Faster multicollision attack on Davies-Meyer hash function scheme implementing Simeck32/64 block cipher algorithm

F Wijitrisnanto$^{1}$ and B H Susanti$^{2}$

$^{1,2}$Sekolah Tinggi Sandi Negara, Ciseeng, Bogor 16120, Indonesia
$^{1}$fajar.w96@gmail.com, $^{2}$bety.hayat@stsn-nci.ac.id

Abstract. Davies-Meyer is one scheme among 12 compression functions found through systematic analysis by Preneel et al. to be provably secure under black-box analysis. But in the development, this scheme proved to be vulnerable to fixed-point attack. With this vulnerability, it is possible to implement one of attack in an iterated hash function that exploits fixed-point weakness named faster multicollision attack. Implementing Simeck32/64 as an underlying block cipher, the attack induced by firstly searching for fixed-point collisions. To accomplish this finding stage based on Yuval’s birthday attack, a sum of $2^{n/2}$ different fixed-point sequences are needed. Based on that, two sets of $2^{16}$ different inputs are generated by modifying 16 bits of least significant bits from each pair of five input samples to find a collision between them. The final result makes an outstanding fact, with 4.194.304 total collisions obtained from five samples and 16 different IV values that already produced before in fixed-point collision finding stage. These facts conclude that Davies-Meyer scheme is not resistance against faster multicollision attack because of its fixed-point weakness.

1. Introduction

A hash function is an algorithm used in cryptography to mainly protect data integrity. This algorithm takes an arbitrary size input and produces a certain fixed length of output called hash value or digest. If a hash function is secure, two distinct pieces of data should always have different hashes. A secure hash function then should behave like a truly random function, which could be indicated by fulfilling the properties of preimage resistance and collision resistance. In collision resistance case, the attacker should not be able to find two distinct messages, $M$ and $M'$, that hash to the same value [1].

A new method to construct a one-way hash function from random block cipher suggested by Merkle as a new way to build hash function [2]. However, a straight usage of a block cipher leads to the problem that the input can be recovered easily with aid of the decryption function of the block cipher. To avoid this, Merkle-Damgård construction was built based on Merkle’s thesis [3], and the properties were already proved by Damgård [4] that leads to iterated hash model. After all, this so-called hash functions based on block cipher later divided into two form i.e. single block length and double block length hash function [5]. Single length hash functions output approximately the same number of bits as processed by the underlying block cipher, such as Davies-Meyer scheme [6].

Davies-Meyer scheme has been attributed by Davies to Meyer; apparently known to Meyer and Matyas circa 1979. This particular scheme is among 12 compression functions found through systematic analysis by Preneel et al. to be provably secure under black-box analysis, with the addition being vulnerable to fixed-point attack [5]. The following research then claimed that Davies-Meyer has a weakness in preimage resistance property, which leads to the creation of fixed-points [7].
A new cryptanalysis technique against hash functions especially in an iterated model like Merkle-Damgård was first introduced by Joux [8] called multicollision attack. This technique targets the main security requirement for a hash function, collision resistance. It shows that finding multicollision, i.e. $r$-tuples of messages that have the same hash value, is not much harder than finding ordinary collisions [8]. Based on this finding, Aumasson [9] innovated a new way called faster multicollision that reduce the cost of Joux’s multicollision in time from $[log_2 r] 2^{n/2}$ to $2^{n/2}$. However, this simple technique requires two conditions; the IV can be chosen by the attacker and the compression function admits easily found fixed-points [9].

Considering the facts that Davies-Meyer scheme which is an iterative hash function is vulnerable against fixed-point attack, then it could be assumed that the scheme should be vulnerable against faster multicollision. To apply this attack, Simeck algorithm is used as a building block cipher within Davies-Meyer scheme. Simeck is a family of lightweight block ciphers that combines the good design components from both Simon and Speck proposed by NSA [10], in order to devise even more compact and efficient block ciphers [11]. The purpose of this research is to empirically prove that Davies-Meyer hash function scheme is vulnerable to faster multicollision based on the former assumption of its weakness against fixed-point.

The paper is organized as follows. In section 2, we recall some basic concepts and theories related to this work. Section 3 detailly explains the methodology used in this attack. In section 4, the result of the attack is presented. Finally, section 5 concludes all efforts that have been done.

2. Related Theories

2.1. Hash function

A hash function $h$ accepts a variable-length block of data $M$ as input and produces a fixed-size hash value $H = h(M)$. Good hash function has the property that the results of applying the function to a large set of inputs will produce outputs that are evenly distributed and apparently random. The kind of hash function needed for security applications is referred to as a cryptographic hash function. This hash function is often used to determine whether or not data has changed [12]. Hash function as a minimum requirement has the following two properties, i.e:

- **Compression.** $h$ maps an input $x$ of arbitrary finite bitlength, to an output $h(x)$ of fixed bitlength $n$.
- **Ease of computation.** Given $H$ and an input $x$, $h(x)$ is easy to compute.

The term *hash function* that we used implies an unkeyed hash function. We use this term for general discussion. As based on functional classification, two types of hash functions are considered in the form of modification detection codes (MDCs) and message authentication codes (MACs). MDC built to provide a representative image of a message that maintains data integrity assurances. Otherwise, MAC’s purposes are to facilitate assurances regarding both the source of a message and its integrity [5].

2.2. Davies-Meyer hash function scheme

In 1993 Bart Preneel, Rene Govaerts and Joos Vandewalle (PGV) made a general model of round function that is used in block cipher-based hash function in single length scheme. They already made 64 different constructions based on all possible combinations between input, output, and constants. From all these models, Davies-Meyer scheme mentioned being one of the secure schemes in PGV. But, this claim soon rejected by Black [7] that proved easily found fixed points exist in such a scheme. Davies-Meyer scheme produces hash value shown at Figure 1.

- Input the message value $M$ with arbitrary length. Make sure the length is equals to multiple of $k$ ($k$ is the size of input key) by implementing padding rules.
- Divide the message into $j$ blocks per $k$-bit, denoted by $x_1, x_2, ..., x_j$.
- For every block, the input $x_i$ serve as the key to $E$. The previous hash-value $H_{i-1}$ serves as the plaintext to be handled with $n$ bit-length. The output $H_i$ is then concatenated with the previous output $H_{i-1}$ with aid of the exclusive-or (XOR) operator. The final output is defined by an iterated formula, $H_i = E_{x_i}(H_{i-1}) \oplus H_{i-1}$ for $1 \leq i \leq j$, $H_0 = IV$. 

---

2
2.3. Simeck lightweight block cipher

In this research, lightweight block cipher Simeck is denoted by Simeck2n/mn, where n is the word size, 2n is the block size, and mn is the key size. Since only Simeck32/64 used in this term, the block size and key size become 32 and 64 bits respectively. The round function and the key schedule algorithm follow the Feistel structure. A plaintext to be encrypted is divided into two words, \( l_0 \) and \( r_0 \), that each of them represent \( n \) bit of most significant and least significant bits. These words are processed by Simeck round function for certain number of rounds, and the outputs \( l_T \) and \( r_T \) are concatenated to form the ciphertext after \( T \) rounds.

Figure 1. Davies-Meyer scheme (Menezes et al., 1997)

\[ R_{k_i}(l_i, r_i) = (r_i \oplus f(l_i) \oplus k_i, l_i) \]

The round function defined by the formula, \( R_{k_i}(l_i, r_i) = (r_i \oplus f(l_i) \oplus k_i, l_i) \) as illustrated in Figure 2. The function \( f \) is defined by \( f(x) = (x \oplus (x \ll 5)) \oplus (x \ll 1) \). To generate the round key \( k_i \) from master key \( K \), the master key is first segmented into four words and loaded as the initial states \( (t_2, t_3, t_0, k_0) \) of the feedback shift registers shown in Figure 3. The updating operation, the value of constant \( C \), and other details can be seen in [11].

Figure 2. The round function of Simeck (Yang et al., 2015)
2.4. Faster multicollision attack
In an iterated hash function scheme, it could be assumed that if a collision occurs between a single block-sized message, then it is possible to find such collisions in the next blocks. The finding of these collisions led to the term of multicollision. A $r$-collision is a sequence of $r$ messages, $M^{(1)}, M^{(2)}, \ldots, M^{(r)}$ such that they have exactly the same hash value, $h(M^{(1)}) = h(M^{(2)}) = \cdots = h(M^{(r)})$. Joux’s proposal to find this $r$-collisions requires $2^{n(r-1)/r}$ iteration of counting hash values. As an upgrade, Aumasson’s faster multicollision develop this technique with the addition of some parameters. The fundamental idea of the method is to find a fixed-point collision for targeted hash function. Assume $FP_f(M)$ represents fixed-point of function $f$ and message $M$, then a fixed-point collision is a couple $(M, M')$ such that $FP_f(M) = FP_f(M') = H_0$. After this finding, as much as $2^k$-collisions could be built using a set of message blocks $\{M, M'\}^k$ followed by an arbitrary block $M^*$, where $k$ stands for the number of block included. Figure 4 shows the process of faster multicollision.

Figure 4. Faster multicollision scheme (Aumasson, 2008)

3. Research method
3.1. Attack’s stages
This research exploits the weakness of Davies-Meyer scheme that already explained by Menezes et al. [5] and Aumasson [1] to build fixed-point sequences. Each of these sequences consists of IV and corresponding hash value that share the same value, IV = $H_i$, whatever the value of hash’s input $x_i$ used. After that, we try to find collisions between fixed-points produced before using Yuval’s birthday attack method, such that different value of hash inputs, $x_i \neq x_i'$ will result in the same hash $H_i$. Finally, from fixed-point collisions that have been found, faster multicollision technique could be applied.

The method used to build fixed-point works as described in Figure 5. Apply $D_k$ as a decryption phase of a block cipher that require two inputs: message block $x_i$ and encryption result $E_k(x_i, H_{i-1})$.
0000 0000H. The value of \(H_{i-1}\) produced will be XORed with \(E_k(x_i, H_{i-1})\) to build \(H_i\) that fulfil fixed-point condition.

![Figure 5. Generation of fixed-point in Davies-Meyer scheme (Aumasson, 2017)](image)

In this attack, Simeck32/64 was implemented within Davies-Meyer scheme as a compression function. Only one block length of hash input size is required (64 bit). The application of this lightweight block cipher works as follows. In this DM-SIMON32/64 scheme, IV (when \(i = 0\)) or \(H_{i-1}\) (when \(1 < i\)) is used as a message input for Simeck block cipher. Meanwhile, input for this scheme, \(x_i\), played a role as a key for Simeck block cipher.

3.2. Sample and sampling technique
To accomplish fixed-point collision finding stage based on Yuval’s birthday attack, a sum of \(2 \times 2^{n/2}\) different fixed-point sequences are needed. Based on that, two sets of \(2^{16}\) different inputs are generated by modifying 16 bits of least significant bits (LSB) from each pair of input samples. Five samples as shown in Table 1 are used in this stage to extend attack result range. As a result of combining all possible pairs amongst entire samples, the total sum of modified input pairs examined with Yuval’s algorithm equal to \(C_2^5\).

| No. | Input \(x_i\) (64 bit)                 |
|-----|--------------------------------------|
| 1.  | 0000 ... 0000\(_H\)                  |
| 2.  | 1111 ... 1111\(_H\)                  |
| 3.  | 2222 ... 2222\(_H\)                  |
| 4.  | 3333 ... 3333\(_H\)                  |
| 5.  | 4444 ... 4444\(_H\)                  |

This attack is conducted by trying to find a collision between two sets of \(2^{16}\) or 65,536 different fixed-point inputs. So, these inputs work as the independent variable. That goes along with Simeck’s encryption \(E_k(x_i, H_{i-1}) = 0000 0000H\) as a control variable. With this fact, experimental technique of research is applied.

4. Implementation of faster multicollision attack on Davies-Meyer hash function scheme
4.1. Construction of fixed-point collisions
The work is done by first collecting all fixed-point across the samples. As a result, a sum of 327.680 fixed points successfully generated by adding each sample’s 65,536 fixed points between one another. This value proofs fixed-point weakness in Davies-Meyer scheme as shown in Table 2.
The fixed-point vulnerability that already stated before could be explained by a simple proof. Take a look at the decryption stage in Figure 5, that shows the value of $D_k(M, E_k(M))$ used as IV. As an effect when Davies-Meyer scheme used, we get the following equation:

\[
H = E_k(M, IV) \oplus IV
\]

\[
= E_k\left( M, D_k(M, E_k(M)) \right) \oplus IV
\]

\[
= 00000000_H \oplus D_k(M, 00000000_H)
\]

\[
= D_k(M, 00000000_H)
\]

\[
= IV.
\]

The second step is to find a collision between such a great number of fixed-points generated before. It is done by finding a couple of messages $M_1$ and $M'_1$ such that $FP_r(M_1) = FP_r(M'_1)$. The fixed-point collision results finally reach 16 collisions, with all samples included in the search. The details are listed in Table 3.

### 4.2. Faster multicolission attack on Davies-Meyer scheme

The output of the fixed-point collision gained in previous steps then can be used to generate multicolission. We use the couple $M_1$ and $M'_1$ as first block input message and combine them with the corresponding IV to fulfill fixed-point condition. Then, use exactly the same message inputs in the second block, which will give the result of another fixed-point collisions. Finally, concatenate the first and second block messages and get four different messages-combination with the format of $M_1 M_1', M_1' M_1, M_1 M_1'$, and $M_1' M_1'$. After this stage implemented, we got 64 collisions as shown in Table 4. It happened because each 16 different IVs generate four different messages, then it is trivial to get such 64 collisions.

After obtaining multicolission in the first form, next step would be adding arbitrary block message $M^*$ in the third block. As a result, new messages generated with format $M_1 M_1^* M^*$, $M_1' M_1 M^*$, $M_1 M_1' M^*$, and $M_1' M_1' M^*$ each with the size of 192 bit. In this research, a total sum of $M^*$ included equal to 65,536 with a purpose to use all possible sample in the range of $2^{16}$. The impact of this parameter is, every IV values will generate $65,536 * 4 = 262,144$ collisions. This step produces a total of 4,194,304 collisions.

| No. | IV          | $M$ (64 bit) | Hash $H$         |
|-----|-------------|--------------|-----------------|
| 1   | ee463aaf    | 000000000000000000 | ee463aaf       |
| 2   | 38d880e7    | 000000000000000001 | 38d880e7       |
| 3   | 336a3eac    | 000000000000000002 | 336a3eac       |
|     |             |              |                 |
| 65536| 9b882e11    | 00000000000000000f | 9b882e11       |
| 65537| a1a2e453    | 111111111111100000 | a1a2e453       |
|     |             |              |                 |
| 131072| d0ac0155   | 111111111111111111 | d0ac0155       |
| 262145| a76b53ce    | 444444444444000000 | a76b53ce       |
|     |             |              |                 |
| 327680| e650a639    | 44444444444444444f | e650a639       |

| No. | IV          | $M$ (64 bit) | Hash $H$         |
|-----|-------------|--------------|-----------------|
| 1   | a1e04ebb    | 000000000000000092 | a1e04ebb       |
|     |             |              |                 |
| 2   | 5fcede29e   | 000000000000000070 | 5fcede29e      |
|     |             |              |                 |
| 15  | ac84e1d0    | 3333333333333377 | ac84e1d0        |
|     |             |              |                 |
| 16  | 9429fd59    | 3333333333333333fc9 | 9429fd59      |
|     |             |              |                 |
as the final sum between 16 different IV value which represented by $16 \times 262,144 = 4,194,304$ as shown in Table 5.

The main factor of this great gain is the implementation of arbitrary block, $M^*$ which value can be flexibly changed base on certain condition. Before the implementation, we got 64 collisions, but after modifying the third block with $M^*$, the output rises in number as much as 65,536 or $2^{16}$ times greater than previous multicollision. These arguments conclude the fact that Davies-Meyer scheme is not resistance against faster multicollision attack because of its fixed-point weakness.

### Table 4. Faster multicollision attack results without $M^*$.

| No | IV       | First Block | Second Block | Hash H   |
|----|----------|-------------|--------------|----------|
| 1. | a1c04ebb | 00000000000006c92 | 00000000000006c92 | a1c04ebb |
| 2. | 5fcfe29e | 1111111111111b3e | 1111111111111b3e | 5fcfe29e |
| 3. | a1c04ebb | 00000000000006c92 | 00000000000006c92 | a1c04ebb |
| 4. | 0000000000000000 | 0000000000000000 | 0000000000000000 |

### Table 5. Faster multicollision attack results with $M^*$.

| No  | IV       | First Block | Second Block | Third Block ($M^*$) | Hash H   |
|-----|----------|-------------|--------------|---------------------|----------|
| 1   | 0000000000000000 | 0000000000000000 | 0000000000000000 | 0000000000000000 | c6876f2d |
| 2   | 0000000000000000 | 0000000000000000 | 0000000000000000 | 0000000000000000 | 4854836c |
| 3   | a1c04ebb | 0000000000000000 | 0000000000000000 | 0000000000000000 | 9429fd59 |
| 4   | 0000000000000000 | 0000000000000000 | 0000000000000000 | 0000000000000000 | 30f74b8e |

### 5. Conclusion

We arranged a multicollision attack following Aumasson’s method to prove that Davies-Meyer scheme that has easily found fixed-point property and iterated structure is vulnerable against this attack. There are 65,536 modifications of five input samples included to perform the Yuval’s birthday attack in order
to find fixed-point collisions. At the first strike, 16 collisions between fixed-points gained with 16 different IVs. After implementing multicollision stage, we got a total of 64 collisions in the record. The final result decides that there are 4,194,304 total collisions obtained from 16 different IVs that already produced within fixed-point collision finding stage. With such a great number of collisions, it can be concluded that Davies-Meyer scheme that implements Simeck32/64 as its founding block cipher is not resistance against fixed-point attack, collision attack, and multicollision attack using Aumasson’s method.

6. Further research
We see an opportunity to try a different research by choosing another scheme of iterated hash function as a target. This trial will be harder to fulfil because the attacker has to find the unique fixed-point weakness for each scheme. Another option is to change the collision finding algorithm we used, to find a better attack complexity.

Acknowledgements
This research is supported by Sekolah Tinggi Sandi Negara, Bogor, Jawa Barat. We would like to thank the reviewers who already helped us to be a better writer.

References
[1] J. Aumasson, Serious Cryptography. No Starch Press, Inc., 2017.
[2] R. C. Merkle, “One Way Hash Functions and DES,” Adv. Cryptol. - CRYPTO ’89, LNCS 435, pp. 428–446, 1989.
[3] R. C. Merkle, “Secrecy, Authentication, and Public Key Systems,” 1979.
[4] I. B. Damgård, “A Design Principle for Hash Functions.” 1989.
[5] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, Handbook of Applied Cryptography, 1st ed. Boca Raton: CRC Press, 1997.
[6] T. Bartkewitz, “Building Hash Functions from Block Ciphers, Their Security and Implementation Properties,” 2009.
[7] J. Black, “An Analysis of the Blockcipher-Based Hash Functions from PGV,” vol. 1, pp. 519–545, 2010.
[8] A. Joux, “Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions,” CRYPTO, vol. 1, no. 0, 2004.
[9] J. Aumasson, “Faster Multicollisions”, INDOCRYPT, vol. 0, no. 113329, 2008.
[10] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, “The Simon and Speck Families of Lightweight Block Ciphers,” 52nd ACM/EDAC/IEEE Des. Autom. Conf., 2013.
[11] G. Yang, B. Zhu, V. Suder, M. D. Aagaard, and G. Gong, “The Simeck Family of Lightweight Block Ciphers,” no. June 2015.
[12] W. Stallings, Cryptography and Network Security Principles and Practice Global Edition, Seventh Ed. Edinburgh Gate: Pearson Education Limited, 2017.
[13] G. Yang, B. Zhu, V. Suder, M. D. Aagaard, and G. Gong, “The Simeck Family of Lightweight Block Ciphers,” no. June 2015.