A Modified Approach For Cryptograpic Hash Function Based On MD5 Algorithm

Amita Pandey, Padma Bonde

Department Of Computer Science

Shri ShankaraCharya Institute Of Technology & Management

Abstract:

Cryptographic hash function is used for creating the message digest. Message digest is a fixed length output of any variable length input. Hash functions are the tools which are used in digital signatures, digital time stamping and assuring the integrity of the messages. The absence of unintended changes or any alteration in some data between two updates of a data record is called data integrity. In this paper we have developed a new procedure for creating the message digest of any message. A comparison is done between the existing cryptographic hashing tool MD5 and our proposed optimized algorithm. In our algorithm the time lapsed to create a hash function from a plain text is less than the existing hashing tool MD5.

Keywords: Cryptography,Message Digest,Message Integrity,MD5,RSA.

INTRODUCTION

In today's world where we access information through internet, it is essential to give right information to right people at right time in a secure manner. Hence so as to maintain security authentication, authorization, integrity, non repudiation and confidentiality are essential. So integrity is one of our major issues nowadays. Integrity is maintained when the information that has been sent actually is the replica of the information which has been received. Suppose one person is sending a critical data through an insecure channel to the other person who is sitting at some other location then it may be possible that any third party can retrieve the message and modify it and then passes it to the destination. This may lead to many undesirable consequences and the company may suffer a big monetary loss. With the increase in network speed, incremented processing speed is also required for encryption, authentication and integration. Nowadays many hash functions are available for this purpose, such as- MD4 [1], MD5 and SHA-1.A cryptographic hash function is an algorithm that takes a variable length block of data as an input and returns a fixed-size bit string as an output. These functions are one-way hash functions. Once generated hash value cannot retrieve the original input message.The hash value will change with the modification in the input message. The input data to be encoded are often

referred to as the "message," and the digest orsimply digests

Hash Function possesses three properties.

- **Preimage:** It is infeasible to find out the original message from its message digest or hash. This concept is related to that of irreversible or one-way function .
- **Second preimage:** Given a message y, it is infeasible to find another message y', such that both messages hash to a same message digest. This property is sometimes referred to as weak collision resistance.
- **Collision:** It is infeasible to find two different messages, which hash to the same message digest. This property is sometimes referred to as strong collision resistance.

WORKING METHODOLOGY

As we know hash functions are one way functions which are used to provide security against data alterations but they cannot ensure sufficient security against some of the attacks. Suppose a sender combines a message M with its hash function H(M) and sends it to the receiver then it may be possible that any third party changes the message M' and combines it with its new hash function H(M') and sends it to the receiver. Now when the receiver receives the message M' and re-evaluates its hash function then it exactly matches with the function which has been received. But this is not true; our original message is different than the received one.To overcome this hash value is known as the message

problem the sender may combine the encrypted message with its hash function and then send it. At the receiver end, one needs to decrypt the encrypted message with the specified key and then re-evaluate its hash function. We consider that the received message is unaltered when both the functions are same. If anyone tries to change the message then the receiver would not be able to decrypt the message with the same private key. Hence no third party can change the message as done earlier because here we are using encryption technique in which private key is only known to sender and receiver. But attacker may combine a hash function of a different message with the original encrypted message then send it. Although the receiver receives the original message but it is interpreted as the wrong one because recalculated hash function does not matches with the received one.

Hence so as to ensure security of the message we are combining the message with its hash function MD5 and then encrypt it by using encryption technique named RSA.

The MD5 algorithm first divides the input in blocks of 512 bits each. 64 Bits are inserted at the end of the last block. These 64 bits are used to record the length of the original input. If the last block is less than 512 bits, some extra bits are 'padded' to the end. Next, each block is divided into 16 words of 32 bits each. These are denoted as M_0 ... M_{15}. MD5 uses a buffer that is made up of four words that are each 32 bits long. These

words are called A, B, C and D. They are initialized as
word A: 01 23 45 67
word B: 89 ab cd ef
word C: fe dc ba 98
word D: 76 54 32 10
MD5 further uses a table K that has 64 elements. Element number i is indicated as K_i. The table is computed beforehand to speed up the computations. The elements are computed using the mathematical sin function:
$K_i = abs(sin(i + 1)) * 2^{32}$
In addition MD5 uses four auxiliary functions that each take as input three 32-bit words and produce as output one 32-bit word. They apply the logical operators and, or, not and xor to the input bits.

F(X,Y,Z) = (X and Y) or (not(X) and Z)

G(X,Y,Z) = (X and Z) or (Y and not(Z))

H(X,Y,Z) = X xor Y xor Z

I(X,Y,Z) = Y xor (X or not(Z))

The contents of the four buffers (A, B, C and D) are now mixed with the words of the input, using the four auxiliary functions (F, G, H and I). There are four *rounds*, each involves 16 basic *operations*.
After all rounds have been performed, the buffers A, B, C and D contain the MD5 digest of the original input.

The RSA algorithm involves multiplying two large prime numbers (a prime number is a number divisible only by that number and 1) and through additional operations deriving a set of two numbers that constitutes the public key and another set that is the private key. Once the keys have been developed, the original prime numbers are no longer important and can be discarded. Both the public and the private keys are needed for encryption /decryption but only the owner of a private key ever needs to know it. Using the RSA system, the private key never needs to be sent across the Internet.The private key is used to decrypt text that has been encrypted with the public key. Thus, if I send you a message, I can find out your public key (but not your private key) from a central administrator and encrypt a message to you using your public key. When you receive it, you decrypt it with your private key. In addition to encrypting messages (which ensures privacy), you can authenticate yourself to me (so I know that it is really you who sent the message) by using your private key to encrypt a digital certificate. When I receive it, I can use your public key to decrypt it.

Our Design principal for the algorithm may be stated as: “make use of already proven techniques and build stronger one.” For this-

- We base our hash function MD5
- MD5 Serves the purpose of providing basic building block for hash algorithm.
- RSA provides keyed function that copulates source authentication along with message integrity.
- The technique provides a solution for unauthorized changes in original message and receiving it by receiver assuming that is coming from original sender only in lesser time.

The proposed algorithm may be stated as:
Step 1: Begin
Step 2: Append padding bits to the message

Step 3: Append original message length to the O/P of Step2. And get 512 bit L blocks of message.
Step 4: Initialize MD buffer (128 bit)
Step 5: Repeat Steps 6 to 10 for all L blocks
Step 6: Generate 128bit digest of ith block
Step 7: Divide O/P of Step 6 into two blocks of 64 bits each.
Step 8: Encrypt both blocks (O/P of Step7) using RSA.
Step 9: Concatenate 64 bit outputs of Step 8.
Step 10: Use the O/P of Step 9 as CV for next 512 bit block.
Step 11: Use final O/P of last Lth block as hash value to be transmitted to receiver for message integrity along with authentication.
Step 12: End

CONCLUSION

An effort has been made to understand the existing cryptographic hash function MD5.Our basic aim is to address the issue of encrypting and securing the data in minimum time or we can say that reducing the time complexity. The focus of discussion was to create and implement a new algorithm, which is easier to optimize and analyze for security or data integrity. Keeping this goal in our mind the new algorithm has been designed. It uses the existing algorithm MD5 and RSA(which is used for encrypting the data).We have compared the MD5 algorithm with our algorithm and the result obtained from this project work is that our algorithm is less time consuming then MD5 algorithm. Our algorithm is used for applications where data integrity and message/sender authentication is required.

REFERENCES

[1]"A SECURED CRYPTOGRAPHIC HASHING ALGORITHM"Prof. Rakesh Mohanty, Niharjyoti Sarangi, Sukant Kumar Bishi

[2]International Journal on Cryptography and Information Security(IJCIS),Vol.2, No.1,March 2012 "CRYPTANALYZING OF MESSAGE DIGEST ALGORITHMS MD4 AND MD5"

[3]International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-4, March 2013 "Cryptographic Hash Functions: SHA Family"

[4]Richa Purohit et al ,Int.J.Computer Technology & Applications,Vol 3 (5), 1715-1719 ISSN:2229-6093"Strenthening Hash Functions using Block Symmetric Key Encryption Algorithm"

[5]R. Rivest, "The MD4 Message Digest Algorithm", Procesedings of CRYPTO'90, August 1990.

[6]Steven M. B., and E. K. Rescorla. "Deploying a new hash algorithm", Proceedings of NDSS '06, 2006.

[7]R. Rivest. "The MD5 Message-Digest Algorithm". RFC 1321, April 1992.

[8]National Institute of Standards and Technology, U.S. Department of Commerce. Secure Hash Standard, 2002. FIPS PUB 180-2.

[9]R.L. Rivest. MD4 Message Digest Algorithm. RFC 1186, October 1990.

[10]Diffie W., M.E. Hellman. "New directions in cryptography".

[11]Shafi Goldwasser, Silvio Micali. "Probabilistic Encryption".

[12]Forouzan, Behrouz. "Cryptography and Network Security"1/e.