Observations on the Shabal keyed permutation

Lars R. Knudsen Krystian Matusiewicz Søren S. Thomsen

April 7, 2009

Abstract

In this note we show that the permutation $\mathcal{P}$ used in the Shabal hash function, which is a candidate in the SHA-3 competition, has some non-random properties. As an example, it is easy to find a number of fixed points in the permutation. Moreover, large key-multi-collisions can be easily found; these are multi-collisions where only the key input contains a difference. All observations are easily verified, and most of them are independent of the choice of security parameters. Our observations, on the other hand, do not seem extensible to the full hash function.

1 Introduction

Shabal [3] is a candidate for the SHA-3 hash function competition organised by NIST [4]. In this note we describe some observations on the keyed permutation $\mathcal{P}$ used in Shabal. We note that $\mathcal{P}$ was already claimed to be non-ideal by Aumasson [1], but our observations are (apparently) different and easy to verify.

For a detailed description of Shabal, we refer to the Shabal specification [3]. We now briefly describe the permutation $\mathcal{P}$.

2 Description of the permutation

Shabal is a hash function based on a keyed permutation $\mathcal{P}$. We shall not describe the mode of operation of this keyed permutation, since our results only have to do with the permutation.

The permutation takes four inputs, where two of them are used as key material (hence, they are not updated). We denote by C and M two vectors of 16 32-bit words; these form the key material. Two vectors A and B, of 12 and 16 words respectively, are updated by the permutation.

The first thing that happens in the permutation is that each word of B is left-rotated by 17 positions. Hence, the following loop is carried out.

for $i = 0$ to 15 **do**
  $B[i] \leftarrow B[i] \lll 17$
end for

Then, an inner loop updates A and B as follows.

for $i = 0$ to 2 **do**
  for $j = 0$ to 15 **do**
    $A[j + 16i \bmod 12] \leftarrow \mathcal{U}(A[j + 16i \bmod 12] \oplus \mathcal{V}(A[j - 1 + 16i \bmod 12] \lll 15)$
      $\oplus C[8 - j \bmod 16])$
    $\oplus B[j + 13 \bmod 16] \oplus (B[j + 9 \bmod 16] \wedge \overline{B[j + 6 \bmod 16]})$
    $\oplus M[j]$

$B[j] \leftarrow (B[j] \lll 1) \oplus \overline{A[j + 16i \bmod 12]}$
end for
end for

Here, $\mathcal{U}$ maps x to $3x \bmod 2^{32}$, and $\mathcal{V}$ maps x to $5x \bmod 2^{32}$. A bar over a value means the bitwise complement of that value, and '$\wedge$' means logical AND.

Finally, A is updated by C as follows.

for $i = 0$ to 35 **do**
$A[i \bmod 12] \leftarrow A[i \bmod 12] + C[i + 3 \bmod 16]$
end for

The addition is modulo 2^{32}.

3 Observations

We describe how to find fixed points and other properties in the permutation.

3.1 Conserving the state

Choose arbitrary b, and compute $\tilde{b} = b \lll 17$. Choose $a = \overline{\tilde{b} \oplus (\tilde{b} \lll 1)}$. Now set $A[i] \leftarrow a$ for all i, $0 \le i < 12$, and set $B[i] \leftarrow b$ for all i, $0 \le i < 16$. Compute

$$M[i] \leftarrow a \oplus \tilde{b} \oplus \mathcal{U}(a \oplus \mathcal{V}(a \lll 15) \oplus C[8 - i \bmod 16])$$

for all i, $0 \le i < 16$.

Now, in the inner loop one gets $A[j + 16i \bmod 12] \leftarrow A[j + 16i \bmod 12]$, and due to the choice of a, one also gets $B[j] \leftarrow B[j]$. This happens for all (i, j). Hence, the inner loop has no effect on A and B.

Note that all words of A are equal. The same is true for B. If $b = 0$ or $b = -1 \bmod 2^{32}$, then the initial loop does not change B. In any case, all words in B remain equal throughout $\mathcal{P}$. If $C[i] = C[j]$ for all i, j, then all words in A also remain equal throughout $\mathcal{P}$.

With $b \in \{0, -1\}$ and $C[i] = 0$ for all i, we have a fixed point. Note that this technique works for any choice of the tunable security parameters, which are the number of words in A, and the number of rounds.

3.2 An extension

Instead of having all words in A and B equal, we can work with sets of four words as follows.

Set $b[i]$, $0 \le i < 4$, to arbitrary values. Compute $\tilde{b}[i] = b[i] \lll 17$. Set $a[i] \leftarrow \overline{\tilde{b}[i] \oplus (\tilde{b}[i] \lll 1)}$. Set $A[i] \leftarrow a[i \bmod 4]$ for all i, $0 \le i < 12$, and set $B[i] \leftarrow b[i \bmod 4]$ for all i, $0 \le i < 16$. Moreover, compute

$$\begin{aligned} M[i] \quad \leftarrow \quad & a[i \bmod 4] \oplus (\tilde{b}[i + 1 \bmod 4] \wedge \tilde{b}[i + 2 \bmod 4]) \oplus \\ & \mathcal{U}(a[i \bmod 4] \oplus \mathcal{V}(a[i - 1 \bmod 4] \lll 15) \oplus C[8 - i \bmod 16]) \end{aligned}$$

for all i, $0 \le i < 16$. Now, again, the inner loop preserves the state, regardless of the values of the $C[i]$ and the $b[i]$.

3.3 Key-collisions

It is clear that collisions in $\mathcal{P}$ are easy to find: choose a key (M, C) and an output (A^*, B^*), and compute the corresponding input (A, B) by inverting $\mathcal{P}$. Doing this for two different keys will produce a collision. However, one can also easily find collisions where only the key input is varied; the "plaintext" input (A, B) is fixed. Moreover, many keys producing the same output can be easily found, leading to what may be termed key-multi-collisions. We now describe how to do this.

The last loop in $\mathcal{P}$ can be described as $A \leftarrow A + \mathbf{D} \cdot C$, where $\mathbf{D}$ is a 12×16 matrix, and A and C are seen as column vectors. The following four vectors form a basis for the nullspace of $\mathbf{D}$.

$$(1, 0, 0, 0, -2, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0)^{\mathrm{T}}$$
$$(0, 1, 0, 0, 0, -2, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0)^{\mathrm{T}}$$
$$(0, 0, 1, 0, 0, 0, -2, 0, 0, 0, 1, 0, 0, 0, 1, 0)^{\mathrm{T}}$$
$$(0, 0, 0, 2, 0, 0, 0, 2^{31} - 1, 0, 0, 0, 2^{31} - 1, 0, 0, 0, 2^{31} - 1)^{\mathrm{T}}$$

(found using Magma [2]). Hence, there are many ways to choose a difference in C such that the mapping has no effect. This means that key-collisions can be found as follows.

Select A and B as described above (any choice of the values $b[i]$ can be used). Choose two values of the vector C, with a difference that is a non-zero linear combination of the four basis vectors above. Choose the message words $M[i]$ accordingly, as described above. Now there is a key-collision in $\mathcal{P}$.

In fact, we can find multi-collisions of size 2^{128} by keeping A and B fixed, and varying C in 2^{128} ways (the four basis vectors can each be varied in 2^{32} ways, all combinations being distinct).

Values of C whose differences are in the nullspace of $\mathbf{D}$ form an equivalence class. There are $2^{512-128} = 2^{384}$ such equivalence classes. Moreover, the four free words of B can be chosen in $2^{4\times 32} = 2^{128}$ different ways. Hence, we can construct a whopping total of $2^{384+128} = 2^{512}$ multi-collisions of size 2^{128}.

4 Examples

We now give a few examples of the observations described above.

4.1 Fixed points

Choose $B[i] = -1 \bmod 2^{32}$ and $A[i] = -1 \bmod 2^{32}$ for all valid i. Choose $C[i] = 0$ and $M[i] = 12$ for all i, $0 \leq i < 16$. Now, $\mathcal{P}_{M,C}(A, B) = (A, B) = (\mathsf{ff} \ldots \mathsf{ff}, \mathsf{ff} \ldots \mathsf{ff})$ (in hexadecimal). With $B[i] = 0$ and $M[i] = -13 \bmod 2^{32}$, we also have a fixed point.

4.2 Key-collisions

We can find key-collisions with the same choice of A and B as in the previous example. We may, for instance, choose $B[i] = -1$ and $A[i] = -1$. We choose, e.g., $C[i] = 0$ for all i except $i \in \{0, 4, 8, 12\}$, and we choose $C[0] = C[8] = C[12] = 1$, and $C[4] = -2$. We choose $M[i] = 12$ for all i except $i \in \{0, 4, 8, 12\}$, and we choose $M[0] = M[8] = M[12] = 15$, and $M[4] = -18$. Now, we again have $\mathcal{P}_{M,C}(A, B) = (A, B) = (\mathsf{ff} \ldots \mathsf{ff}, \mathsf{ff} \ldots \mathsf{ff})$. A third input that leads to the same output is $C[0] = C[8] = C[12] = -1$, $C[4] = 2$, $M[0] = M[8] = M[12] = -15$, $M[4] = 18$ (all other input words are unchanged).

5 Conclusion

We described some simple observations on the Shabal keyed permutation $\mathcal{P}$. These include simple methods of finding (many) fixed points in the permutations, and methods of constructing many large multi-collisions where only the key input contains a difference. We do not claim that these observations directly lead to attacks on Shabal.

References

[1] Jean-Philippe Aumasson. On the pseudorandomness of Shabal's keyed permutation. Available: `http://131002.net/data/papers/Aum09.pdf` (2009/04/03).

[2] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997.

[3] Emmanuel Bresson, Anne Canteaut, Benoît Chevallier-Mames, Christophe Clavier, Thomas Fuhr, Aline Gouget, Thomas Icart, Jean-François Misarsky, Marìa Naya-Plasencia, Pascal Paillier, Thomas Pornin, Jean-René Reinhard, Céline Thuillet, and Marion Videau. Shabal, a Submission to NIST's Cryptographic Hash Algorithm Competition. Available: `http://ehash.iaik.tugraz.at/uploads/6/6c/Shabal.pdf` (2009/04/01).

[4] National Institute of Standards and Technology. Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family. *Federal Register*, 27(212):62212–62220, November 2007. Available: `http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf` (2008/10/17).