ON A CONSTRUCTION OF STREAM-CIPHER-BASED HASH FUNCTIONS

Yuto Nakano, Jun Kurihara, Shinsaku Kiyomoto and Toshiaki Tanaka
KDDI R&D Laboratories Inc., 2-1-15, Ohara, Fujimino, Saitama, Japan

Keywords: Hash function, Stream cipher, Collision resistance, Second preimage resistance, Preimage resistance.

Abstract: Hash functions using stream ciphers as components perform fast on a variety of platforms. However, the security and the design policy of stream-cipher-based hash functions (SCHs) have not yet been studied sufficiently. In this paper, we analyze its design criteria based on a ideal function of SCHs. First, we show that attacks against a stream cipher can also be threats against SCHs. Then we discuss the security on each phase of SCH; message injection, blank rounds, and hash generation with this function. Finally we derive the necessary conditions on the stream cipher function for an SCH to be secure.

1 INTRODUCTION

Hash functions are used in many cryptographic applications, *e.g.*, message authentication codes, digital signatures and authentication protocols. Standard hash functions MD5 (Rivest, 1992) and SHA-1 (NIST, 1995) have been demonstrated that these functions are not collision resistant (Wang et al., 2005; Wang and Yu, 2005). Many constructions for new hash functions have been presented as alternatives to these ordinary algorithms. Some hash functions such as Abacus (Sholer, 2008) and Boole (Rose, 2008) are based on stream ciphers, and shows that they have good performance on a variety of platforms. A noteworthy advantage of stream cipher-based hash functions is that the algorithm is used not only for generating a hash value but also for encrypting/decrypting data as a stream cipher. We can reduce total costs of implementations for a hash function and a symmetric-key encryption algorithm with a stream-cipher-based hash function. Thus, it is a reasonable solution for resource constraint devices.

The general construction of stream-cipher-based hash functions (SCHs) was first introduced by Golić (Golić, 2001) as a mode of operation of stream ciphers. An SCH consists of a stream cipher function and an additional function that inputs a message into the internal state of the stream cipher function. Therefore, a model of an SCH consists of a pre-computation function and a stream cipher function. The stream cipher function is a core component of SCHs and an appropriate algorithm is selected from among existing stream cipher algorithms. The pre-computation function is used to absorb the message into the internal state of the stream cipher function. On the other hand, a well developed evaluation design of the security of SCHs is yet to be produced, and design principles of secure SCHs have not been established.

In this paper, we define a stream cipher function that consists of three rounds; message injection, blank rounds, and hash generation. With this function, we show that an attack on the keystream feedback mode and an attack against a self-synchronizing stream cipher can also be employed as an attack against SCHs. Furthermore, we analyze the security of each phase from an attack perspective. As a result of our work, following criteria can be formulated:

- The size of the internal state is larger than the size the of hash value. $|S| > |H|$ is required for collision, second preimage, and preimage resistance, where $|S|$ is the size of the internal state and $|H|$ is the length of the hash value.
- The message injection has to be collision resistant. The computational cost of the collision attack against message injection has to be equal to or more than $2^{|H|/2}$, and an adversary is not able to control the internal state of the stream cipher function.
- Blank rounds are preimage resistant. That is, deriving the internal state after message injection by going blank rounds backwardly requires $2^{|H|}$ computational cost. Furthermore, the transition of the internal state is cyclic and its period is more than or equal to $2^{|H|}$.

Nakano Y., Kurihara J., Kiyomoto S. and Tanaka T. (2010).
ON A CONSTRUCTION OF STREAM-CIPHER-BASED HASH FUNCTIONS.
In *Proceedings of the International Conference on Security and Cryptography*, pages 334-343
DOI: 10.5220/0002939703340343

- The hash value has to be affected by the whole internal state. Suppose that the internal state size is $|S|$ and the hash size is $|H|$, the hash value bit has to be decided by at least an $|S|/|H|$-bit internal state.
- Suppose that an x-bit internal state can be controlled by an x-bit message and this operation does not affect other bits, then the whole internal state can be controlled with an $|S|/x$-bit message. Here we assume that one bit message is xored with n bits of the internal state at equal intervals. Then the adversary can control up to an $|S|/n$-bit internal state. Hence the condition $s(n-1) \geq nh$ is required for a message injection to be collision resistant.

2 PRELIMINARIES

We provide definitions and summarizes related works in this section.

2.1 Definitions

Throughout the paper, we use the following notations.

- M: input message
- m_t: input message to the stream cipher function at time t
- S_t: internal state of the stream cipher function at time t
- o_t: output of the stream cipher function at time t
- $|x|$: bit length of x (x can be M, S, H)
- $\oplus$: bitwise exclusive-or
- S_{msg}, S'_{msg}: internal state of stream cipher function after message M or M' are injected, respectively
- S_{blk}, S'_{blk}: internal state of stream cipher function after blank rounds derived from S_{msg} or S'_{msg}, respectively
- Δ_{msg}: internal state difference after message injection $\Delta_{msg} = S'_{msg} - S_{msg}$
- Δ_{blk}: internal state difference after blank rounds $\Delta_{blk} = S'_{blk} - S_{blk}$

2.2 Security Definitions for Hash Functions

Security requirements for hash functions are collision resistance, second pre-image resistance, and pre-image resistance (Menezes et al., 1996). Let M and M' be messages, $|H|$ be hash length, h be a hash function, and the symbol $||$ be concatenation of data.

Collision Resistance. Finding M and M' such that $h(M) = h(M')$ and $M \neq M'$ requires $2^{|H|/2}$ hash operations.

Second Pre-image Resistance. Finding M for given M' such that $h(M) = h(M')$ and $M \neq M'$ requires $2^{|H|}$ hash operations.

Pre-image Resistance. Finding M from $h(M)$ requires $2^{|H|}$ hash operations.

Length-extension Security. This requirement has been proposed in NIST SHA-3 competition. Given $h(M)$, the complexity of finding (z,x) such that $x = h(M||z)$ should be greater than or equal to either the complexity of guessing M itself or $2^{|H|}$.

2.3 Related Work

Here we introduce important research related to the construction of SCHs.

2.3.1 Self-synchronizing Stream Cipher

A self-synchronizing stream cipher (SSSC) is one in which the keystream is generated from the secret key and a fixed number of previous ciphertext bits (Menezes et al., 1996).

Let p_t, c_t, and z_t be a plaintext bit, a ciphertext bit, and a keystream bit at time t, respectively. The keystream at time t depends on the secret key K and previous n_{me} ciphertext bits $c_{t-n_{me}}, \ldots, c_{t-1}$, where n_{me} is called *input memory*. The keystream bit is described by

$$z_t = f_c(c_{t-n_{me}}, c_{t-(n_{me}-1)}, \ldots, c_{t-1}, K), \qquad (1)$$

where f_c is the function defined by the stream cipher. When the ciphertext is decrypted, the keystream is generated in the same manner as encryption. For the first keystream bit, the previous n_{me} ciphertext bits do not exist. Therefore, n_{me}-bit Initial Vector(IV) has to be defined as $IV = c_{n_{me}}, c_{(n_{me}-1)}, \ldots, c_0$.

2.3.2 Keystream Feedback Mode

Keystream feedback mode is employed in the initialization of many stream ciphers such as SNOW (Ekdahl and Johansson, 2002). In the initialization of stream ciphers, an initial key and an IV are loaded into the internal state of the stream cipher. After loading the key and the IV, the stream cipher is clocked a specified number of times. In the initialization process, keystream bits are fed back to the internal state to enable the further diffusion of the key and the IV.

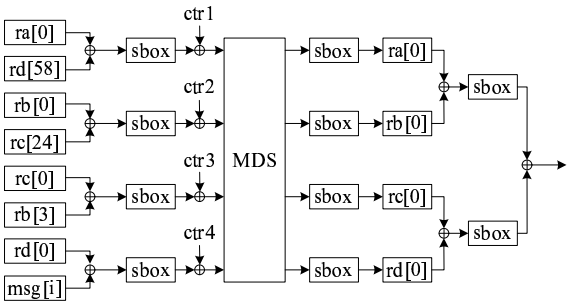

Figure 1: Schematic of Abacus.

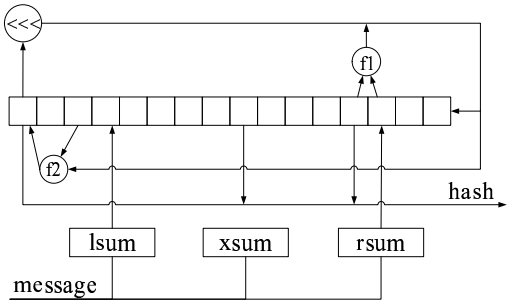

Figure 2: Schematic of Boole.

2.3.3 Golić's Construction

Golić (Golić, 2001) showed how to convert a keystream generator into a stream cipher with memory (SCM) mode and how to built hash functions with SCM mode. When a feedback shift register (FSR) based keystream generator is used, SCM mode is easily converted by adding the plaintext bit to the feedback bit of the FSR. The following is the scheme of Golić's construction.

1. Generate a keystream by feeding back a message into the keystream generator and encrypting the message.
2. Another encryption is applied; this time the ciphertext obtained in step 1. is used in reverse order.
3. Generate a keystream by feeding back a constant value (if possible, set all constants to zero) for αm times, where α is not a large integer.
4. Output the last h successive bits of the keystream obtained in step 3. as the hash value.

In step 1, using a fixed and known key, SCM mode is clocked m times with an m-bit message and the corresponding m-bit ciphertext is memorized. SCM mode is clocked another m times with the m-bit ciphertext in the reverse order in step 2. SCM mode is clocked αm times where α is not a large integer (*e.g.*, three), and the last h successive ciphertext bits (or keystream bits) are output as the hash value.

As the ciphertext in reverse order is used in step 2, this scheme requires an amount of memory that is the same as the message size.

2.3.4 Abacus

Abacus is a family of hash functions submitted to NIST for consideration as SHA-3. Abacus has four registers (`ra, rb, rc, rd`) as shown in Fig 1. We only show the absorb phase. Refer to (Sholer, 2008) for detailed information.

The absorb phase processes one byte of a message in a round. First, the values of four registers and a byte message are Exclusive-ORed(XORed) and the results go through S-Boxes. In the next step, four counters are combined with registers. Then, the maximum distance separable (MDS) matrix function is applied and four bytes are output. Four bytes of registers go through the S-Boxes again.

ATTACKS ON ABACUS. Two second pre-image attacks against Abacus were proposed by Nikolić et al. (Nikolić and Khovratovich, 2008) and Wilson (Wilson, 2008) independently. In the paper (Wilson, 2008), Wilson also showed a collision attack.

In the absorb phase of Abacus, message bytes are XORed with the register `rd[0]`. At this point, the adversary can set the value of `rd[0]` to zero. Once the value of `rd[0]` is determined, it remains the same in the following 88 rounds. Thus, with an 89-byte message, the adversary can fix the whole register `rd`.

Then the attack uses a meet-in-the-middle (MITM) approach (Menezes et al., 1996). Since whole values of the register `rd` are set to zeros the internal state of the Abacus is reduced to $1+5+37=43$ bytes. Therefore the MITM approach only requires $2^{8\times 43/2} = 2^{172}$ computation.

Vulnerabilities of Abacus are that the register value can be determined from the input message and that the hash output function is not a one-way function.

2.3.5 Boole

Boole (Rose, 2008) is a family of hash functions submitted to NIST for SHA-3 competition. The schematic of Boole is shown in Figure 2. Boole is constructed from a non-linear feedback shift register, input accumulators, and an output filter function. Boole consists of three phases, *i.e.*, an input phase, a mixing phase, and an output phase. The state transition function of the register, referred to as a cycle, transforms state S_t into S_{t+1}.

A message word is input to three word accumulators, and the accumulators are updated in the input phase. Then the register is cycled once. After the in-

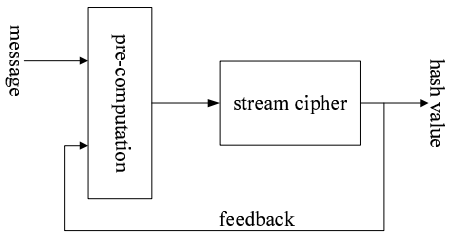

Figure 3: Model of SCH

put phase, the register is mixed with three accumulators, then the register is cycled 16 times, and the hash value is generated.

ATTACK ON BOOLE. A pre-image attack against Boole is presented by Nikolić (Mendel et al., 2009). This attack uses the MITM method. The value of ten registers can be determined from the message and the target hash. Therefore the search space is reduced to nine registers. Each register is one byte, hence the complexity of the attack is $2^{\frac{9n}{16}}$.

Boole is vulnerable in two ways: the register value can be determined from the input message and the hash output function is not a one-way function.

A collision attack on Boole is also proposed (Mendel et al., 2009). Since two boolean functions used in Boole are not invertible, the collision can be constructed in these functions.

3 CONSTRUCTION OF SCHS

In this section we present a general model of SCH, which is shown as Figure3. An SCH is modeled with two components: a pre-computation function and a stream cipher function. This is the generalized model of the construction of the hash functions based on stream ciphers.

3.1 Pre-computation Function

A pre-computation function is appended to a stream cipher for constructing an SCH. The pre-computation function is the part which takes an input message and an intermediate hash value as an input and determines the internal state of the stream cipher function. The stream cipher function plays the role of diffusing the message into the internal state. The hash value is a certain length of keystream which is produced by the stream cipher function. Generally, keys and IVs are set to a constant value, usually to zero, and the message is loaded to the internal state of the stream cipher function. Golić's construction consists of a first-in-last-out (FILO) buffer and a stream cipher. Therefore it is obvious that the FILO buffer should be treated as pre-computation function and the stream cipher is the stream cipher function.

Generally, the pre-computation function requires very few computations; for example, the pre-computation function in Abacus is defined as the registers where the message is XORed. Namely the operation $rd[0] \oplus msg[i]$ is the pre-computation function in Abacus. In the case of Boole, three accumulators; lsum, xsum, and rsum, are used to input a message into the non-linear feedback shift register. Hence operations which use these accumulators are defined as the pre-computation function.

3.2 Hash Value Generation

SCHs execute three phases: message injection, *blank rounds*, and hash generation. Here, the message is loaded into the internal state of the stream cipher in the message injection. The blank round is an operation in which the stream cipher function is clocked by feeding back the output without outputting the keystream. This phase is a similar operation to the initialization of the stream cipher. The stream cipher outputs a keystream as a hash value in the hash generation.

We define a stream cipher with a pre-computation function at time t as $f(m_t, S_t, o_t)$. In the stream cipher function, the internal state is updated and the keystream (hash value) is generated from the message, internal state, and feedback of output. The operation can be denoted as:

$$f(m_t, S_t, o_t) \rightarrow S_{t+1}, o_{t+1}. \quad (2)$$

We denote message injection, blank rounds, and hash generation by Eq. (2) as follows:

$$f(m_t, S_t, o_t) \rightarrow S_{t+1}, o_{t+1}, \quad (3)$$
$$f(0, S_t, o_t) \rightarrow S_{t+1}, o_{t+1}, \quad (4)$$
$$f(0, S_t, 0) \rightarrow S_{t+1}, o_{t+1}. \quad (5)$$

The internal state is updated from message, previous state, and feedback of output in the message injection. In blank rounds, the internal state is updated from the previous state, and output. During hash generation, only the previous state is used to update the state and output the hash value.

4 ATTACKS AGAINST SCH

In this section, we considere the relation between attacks against stream ciphers and attacks against SCHs. We also consider general attacks against hash functions to derive secure construction of SCHs.

4.1 Relation to the Attack against a Stream Cipher

Since blank rounds of an SCH and the initialization of stream ciphers have the same structure, chosen IV/key attack against stream ciphers can be applied to the attack against SCH. Wu et al. (Wu and Preneel, 2007) presented the chosen IV attack against Py (Biham and Seberry, 2005) and Pypy (Biham and Seberry, 2006) in which an adversary can find an identical keystream using IVs which have special differences. In the initialization of keystream feedback mode, the internal state is updated after the key and IV setup as follows:

$$f(S_t, z_t) \rightarrow S_{t+1}, z_{t+1}, \quad (6)$$

where S_t and z_t are the internal state and keystream at time t. Blank rounds in an SCH can be denoted as Eq. (4). There is no message to be injected into the internal state during blank rounds, Eq. (4) can be modified as:

$$f(S_t, o_t) \rightarrow S_{t+1}, o_{t+1}. \quad (7)$$

From the Eq. (6) and (7), an attack against keystream feedback mode can be used against blank rounds of an SCH. In the case where the attack proposed by Wu et al. is applied to the SCH, an adversary can find two identical hash values from different messages; this leads to a collision and second preimage attack against SCHs. If there is an attack that reveals the secret key from the keystream, such attack leads to the preimage attack against SCHs.

Blank rounds also have a relation to SSSC. In Eq. (1) c_{t-1} can be considered as an output feedback at time t and $(c_{t-n_{me}}, c_{t-(n_{me}-1)}, \ldots, c_{t-2})$ as an internal state. At time $t+1$, c_t will be fed back and $(c_{t-(n_{me}-1)}, \ldots, c_{t-1})$ will be the internal state. Let S_t be $S_t = (c_{t-n_{me}}, c_{t-(n_{me}-1)}, \ldots, c_{t-2})$, then the generation of the keystream can be denoted as:

$$z_{t+1} = f(S_t, p_t \oplus z_t),$$

and the internal state is updated at the same time:

$$f(S_t, p_t \oplus z_t) \rightarrow S_{t+1}, p_{t+1} \oplus z_{t+1}. \quad (8)$$

Hence an attack against SSSCs can be regarded as an attack against SCHs from Eq. (7) and (8).

An attack aganist an SSSC which controls the keystream or ciphertext with the fed-back ciphertext should be considered. If there is an attack such that identical ciphertexts can be obtained from different plaintexts, this attack can be converted to the collision attack against SCH. Moreover, if the adversary can set the ciphertext arbitrarily, it is considered to be a second preimage attack.

4.2 Collision Attacks

We consider collision attacks against each phase; the message injection, blank rounds, and the hash generation in an SCH.

4.2.1 Message Injection

We set the message difference which is canceled after all message bits are input to the internal state in the attack against message injection. Let M and M' be a pair of collision messages. Then for the two internal states S_{msg} and S'_{msg}, $S_{msg} = S'_{msg}$ holds and no input exists except the output feedback after the message injection. Hence $S_{blk} = S'_{blk}$ and $H = H'$ are obtained.

4.2.2 Blank Rounds

In the collision attack against blank rounds, the difference canceled during the blank rounds is set to the pair of messages. Suppose two internal states S_{blk}, S'_{blk} are obtained after applying blank rounds to S_{msg}, S'_{msg}. The collision attack against blank rounds is successful if $S_{blk} = S'_{blk}$.

First, an adversary searches for a point at which two internal states collide; we call this point c. Once internal states pass point c during j times blank rounds, then the internal states remain the same for the rest of the blank rounds, and this leads to a collision. Set two internal states S_{msg}, S'_{msg} which pass the point c at i-th clock of blank rounds, and derive two messages M, M' from S_{msg}, S'_{msg}. Since the adversary has j choices for i, one point c means j collision pairs.

4.2.3 Hash Generation

First, an adversary has to find the difference Δ_{blk} which is canceled during $f(0, S_t, 0) \rightarrow S_{t+1}, o_{t+1}$. Second, he has to find the difference Δ_{msg} which leads to Δ_{blk}. Then he derives collision messages M, M' from S_{msg}, S'_{msg}.

4.2.4 Multi-collision Attack

A. Joux (Joux, 2004) presented a multi-collision attack on iterated hash functions. Constructiong 2^t collisions only costs t times as much as an ordinary collision.

4.3 Second Preimage Attacks

As second preimage attacks, attacks on message injection, blank rounds, and hash generation have to be considered.

4.3.1 Message Injection

In the second preimage attack against message injection, an adversary tries to find another message M' which satisfies $S_{msg} = S'_{msg}$ for the given message M. Two internal states derived from two messages are identical and so are the hash values.

4.3.2 Blank Rounds

By applying j clock of blank rounds internal states S_{blk}, S'_{blk} are obtained from S_{msg}, S'_{msg}. If blank rounds have collision in them, a second preimage attack becomes possible. Similar to the collision attack, an adversary searches for the point c. Given the internal state S_{msg}, and it passes the point c during j clocks of blank rounds, then another internal state S'_{msg} can be found. Thus a second preimage attack against blank rounds can be performed.

4.3.3 Hash Generation

Suppose that h-bit hash is generated and the internal state is updated every one clock of the hash generation. If $|S| > |H|$, then collision occurs by compressing the $|S|$-bit internal state to the $|H|$-bit hash. By using this property, a second preimage attack becomes possible. Specifically, derive internal states S_{blk} and S'_{blk} for given H then a second preimage of M can be found by inverting blank rounds and the message injection.

4.4 Preimage Attacks

Preimage attacks consist of three phases: against the hash generation, blank rounds, and message injection.

In hash generation, since the $|S|$-bit internal state is compressed into the $|H|$-bit hash, it is impossible to derive all bits on S_{blk} only from H.

An adversary tries to invert blank rounds to derive S_{msg} from S_{blk}. Let p be the probability at which all $|S|$ bits of the internal state can be correctly guessed by going through one clock of blank rounds in reverse order. Then the probability to derive S_{msg} from S_{blk} is given as p^j.

The adversary derives the initial state S_0 from the internal state after the message injection S_{msg} in the preimage attack against message injection. Since all bits of S_{msg} are known at this point, it would be possible to guess a message M from S_{msg}.

4.5 Length-extension Attacks

The adversary has to compute the hash value $x = H(M||y)$ without M. Thus, the adversary tries to obtain S_{msg} or S_{blk} for the message $M||y$ using the hash value $H(M)$.

5 CONSTRUCTING A SECURE SCH

In this section, we propose a construction of an SCH which has collision, second preimage, and preimage resistance.

5.1 Collision Resistance

We discuss collision resistance and collision resistant SCHs.

5.1.1 Collision Resistant SCH

We consider the following three cases as collision attacks against SCHs:

1. Against message injection
   Decide message difference which leads to an identical internal state after all message bits are injected.
2. Against blank rounds
   The internal state difference caused by messages is canceled during blank rounds.
3. Against hash generation
   The difference given by messages goes through blank rounds and is canceled during hash generation.

Theorem 1. *Suppose $|S| > |H|$. Given the collision-resistant message injection, SCHs constructed with blank rounds whose period of the transition of the internal state is longer than $2^{|H|}$ and hash generation which uses the whole internal state to derive the hash value. Then such SCHs are collision resistant.*

Proof. It is obvious that the condition $|S| > |H|$ is required. SCHs which satisfy lemmas 1, 2, and 3 are secure against case 1, 2, and 3 attacks. Therefore these SCHs are collision resistant. □

Lemma 1. *If a message injection has collision resistance, then SCHs can withstant a case 1 attack.*

Proof. If there is a collision attack, which is more efficient than a birthday attack, targeting the internal state after message injection, then an adversary can find two different messages M, M' which have the same internal states $S_{msg} = S'_{msg}$. With two identical internal states after the message injection, internal states during blank rounds and hash generation must

be also identical and two hash values collide. Hence message injection has to be collision resistant. □

Lemma 2. *If the assumption in Lemma 1 holds and the internal state transition during blank rounds is updated cyclically and has a period of $2^{|H|}$ or longer, SCHs are secure against case 1 and 2 attacks.*

Proof. If the internal state transition during blank rounds is cyclic and its period is $2^{|H|}$ or longer, there are at least $2^{|H|}$ candidates for the value of the internal state. Therefore the computational cost required for the collision attack must exceed $2^{|H|/2}$. Furthermore, as the internal state is not compressed in blank rounds, collision resistance of blank rounds is guaranteed.

Hence the assumption of Lemma 1 holds and transition of the internal state in blank rounds has longer period than $2^{|H|}$, SCHs are secure against case 1 and 2 attacks. □

Lemma 3. *If assumptions in Lemma 2 hold and the hash value is affected by the whole internal state, SCHs are secure against case 1, 2, and 3 attacks.*

Proof. In the hash generation, if all bits of the internal state affect the hash value, any difference of the internal state is not canceled, and the slight difference of the internal state invokes significant changes in the hash value. This ensures that the hash generation is collision resistant.

Hence if assumptions of Lemma 1 and 2 hold and the hash value is affected by the whole internal state, SCHs are secure against case 1, 2, and 3 attacks. □

5.1.2 Condition for Collision Resistant SCH

Here we discuss how to build a collision resistant stream cipher function which satisfies the condition we presume to prove Theorem 1.

- **Collision Resistant Message Injection.** Suppose that an x-bit internal state can be controlled by an x-bit message and this operation does not affect other bits, then the whole internal state can be controlled with $|S|/x$-bit message. Hence multiple bits of the internal state have to be influenced by one bit message. In addition to this, the message needs to be used more than once to avoid the internal state being controlled.

  By fixing the internal state of the stream cipher function, the computational cost of attacks can be reduced. This method is applied by attacks against Abacus and Boole (Nikolić and Khovratovich, 2008; Wilson, 2008; Nikolić, 2008). Hence the security of an SCH is greatly dependent on whether the adversary can control the internal state or not. We suppose that the message is input into the internal state by xoring with the value in the internal state. Here one bit message is xored with n bits of the internal state at equal intervals. Then the adversary can control up to an $|S|/n$-bit internal state.

  $$2^{\frac{|S|-|S|/n}{2}} = 2^{\frac{|S|(n-1)}{2n}} \geq 2^{|H|/2}. \quad (9)$$

  It can be shown that the fewer points that are used in the message injection, the less computational cost is required from Eq. (9). When $n = 2$, the computational cost of finding a collision is $2^{|S|/4}$, and when $n = |S|$, the cost is $2^{|S|-1}$.

  The following condition is derived for a message injection to be collision resistant;

  $$|S|(n-1) \geq n|H|. \quad (10)$$

  When $n = 1$, the Eq. (10) does not hold. Therefore n has to be equal to or greater than 2, $n \geq 2$.

- **Collision Resistant Blank Rounds.** Suppose that the transition of the internal state in blank rounds has the period which is longer than $2^{|H|}$. If the internal state after the message injection S_{msg} is different, then the internal state after blank rounds S_{blk} is also different, since the number of blank rounds j ($j < 2^{|H|}$) is independent of the message length. Hence the difference in the internal state is not canceled during blank rounds. The transition of the internal state in blank rounds does not always have the period longer than $2^{|H|}$ in order to satisfy the collision resistance. If the period i of the operations performed in blank rounds is larger than the number of blank rounds j and the transition does not have the collision point of the internal state, then such blank rounds imply collision resistance. In order for blank rounds to have the period longer than $2^{|H|}$, a linear feedback shift register (LFSR) of maximum length sequence can be used as one of option. However LFSRs are vulnerable to correlation attacks, hence other approaches should be considered.

  Blank rounds with $i > j$ ensure the collision resistance and second preimage resistance of blank rounds.

- **Collision Resistant Hash Generation.** The internal state is compressed to derive the hash value in the hash generation. In this process, all bits of the internal state have to affect the hash value, otherwise collisions can occur by putting differences on bits which do not affect the hash value. In order for the hash value to be affected by the whole internal state, more than one bit of the state has to

be used to generate a bit of hash value. This can be easily achieved with an n-to-1 filter function. As an example, suppose the internal state size is double the hash size. Then two internal state bits have to be used to generate a 1-bit hash.

5.1.3 Multi-collision Resistance

Here we give a theorem and proof that collision resistant SCHs are secure against multi-collision attacks.

Theorem 2. *Collision resistant SCHs are also multi-collision resistant.*

Proof. Since the internal state size is larger than the hash size. If the internal state of the stream cipher w is large enough, there is no efficient way to find the internal state collision, hence a multi-collision attack does not affect the security of SCHs. □

5.2 Second Preimage Resistance

In this section, we consider constructions of second preimage resistant SCHs.

Theorem 3. *Collision resistant hash functions are also second preimage resistant.*

Proof. Generally a collision resistant hash function is also second preimage resistant (Menezes et al., 1996). A SCH, which satisfies Theorem 1, woudl be second preimage resistant. Hence collision resistant SCHs imply second preimage resistant SCHs. □

5.3 Preimage Resistance

In this section, a preimage attack and preimage resistance will be explained.

5.3.1 Preimage Resistant SCH

Preimage attack against SCH consists of following three steps:

1. Against hash generation
   Find the internal state S_{blk} from the hash value H.
2. Against blank rounds
   Find the internal state S_{msg} from S_{blk}.
3. Against message injection
   Derive the message M from the internal state S_{msg}.

Theorem 4. *If the computational cost of deriving the internal state S_{msg} from S_{blk} is more than $2^{|H|}$, or $|S| > |H|$, then SCHs have preimage resistance.*

Proof. SCHs which satisfy lemmas 4 or 5 are secure against attacks in steps 1 or 2, respectively. When an adversary tries to find a preimage of the message, the adversary has to compute steps 1 to 3, which are described at the beginning of this section. We assumed that $|S| > |H|$ in Theorem 1, therefore step 1 of the attack is infeasible. Hence, collision resistant SCHs imply preimage resistant SCHs. □

Lemma 4. *If the internal state size $|S|$ is larger than the hash size, $|S| > |H|$, then the SCH is secure against the step 1 of the attack.*

Proof. The information of $|S| - |H|$ bits of the internal state will be perished by compressing the $|S|$-bit internal state to the $|H|$-bit hash value. Preimage resistance holds because of this property. □

Lemma 5. *If blank rounds have preimage resistance, then the SCH is secure against step 2 of the attack.*

Proof. Let the probability of recovering the internal state by inverting one clock of blank rounds be p, then inverting j-clock of blank rounds and recovering S_{msg} from S_{blk} can be denoted as p^j. If $p^j < 2^{-|H|}$ holds, then blank rounds are preimage resistant.

Since S_{msg} is known, it is possible to invert message injection to gain the input message, because we have to assume the case where $|M| > |S|$ as well as the case where $|M| \leq |S|$. Hence, message injection with preimage resistance hardly exists. □

5.3.2 Condition for Preimage Resistant SCH

Here we discuss how to build a preimage resistant stream cipher function which satisfies the condition we presume to prove Theorem 4.

There are two ways for blank rounds to be preimage resistant.

1. Collision occurs during blank rounds.
2. For j-clock blank rounds, $p^j < 2^{-|H|}$ holds, where p is the probability of reverting one-clock of blank rounds and guessing the internal state correctly.

The existence of collision in blank rounds means they are vulnerable to collision and second preimage attacks. Hence, method 2 is appropriate and the stream cipher function has to satisfy $p^j < 2^{-|H|}$ with the probability p and the number of blank rounds j.

The existence of collision is inevitable if the size of the internal state is larger than that of the hash value. However, the collision ensures preimage resistance in the hash generation.

5.4 Length-extension Security

We give the theorem and proof that a preimage resistant SCH is secure against the length-extension attack.

Theorem 5. *Preimage resistant SCHs are secure against a length-extension attack.*

Proof. In SCHs, the hash value $H(M||z)$ is computed independently of the hash value $H(M)$, because of blank rounds. If an adversary cannot obtain M from $H(M)$ efficiently, the adversary cannot compute $H(M||z)$. Thus, an SCH is secure against a length-extension attack, where the SCH has preimage resistance. □

6 DISCUSSION

Both attacks against Abacus and Boole exploit the fact that an adversary can easily cause collisions of the internal state by tweaking messages. Moreover, they also exploit the fact that the internal state S_{blk} which produces the desired hash value can be derived. In fact, these ciphers do not satisfy our criterion, i.e., "the message injection has to be collision resistant". Thus, this evidence supports the feasibility of our criterion.

Relations to the attacks against stream ciphers also have to be considered. Internal state recovery attacks such as guess-and-determine attacks (Bleichenbacher and Patel, 1999) and correlation attacks (Hawkes and Rose, 2002; Meier and Staffelbach, 1988) obtain internal state bits from keystream bits. These attacks can be used for preimage attacks on SCHs. That is, an attacker can obtain internal state bits from the hash value if the stream cipher used for the SCH is vulnerable against an internal state recovery attack. Where we select a stream cipher algorithm for constructing an SCH, it is a mandatory condition that the stream cipher is secure against existing attacks. Recovering the internal state of the stream cipher function from h-bit keystream incurs a computational cost of at least $2^{|H|}$. If there is an attack which recovers the internal state of the stream cipher function with a computation cost of less than $2^{|H|}$ computational cost, this attack can lead to a pre-image attack against the SCH.

Distinguishing attacks (Coppersmith et al., 2002) are another possible approach for recovering internal state bits. A linear approximation of the nonlinear process is obtained by a distinguishing attack and it is applicable to recovering internal state bits. The internal state bits can be determined to solve a system of linear equations. The attacks require a much longer keystream than the h-bit hash value, hence, it is rarely applied to a pre-image attack on the SCH.

Next, we discuss relationships between two specific attacks on stream ciphers and our criteria for constructing SCHs. Two types of attack as shown in Sect. 4 are relevant to collisions in the blank round. For example, the chosen IV attack against Py and Pypy corresponds to the collision attack against blank rounds in an SCH. In the chosen IV attack against Py and Pypy, identical keystreams can be generated from different IVs. Keystreams correspond to hash values and IVs correspond to internal states. Initialization of Py and Pypy and blank rounds in an SCH are basically the same operations, hence this attack can be applied to collision and second preimage attack against an SCH. From this view-point, we should use a stream cipher that is secure against the attacks when constructing an SCH. Furthermore, security against these attacks is strongly related to our criterion; blank rounds have to be executed as the transition of the internal state has the longer period than $2^{|H|}$.

An attack in which the adversary tries to control the output keystream by inputting a certain ciphertext can be considered to be an attack against an SSSC (Joux and Muller, 2003; Joux and Muller, 2006). This type of attacks can be applied to attacks against blank rounds of SCHs as discussed in Sect. 4.1. Hence, the stream cipher functions have to resist this type of attacks in order to be collision and second preimage resistant.

7 CONCLUSIONS

In this paper, we introduced a model of SCH, consisting of a pre-computation function and a stream cipher function. Then we defined the stream cipher function with pre-computation as the one which updates an internal state and outputs a hash value from three inputs: a message, an internal state, and feedback. Then we showed that the keystream feedback mode and the self-synchronizing stream cipher are equivalent to SCHs. Therefore, attacks against these stream ciphers can be various attacks against SCHs. Furthermore we described each phase of SCH with the function we defined, and considered attacks against each phase. As a result of our work, the following necessary conditions can be obtained:

- $|S| > |H|$ is required for collision, second preimage, and preimage resistance.
- The computational cost of the collision attack on message injection has to be equal or more than $2^{|H|/2}$, and an adversary is not able to control the internal state of the stream cipher function.

- The internal state transition during blank rounds has a period of at least $2^{|H|}$.
- Deriving the internal state after the message injection by going through one clock of blank rounds in reverse order incurs a computational cost of $2^{|H|}$.
- When one bit message is xored with n bits of the internal state at equal intervals, then the condition $|S|(n-1) \geq n|H|$ is required for message injection to be collision resistant, where n is the number of bits of the internal state which are xored with one bit message.

In this paper we proposed the conditions for secure SCHs. We showed that conditions for second preimage and preimage resistant SCH are included in the conditions for a collision resistant SCH. Furthermore, the condition for length-extension security is included in the condition for the preimage resistant SCH. Thus, we can focus on the conditions for the collision resistant SCH, when we design the SCH.

Consideration of a concrete algorithm for a collision resistant SCH should be the subject of future work. Especially, we will consider a secure design of pre-computation and message injection mechanism for a SCH in the next stage.

REFERENCES

Biham, E. and Seberry, J. (2005). Py : A Fast and Secure Stream Cipher using Rolling Arrays. eSTREAM, ECRYPT Stream Cipher Project.

Biham, E. and Seberry, J. (2006). Pypy: Another Version of Py. eSTREAM, ECRYPT Stream Cipher Project.

Bleichenbacher, D. and Patel, S. (1999). Sober crytanalysis. In Knudsen, L. R., editor, *FSE*, volume 1636 of *LNCS*, pages 305–316. Springer.

Coppersmith, D., Halevi, S., and Jutla, C. S. (2002). Cryptanalysis of stream ciphers with linear masking. In Yung, M., editor, *CRYPTO*, volume 2442 of *LNCS*, pages 515–532. Springer.

Ekdahl, P. and Johansson, T. (2002). A New Version of the Stream Cipher SNOW. In Nyberg, K. and Heys, H. M., editors, *SAC*, volume 2595 of *LNCS*, pages 47–61. Springer.

Golić, J. D. (2001). Modes of Operation of Stream Ciphers. In Stinson, D. R. and Tavares, S. E., editors, *SAC*, volume 2012 of *LNCS*, pages 233–247. Springer.

Hawkes, P. and Rose, G. G. (2002). Guess-and-Determine Attacks on SNOW. In Nyberg, K. and Heys, H. M., editors, *SAC*, volume 2595 of *LNCS*, pages 37–46. Springer.

Joux, A. (2004). Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions. In Franklin, M. K., editor, *CRYPTO*, volume 3152 of *LNCS*, pages 306–316. Springer.

Joux, A. and Muller, F. (2003). Loosening the KNOT. In Johansson, T., editor, *FSE*, volume 2887 of *LNCS*, pages 87–99. Springer.

Joux, A. and Muller, F. (2006). Chosen-Ciphertext Attacks Against MOSQUITO. In Robshaw, M. J. B., editor, *FSE*, volume 4047 of *LNCS*, pages 390–404. Springer.

Meier, W. and Staffelbach, O. (1988). Fast Correlation Attacks on Stream Ciphers. In *EUROCRYPT*, volume 330 of *LNCS*, pages 301–314. Springer.

Mendel, F., Nad, T., and Schläffer, M. (2009). Collision Attack on Boole. In Abdalla, M., Pointcheval, D., Fouque, P.-A., and Vergnaud, D., editors, *ACNS*, volume 5536 of *LNCS*, pages 369–381.

Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A. (1996). *Handbook of Applied Cryptography*. CRC Press.

Nikolić, I. (2008). Preimage attack on Boolen. The ECRYPT Hash Function Website, http://ehash.iaik.tugraz.at/uploads/2/2f/Boole.pdf

Nikolić, I. and Khovratovich, D. (2008). Second preimage attack on Abacus. available online, http://lj.streamclub.ru/papers/hash/abacus.pdf

NIST (1995). Secure hash standard. FIPS180-1.

Rivest, R. (1992). The MD5 message digest algorithm. RFC1321.

Rose, G. G. (2008). Design and primitive specification for Boole. submission to NIST, http://seergrog.net/BoolePaper.pdf

Sholer, N. (2008). Abacus a candidate for SHA-3. submission to NIST, http://ehash.iaik.tugraz.at/uploads/b/be/Abacus.pdf

Wang, X., Yin, Y. L., and Yu, H. (2005). Finding collisions in the full SHA-1. In Shoup, V., editor, *CRYPTO*, volume 3621 of *LNCS*, pages 17–36. Springer.

Wang, X. and Yu, H. (2005). How to break MD5 and Other Hash Functions. In Cramer, R., editor, *EUROCRYPT*, volume 3494 of *LNCS*, pages 19–35. Springer.

Wilson, D. (2008). A second-preimage and collision attack on Abacus. available online, http://web.mit.edu/dwilson/www/hash/abacus_attack.pdf

Wu, H. and Preneel, B. (2007). Differential Cryptanalysis of the Stream Ciphers Py, Py6 and Pypy. In Naor, M., editor, *EUROCRYPT*, volume 4515 of *LNCS*, pages 276–290. Springer.