Collisions and Preimages for Sarmal

Florian Mendel and Martin Schläffer

Institute for Applied Information Processing and Communications (IAIK)
Graz University of Technology, Inffeldgasse 16a, A-8010 Graz, Austria.

{florian.mendel,martin.schlaeffer}@iaik.tugraz.at

Abstract. In this paper, we show a collision attack on the hash function of Sarmal with different salt. The attack has a complexity of $2^{n/3}$ compression function evaluations and memory requirement of $2^{n/3}$. Since the salt of Sarmal is only 256 bits the attack works only for variants of Sarmal up to 384 bits. Note that we can choose the messages in our attack and hence we can even construct meaningful collisions for the hash function.
Furthermore, we show how to construct preimages for Sarmal faster than brute force search if the attacker can choose the salt value. The attack works for all output sizes of Sarmal and with a complexity of $2^{n/2+x}$ compression function evaluations and memory requirements of $2^{n/2-x}$ for $n = \{224, 256\}$, and $2^{n-128+x}$ compression function evaluations and memory requirements of 2^{128-x} for $n = \{384, 512\}$.

1 Description of Sarmal

The hash function Sarmal is an iterated hash function based on the HAIFA framework. It processes message blocks of 512 bits and produces a hash value of 224, 256, 384, or 512 bits. If the message length is not a multiple of 512, an unambiguous padding method is applied. For the description of the padding method we refer to [4]. Let $M = M_1 \| M_2 \| \cdots \| M_t$ be a t-block message (after padding). The hash value h is computed as follows:

$$\begin{aligned} H_0 &= IV \\ H_i &= f(H_{i-1}, M_i, S, i) \quad \text{for } 0 < i \leq t \\ h &= \mathtt{trunc}_n(H_t) \end{aligned}$$

where $\mathtt{trunc}_n$ denotes the truncation to n bits. IV is a predefined initial value and s is the salt of 256-bits. The compression function f of Sarmal basically consist of two streams α and β. Let M_i the i-th message block of 512 bits, $H_{i-1} = h_0 \| h_1$ the previous chaining value of 512 bits, $S = s_0 || s_1$ the 256-bit salt, and t the 64-bit block counter. Then the compression function f is computed as follows:

$$f(h_0 \| h_1, M_i, s_0 \| s_1, t) = \alpha(h_0, M_i, s_0, t) \oplus \beta(h_1, M_i, s_1, t) \oplus h_0 \| h_1 \qquad (1)$$

For a detailed description of α and β we refer to [4], since we do not need it for our attack.

2 Collision for Sarmal with different salt

In this section, we present a collision for the hash function Sarmal with different salt. For the sake of simplicity, we show how the collision attack for Sarmal works for a single message block. First, we choose two arbitrary different message blocks M_1 and M_1'. To get a collision we require that:

$$f(H_0, M_1, S, 1) \oplus f(H_0, M_1', S', 1) = 0$$

Using Equation (1) we get:

$$\begin{aligned}\alpha(h_0, M_1, s_0, 1) \oplus \beta(h_1, M_1, s_1, 1) \oplus h_0\|h_1 \oplus \alpha(h_0, M'_1, s'_0, 1) \oplus \beta(h_1, M'_1, s'_1, 1) \oplus h_0\|h_1 = \\ \alpha(h_0, M_1, s_0, 1) \oplus \beta(h_1, M_1, s_1, 1) \oplus \alpha(h_0, M'_1, s'_0, 1) \oplus \beta(h_1, M'_1, s'_1, 1) = 0\end{aligned}$$

Since h_0, h_1, M_1, and M'_1 are fixed in the attack, the above equation can be rewritten as:

$$u(s_0) \oplus v(s_1) \oplus w(s'_0) \oplus z(s'_1) = 0 \quad (2)$$

with

$$\begin{aligned}u(s_0) &= \alpha(h_0, M_1, s_0, 1)\\ v(s_1) &= \beta(h_1, M_1, s_1, 1)\\ w(s'_0) &= \alpha(h_0, M'_1, s'_0, 1)\\ z(s'_1) &= \beta(h_1, M'_1, s'_1, 1)\end{aligned}$$

In order to construct a collision for Sarmal with different salt, we have to solve equation (2). This can be done by using the generalized birthday attack [5]. Wagner shows that this system can be solved with a complexity of about $2^{n/3}$ computations and memory.

Note that we can independently choose 2^{128} values for each of s_0, s_1, s'_0, s'_1. Hence, this attack works only for variants of Sarmal with an output size up to $n = 128 \cdot 3 = 384$ bits. In other words, the attack is not applicable to Sarmal-512 at the moment. However, it can be used to construct collisions for Sarmal-224, Sarmal-256 and Sarmal-384.

Table 1. Summary of results.

	complexity	memory
Sarmal-224	$2^{74.7}$	$2^{74.7}$
Sarmal-256	$2^{85.4}$	$2^{85.4}$
Sarmal-384	2^{128}	2^{128}

By allowing differences in the chaining variables as well, we can construct pseudo-collisions for all output sizes of Sarmal, with $2^{n/3}$ computations and memory. Note that we can choose the messages in our attack and hence, we can even construct meaningful collisions for the hash function.

3 Preimage for Sarmal with chosen salt

In [3], Nikolic showed a preimage attack for Sarmal-512. The attack has a complexity of about 2^{384+x} and memory requirement of 2^{128-x}. However, to date no preimage attack on the other output sizes of Sarmal is known. In this section, we show how the preimage attack for Sarmal can be extended to all output sizes, if the attacker can choose the salt value in the preimage attack. The attack works similar as the collision attack. Suppose we seek a preimage of h for Sarmal. For the sake of simplicity, we show how the attack works for a single message block. First we choose an arbitrary message block M_1 with correct padding. To get a preimage we require that:

$$f(H_0, M_1, S, 1) = h$$

Using Equation (1) we get:

$$\alpha(h_0, M_1, s_0, 1) \oplus \beta(h_1, M_1, s_1, 1) \oplus h_0 \| h_1 = h$$

Since h_0, h_1, and M_1 are fixed in the attack, the above equation can be rewritten as:

$$u(s_0) \oplus v(s_1) = h \oplus h_0 \| h_1 = h^* \tag{3}$$

with

$$u(s_0) = \alpha(h_0, M_1, s_0, 1)$$
$$v(s_1) = \beta(h_1, M_1, s_1, 1)$$

In order to construct a preimage for Sarmal, we have to solve equation (3). This can be done by using a birthday attack. Since we assume in the attack that we can choose the salt, we can independently choose 2^{128} values for s_0 and 2^{128} values for s_1. Hence, we can construct preimages for Sarmal with all output sizes faster than brute force search. Since, the two salt values s_0 and s_1 are only 128 bits, we get only 2^{256} candidates for h in the birthday attack. In other words the attack succeeds only with a probability of 2^{-128} and 2^{-256} for Sarmal-384 and Sarmal-512, respectively. Hence, we have to repeat the attack 2^{128} and 2^{256} times with different choices for the message to find a preimage for Saraml-384 and Sarmal-512, respectively. The complexities and memory requirements for all output sizes of Sarmal are summarized in Table 2 with different time-memory trade offs. Note that more efficient memory-less variants of the attacks might be devised by using cycle-finding algorithms [1, 2].

Table 2. Summary of results with $x \leq 112$ for $n = 224$ and $x \leq 128$ for $n = \{256, 284, 512\}$.

	complexity	memory
Sarmal-224	2^{112+x}	2^{112-x}
Sarmal-256	2^{128+x}	2^{128-x}
Sarmal-384	2^{256+x}	2^{128-x}
Sarmal-512	2^{384+x}	2^{128-x}

References

1. Richard P. Brent. An improved Monte Carlo factorization algorithm. *BIT Numerical Mathematics*, 20(2):176–184, June 1980.
2. Robert W. Floyd. Nondeterministic Algorithms. *Journal of the Association for Computing Machinery*, 14(4):636–644, October 1967.
3. Ivica Nikolic. Preimage attack on Sarmal-512. Available online, 2008.
4. Kerem Varici, Onur Özen, and Çelebi Kocair. Sarmal: SHA-3 Proposal. Submission to NIST, 2008.
5. David Wagner. A Generalized Birthday Problem. In Moti Yung, editor, *CRYPTO*, volume 2442 of *LNCS*, pages 288–303. Springer, 2002.