Hashing Algorithm: MD5

Shweta Mishra[1] Shikha Mishra[2] Nilesh Kumar[3]
[1, 2, 3]Department of Computer Science & Engineering
[1]Echelon Institute of Technology, Faridabad, India [2, 3] JB Knowledge Park, Faridabad, India

Abstract—a message digest is a cryptographic hash function containing a string of digits created by a one-way hashing formula. Message digests are designed to protect the integrity of a piece of data or media to detect changes and alterations to any part of a message. In this paper, we have explained the hashing algorithm of MD5 and also proposed how to use it for file transmission and for hashing any string.

General Terms: Security, Algorithms, Auxiliary Functions

Key words: Hash function, MD.

I. INTRODUCTION

Message Digest is a type of cryptography utilizing hash values that can warn the copyright owner of any modifications applied to their work. Message digest hash numbers represent specific files containing the protected works. One message digest is assigned to particular data content. It can reference a change made deliberately or accidentally, but it prompts the owner to identify the modification as well as the individual(s) making the change. Message digests are algorithmic numbers. MD5 was the last in a succession of cryptographic hash functions designed by Ron Rivest in the early 1990s [1].

It is a widely-used well-known 128-bit iterated hash function, used in various applications including SSL/TLS, IPsec, and many other cryptographic protocols. It is also commonly-used in implementations of time stamping mechanisms, commitment schemes, and integrity-checking applications for online software, distributed systems, and random-number generation.

Message digest functions are used to produce digital summaries of information called message digests. Message digests are commonly 128 bits to 160 bits in length and provide a digital identifier for each digital file or document. Message digest functions are mathematical functions that process information to produce a different message digest for each unique document [2]. Identical documents have the same message digest; but if even one of the bits for the document changes, the message digest changes.

Because message digests are much shorter than the data from which the digests are generated and the digests have a finite length, duplicate message digests called collisions can exist for different data sets. However, good message digest functions use one-way functions to ensure that it is mathematically and computationally infeasible to reverse the message digest process and discover the original data. Finding collisions for good message digest functions is also mathematically and computationally infeasible but possible given enough time and computational effort. However, even if an attacker discovers a collision, it is highly improbable that the collision could be useful.

II. NETWORK SECURITY AND MESSAGE DIGEST

Security and privacy is a growing concern in the Internet community, due to the Internet's rapid growth and the desire to conduct business over it safely. This desire has led to the advent of several proposals for security standards, such as secure IP, Secure HTTP (SHTTP), and the Secure Socket Layer (SSL). Thus, the need to use encryption protocols such as DES and RSA is increasing. One problem with using cryptographic protocols is the fact that they are slow. An important question then is whether security can be provided at gigabit speeds.

The standard set of algorithms required to secure a connection includes a bulk encryption algorithm such as DES, a cryptographic message digest such as MD5, a key exchange algorithm such as Diffie-Hellman to securely distribute a private key, and some form of digital signature algorithm to authenticate the parties, such as RSA. The encryption and hash digest algorithms must be applied to every packet going across a link to ensure confidentiality, and therefore the performance of these algorithms directly affects the achievable throughput of an application. MD5 is a message digest algorithm used for authentication and message integrity. MD5 is a "required option" for secure IP; by required option we mean that an application's use of MD5 in IP is optional, but an implementation must support use of that option. MD5 is also the default message digest algorithm proposed for SHTTP; and is also used in SSL.

III. MD5 ALGORITHM

The MD5 algorithm is designed to be quite fast on 32-bit machines. In addition, the MD5 algorithm does not require any large substitution tables; the algorithm can be coded quite compactly. The MD5 algorithm is an extension of the MD4 message-digest algorithm MD5 is slightly slower than MD4, but is more "conservative" in design.

MD5 algorithm takes as input a message of arbitrary length and produces as output a 128 bit message digest of the input. The authentication algorithm computes a digest of the entire data of the message, used for authentication. Typically, the message digest is registered with a trusted third-party, or encrypted via other means. The digest is used by the receiver to verify the contents of a message. It can also be used to encrypt the contents of a message, via a second pass over the data by another algorithm. MD5 requires that both the sender and receiver compute the digest of the entire body of a message [3] [4].

Suppose a b-bit message as input, and that we need to find its message digest.

Step. 1 : Append padded bits--
The message is padded so that its length is congruent to 448, modulo 512. A single "1" bit is appended to the message, and then "0" bits are appended so that the length in bits equals 448 modulo 512.

Step. 2 : Append length--
A 64 bit representation of b is appended to the result of the previous step. The resulting message has a length that is an exact multiple of 512 bits.

Step. 3 : Initialize MD Buffer--
A four-word buffer (A, B, C, D) is used to compute the message digest. Here each of A, B, C, D is a 32 bit register. These registers are initialized to the following values in hexadecimal:
Word A: 01 23 45 67
Word B: 89 ab cd ef
Word C: fe dc ba 98
word D: 76 54 32 10

Step. 4 : Process message in 16-word blocks—
Four auxiliary functions that take as input three 32-bit words and produce as output one 32-bit word:
F(X,Y,Z) = XY v not(X) Z
G(X,Y,Z) = XZ v Y not(Z)
H(X,Y,Z) = X xor Y xor Z
I(X,Y,Z) = Y xor (X v not(Z))
If the bits of X, Y, and Z are independent and unbiased, the each bit of F(X,Y,Z), G(X,Y,Z), H(X,Y,Z), and I(X,Y,Z) will be independent and unbiased.

Step. 5 : Output—
The message digest produced as output is A, B, C, D. That is, output begins with the low-order byte of A, and end with the high-order byte of D[5].

IV. MD5 HELPER FUNCTIONS

A. The Buffer

MD5 uses a buffer that is made up of four words, each 32 bits long. These words are called A, B, C and D. They are initialized as:

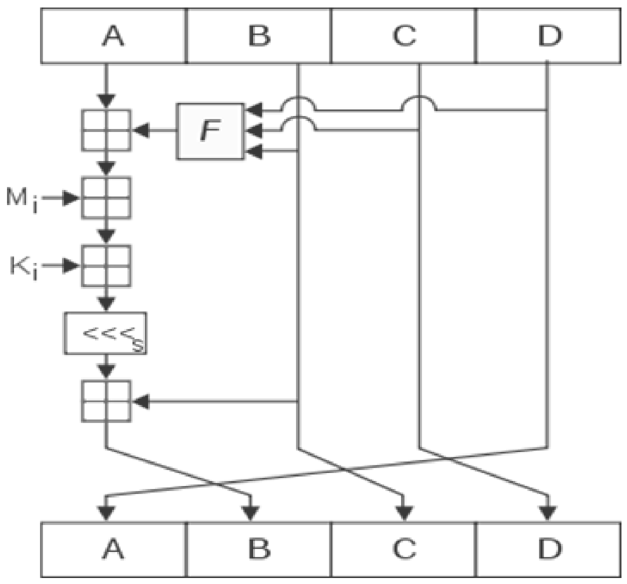

Fig. 1: MD5 Functions

Word A: 01 23 45 67
Word B: 89 ab cd ef
Word C: fe dc ba 98
Word D: 76 54 32 10

B. The Table

MD5 uses a table K that has 64 elements. The table is computed beforehand to increase the speed of the computation.

C. Auxiliary Functions(F)

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.

V. MD5 IN FILE TRANSMISSION

MD5 digests have been widely used in the software world to provide some assurance that a transferred file has arrived intact. , file servers often provide a pre-computed MD5 checksum for the files, so that a user can compare the checksum of the downloaded file to it. However, now that it is easy to generate MD5 collisions, it is possible for the person who created the file to create a second file with the same checksum, so this technique cannot protect against some forms of malicious tampering. Also, in some cases, the checksum cannot be trusted in which MD5 can only provide error-checking functionality.

Hashing algorithm is applied on the file that to be transmitted. Hashing value is then generated. When the file is sent to the user or receiver one of the two conditions arises. If the hash value received and the hash value calculated matches then the file is correctly received by the user but if those two values don't match then the received file have some error. In this way MD5 is used for authentication purpose.

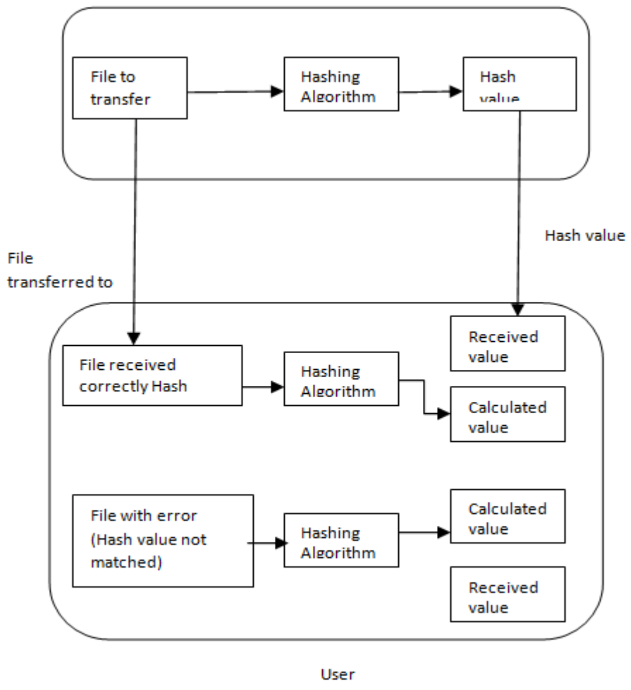

Fig. 2: MD5 in file transmission

VI. CONCLUSION

Thus we saw that how message digest 5 is used for authenticating any file and how it is used for hashing any string using some helper functions. In my future work I would be implementing this file transmission by using two different algorithms and would like to show which one is better by comparing their speeds and complexity. We would also like to show how the various attacks are applied on MD5 and what are their effects.

REFERENCES

[1] http://theory.lcs.mit.edu/~rivest/
[2] Bruce Schneir "Applied Cryptography: Protocols, Algorithms, and Source Code in C"; 18.5 MD5 (pp. 436-441)
[3] Ronald Rivest: The MD5 Message Digest Algorithm, RFC1321, April 1992, ftp://ftp.rfc-editor.org/innotes/rfc1321.txt
[4] R.L. Rivest. The MD5 Message-Digest Algorithm. *RFC1321, MIT Laboratory for Computer Science and RSA Data Security, Inc.*, April 1992.
[5] Amphion. CS5315, High Performance Message Digest 5 Algorithm (MD5) Core. Datasheet, URL: http://www.amphion.com/acrobat/DS5315.pdf, (visited September 9, 2004).
[6] J. Black, M. Cochran, T. Highland: A Study of the MD5 Attacks: Insights and Improvements, March 3, 2006