SHA-3 Submission: Skein Hash Function Family

Jarrick Goldhamer

Abstract— **Moore's law defines the approximate life span of modern factoring based security algorithms. As such, there is a constant need for new and improved standards in hash functions. The next generation, SHA-3 hash functions are currently under international review to find secure algorithms that provide signifcantly greater security and improved efficiency over SHA-2 hash functions. The following paper examines the Skein Hash Function, one of the 14 candidates chosen to be part of the second round of SHA-3 examination.**

I. Introduction

The Skein Hash Function is based upon a tweakable block cipher named Threefish, a Unique Block Iteration (UBI) and an Optional Argument System [1]. This design allows for messages of arbitrary length to be compressed to fixed output sizes. Many precautions were taken in the development of Skein to protect against known cryptanalytic attacks used against existing block ciphers. The core of the Skein proposal's security is a minimum of 72 rounds of the MIX function of Threefish which only uses basic arithmetic operations and shifts to allow for quick and efficient implementations on a wide variety of platforms, while being optimized for modern 64-bit processors [1]. With no table look-ups involved, each round using constant time and most computations capable of being completed in a modern processor's registers, Threefish is not likely to leak any information to current side channel attacks [1].

II. Design

The Skein Hash Function was broken down into three basic components by its developers:

- Threefish: a tweakable block cipher at the core of Skein, defined with a 256-, 512-, and 1024-bit block size [1].

- Unique Block Iteration (UBI): a chaining mode that uses Threefish to build a compression function that maps an arbitrary input size to a fixed output size [1].

- Optional Argument System: to allow Skein to support a variety of optional features without imposing any overhead on implementations and applications that do not use the features [1].

A. *Threefish*

Threefish has three defined block size modes of 256-bits, 512-bits, and 1024-bits [1]. The key length for each must be equal to the block size and the tweak is always 128-bits [1]. At the heart of Threefish is the notion that increasing the number of rounds is more secure than increasing the complexity of rounds when considering diffusion observed with comparable time performance.

Authors are with the Department of Computer Science, University of California, Santa Barbara, CA 93106. E-mail: jgoldhamer@cs.ucsb.edu

A.1 The MIX function

The MIX function of Threefish is an addition, a shift and an XOR of two 64-bit words. These three functions were chosen due to the fact that this combination can be computed in a single clock cycle on modern processors [1]. (Other designs included 2 rotates, but the designers found that this was not possible in one clock cycle despite documentation saying otherwise [1].) For both 256 and 512-bit modes, Threefish runs through 72 rounds with a subkey injected every four rounds [1]. Each round consists of four MIX functions followed by a permutation of each of the 64-bit words [1]. The rotation constants and round permutations are different for each version of ThreeFish, but the permutation remains the same for each iteration. For the 1024-bit mode, each round consists of eight MIX functions and it completes 80 rounds [1]. The added rounds are due to the fact that complete diffusion requires one extra round for the this size. Each input bit position affects every output bit position in 10 rounds for Skein-512 (9 rounds for Skein-256 and 11 rounds for Skein-1024).

A.2 Key Scheduling

The Three fish key schedule was inspired by the simplicity of the Skipjack block cipher [1]. Skipjack has an 80 bit key and 64 bit data blocks and was developed by the NSA [2]. Three Fish relies on a 128 bit tweak, a key equal to the length of the output block size and a counter to produce the subkeys [1]. The key scheduler begins by creating two additional words. One which is the XOR of the two words of the tweak value and one that is the XOR of all the words of the key XOR with the constant $\frac{2^{64}}{3}$ to ensure this is not all zeros [1]. These words are then used in conjunction with a counter to create the subkeys based on addition modulo 2^{64} and injected after ever four rounds in Skein. Each subkey is a combination of all but one of the extended key words, two of the three extended tweak words, and the subkey number [1]. The key schedule is defined as follows [1]:

$$\begin{array}{ll}
k_{s,i} := k_{(s+i)\ mod\ (N_w+1)} & \text{for i - 0,...,}N_w\text{-4} \\
k_{s,i} := k_{(s+i)\ mod\ (N_w+1)} + t_{s\ mod\ 3} & \text{for i = } N_w \text{ - 3} \\
k_{s,i} := k_{(s+i)\ mod\ (N_w+1)} + t_{s+1\ mod\ 3} & \text{for i = } N_w \text{ - 2} \\
k_{s,i} := k_{(s+i)\ mod\ (N_w+1)} + \text{s} & \text{for i = } N_w \text{ - 1}
\end{array}$$

Where $k_{s,i}$ represents the words of subkey s, N_w represents the number of words in the state and t_i represents the words of the tweak [1].

Due to the interdependence of key, tweak and subkey number, three interesting qualities emerge about key regeneration:

• With a known tweak and subkey number it is possible to extract the full key [1].

• With a known key and subkey number it is possible to extract the full tweak [1].

• With any two consecutive subkeys, it is possible to extract the full key, tweak, and subkey number [1].

B. Unique Block Iteration

The Unique Block Iteration (UBI) allows for messages of arbitrary length to be hashed to fixed output sizes. The message is broken into predetermined block sizes (64 bytes for Skein-512) and padded to fit accordingly [1]. Each block is encoded with its tweak value and the previous block output in a chain to get the result. The tweak value of each block includes information about first/last block and how many bytes have been encoded previously which ensures every block will be encoded differently even if the message inputs are the same [1]. The tweak value also includes information about the many different types of UBI mode that store the information about the Optional Argument System.

C. Optional Argument System

The tweak's type bits can be used to inform which of eight options are being implemented in a given use of Skein. The two required aspects are the following:

• The configuration clock: a 32-byte configuration string with information about the desired output length and tree hashing options.

• The output transformation: allows for output length up to 2^{64}. The output transformation is the aspect of Skein which guarantees randomness [1].

Beyond these are six optional aspects that can be triggered in the tweak type. They include allowing for Message Authentication Code or a Key Derivation Function, Personalization and Public Keys [1]. More options are available and individual implementations of Skein can implement new modes that can be triggered with their own unique type arguments [1].

III. Security

The first decision in creating a new hash function to was the type of cipher to use. The authors of Skein decided to create a new block cipher because they are well understood and the core aspects of their security have been analyzed in the past. In order to ensure security the following implementation choices were made:

• Threefish uses a large enough block size to eliminate all collision-style attacks and provide high security even when processing large amounts of data [1].

• The design of the MIX function avoids look-up tables to avoid side channel attacks stealing the cipher key [1].

• Each round is of constant time to avoid leaking information to timing attacks [1].

• The input to the UBI hash function is the same as the plaintext input to the block cipher. Since the attacker has the greatest control over the message input, this provides an additional level of security [1].

• For digital signatures, the option of hashing the public key. This forces message hashes to depend on the public key, and proves that someone with access to the actual document intended to have it signed by that key. The public key in the input to the hash also means that in the case of the discovery of a collision finding attack on the hash function, the attacker has to reinvest effort for every public key that it wants to attack [1].

IV. Attacks

There have been many attempts to find weakness in the Skein Hash Function during the review periods. Due to the simplicity of the MIX function, a majority of attempted attacks focus on versions of reduced round Threefish. These attempts rely mostly on properties of systems comprised of only additions, rotations and XORs.

A paper which claims to be the first third-party analysis of Skein, investigated near collisions, distinguishers, impossible differentials, key recovery using related-key differential and boomerang attacks [3]. The results were based on the original proposal which has since been revised with some new rotational constants in the latest version. The authors admit that none of their attacks extend to the full implementation defined in the Skein algorithm, but a summary of their results can be seen in table I.

TABLE I
Attacks of Original Skein Proposal

Rounds	Time	Type
16	2^{6}	459-bit near-collision
17	2^{34}	434-bit near-collision
21	$2^{3.4}$	459-bit near-collision
21	–	related-key impossible differential
25	$2^{416.6}$	related-key key recovery
26	$2^{507.8}$	related-key key recovery
34	2^{398}	related-key boomerang distinguisher
35	2^{478}	known-related-key boomerang distinguisher

A later paper, Improved Related-Key Boomerang Attacks on Round-Reduced Threefish-512 by Jiazhe Chen and Keting Jia [4], reveals results on similar attacks taking into account the new rotational constants. By fixing some bit pairs and running their distinguisher sufficiently many times they succeeded in recovering one key bit at a time. Their results up to 34 rounds of Threefish can be seen in Table II, further rounds were not able to be achieved with this method [4].

TABLE II
Related-Key Boomerang Key Recovery With New Rotational Constants

Rounds	Time	Keys
32	2^{195}	4
33	$2^{324.6}$	4
34	$2^{474.4}$	4

A paper by Dmitry Khovratovich and Ivica Nikolić [5] explores the security of such systems and outline attacks on reduced or simplified versions of Threefish. A key component to their analysis is that addition provides diffusion and non-linearity, while XOR does not [5]. While the developers of Skein chose to design which relies on the security of a massive amount of rounds, Khovratovich and Nikolić point out that this could be a false security. They state that although it is a common belief that the mixture of addition, rotation and XOR for many rounds is sufficient, there is no formal theory whether all three operations are sufficient for the task [5]. The complexity of their rotational attack depends only on the number of modular additions, and does not depend on the number of XORs and rotations, nor on the rotation amounts [5].

V. Conclusions

After reviewing the available information on the implementation and security of the Skein Hash Function proposal, it appears to be a strong candidate for consideration by the National Institute of Standards and Technology. The algorithm was cleverly designed to have time constant rounds and avoid table look-ups in order to avoid many security flaws known today, but is also capable of being implemented very quickly on a wide variety of platforms due to the simplicity of the MIX function. The designer's choice to optimize their algorithm's performance for 64-bit platforms represents the necessary progression needed in new standards. The Optional Argument System and its ability to allow the algorithm to easily adapt to new and advanced uses is another component which gives Skein an edge on the competition. Finally is Skein's security, although block ciphers are nothing new in the world of cryptography, this algorithm provides plenty of security against any known computer system today.

References

[1] N. Ferguson, S. Lucks, B. Scheiner, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker, "The Skein Hash Function Family," Sept. 2009, `http://www.schneier.com/skein1.2.pdf`.

[2] US Department of Commerce Technology Administration/National Institute of Standards and Technology, "Skipjack and kea algorithm specifications," May 1998, `http://csrc.nist.gov/groups/ST/toolkit/documents/skipjack/skipjack.pdf`.

[3] J. Aumasson, Ç. Çalk, W. Meier, O. Özen, R. Phan, and K. Varc, "Improved Cyptanalysis of Skein," Sept. 2009, `http://eprint.iacr.org/2009/438.pdf`.

[4] J. Chen and K. Jia, "Improved Related-Key Boomerang Attacks on Round-Reduced Threefish-512," *Cryptology ePrint Archive*, 2009, `http://eprint.iacr.org/2009/526.pdf`.

[5] D. Khovratovich and I. Nikolić, "Rotational Cryptanalysis of ARX," 2010, Fast Software Encryption conference, `http://www.skein-hash.info/sites/default/files/axr.pdf`.