. Rebound attack on the full LANE compression function. In *ASIACRYPT'09*, volume 5912 of *Lecture Notes in Computer Science*, pages 106–125. Springer, 2009.

15. F. Mendel, T. Peyrin, C. Rechberger, and M. Schläffer. Improved cryptanalysis of the reduced Grøstl compression function, ECHO permutation and AES block cipher. In *Selected Areas in Cryptography'09*, volume 5867 of *Lecture Notes in Computer Science*, pages 16–35. Springer, 2009.
16. F. Mendel, C. Rechberger, M. Schläffer, and S. S. Thomsen. The rebound attack: Cryptanalysis of reduced Whirlpool and Grøstl. In *FSE'09*, volume 5665 of *Lecture Notes in Computer Science*, pages 260–276. Springer, 2009.
17. Y. Naito, Y. Sasaki, T. Shimoyama, J. Yajima, N. Kunihiro, and K. Ohta. Improved collision search for SHA-0. In *ASIACRYPT'06*, volume 4284 of *Lecture Notes in Computer Science*, pages 21–36. Springer, 2006.
18. I. Nikolić, J. Pieprzyk, P. Sokolowski, and R. Steinfeld. Rotational cryptanalysis of (modified) versions of BMW and SIMD. Available online at `https://cryptolux.org/mediawiki/uploads/0/07/Rotational_distinguishers_(Nikolic,_Pieprzyk,_Sokolowski,_Steinfeld).pdf`, 2010.
19. M. Stevens. On collisions for MD5. Master's thesis, Eindhoven University of Technology, Eindhoven, Netherlands, 2007.
20. X. Wang, Y. L. Yin, and H. Yu. Finding Collisions in the Full SHA-1. In V. Shoup, editor, *CRYPTO*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer, 2005.
21. X. Wang and H. Yu. How to Break MD5 and Other Hash Functions. In R. Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.
22. S. Wu. Semi-free start collision for 12-round Cheetah-256. NIST mailing list (local link), 2009.

# A Details

**Table 6.** Neutral bits in the acceleration phase. These are used in an inside-out manner, with those computations being 8 rounds apart. A single 64-bit word is used, enumeration is from 0 (LSB) to 63 (MSB). The probabilites are measured over 100 right pairs over two rounds backwards and three rounds forwards direction for Skein-256.

| bit | prob. | bit | prob. | bit | prob. | bit | prob. | bit | prob. | bit | prob. | bit | prob. | bit | prob. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $7-17$ | 1.00 | 18 | 0.99 | 19 | 1.00 | 20 | 0.99 | 21 | 1.00 | 22 | 0.99 | 23 | 1.00 | 24 | 0.99 |
| 25 | 0.95 | 26 | 0.94 | 27 | 0.93 | 28 | 0.82 | 31 | 0.79 | 33 | 0.86 | 36 | 0.77 | $38-45$ | 1.00 |
| 46 | 0.99 | 47 | 1.00 | 48 | 0.99 | 49 | 0.98 | 50 | 0.97 | 51 | 0.96 | 52 | 0.96 | 53 | 0.96 |
| 54 | 0.90 | 55 | 0.84 | | | | | | | | | | | | |

**Table 7.** Round-by-round rotational probabilities for Skein-256

| Rounds | 1-2 | 3-5 | 6-9 | 10-13 | 14-17 | 18-21 |
|---|---|---|---|---|---|---|
| Prob. $log_2$ | – | $-15.13$ | $-21.97$ | $-21.84$ | $-24.44$ | $-24.69$ |
| Rounds | 22-25 | 26-29 | 30-33 | 34-37 | 38-41 | 42 |
| Prob. $log_2$ | $-23.83$ | $-26.09$ | $-23.44$ | $-31.75$ | $-27.09$ | $-3.3$ |

**Table 8.** Round-by-round rotational probabilities for Skein-512

| Rounds | 1-2 | 3 | 4-5 | 6-7 | 8-9 | 10-11 | 12-13 | 14-15 |
|---|---|---|---|---|---|---|---|---|
| Prob. $log_2$ | – | $-6.7$ | $-26.35$ | $-17.05$ | $-26.21$ | $-17.05$ | $-24.26$ | $-17.05$ |
| Rounds | 16-17 | 18-19 | 20-21 | 22-23 | 24-25 | 26-27 | 28-29 | 30-31 |
| Prob. $log_2$ | $-28.26$ | $-17.05$ | $-28.29$ | $-17.05$ | $-23.79$ | $-17.05$ | $-23.56$ | $-17.05$ |
| Rounds | 32-33 | 34-35 | 36-37 | 38-39 | 40-41 | 42-43 | 44-45 | 46 |
| Prob. $log_2$ | $-27.18$ | $-17.05$ | $-32.23$ | $-17.05$ | $-35.17$ | $-17.05$ | $-31.86$ | $-6.7$ |

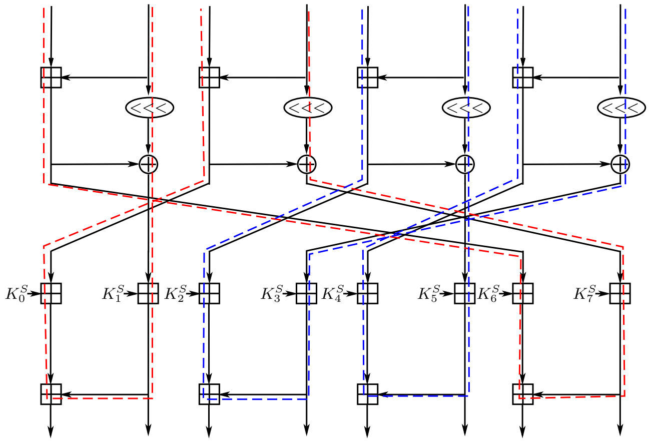


**Fig. 3.** Double subkey round in Skein-512 divided into two nonintersecting halves – red and blue.