Analysis of Various Hash Function

Priyamvada Saxena
Department of Electrical Engineering,
Centre of Excellence,
Veermata Technological Institute,
Mumbai, India-4000019

Abstract:- Hash function has become a basic tool of modern networking. It is a mathematical tool used for identity authorization, secure flow of data in the network. Hashing is also used in the encryption and decryption of digital signatures. It is a tool that converts numerical input value into a compressed numerical value. Input to a hash module is of arbitrary length but its output is of fixed length. This paper presents the analysis and theoretical overview of various hash function and time conversion analysis of all the hash algorithms.

Keywords:- *cryptography, hash function, time conversion analysis.*

I. INTRODUCTION

Data Security is an important element of data communication. Encryption techniques play an important role in data security. They enhance data privacy by making the data unreadable or impossible to break the data which can be understandable. But when data is shared or transferred it becomes extremely important for data authentication and authorization. A hash function is a tool that can be used to map a given or defined data of arbitrary size to a data of fixed size. It is also known as one-way cryptography. It would map data from one form to another. The text obtained can also be called as hashed text as seen in fig 1.

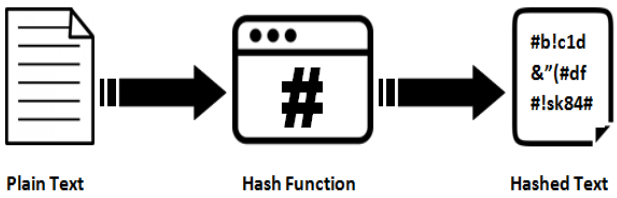

Fig1:- Hash algorithm.

It also used in computer software. It can also be used to identify secret information. It is also collision-resistant, which means that it is very hard to find data that any data will generate the same hash value. It consists of certain properties. The first one is Pre-Image Resistance. This means that should be hard to compute reverse of hash function .In other words, if a hash module h produces a hash value z, then it should be a difficult to compute to find any input value x that hashes to z. It can be used to protect against a hacker who only has a hash value and is trying to find the input. The next property is Collision Resistance. This means it should be hard to find two different inputs that result in the same hash value. It is also referred to as collision free hash function. In other words, for hash h, it is hard to find any two different inputs l and m such that h(l) = h(m).There are two types of hash function that is presented in this paper. Non- cryptographic and cryptographic hash function.

Non-cryptographic hash function produces a hash value h of fixed length of any length of message digest M. This can be seen in fig2.They are faster than the cryptographic hash function.

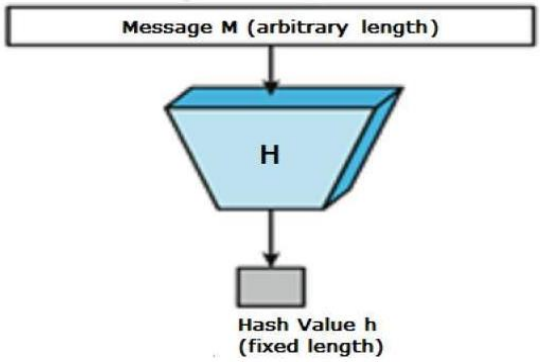

Fig 2:- Non-cryptographic hash function.

Cryptographic hash function is a special class of hash function that is used extensively in cryptography. It is a mathematical tool that maps data of arbitrary length to a bit string of a fixed size (a hash) and is a one-way function. The property one-way function means that it is impossible to invert. It has various properties such as it is deterministic that is the same message always results in the same hash value, it computes the hash value for any given message quickly, a minute change in the message should change the hash value so extensively that the new hash value appears has no relation with the old hash value. This can be explained in the figure3.

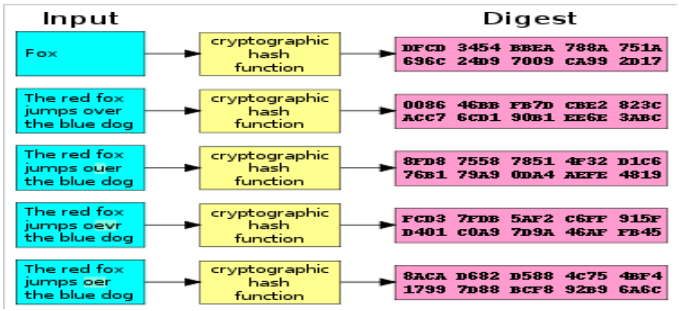

Fig 3:- Cryptographic Hash function.

This technique has wide range of applications such as verifying the integrity of files or messages, password verification, proof-of-work.

II. OVERVIEW OF ALGORITHMS

The aim is to study both non-cryptographic and cryptographic algorithms. The algorithms that have detailed analysis in this paper are MD2, MD4, MD5, Ripemd, Murmur, Spooky, City Hash, xx Hash.

A. MD2 Algorithm.

MD2 Message-Digest Algorithm is a cryptographic hash function. It was developed in the year 1989 by Ronald Rivest. The algorithm is used for 8-bit computers. The message digest is of 128 bits. Thirty-two-digit hexadecimal numbers are used for representing it. The MD2 algorithm uses a message of any length and an output of a 128-bit message digest is produced. It is seen that it is impossible to produce two messages with same message digest. The algorithm involves the following procedure: appending padding bytes i.e. The message is "padded" so that its length (in bytes) is congruent to 0, modulo 16, appending checksum i.e.16-byte checksum of the message is appended, initializing the message digest buffer for computing the message digest i.e. A 48-byte buffer X is used to compute the message digest, processing the message in 16-byte blocks and finally producing the output.

B. MD4 Algorithm.

MD4 Message-Digest Algorithm is a cryptographic hash function. It was developed in 1990 by Ronald Rivest after the success of MD2. The message digest length is 128 bits. MD4 was aimed at 32-bit machines. MD4 is meant to be fast, which means taking a few risks regarding security. The algorithm involves the following procedure: appending padding bits i.e. Message is "padded" so that its length (in bits) is congruent to 448, modulo 512, append length i.e. 64-bit is appended to the result of the previous step, initialize MD buffer i.e. four-word buffer (A,B,C,D) is used to compute the message digest. In this each of A, B, C, D is a 32-bit register. These registers are initialized to the following values in hexadecimal, low-order bytes first: word A: 01 23 45 67 word B: 89 ab cd ef word C: fe dc ba 98 word D: 76 54 32 10, processing the message in 16-byte blocks and finally producing the output.

C. MD5 Algorithm

MD5 Message-Digest Algorithm is a cryptographic hash function. It was developed in 1991 by Ronald Rivest after the success of MD4. It is a widely used hash function .It produces a 128-bit hash value. The algorithm involves the following procedure: appending padding bits i.e. Message is "padded" so that its length (in bits) is congruent to 448, modulo 512, append length i.e. 64-bit is appended to the result of the previous step, initialize MD buffer i.e. four-word buffer (A,B,C,D) is used to compute the message digest. In this each of A, B, C, D is a 32-bit register. These registers are initialized to the following values in hexadecimal, low-order bytes first: word A: 01 23 45 67 word B: 89 ab cd ef word C: fe dc ba 98 word D: 76 54 32 10, processing the message in 16-byte blocks and finally producing the output. The difference exists only in the processing the message.

D. Ripemd Hash

RIPEMD RACE Integrity Primitives Evaluation Message Digest is a cryptographic hash function. It is developed in Leuven, Belgium. It was developed by Hans Dobbertin, Antoon Bosselaers and Bart Preneel at the COSIC research group at the Katholieke Universiteit Leuven. It was first published in 1996. RIPEMD was based upon the design principles used in MD4. The RIPEMD hashes are typically represented as 40-digit hexadecimal numbers.

E. Murmur Hash

Murmur Hash is a non-cryptographic hash function. It was developed by Austin Appleby in the year 2008. It is suitable for general hash-based lookup. The version which is currently used is MurmurHash3 it yields a 32-bit or 128-bit hash value. MurmurHash2 is an older version that yields a 32-bit or 64-bit value. It has been adopted in a number of open-source projects. It can also be vulnerable to attack if a user can chooses input in such a way to intentionally cause hash collisions.

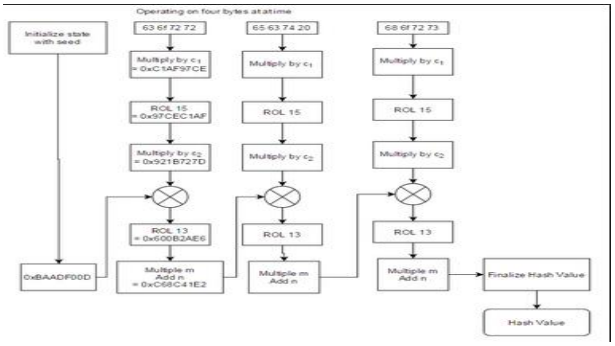

Fig 4:- Murmur Hash

It mainly uses just three steps multiplication, rotate and xor. This can be seen in figure4.

F. Cityhash

City Hash is a non-cryptographic hash function. It is designed for fast hashing of strings. Google developed the algorithm in-house starting in 2010 and was released in 2011. It hash strings to 64- and 128-bit hash codes. The only disadvantage of this approach is that the code is more complicated than most popular other algorithms. The algorithm is partly based on Murmur Hash and it claims to be faster.

G. Spookyhash

SpookyHash is a non-cryptographic hash function producing 128-bit hash values for byte arrays of any length. It can produce 64-bit and 32-bit hash values too. It was created by Bob Jenkins. It is one of the fastest hashing functions. It also uses functions like addition, rotation. In C language and in python there is a module called spooky which let us apply the spooky has function.

H. Xxhash

XXHash is an Extremely fast Hash algorithm, running at RAM speed limits. Code is highly portable. The algorithm takes a message as input of arbitrary length and an optional seed value and then produces an output of 32 or 64-bit as digest. It is primarily designed for high speed. It is a non-cryptographic library and is not meant to avoid intentional collisions, or to prevent producing a message with predefined digest. This algorithm also uses addition, rotation, shifting and xor method.

III. SIMULATION RESULTS AND ANALYSIS

The comparison in the performance of the algorithms was conduction with hashing of a string. The performance

parameter that is needed to be calculated is the time taken for hashing a string.

The simulation was conducted on Linux platform. A string was selected. Python was used as the language of choice for implementation. For each algorithm time was calculated and comparison is made.

The first technique is MD2 and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig5.

Fig 5:- Output of MD2 Algorithm.

The time taken for hashing the string using MD2 technique is 0.002099sec.

The next technique is MD4 and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig6.

Fig 6:- Output of MD4 Hash algorithm.

The time taken for hashing the string using MD4 technique is 0.002097sec.

The next technique is MD5 and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig7.

Fig 7:- Output of MD5 Hash algorithm.

The time taken for hashing the string using MD5 technique is 0.00933sec.

The next technique is RIPEMD and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig8.

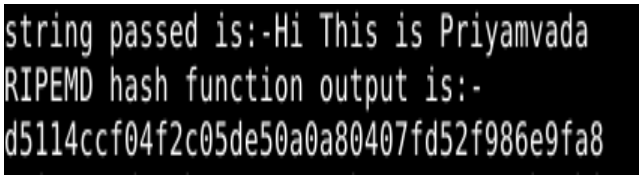

Fig 8:- Output of RIPEMD Hash algorithm.

The time taken for hashing the string using RIPEMD technique is 0.002029sec.

The next technique is MURMUR HASH and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig9.

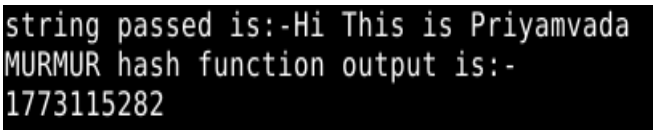

Fig 9:- Output of MURMUR Hash algorithm.

The time taken for hashing the string using MURMURHASH technique is 0.000826sec.

The next technique is CITYHASH and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig10.

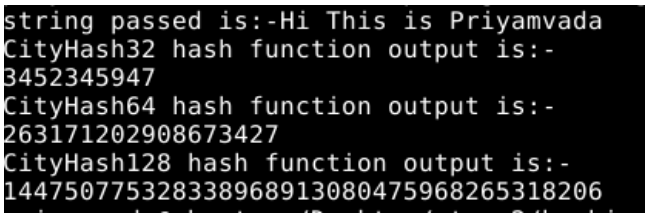

Fig 10:- Output of CITYHASH Algorithm

There are three CITYHASH function used. The time taken for hashing the string using CITYHASH technique is 0.000851sec.

The next technique is SPOOKYHASH and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig11.

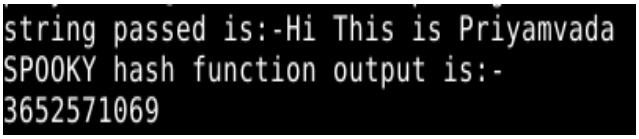

Fig 11:- Output of SPOOKYHASH Algorithm

The time taken for hashing the string using SPOOKYHASH technique is 0.00092sec.

The next technique is XXYHASH and string to be passed on is “Hi This is Priyamvada”. The output can be seen in the fig12.

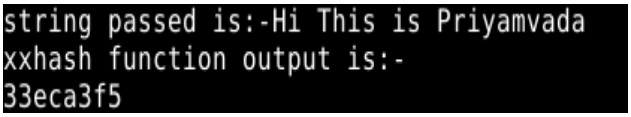

Fig 12:- Output of XXHASH Algorithm

The time taken for hashing the string using XXHASH technique is 0.000859sec.

The comparison between various technique is shown in the below table:-

TECHNIQUE	HASHING TIME(IN SEC)
MD2	0.002099
MD4	0.002097
MD5	0.00933
RIPEMD	0.002029
MURMURHASH	0.000826
CITYHASH	0.000851
SPOOKYHASH	0.00092
XXHASH	0.00085

The fastest cryptographic hash function is the RIPEMD technique and in non-cryptographic hash function is the xxhash technique.

IV. CONCLUSION

This paper presents the performance characteristics of some algorithms. From the results it can be seen that RIPEMD and xxhash technique are the best technique in terms of less time for hashing the string. In future it can be extended for security in data transmission and other techniques can be used and compare their performance.

REFERENCES

[1] Guang Cheng andYang Yan "Evaluation and Design of Non-cryptographic Hash Functions for Network Data Stream Algorithms",IEEE, 2017 3rd International Conference on Big Data Computing and Communications.

[2] Liu Jian-dong, Tian Ye, Wang Shu-hong, Yang Kai "A Fast New One-Way Cryptographic Hash Function",IEEE.

[3] Abdulaziz Ali Alkandari, Imad Fakhri Al-shaikhli and Mohammad A. Alahmad "Cryptographic Hash Function: A High Level View",IEEE, 2013 International Conference on Informatics and Creative Multimedia.

[4] Puliparambil Megha Mukundan1 , Sindhu Manayankath1, Chungath Srinivasan1, Madathil Sethumadhavan1 "Hash-One: a lightweight cryptographic hash function",IEEE, Special Issue: Lightweight and Energy-Efficient Security Solutions for Mobile Computing Devices.

[5] Dexi Wang, Yu Jiang, Houbing Song, Fei He, Ming Gu and Jiaguang Sun "Verification of implementations of cryptographic hash functions",IEEE.