Theoretical Survey on Secure Hash Functions and issues

K. Saravanan [1] and A. Senthilkumar [2]
[1] Research scholar, Anna University Chennai and Assistant Professor, Department of Electronics and Communication, Nehru institute of technology, Coimbatore, Tamilnadu, India

[2] Professor and Head, Department of Electrical and Electronics, Dr. Mahalingam college of Engineering and Technology, Pollachi, Tamilnadu, India

Abstract

Security is a very important issue, which has attracted the interest of the research community, at a great factor. Hash functions belong to the category of encryption algorithms and are included in almost all cryptographic schemes and security protocols for providing authentication services. Cryptographic hash functions are primitives and building blocks that are used to provide information security. A cryptographic hash function is one which converts an input data of arbitrary length into a fixed-length output. Bloom filters are hash based structures which have a certain degree of accuracy for considerable savings in memory. Research is being done to increase the performance by modifying the structure of hash functions and enabling it to operate in the increasing network speeds, thus variant Hashing algorithms are being introduced. The aim of this paper is to survey the cryptographic hashing algorithms and applications to benefit the research community.

Keywords: *Security, Hash function, Cryptographic hash function, signature, FPGA.*

1. Introduction

Hash function gained high significance for their role in effective mapping of elements in hash tables. Basic functions of hash tables are insertion, look-up and deletion of data records. Various applications of hash function are digital signature, data integrity, password verification and other cryptographic protocols. Cryptographic hash functions are primitive building blocks utilized in the schemes that are used to provide information security. The cryptographic hash functions on their own do not typically provide full information security; however, they play a critical role in the schemes that do provide information security. Hence the security and speed of the cryptographic hash function can significantly impact the overall security and computational efficiency of an information security scheme. A cryptographic hash function is one which converts an input data of arbitrary length into a fixed-length output. Mathematically, a hash function [1] is defined as follows,

$$H: \{0, 1\}^* \rightarrow \{0, 1\}^n$$

In this notation, $\{0, 1\}^*$ refers to the set of binary elements of any length including the empty string while $\{0, 1\}^n$ refers to the set of binary elements of length n. Thus, the hash function maps a set of binary elements of arbitrary length to a set of binary elements of fixed length.

As mentioned earlier, hash functions are used in certain information security schemes like password storage application, digital signatures, Message Authentication Codes (MACs) and digital image watermarking. In password storage application, the password entered by a user at the first log-in is not stored in the computer system; rather the hash of the password is stored. In digital signatures, hash functions are used to improve the efficiency (speed) and reduce the bandwidth of the scheme. The digital signature provides a means of demonstrating the authenticity of a message. A Message Authentication Code (MAC) can be used to verify that a received message is identical to the one that was sent. That is, it can be used to verify that a message has not been corrupted or manipulated in transit. Another important application of hash functions is found in digital image watermarking. Digital image watermarking is the process of embedding information into a digital image. This serves the purpose of facilitating the detection of image manipulation.

2. SHA algorithms

The SHA [2] series of algorithms stand for "Secure Hash Algorithm" they were developed by NIST. Due to the avalanche effect even a small change in the data to be encrypted will probably result in a very different hash string. Because the SHA algorithms show signs of the avalanche effect they are believed to have quite a good randomization feature. SHA algorithms were based upon the MD4 and MD5 algorithms developed by Ron Rivest. SHA was released by the national security authority as a US government standard.

2.1 SHA-0

SHA-0 is officially known as SHA, it was the first incarnation of the secure hashing algorithm. This first version was withdrawn soon after release due to weaknesses in the design. SHA-1 was released a couple of years later that fixed these problems.

2.2 SHA-1

SHA-1 is a popular hashing algorithm released in 1994, and was developed by NIST. SHA-1 is similar to MD4 and MD5 hashing algorithms, and is more secure Also it is considered as MD5's successor. SHA-1 produces a 160 bit hash. The SHA-1 algorithm is featured in a large number of security protocols and applications.

2.3 SHA-2

SHA-2 is based closely upon the SHA-1 Algorithm. SHA-2 actually combines the SHA-224, SHA-256, SHA -384 and SHA-512 algorithms.

3. MD hash functions

The MD [3] family of hashing algorithms was designed by Ron Rivest during the late 1980's and early 1990's. MD actually stands for Message Digest.

3.1 MD2

MD2 was optimized to run on 8-bit computers and generates a 128-bit hash value. The hashes are generally displayed as hexadecimal string which is 32 characters long.

3.2 MD5

It is basically a secure version of MD4 which is a little faster than MD5.MD5 has been widely used as a secure hash algorithm particularly in Internet-standard message authentication. The algorithm takes as input a message of arbitrary length and produces as output a 128-bit message digest of the input.MD5 hash function is mainly intended for digital signature applications. The processing involves the following steps: padding, appending length, initialize the MD buffer, process message in 16-word blocks and output function.

4. Haval hash function

HAVAL is a one-way hashing algorithm that produces output message digests of 128, 160, 192, 224 and 256-bits [4]. The 1024-bit input message has different number of passes that can be 3, 4, or 5. Combination of the two variable parameters, pass and output length, provides different levels of security. Two messages will collide each other when they are compressed to the same message digest. There are two possibilities for a pair of message to collide in a HAVAL function: the number of passes the messages is processed can be the same or differ and also like many others hashing algorithms HAVAL could not be formally proved to be secure. HAVAL operation defines at least three passes.

5. Tiger hash function

Tiger [5] is a cryptographic hash function proposed by Anderson and Biham in 1996. This hash function processes 512-bit blocks and produces a 192-bit hash value. The tiger hash function uses a block-cipher-based compression function and the Davies- Mayer-type feed-forward structure [6].The 8-bit input and 64-bit output S-boxes of Tiger for the compression function provide faster diffusion in comparison with integer arithmetic. It consists of two blocks namely, key schedule and state update transformation block. The key schedule is an invertible function in which changing a small number of bits in the message will affect a lot of bits in the next pass. In state update transformation of Tiger hash function, in each eight rounds, one 64-bit word X is used to update the three state variables (A, B and C).
The basic attack strategy on Tiger, based on the work of Kelsey and Lucks on round -reduced variants involves the following steps: Finding a good Characteristic for the Key Schedule of Tiger and Message Modification by Meet-in-the-Middle.

6. Cuckoo hash function

Cuckoo Hashing [11] is a dynamization of a static dictionary .The dictionary uses two hash tables, T1 and T2, each consisting of r words, and two hash functions h1, h2 : $U = \{0 \ldots r-1\}$. Every key x, two values are stored either in cell h1(x) of T1 or in cell h2(x) of T2, but never in both. Cuckoo Hashing function method insertion and deletion is used to move the keys either to the table or from the table. It is based on the possibilities of the moving keys. Two way hashing method are used to two instances of Chained hashing. A key is inserted in one of the two hash tables, namely the one where it hashes to the shorter chain. If the

chain is long re-hash is needed. Finally the large amount of space size is needed for dynamic perfect hashing.

7. Whirlpool hash function

The Whirlpool hash function [7] was chosen as part of the New European Schemes for Signatures, Integrity and Encryption (NESSIE) in February 2003.The Whirlpool hash function produces an output of 512-bits and has been included in the ISO/IEC 10118-3 standard. Whirlpool, which was designed by Baretto and Rijmen, is a collision-resistant hash function which produces a hash code of 512 bits for an input message of maximum length less than 2^256 bits. It is one of the unbroken methods for constructing a hash function from a block cipher and is provably secure. Whirlpool hash function comprises message padding and 10 iterations of the block cipher, W and this cipher is similar to the AES symmetric key algorithm and each iteration or round consists of a non-linear layer, a cyclic permutation, a linear diffusion layer, key addition and a key schedule. Given a message consisting of a sequence of blocks m1, m2. . . mt, the whirlpool hash function is expressed as follows:

Ho= initial value.
Hi = E (Hi-1, mi) XOR Hi-1 XOR mi = intermediate value.
Ht = hash code value.

The encryption key input for each iteration i is the intermediate hash Hi-1 value from the previous iteration and the plaintext is the current message block mi.
The output for this iteration (Hi) consists of the bitwise XOR of the current message block the intermediate hash value from the previous iteration, and the output from W.

The overall processing of a message to produce a digest consists of the following steps: append padding bits, append length, initialize hash length and process the message in 512 bit (64- byte) blocks.

8. JH hash function

JH [8], a new hash function, introduced in 2008 ensure high collision resistance and security because of its two features.
It has a novel structure to construct a new compression function from a bijective function and a generalized advanced encryption standard (AES) methodology that allows the easy construction of large-block ciphers from smaller components also in addition, round constants are used instead of round keys. JH hash family includes four types which are, JH-224, JH-256, JH-384 and JH-512 and all these types use the new compression function,F8.For every input message block, a 1024-bit output is generated that is truncated producing a message digest with 224 (JH-224), 256 (JH-256), 384 (JH-384) or 512 bits (JH-512). The JH hash function consists of five steps, which are: Padding the initial message, M, passing the padded message into message blocks, setting the initial hash value, H(0), computing the hash value H (N) and finally generating the message digest by truncating H (N).

9. Blake hash function

The Blake hash function [9], one of the final round candidates in the competition organized by NIST is proposed by Jean-Philippe Aumasson, Luca Henzen, Willi Meier and Raphael. Blake hash function involves BLAKE-28, BLAKE-32, BLAKE-48, BLAKE 64 to BLAKE-224, BLAKE-256, BLAKE-384 12 and BLAKE-512 respectively and these new names indicate the digest sizes for the various versions of the Blake hash function. These digest sizes are similar to that of SHA-2 and makes it easy to directly substitute Blake in applications utilizing SHA-2. This hash function was designed with the intent of making it capable of running at high speed. It has a relatively simple algorithm and its compression function is a modified "double round" version of Bernstein's stream cipher "Chacha". Chacha [10] was designed to be immune to all kinds of side-channel attacks and Blake automatically inherited this property. Some of the techniques used for the speed optimization of Blake hash function are parallelism, pipelining (in an area of the algorithm where pipelining is feasible) and the use of carry-save adders in the compression function. Each compression takes a number of rounds or in other words, a number of clock cycles. In Blake-256 (the typical implementation of Blake which gives a 256 bit digest), each compression requires 14 clock cycles.
It is useful in comparing various hash function implementations.

Table 1: Relative comparisons of hash functions

Hash Functions	Area	Device	Speed (Mbps)
MD5 [14]	1369	Virtex-II	467
SHA-1 [13]	3040	Altera	143
SHA-2 [15]	2734	Virtex-IV	854
WHIRPOOL [7]	4908	Virtex-IV	4710
HAVAL [4]	1708	Virtex	196

It is not always fair to judge the results as such because of variations in the implementation devices but it reveals a better perspective in terms of relative comparisons

between the hash functions for being informative to next level of researches.

10. Conclusions

In this survey we have discussed about different cryptographic hash algorithms and the basic logic behind them. It is easy to compute the hash value for any given messages and it is infeasible to modify a message without changing the hash. Cryptographic hash functions have many information security applications, notably in digital signatures, message authentication codes (MACs), and other forms of authentication. It is specific that cryptographic hashing algorithms can be made non cryptographic by making suitable modifications in the hashing schemes. This review paper will benefit the research community working in the fields of networking and information security.

References

[1] Olakunle and Esuruoso, "High Speed FPGA Implementation of Cryptographic Hash Function" Electronic Theses and Dissertations, Paper 124, (2011).

[2] N. Sklavos," Towards to SHA-3 Hashing Standard for Secure Communications: On the Hardware Evaluation Development", IEEE Latin America transactions, Vol. 10, No. 1, Jan. 2012.

[3] Janaka Deepakumara, Howard M. Heys and R. Venkatesan," FPGA implementation of MD5 hash algorithm" Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering (CCECE 2001), Toronto, Ontario, May 2001.

[4] N. Sklavos and C. Efstathiou," On the FPGA Implementation of HAVAL Hash Function" The IEEE Region 8 EUROCON 2007, International Conference on "Computer as a Tool", (IEEE EUROCON'05), Belgrade, Serbia & Montenegro, November 21-24, 2005.

[5]Florian Mendel and Vincent Rijmen," Cryptanalysis of the Tiger Hash Function" International Association for Cryptologic Research , ASIACRYPT 2007, pp. 536–550,

[6] Akashi Satoh, Nicolas Sklavos," Compact and High-Speed Hardware Architectures for Hash Function Tiger", Proceedings of IEEE International Symposium on Circuits & Systems (IEEE ISCAS'09), Taiwan, May 24-27, 2009.

[7] Máire McLoone, Ciaran McIvor, Aidan Savage," High-Speed Hardware Architectures of the Whirlpool Hash Function", Proceedings of International Conference on Field Programmable Technology 2005

[8] George S. Athanasiou, Harris E. Michail, George Theodoridis, Costas E. Goutis," High-performance FPGA implementations of the cryptographic hash function JH", IET Comput. Digit. Tech., 2013, Vol. 7, Iss. 1, pp. 29–40.

[10] J.P. Aumasson, L. Henzen, M. Willi and C.W.R. Phan, SHA-3 Proposal BLAKE, January 11, 2011. Available: http://www.131002.net/blake.

[11] L. Devroye and P. Morin. Cuckoo Hashing: Further Analysis. Information Processing Letters, 86(4) pp 215-219, 2003.

[12] Ross J. Anderson and Eli Biham. TIGER: A Fast New Hash Function. In Dieter Gollmann, editor, FSE,volume 1039 of Lecture Notes in Computer Science, pages 89–97. Springer, 1996.

[13] M. Y. Wang, C. P. Su, C. T. Huang, and C. W. Wu, "An HMAC Processor with Integrated SHA-1 and MD5 Algorithms", proceedings of the conference on ASIA South Pacific Design Automation: Electronic Design and Solution Fair, pp. 456-458, Yokohama, Japan, 2004.

[14] M.J. Diez, et al., "Hash Algorithm for cryptographic Protocols: FPGA Implementations", proceedings of TELFOR 2002, pp. 26-28, Belgrade, November 2002.

[15] M. McLoone, and J.V. McCanny, "Efficient Single-Chip Implementation of SHA-384 & SHA-512", proceedings of IEEE International Conferece on Field ProgrammableTechnology (FPT), pp. 311-314, Hong Kong, 2002.