World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:4, No:1, 2010

Performance Evaluation of Popular Hash Functions

Sheena Mathew, K. Poulose Jacob

***Abstract*—**This paper describes the results of an extensive study and comparison of popular hash functions SHA-1, SHA-256, RIPEMD-160 and RIPEMD-320 with JERIM-320, a 320-bit hash function. The compression functions of hash functions like SHA-1 and SHA-256 are designed using serial successive iteration whereas those like RIPEMD-160 and RIPEMD-320 are designed using two parallel lines of message processing. JERIM-320 uses four parallel lines of message processing resulting in higher level of security than other hash functions at comparable speed and memory requirement. The performance evaluation of these methods has been done by using practical implementation and also by using step computation methods. JERIM-320 proves to be secure and ensures the integrity of messages at a higher degree. The focus of this work is to establish JERIM-320 as an alternative of the present day hash functions for the fast growing internet applications.

***Keywords*—**Cryptography, Hash function, JERIM-320, Message integrity

I. Introduction

DEPENDENCY of modern human life on electronic communication is increasing day by day. The convenience, speed and cost effectiveness of this communication channel is leading to rapid growth and spreading of internet enabled services into almost all walks of human life. On the other hand, the enormous applications in financial sector together with the increasing network base and accessibility from any nook and corner of the world has necessitated the need for increased network security also. The threats in ensuring confidentiality, integrity and authenticity of internet transactions call for greater focus of the research community as a challenging goal.

The integrity of the message sent from the sender to the receiver can be verified using cryptographic hash function. Hash function takes a variable size message as input and returns a fixed size string as output, which is called the hash code. The hash code is a concise representation of the longer message or document from which it was computed. Hash functions are important components in many cryptographic applications and security protocol suites. The most important uses are in the protection of information authentication and as a tool for digital signature schemes.

Sheena Mathew and K. Poulose Jacob are with the Department of Computer Science, Cochin University of Science and Technology, Kochi, Kerala, India. (e-mail: sheenamathew@cusat.ac.in, kpj@cusat.ac.in).

II. Design Factors with respect to JERIM-320

MD5 [1], SHA-1 [2] and RIPEMD algorithms [3] are popularly used for generating hash codes. But these algorithms have been broken at various levels [4]-[8]. The SHA-2 hash functions are quite resistant against those attack techniques which had been used to attack MD4 [9], MD5 and SHA-1. Another alternative, RIPEMD-family [3], has a different approach for designing a secure hash function. Here the attacker who tries to break the algorithm should try simultaneously at two ways where the message difference passes. This design strategy is still successful because so far there is no effective attack on RIPEMD-family except the first proposal of RIPEMD.

As a result of a large number of attacks on hash functions such as MD5 and SHA-1 of the so called MD4 family, and also general attacks on the typical construction method [10],[11] there is an increasing need for developing alternate designs based on new principles for future hash functions.

Several attacks on hash functions are focused on improving the difference of intermediate values which are caused by the difference in the message. In this context, a hash function can be considered secure, if it is computationally hard to alleviate such difference in its compression function. The design of the hash algorithm JERIM-320 [12] has been done based on these findings. In the design criteria, more emphasis can be seen for security over speed. The marginal reduction in speed of JERIM-320 can be neglected in the light of today's high computing power. The efficiency of the new hash function is its design based on potential parallelism. In this contest, the performance of JERIM-320 is compared with the some of the popular hash functions and presented in the subsequent sections.

III. Review of Popular Hash Functions and JERIM-320

A. SHA-1, SHA-256, RIPEMD-160 and RIPEMD-320

Here the hash functions SHA-1, SHA-256, RIPEMD-160 and RIPEMD-320 are briefly described.

The general skeleton of these hash functions shows their similarities and consists of the following steps:

1. Initialization: In this step some constant values are defined. These constants include initial chaining values (IVs), order of accessing message words, additive constants and the number of bits for rotation in each step.
2. Preprocessing: The message to be hashed has to be of length divisible by 512. The message is appended with a

single bit of value '1', followed by the required number of 0's to make the message length 64 bits less than a multiple of 512 bit blocks, each of which consists of sixteen 32-bit words. In addition, a number of chaining variables are initialized in this step during the initial IVs.

3. Processing: This is the heart of the algorithm, where each 512-bit block is processed in a step. Each step consists of the following sub steps
   a. Initialize working variables with the current values of the chaining variables.
   b. Update the working variables using some computation in rounds. Each round has almost the same computation in all the steps.
   c. Update the chaining variables.
4. Completion: The final hash value is composed by appending the chaining variables.

The main differences between these four hash functions are as follows:

1. For each 512-bit block, SHA-1 has 4 rounds of 20 steps each, SHA-256 has 4 rounds of 16 steps each, and RIPEMD-160 and RIPEMD-320 have 5 rounds, each of 16 steps.
2. Working variables are not updated in the same way in compression steps.
3. RIPEMD-160 and RIPEMD-320 use two parallel processing blocks for each 512 bit message block. The other two hash functions use only one processing block each.
4. After processing each 512-bit message block, SHA-1, SHA-256 and RIPEMD-320 update the chaining variables in the same way, whereas RIPEMD-160 uses a different way to update them.
5. The final hash value of SHA-1 is 160-bit long, SHA-256 is 256-bit long and RIPEMD-160 is 160-bit long; whereas RIPEMD-320 is 320-bit long.

B. *JERIM-320*

1) *Structure of JERIM-320*

JERIM-320 consists of four parallel branches B1, B2, B3 and B4. The initial chaining variable CV_i is given as input to the compression functions. CV_i consists of 10 registers A,B,C,D,E,F,G,H,I and J.

Each successive 512-bit message block M is divided into sixteen 32 bit sub blocks $M_0, M_1, \ldots, M_{15}$ given as $\Sigma_i(M)$ as input to all four branches and a computation is done to update CV_i to CV_{i+1} as

$CV_{i+1}=CV_i$^ ((B1output ^ B2output) + (B3output ^ B4output)). Finally the message is transformed into the 320 bit hash value.

2) *Single Step Operations*

Five rounds are used in JERIM-320 for each 512-bit message block. The sixteen 32-bit sub blocks of the 512-bit block in each round are processed in four parallel branches. The inputs to each single step operations are the sixteen sub blocks, the chaining variables A1, B1,…J1, A2, B2, …J2, A3, B3,….J3, A4, B4,…..J4 of each branch and the constants $K_{[t]}$. Order of message words, shift values, Boolean functions and constants in each branch and each round are different. There are 16 single step iterations in each round and in all the four branches. The output of each iteration is copied again into the chaining variables A1, B1,…J1; A2, B2, …J2; A3, B3,….J3; A4, B4,…..J4 and so on.

C. *A Brief Comparison of Hash Functions*

A brief overview of the above discussions is summarized in Table I

TABLE I
HASH FUNCTIONS AT A GLANCE

ALGORITHM	SHA-1	SHA-256	RIPEMD-160	RIPEMD-320	JERIM-320
Block size (bits)	512	512	512	512	512
Word size (bits)	32	32	32	32	32
Output size (bits)	160	256	160	320	320
Rounds	80	64	80	80	80
Serial / parallel iteration	Serial	Serial	Parallel (2 lines)	Parallel (2 lines)	Parallel 4 lines)
Max. message size (bits)	2^{64}-1	2^{64}-1	2^{64}-1	2^{64}-1	2^{64}-1
Operations	+, and, or, xor, rotl, not	+, and, or, xor, shr, rotr, not	+, and, or, xor, rotl, not	+, and, or, xor, rotl, not	+, and, or, xor, rotl, not
Collision	Yes (in 2005)	None yet	None yet	None yet	None yet

IV. PERFORMANCE EVALUATION

In this section the performance evaluation of the hash functions is done by using practical implementations and by using single step computations. The total number of operations, memory requirements and the speed performance of SHA-1, SHA-256, RIPEMD-160, RIPEMD-320 and JERIM-320 were compared. The evaluation was done using Pentium IV processor, Linux operating system and C compiler

A. *Practical Implementation*

As shown in Table II the total number of operations used in JERIM-320 is 7 times that of SHA-1, 3.7 times that of SHA-256 and 4 times that of RIPEMD-160 and RIPEMD-320. This is because of the hash function JERIM-320 making use of four parallel lines of message processing and hence the variables and computations in JERIM-320 will be more compared to other hash functions mentioned here. These multiple operations on the message blocks in JERIM-320 will result in much higher security than other hash functions.

TABLEII
COMPARISON OF NUMBER OF OPERATIONS OF SHA-1, SHA-256, RIPEMD-160, RIPEMD-320 AND JERIM-320

OPERATION	SHA-1	SHA-256	RIPEMD - 160	RIPEMD-320	JERIM-320
Addition	12	20	20	20	42
Bitwise operation (^,V, Λ,¬)	18	27	36	36	187
Shift operation	7	23	9	9	33
Total number of operations	37	70	65	65	262

As shown in Table III, the memory requirement for JERIM-320 is more and the speed is less than those of SHA-1, SHA-256, RIPEMD-160 and RIPEMD-320. These are because of the increased number of Boolean functions, the need for other operations like add and shift as well as the greater number of lines of message processing used in JERIM-320. Even though the speed of JERIM-320 is less than that of the other hash functions, it is very much within the acceptable limits and hence the advantages due to increase in the security overcomes the disadvantage in speed.

TABLE III
PERFORMANCE COMPARISON BETWEEN SHA-1, SHA-256, RIPEMD-160, RIPEMD-320 AND JERIM-320

ALGORITHM	SPEED (MBPS)	MEMORY REQUIREMENT (BYTES)
SHA-1	60.89	6533
SHA-256	55.93	7214
RIPEMD-160	35.89	8679
RIPEMD-320	35.63	8927
JERIM-320	14.01	12003

B. *Single Step Computation*

The single step computations for comparison of speed of the five hash functions are as follows:

The step operation of SHA-1 consists of 4 additions, 2 shifts and a Boolean function. The Boolean function consists of 3 unit operations, and the step operations consist of 80 steps (4 rounds * 20 iterations). That is 1 (stream) * 80 (steps) * 9 (step operations) = 720 (unit operations)

The step operation of SHA-256 consists of 7 additions, 2 summations and 2 Boolean functions. Each Boolean function and summation consists of 3 unit operations, and the step operation consists of 64 steps. That is

1 (stream) * 64 (steps) * 19(step operations) = 1216 (unit operations)

RIPEMD-160 consists of 4 additions, 2 circular shifts and a Boolean function. The Boolean function consists of 3 unit operations.

2(streams) * 80(steps) * 9(step operations) = 1440 (unit operations).

RIPEMD-320 consists of 4 additions, 2 circular shifts and a Boolean function. The Boolean function consists of 3 unit operations.

2(streams) * 80(steps) * 9(step operations) = 1440 (unit operations).

The step operation of JERIM-320 consists of 5 additions, 4 XORs, 4 shift and 2 Boolean functions. Each Boolean function consists of 3 unit operations, and the step operation consists of 80 steps (5 rounds * 16 iterations). That is 4 (streams) * 80 (steps) * 19 (step operations) = 6080 (unit operations).

From the above computations it can be seen that JERIM-320 has 8.4 times unit operations as compared to SHA-1, 5 times as compared to SHA-256 and 4.2 times as compared to RIPEMD-160 and RIPEMD-320. Due to this, the hash code produced in JERIM-320 will be much more secure than the other hash functions.

V. CONCLUSION

Various cryptographic hashes are analysed in this paper along with a high security hash function JERIM-320 using practical implementations and using single step computations. The core strength of JERIM-320 is the four parallel lines with five rounds of processing, which provide a strong nonlinear avalanche plus more number of register operations that increase diffusion in its output and make differential attacks difficult. Thus it is more secure than most of the existing popular hash functions, which are based on serial iterations. Due to the more number of operations performed in each message block, JERIM-320 produces much more secure hash code compared to other hash functions. Since message integrity is an important security service in today's high-speed network protocols and also because of the confidence level with respect to the current candidates like SHA-1 is coming down, new hash schemes become a necessity. A more secure hash code JERIM-320 can definitely be used as a substitute.

REFERENCES

[1] R.L.Rivest, "The MD5 message digest algorithm", (RFC 1320), Internet Activities Board, Internet Privacy Task Force. (1992)

[2] National Institute of Standards and Technology (NIST), FIPS-180-2: Secure Hash Standard. Available: http://csrc.nist.gov/publications/fips/fips 180-2/fips 180-2.pdf. (2002)

[3] H. Dobbertin, A.Bosselaers, B.Preneel, "RIPEMD-160, a strengthened version of RIPEMD". Fast Software Encryption , LNCS 1039, Springer-Verlag, pp. 71-82. (1996)

[4] E.Biham and R.Chen, A.Joux, P.Carribault, C.Lemuet, and W.Jalby, "Collissions of SHA-0 and Reduced SHA-1", Advances in Cryptology-EUROCRYPT, LNCS 3494, Springer- Verlag, pp. 36-57. (2005)

[5] F.Chabaud and A.Joux, "Differential Collissions in SHA-0", Advances in Cryptology- CRYPTO, LNCS 1462, Springer- Verlag, pp. 56-71. (1998)

[6] H. Dobbertin, "Cryptanalysis of MD4", Fast Software Encryption, LNCS 1039, Springer-Verlag, pp. 53-69. (1996)

[7] E.Biham and R.Chen, "Near Collissions of SHA-0", Advances in Cryptology- CRYPTO, LNCS 3152, Springer- Verlag, pp. 290-305. (2004)

[8] Xiaoyun Wang and Hongbo Yu, "How to break MD5 and other hash functions", Advances in Cryptology – EUROCRYPT, LNCS 3494, Springer-Verlag, pp. 19-35. (2005)

[9] R.L.Rivest, "The MD4 message digest algorithm", Advances in Cryptology-CRYPTO, LNCS 537, Springer-Verlag, pp. 303-311. (1990)

[10] Ivan Damgard, "A design principle for hash functions", Advances in Cryptology -CRYPTO, LNCS 435, Springer – Verlag, pp. 416-427. (1989)

[11] Ralph C.Merkle, "One way hash functions and DES", Advances in Cryptology –CRYPTO, LNCS 435, Springer – Verlag, pp. 428-446. (1989)

[12] Sheena Mathew, K. Poulose Jacob, "JERIM-320: A New 320-bit Hash Function with Higher Security", International Journal of Computers, Systems and Signals, Vol. 9, No.1, 2008, pp 31-41.

Sheena Mathew, Reader in Cochin University of Science and Technology (CUSAT), Kochi, Kerala, India has 16 years of teaching experience in Computer Science. She had her graduation from Madurai Kamaraj University and post graduation from the Indian Institute of Science, Bangalore. She is presently a research scholar; her areas of interest being Cryptography and Network Security. She has 9 publications in various international journals and conferences to her credit.

Dr. K. Poulose Jacob, a National Merit Scholar all through, got his degree in Electrical Engineering in 1976 from University of Kerala, followed by his M.Tech. in Digital Electronics and Ph. D. in Computer Engineering from CUSAT, Kochi. He has been teaching at CUSAT since 1980 and currently occupies the position of Professor and Head of the Department of Computer Science. He has served as a Member of the Standing Committee of the UGC on Computer Education and Development. He is on the editorial board of two international journals and has more than 80 papers in various international journals and conferences to his credit. His research interests are in Information Systems Engineering, Intelligent Architectures and Networks.