Length extension attack on narrow-pipe SHA-3 candidates

Danilo Gligoroski

Department of Telematics, Norwegian University of Science and Technology,
O.S.Bragstads plass 2B, N-7491 Trondheim, NORWAY
danilo.gligoroski@item.ntnu.no

Abstract. In this paper we show that narrow-pipe SHA-3 candidates BLAKE-32, BLAKE-64, Hamsi, SHAvite-3-256, SHAvite-3-512, Skein-256-256 and Skein-512-512 do not provide n bits of security where n is the hash output size. The actual security against length extension attack that these functions provide is $n-k$ bits of security, where k is an arbitrary value chosen by the attacker who wants to perform one-time pre-computation of 2^{k+1} compression functions. The attack can be in two variants: 1. The attacker is not collecting the hash values given by the user or 2. The attacker is collecting the hash values given by the user. In any case, the attacker does not know the content of the hashed messages. The optimal value for this attack from the perspective of minimizing the number calls to the compression function and increasing the probability of the successful attack is achieved when k has a value $k = \frac{n}{2}$, thus reducing the security against the length-extension attack from n to $\frac{n}{2}$ bits.

1 Introduction

The usefulness of the concept of the cryptographic hash functions have been confirmed in practice with the fact that they have became the fundamental building part of the modern cryptography and information security and their presence is evident in numerous protocols and schemes such as: digital signatures, commitment schemes, password protection schemes, in algorithms for checking the data integrity, key derivation functions and cryptographic random number generators, authentication schemes and many others.

The most used family of hash functions is the family called "SHA", as a worldwide accepted industry standard for cryptographic hash functions. They have been designed by the National Security Agency (NSA) and published by the National Institute of Standards and Technology (NIST) as a U.S. Federal Information Processing Standard [1, 2]. The acronym SHA stands for Secure Hash Algorithm. There are two types of SHA algorithms: SHA-1, and SHA-2, and although they have some similarities,

they have also significant differences. SHA-1 is the most used member of the SHA hash family, employed in countless different applications and protocols. However, in 2005, a very significant theoretical development in detecting some security flaws in SHA-1 has been made by Wang et.al [3].

SHA-2 is actually a family of its own, consisting of four algorithms that differ from each other by different digest size, different initial values and different word size. The digest sizes are: 224, 256, 384 and 512 bits. Although no attacks have yet been reported on the SHA-2 variants, they are algorithmically similar to SHA-1, and NIST have felt the need for and made efforts to develop an improved new family of hash functions [4]. The new hash standard SHA-3, is currently under development - the function will be selected via an open competition running between 2008 and 2012. In the First Round there were 51 proposals [5] and in July 2009 NIST has chosen 14 Second Round candidates [6].

In their call for the SHA-3 competition [4], NIST has defined several security requirements such as collision resistance of $\frac{n}{2}$ bits, preimage resistance of n bits, resistance against second preimages of 2^{n-k} bits for messages long 2^k bits and resistance against length-extension attack. However, in the SHA-3 call there is no clear statement how many bits of security should SHA-3 candidates provide against length extension attack.

On my request for clarification submitted to the SHA-3 hash forum list on 12 December 2008, I got the following answer by the NIST representative submitted to the hash forum list on 14 January 2009:

> "We expect the winning n-bit hash function to be able to provide n bits of security against length extension attacks. That is, given $H(M)$, with M wholly or partially unknown to the attacker: the cost of finding (Z, x) so that $x = H(M||Z)$ should be greater than or equal to either the cost of guessing M or 2^n times the cost of computing a typical hash compression function."

In this paper we will show that four SHA-3 candidates that are narrow-pipe designs do not provide n bits of security. Those narrow-pipe designs are: BLAKE-32 and BLAKE-64 [8], Hamsi [9], SHAvite-3-256 and SHAvite-3-512 [10], and narrow-pipe Skein versions (Skein-256-256 and the primary submission variant of Skein, Skein-512-512) [11].

2 A generic modeling of the narrow-pipe iterative finalization

In order to launch a length-extension attack to the narrow-pipe designs we will need to model the finalization of the iterative process that narrow-pipe designs do when they are processing messages. Moreover, we will assume that the length of the digested messages is such that the final processed block does not have any bits from the message but is a constant $PADDING$ that consist only from the bits defined by the padding rule of that hash function.

In that case, the modeling of the narrow-pipe hash designs can be expressed by the following expression:

$$H = f(parameters, \mathbf{compress}(H_{chain}, parameters, PADDING)) \quad (1)$$

and is graphically described in Figure 1.

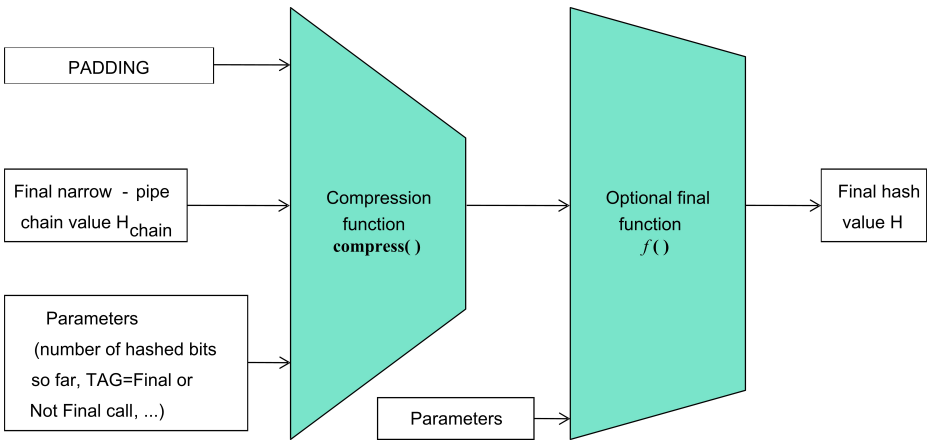

Fig. 1. A graphic representation of narrow-pipe hash designs finalization of the iterative process of message digestion.

Note that in the narrow-pipe designs where the final function $f()$ is missing, we can treat it as the identity function in the expression (1) and that although the parts *parameters* are different for all four designs, it will not change our analysis and our attack.

How narrow-pipe designs are protected from the length-extension attack?

Since the designers of narrow-pipe hash functions are designing the compression function as one-way pseudo-random function, the value H_{chain},

which is the internal state of the hash function, is hidden from the attacker. That means that by just knowing the final hash value H it is infeasible for the attacker to find the preimage H_{chain} that has produced that value H. Consequently, the attacker will have a difficulty to produce a message Z such that only by knowing the value H (where $H = Hash(M)$ and the message M is unknown to the attacker), he/she can produce a valid hash value $x = Hash(M||Z)$.

In what follows our goal will be to show that the attacker can recover that internal state H_{chain} with much less complexity than 2^n calls to the compression function - the complexity that NIST requires in order to claim that the design is offering n bits of security against the length-extension attack.

A generic length-extension attack on narrow-pipe hash functions
1. **One time pre-computation phase** **Step 0.** Fix the length of the messages such that the $PADDING$ block does not posses any message bits. **Step 1.** Produce 2^k pairs (h_{chain}, h) for random values h_{chain} with the expression: $h = f(parameters, \textbf{compress}(h_{chain}, parameters, PADDING))$. This phase has a complexity of 2^k calls to the compression function (or 2^{k+1} calls if the design has a final transformation $f()$). 2. **Query (attack) phase** **Step 2.** Ask the user to produce a hash value $H(M)$ where M is unknown (but its length is fixed in **Step 0**). **Step 3.** If there exists a pre-computed pair (h'_{chain}, h') such that $H(M) = H = h'$, put $H_{chain} = h'_{chain}$, put whatever message block Z and produce a valid $x = H(M||Z)$.

Table 1. A generic length-extension attack on narrow-pipe hash functions

3 Generic length extension attack on narrow-pipe SHA-3 candidates

Our attack is based on the old Merkle's observation [7] that when an adversary is given 2^k distinct target hashes, (second) preimages can be found after hashing about 2^{n-k} messages, instead of expected 2^n different messages. In our attack we use the Merkle's observation not on the whole hash function, but on the two final invocations of the compression func-

tion. In order our attack to work, we will assume that the length of the messages is such that after the padding, the final padded block is without any message bits (which is usual situation when the length of the message is a multiple of 256, 512 or 1024 bits).

A generic description of the length-extension attack on narrow-pipe hash functions is given in Table 1.

Proposition 1. *The probability that the condition in* **Step 3** *is true is* $\frac{1}{2^{n-k}}$.

Proof. The proof is a trivial application of the ratio between the volume of the pre-computed pairs (h_{chain}, h) which has a value 2^k and the volume of all possible hash values of n bits which is 2^n. □

Proposition 2. *For the conditional probability that the corresponding* h'_{chain} *is the actual chaining value* H_{chain} *the following relation holds:*

$$P(H_{chain} = h'_{chain} \mid H(M) = h') \geq 0.58 \times 2^{k-n} \approx 2^{k-n-0.780961}. \quad (2)$$

Proof. (Sketch) It is sufficient to notice that for an ideal random function $g : \{0,1\}^n \to \{0,1\}^n$ that maps n bits to n bits, the probability that an n-bit value has m preimages for the first 8 values of m is approximately given in the Table 2 (the precise analytical expressions for the given probabilities can be a nice exercise in the Elementary Probability courses).

Number of preimages m	Probability P
0	0.36787
1	0.36787
2	0.18394
3	0.06131
4	0.01533
5	0.00307
6	0.00051
7	0.00007

Table 2. The probabilities an n bit value to have m preimages for an ideal random function $g : \{0,1\}^n \to \{0,1\}^n$.

The relation (2) follows directly from Proposition 1 and from Table 2. □

Corollary 1. *After approximately* $2^{n-k+0.780961}$ *queries the attacker should expect one successful length extension.* □

Corollary 2. *The security of narrow-pipe hash designs is upper bounded by the following values:*

$$\max(2^{\frac{n}{2}}, 2^{n-k+0.780961}), \tag{3}$$

where $k \leq n$.

Proof. The minimal number of calls to the compression function of the narrow-pipe for which the attack can be successful is achieved approximately for $\frac{n}{2}$. □

The interpretation of the Corollary 2 is that narrow-pipe hash designs do not offer n bits of security against length-extension attack but just $\frac{n}{2}$ bits of security.

4 Why wide-pipe designs are resistant to our attack?

A natural question is raising about the security of wide-pipe hash designs and their resistance against the described attack in this paper. The reason of the success of our attack is the narrow size of just n bits of the hidden value H_{chain} that our attack is managing to recover with a generic collision search technique.

Since in the wide-pipe hash designs that internal state of the hash function has at least $2n$ bits, the search for the internal collisions would need at least 2^n calls to the compression function which is actually the value that NIST needs for the resistance against the length-extension attack.

5 Conclusions

When it comes to the properties of the hash functions designed by humans, that are trying to mimic the expected properties of ideal random functions (i.e mimicking the random oracle model [12]) the narrow-pipe designs are showing pretty big number of abberations from that model. In this paper we have showed that they are not giving n bits of security against the length-extension attack, that NIST is demanding from SHA-3 candidates. Narrow-pipe designs are offering just $\frac{n}{2}$ bits of security, while in the same time, wide-pipe (double-pipe) hash designs are offering the requested security of n bits against the length-extension attack.

Acknowledgement

I would like to thank Vlastimil Klima, Rune Jensen, Svein Johan Knapskog and Rune Ødegård for their useful comments and suggestions to improve the clarity of text.

References

1. FIPS 180-1, "Secure Hash Standard", Federal Information Processing Standards Publication 180-1, U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, Virginia, April 1995.
2. FIPS 180-2, "Secure Hash Standard", Federal Information Processing Standards Publication 180-2, U.S. Department of Commerce/NIST, National Technical Information Service, Springfield, Virginia, August 2002.
3. X. Wang, Y. L. Yin, H. Yu, "Collision Search Attacks on SHA-1", CRYPTO 2005, LNCS 3621, pp. 17–36, 2005.
4. National Institute of Standards and Technology: "Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family". Federal Register, 27(212):62212–62220, November 2007. Available: `http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf` (2009/04/10).
5. NIST, SHA-3 First Round Candidates , `http://csrc.nist.gov/groups/ST/hash/sha-3/Round1/submissions\_rnd1.html`
6. NIST, SHA-3 Second Round Candidates , `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/submissions\_rnd2.html`
7. R. C. Merkle - Secrecy, authentication, and public key systems, Ph.D. thesis, Stanford University, 1979, pp. 12 - 13, `http://www.merkle.com/papers/Thesis1979.pdf` (2010/08/08).
8. Jean-Philippe Aumasson, Luca Henzen, Willi Meier, Raphael C.-W. Phan: "SHA-3 proposal BLAKE, Submission to NIST (Round 2)". Available: `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/documents/BLAKE_Round2.zip` (2010/05/03).
9. Özgül Kücük: "The Hash Function Hamsi, Submission to NIST (Round 2)". Available: `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/documents/Hamsi_Round2.zip` (2010/05/03).
10. Eli Biham and Orr Dunkelman: "The SHAvite-3 Hash Function, Submission to NIST (Round 2)". Available: `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/documents/SHAvite-3_Round2.zip` (2010/05/03).
11. Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, Jesse Walker: "The Skein Hash Function Family, Submission to NIST (Round 2)". Available: `http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/documents/Skein_Round2.zip` (2010/05/03).
12. M. Bellare and P. Rogaway: "Random oracles are practical: A paradigm for designing efficient protocols," in CCS 93: Proceedings of the 1st ACM conference on Computer and Communications Security, pp. 6273, 1993.