A Minor Change to Lesamnta — Change of Round Constants —

Shoichi Hirose[1], Hidenori Kuwakado[2], Hirotaka Yoshida[3,4]

[1] University of Fukui
hrs_shch@u-fukui.ac.jp
[2] Kobe University
kuwakado@kobe-u.ac.jp
[3] Systems Development Laboratory, Hitachi, Ltd.,
hirotaka.yoshida.qv@hitachi.com
[4] Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC

1 Introduction

Lesamnta is a new family of hash functions submitted to NIST for their cryptographic hash algorithm competition.

In this document, we propose to make a minor change to the specification of Lesamnta by changing round constants. We give a short overview of the minor change to the Lesamnta hash function. We divide our arguments into the following categories:

- The minor change of the Lesamnta hash function
- The motivation of the change
- The design principle for the round constants
- Security against potential attacks on Lesamnta with new round constants
- The impact of the change

Note that we mainly explain about the minor change in the case of Lesamnta-256 because the explanation could be easily adapted to the case of Lesamnta-224/384/512.

2 The Minor Change

We propose to make a minor change to the specification of Lesamnta [2]. The minor change is only the replacement of 32 round constats. No other parts of the specification is changed.

For Lesamnta-224/256, we replace the 32 round constants described in Section 5.1.1 (page 14) of [2] with the following constants.

Round constants of [2]

```
0000000100000000 0000000300000002 0000000500000004 0000000700000006
0000000900000008 0000000b0000000a 0000000d0000000c 0000000f0000000e
0000001100000010 0000001300000012 0000001500000014 0000001700000016
0000001900000018 0000001b0000001a 0000001d0000001c 0000001f0000001e
0000002100000020 0000002300000022 0000002500000024 0000002700000026
0000002900000028 0000002b0000002a 0000002d0000002c 0000002f0000002e
0000003100000030 0000003300000032 0000003500000034 0000003700000036
0000003900000038 0000003b0000003a 0000003d0000003c 0000003f0000003e
```

New round constants

```
9e754700889cfedb 2db4ad503bbd6f80 02db4ad503bbd6f8 e1a70c522758bc4b
2a4989e511412ba9 1e95cf81bff8729e a8c416470af5c6d6 422bb32416c61cb6
4c85497227052110 04c8549722705211 fdf76aa9eba86421 f264994a0735e742
3744e7ab7dab9e3d 6f80451ae2875955 8b86b7ce8c169407 bda476dc1727489b
2f89be4df246d4e4 723dc79b6495eddc 966c38f97a9bdf6b 2d353aafa49d1d9b
2680aa8ac97d71b4 72ad56d717265789 1b1b82729f9e055c 90fe5ca7e52b61e3
ccd6a4153a051757 b9d177e1ac4670ae a2b05dc10bce26f5 8755b643328203fd
648150046675c089 1a79421fa88b3c2c 90e870a1365a3274 79cbdb75a8d423b5
```

For Lesamnta-384/512, we replace the 32 round constants described in Section 5.1.2 (page 15) of [2] with the following constants.

Round constants of [2]

```
00000000000000010000000000000000 00000000000000030000000000000002
00000000000000050000000000000004 00000000000000070000000000000006
00000000000000090000000000000008 000000000000000b000000000000000a
000000000000000d000000000000000c 000000000000000f000000000000000e
00000000000000110000000000000010 00000000000000130000000000000012
00000000000000150000000000000014 00000000000000170000000000000016
00000000000000190000000000000018 000000000000001b000000000000001a
000000000000001d000000000000001c 000000000000001f000000000000001e
00000000000000210000000000000020 00000000000000230000000000000022
00000000000000250000000000000024 00000000000000270000000000000026
00000000000000290000000000000028 000000000000002b000000000000002a
000000000000002d000000000000002c 000000000000002f000000000000002e
00000000000000310000000000000030 00000000000000330000000000000032
00000000000000350000000000000034 00000000000000370000000000000036
00000000000000390000000000000038 000000000000003b000000000000003a
000000000000003d000000000000003c 000000000000003f000000000000003e
```

New round constants

```
f6251864809494cd35cb7fa305acbe7f 78b114d45c0c003757aa6c4b9d98f1bf
b508148e2c0e460802e6cd2af27a24b0 ba220a9a4170d2de29fdd68d717f83f4
fa8e84753153428a0c9d29ba4c07bc9f 97fc92f852b9c3860d30da783d3f6b9d
95b68b70b22784abca19a58a8ca71e4c 48abbc03a30a7ff77422b58cdfd2a9ca
c7c5fa0d1976cfcbbfd178c3b7e94af7 f9c7bdd4fd083fedb7b7be15c8dcc1d3
dfc1d14920cdc088b5635cc6c7e5be34 37dcf3f822ec2133f52f774280cbc7e2
ed519add8adb45eae57e1d138887b7e1 eebfc9e5f47009f492d2f77813921014
159b340651e246363b85e6fe008b602c 2eb05b97b586d5603e4449f6e8e3f514
155b3b9423a3b0eaaf2970408e7011c9 c4acd4dbd5f51d7e0cb6c807b1a503ca
c749fd65c10030a936a9ecbe3c873d5d 58d1aa49ef6ae3f34a0cfccecddc475a
5f343b7343bca903289d46dd90e26da9 a27d71f052fa6d3232a61c086f06e116
17f09d2029b961fe360d4014031eb9db d7b2481063efc7658a41ae3d098b4854
514f4a4a1bc06c61cf87358938b8d9b4 be889af85ebc47add66113773567db05
05e3ea69155b31c85e13ac1129135b54 519d1be862b6d8976253678b149841a7
ac87ca0bc82b2705d736ec2f621c7828 2a47905563e447589bf95efede53f800
002a47905563e447589bf95efede53f8 f6e7f57d574abc562f1ea392b7ffb35b
```

The minor change of Lesamnata is only the above replcement. We will describe how to produce the new round constants in Sect. 4.2.

3 The Motivation for the Minor Change

The motivation why we propose to make a minor change to the specification of Lesamnta by changing round constants is to prevent the attacks described in [3] and one newly-discovered attack which will be described in Appendix A.1. These attacks are summarized in the following:

- Distinguishing attack on Lesamnta's block cipher [3]
  - An adversary can distinguish between Lesamnta-256's block cipher and the ideal cipher by making only two queries.
- Pseudo-collision attack on Lesamnta [3]
  - An adversary can produce a pseudo-collision of Lesamnta-256 with $O(2^{64})$ computations of the compression function.
- Semi-free start collision attack on Lesamnta [4]
  - An adversary can produce a semi-free start collision of Lesamnta-256 with $O(2^{64})$ computations of the compression function. It is a kind of pseudo-collision attack mentioned above.
- Attack using weak messages on Lesamnta
  - An adversary can find a kind of second preimage of Lesamnta-256 with $O(2^{128})$ computations of the compression function if the target message satisfies a certain property.

We observe that all the above attacks are based on some symmetry in the key scheduling function and the message mixing function of Lesamnta. To destroy the symmetry, we have chosen the new round constants. We expect that all the above attacks do not work any more for Lesamnta with the new round constants.

4 Design Principle for the New Round Constants

4.1 Condition for New Round Constants

We here show the condition allowing attacks described in Sect. 3. Since all the attacks uses it, they fail if it does not hold. We have confirmed that the new round constants do not satisfy the condition. The relationship between conditions and attacks are described in Appendix.

Let $\mathtt{C}[r][0]$ and $\mathtt{C}[r][1]$ be the left part and the right part of the r-th round constant in the key scheduling function of $EncComp_{256}$ or $EncComp_{512}$ (see Figs. 18, 28 of [2]). Note that $\mathtt{C}[r][i]$ is a 32-bit string for $EncComp_{256}$ and it is a 64-bit string for $EncComp_{512}$. We define a difference Δ_r as

$$\Delta_r = \mathtt{C}[r][0] \oplus \mathtt{C}[r][1] \tag{1}$$

for $r = 0, 1, \ldots, 31$. The condition is to satisfy all the following equations:

$$\begin{aligned} &\Delta_0 = \Delta_4 = \Delta_8 = \Delta_{12} = \cdots = \Delta_{24} = \Delta_{28}, \\ &\Delta_1 = \Delta_5 = \Delta_9 = \Delta_{13} = \cdots = \Delta_{25} = \Delta_{29}, \\ &\Delta_2 = \Delta_6 = \Delta_{10} = \Delta_{14} = \cdots = \Delta_{26} = \Delta_{30}, \\ &\Delta_3 = \Delta_7 = \Delta_{11} = \Delta_{15} = \cdots = \Delta_{27}. \end{aligned} \tag{2}$$

The condition is used for distinguishing Lesamnta's block ciphers from ideal ciphers. Furthermore, the distinguishing attack can be extended to the pseudo-collision attack and the weak-message attack.

4.2 Generators of New Round Constants

To be free of suspicion of a trapdoor, round constants must be determined in a transparent way. The new round constants for Lesamnta-256 were determined by the algorihtm of Fig. 1. The algorihtm of Fig. 1 is based on the linear feedback shift register (LFSR) of the following primitive polynomial $g(x)$.

$$\begin{aligned} g(x) = {} & x^{64} + x^{61} + x^{58} + x^{55} + x^{47} + x^{46} + x^{42} + x^{41} + x^{39} + x^{38} \\ & + x^{37} + x^{35} + x^{34} + x^{33} + x^{31} + x^{30} + x^{29} + x^{28} + x^{27} + x^{26} \\ & + x^{25} + x^{24} + x^{20} + x^{19} + x^{18} + x^{16} + x^{14} + x^{12} + x^{8} + x^{7} \\ & + x^{2} + x^{1} + 1. \end{aligned}$$

Due to 33 non-zero coefficients, almost half of bits of the internal state may be changed by one operation. Since there are many 33-term primitive polynomials, we adopted the first 33-term primitive polynomial obtained from the decimation of the M sequence produced by the LFSR of $x^{64} + x^{63} + x^{61} + x^{60} + 1$ with the all-one initial state.

Since the round constants $\mathtt{C}[r]$ can be considered as elements of GF(2^{64}), the above algorithm is equivalent to

$$\mathtt{C}[r] = \mathtt{C}[r-1] * \alpha^{J} \text{ over GF}(2^{64}) \text{ for } r = 1, 2, \ldots, 31$$

```
ConstantGenerator256(word C[Nr_comp256]) /* Nr_comp256=32 */
begin
  word c = ffffffffffffffff /* in hexadecimal */
  for i = 0 to Nr_comp256*J-1  /* J = 4 */
   /* Galois-type LFSR */
   if c ∧ 0000000000000001 = 0000000000000001
     c = (c >> 1) ⊕ e18ab8ff77630124
   else
     c = c >> 1
   end if
   if i mod J = 0
     C[i/J] = c
   end if
  end for
end
```

Fig. 1. Pseudocode for generating round constants of Lesamna-224/256.

where α is a root of $g(x)$. We chose $J = 4$. If $J \leq 3$, then there exists an initial state such that almost all $\mathtt{C}[r]$'s satisfy Condition 1.

The new round constants for Lesamnta-384/512 were determined in a similar manner. Specifically, the following 65-term primitive polynomial $g(x)$ and $J = 8$ were used.

$$\begin{aligned} g(x) = {} & x^{128} + x^{124} + x^{121} + x^{120} + x^{119} + x^{117} + x^{116} + x^{114} + x^{112} \\ & + x^{111} + x^{110} + x^{107} + x^{106} + x^{105} + x^{104} + x^{101} + x^{100} + x^{98} \\ & + x^{97} + x^{95} + x^{94} + x^{93} + x^{92} + x^{91} + x^{90} + x^{89} + x^{87} + x^{86} \\ & + x^{84} + x^{82} + x^{81} + x^{79} + x^{78} + x^{76} + x^{74} + x^{73} + x^{70} + x^{69} \\ & + x^{66} + x^{64} + x^{63} + x^{60} + x^{58} + x^{54} + x^{53} + x^{51} + x^{48} + x^{39} \\ & + x^{37} + x^{36} + x^{35} + x^{32} + x^{31} + x^{30} + x^{29} + x^{28} + x^{26} + x^{23} \\ & + x^{21} + x^{18} + x^{17} + x^{15} + x^{9} + x^{8} + 1 \end{aligned}$$

The pseudocode for generating the round constants is shown in Fig. 2. Since there are many 65-term primitive polynomials, we adopted the first 65-term primitive polynomial obtained from the decimation of the M sequence produced by the LFSR of $x^{128} + x^{127} + x^{126} + x^{121} + 1$ with the all-one initial state.

5 Security against Potential Attacks on Lesamnta with New Round Constants

We here consider a hash function family $\mathtt{D}$-Lesamnta which is the same as Lesamnta expept that the round constants $\mathtt{C}[r]$ are replaced by some $\mathtt{D}[r]$. We here present two potential distinguishing attacks on this Lesamnta-like hash function family. These two attacks can not work for Lesamnta with new round constants.

```
ConstantGenerator512(word C[Nr_comp512])  /* Nr_comp512=32 */
begin
  word c = ffffffffffffffffffffffffffffffff
  for i = 0 to Nr_comp512*J-1   /* J = 8 */
   /* Galois-type LFSR */
   if c ∧ 00...01 = 00...01
     c = (c >> 1) ⊕ 89dae79b7f6b6b32ca34805cfa534180
   else
     c = c >> 1
   end if
   if i mod J = 0
     C[i/J] = c
   end if
  end for
end
```

Fig. 2. Pseudocode for generating round constants of Lesamna-384/512.

Condition 1 Let $a = a_0\|a_1\|\dots\|a_7$ and $b = b_0\|b_1\|\dots\|b_7$, where $a_i, b_i \in \{0,1\}^8$. Let $\mathsf{tp}_\ell(a) = b$ be a byte-transposition such that $b_i = a_{i+2\ell \bmod 8}$ for $0 \le \ell \le 3$. Let $\mathsf{rv}(a) = b$ be a byte-transposition such that $b_i = a_{7-i}$.

We have a distinguishing attack on the underlying block cipher of the D-Lesamnta-256 output function if D satisfies the following condition.

Let $\mathtt{D}[r]$ be the 64-bit r-th round constant in the key scheduling function of $EncOut_{256}$. $\mathtt{D}[r]$ is considered as an 8-byte data, that is,

$$\mathtt{D}[r] = \mathtt{D}[r](0) \parallel \mathtt{D}[r](1) \parallel \dots \parallel \mathtt{D}[r](7),$$

where $\mathtt{D}[r](i) \in \{0,1\}^8$. We define a 64-bit difference Λ_r as

$$\Lambda_r = \mathtt{D}[r] \oplus \Pi(\mathtt{D}[r])$$

for $r = 0, 1, \dots, 31$, where Π is any composition of tp_ℓ and rv but $\Pi \ne \mathsf{tp}_0$.

The condition is to satisfy the following equations:

$$\Lambda_k = \Lambda_{k+4} \quad \text{for } 0 \le k \le 26 \ .$$

Notice that round constants of $EncOut_{256}$ are identical to those of $EncComp_{256}$ (see Fig. 20 of [2]).

Since the new round constants for Lesamnta-256 do not satisfy the above condition, Lesamnta-256 is secure against the above attack.

Condition 2 Let $s = (s_{i,j})$ and $t = (t_{i,j})$, where $s_{i,j}, t_{i,j} \in \{0,1\}^8$ and $0 \le i, j \le 3$. Let $\pi_\ell(s) = t$ be a byte-transposition such that $t_{i,j} = s_{i+\ell \bmod 4, j}$ for $0 \le \ell \le 3$. Let $\varpi_\ell(s) = t$ be a byte-transposition such that $t_{i,j} = s_{i,j+\ell \bmod 4}$ for $0 \le \ell \le 3$.

We have a distinguishing attack on the underlying block cipher of the $\mathtt{D}$-Lesamnta-512 output function if $\mathtt{D}$ satisfies the following condition.

Let $\mathtt{D}[r]$ be the 128-bit r-th round constant in the key scheduling function of $EncOut_{512}$. $\mathtt{D}[r]$ is considered as a 16-byte data, that is,

$$\mathtt{D}[r] = \mathtt{D}[r](0) \parallel \mathtt{D}[r](1) \parallel \ldots \parallel \mathtt{D}[r](15),$$

where $\mathtt{D}[r](i) \in \{0,1\}^8$. We see $\mathtt{D}[r] = (\mathtt{D}[r]_{i,j})$, where $\mathtt{D}[r]_{i,j} = \mathtt{D}[r](i+4j)$ for $0 \leq i,j \leq 3$. We define a 128-bit difference Ξ_r as

$$\Xi_r = \mathtt{D}[r] \oplus \Pi(\mathtt{D}[r])$$

for $r = 0, 1, \ldots, 31$, where Π is any composition of π_ℓ and $\varpi_{\ell'}$ but Π is not an identity transposition. The condition is to satisfy the following equations:

$$\Xi_k = \Xi_{k+4} \quad \text{for } 0 \leq k \leq 26 \ .$$

Notice that round constants of $EncOut_{512}$ are identical to those of $EncComp_{512}$ (see Fig. 28 of [2]).

Since the new round constants for Lesamnta-512 do not satisfy the above condition, Lesamnta-512 is secure against the above attack.

6 The Impact of the Change on Lesamnta

6.1 Impact on the Security of Lesamnta

We observe that the change does not cause any impact on resistance against the known attacks on Lesamnta described in [2].

We expect that Lesamnta with the new round constants prevents the attacks described in Sect. 3 which violated the assumptions of the security proofs in the ideal cipher model in [2]. Therefore we believe that these security proofs for Lesamnta would become more meaningful if the round constants are changed to the new ones.

6.2 Impact on the Performance of Lesamnta

For speed-optimized implementations of Lesamnta, the change does not cause any impact on its performance because the round constants are stored in a table.

For area-optimized implementations of Lesamnta, the change may slightly increase the required memory size because on-the-fly generation of round constants may be less useful than storing them in a table, due to the relatively large number of terms in the feedback polynomial used in the LFSR generating them. However, based on our estimation, we expect that storing them in a table does not cause any problem in real applications.

7 Concluding Remarks

We propose to make a minor change to the specification of Lesamnta by changing round constants. The motivation of this change is to prevent the attacks described in [3] and the newly-discovered attack described in this document.

We expect that all of these attacks do not work any more for Lesamnta with the new round constants. We also expect that the change does not cause any significant impact on resistance against the known attacks and on performance of Lesamnta.

Acknowledgments

We would like to thank Charles Bouillaguet, Orr Dunkelman, Gaëtan Leurent, Pierre-Alain Fouque for their excellent analysis on Lesamnta. We would like to thank Mridul Nandi for improving the analysis. We would like to mention the people who gave us feedback and important comments on this work: Kota Ideguchi, Yasuko Fukuzawa, Toru Owada, Bart Preneel. This work was partially supported by the National Institute of Information and Communications Technology, Japan.

References

1. C. Bouillaguet, O. Dunkelman, G. Leurent, and P. A. Fouque, Private communication, 2009.
2. S. Hirose, H. Kuwakado, and H. Yoshida, "SHA-3 proposal: Lesamnta," `http://csrc.nist.gov/groups/ST/hash/sha-3/Round1/documents/Lesamnta.zip%`, October 2008. latest version: `http://www.sdl.hitachi.co.jp/crypto/lesamnta/`.
3. S. Hirose, H. Kuwakado, and H. Yoshida, "Security Analysis of the Compression Function of Lesamnta and its Impact," `http://csrc.nist.gov/groups/ST/hash/sha-3/Round1/documents/LESAMNTA_Comments.pdf`, June 2009.
4. M. Nandi, Private communication, 2009.
5. National Institute of Standards and Technology, "Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (SHA-3) family," `http://csrc.nist.gov/groups/ST/hash/documents/`, November 2007.

8 List of Annexes

A The Attack Methods

A.1 Attack Using Weak Messages on Lesamnta

The pseudo-collision-finding attack might be applicable to an attack using weak messages. The primary idea of this attack was shown in [1].

Consider Lesamnta-256. The algorithm of an adversary that finds a second preimage is described below.

1. Suppose that a t block message and its digest are given. The t-block message is denoted by $\mathtt{mb}^{(0)} \parallel \mathtt{mb}^{(1)} \parallel \ldots \parallel \mathtt{mb}^{(t-1)}$ where $\mathtt{mb}^{(i)}$ is a 256-bit message block.
2. For $i = 0, 1, \ldots, t-1$, compute $\mathtt{chain}^{(i)}$ as

$$\mathtt{chain}^{(i)} = \mathtt{Compression256}(\mathtt{chain}^{(i-1)}, \mathtt{mb}^{(i)}),$$

   where $\mathtt{chain}^{(-1)}$ is the standard initial value.
3. If there is an index i_1 such that

$$\mathtt{chain}^{(i_1)}[j] = \mathtt{chain}^{(i_1)}[j+1]$$

   for $j = 0, 2, 4, 6$, go to the next step. Otherwise output fail.
4. Find $\mathtt{mb}'^{(i_1-1)}$ and $\mathtt{mb}'^{(i_1)}$ such that

$$\begin{aligned} \mathtt{chain}'^{(i_1-1)} &= \mathtt{Compression256}(\mathtt{chain}^{(i_1-2)}, \mathtt{mb}'^{(i_1-1)}), \\ \mathtt{chain}^{(i_1)} &= \mathtt{Compression256}(\mathtt{chain}'^{(i_1-1)}, \mathtt{mb}'^{(i_1)}), \end{aligned}$$

   by using the attack described in [3].

The probability that the condition in step 3 is satisfied is $t/2^{128}$. Since Lesamnta-256 accepts a $(2^{64}-1)$-bit message at most, the probability is less than 2^{-72}. We call a message satisfying the condition in step 3 a *weak message.* We notice that this attack is effective only when a given message is a weak message.