International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 08 | Aug 2021 www.irjet.net p-ISSN: 2395-0072

Message/Data Authentication using Hash Functions

Akhil Chandra Gorakala[1], K V R A Vikram[2], Venkateswarlu K[3]

[1]Student, Department of Computer Science and Engineering, GITAM deemed to be University, Visakhapatnam, Andhra Pradesh, India.
[2]Student, Department of Computer Science and Engineering, GITAM deemed to be University, Visakhapatnam, Andhra Pradesh, India.
[3]Assistant Professor, Department of Computer Science and Engineering, GITAM deemed to be University, Visakhapatnam, Andhra Pradesh, India.

Abstract - Cryptographic hash functions or message authentication have become common in many internet applications or protocols because they are effortless to implement. However, these are based on temporary techniques; as hash functions were not initially discovered for message authentication, they are not much secure. Generally, in message authentication, the underlying cryptographic function should be vital to preventing the data from being forged. In this paper, we present our schemes NMAC(), HMAC(), and digest(), which strengthen the underlying cryptographic hash functions by replacing them with simpler and effective ones.

***Key Words*: Cryptography, Hash functions and Message Authentication**

1. INTRODUCTION

Message Authentication is a way to ensure the safe transmission of the data between two stations when they are communicating through an insecure channel by preventing the data from being forged or tampered with. The receiver can also verify the legitimacy of the data using specific protocols. However, it is impossible to track some seriously potential pirates trying to tamper or modify the data.

Although some methods like using a firewall, Intrusion Prevention System(IPS), Antivirus, or Information Detection System(IDS), they only protect the connection to a certain extent, after which they fail to work. Moreover, as the world has become a global village, a significant amount of data is being transferred each day, containing much sensitive information that the above policies may not entirely protect. There are chances that they might not find some inconspicuous agents, so with the help of proper methods like MAC(Message Authentication Code) and Cryptographic Hash functions, help us track intruders, verify the legitimacy of the message, and the transfer of data would be more secure..

2. Message Authentication Code(MAC)

It is of utmost importance to verify data integrity in a network when two stations communicate over an insecure channel; it is vital to confirm whether data sent by one station is bona fide or not by the other. This Authentication can be done with the help of MAC.

MAC is a small fixed-size block

MAC works on the principle formula:

C(M, K)= MAC

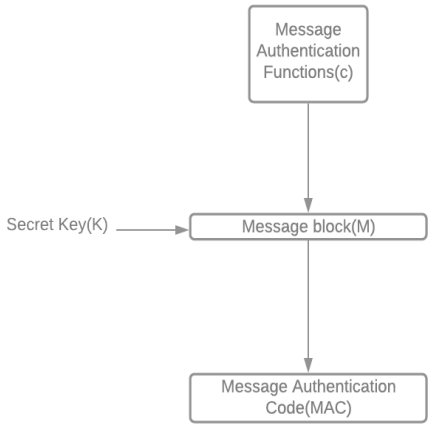
Where,

C= Message authentication Functions

M= Plain text

K= Shared Secret Key

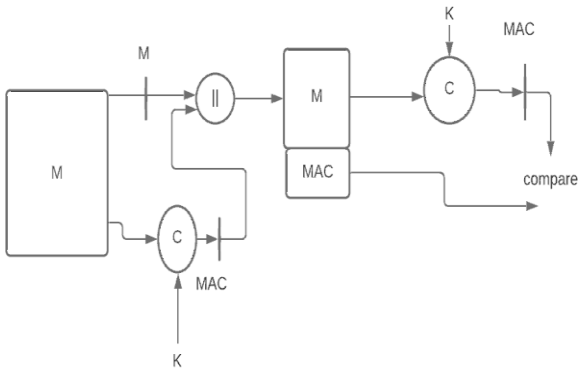
Message authentication functions must be applied to the message(plain text) and the secret key to generate a MAC of fixed-size –



2.1 Basic MAC

Suppose station A wants to send a message to station B over a channel. In that case, station A appends an authentication tag or MAC to the plain text message, calculated by applying some message-authentication functions and using a secret key on the plain text.

At the receiving end(station B) uses the same message-authentication functions and the same secret key to regenerate the MAC and is compared to the appended MAC, on comparison; if both the MAC's are the same, it means that the message is safe and is not forged or tampered with if they differ vice versa.



Here MAC is a fixed-length code.

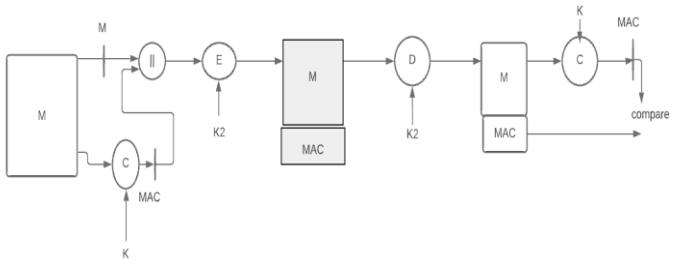
||= appendment

However, this method is not entirely secure. Here, the message is totally in plain text and is not encrypted, so there is a chance of being forged. So, to prevent this, we use a symmetric key.

2.2 Authentication Tied to plain text

To overcome the problem faced in primary MAC, we use a symmetric key to encrypt the data after generating the MAC and is appended to the plain text and is transmitted over the channel.

The same protocols are used to decrypt the data using the symmetric key). Then the MAC is regenerated and is compared to the original MAC from the sender's side appended with the data.



E= Encryption

D= Decryption

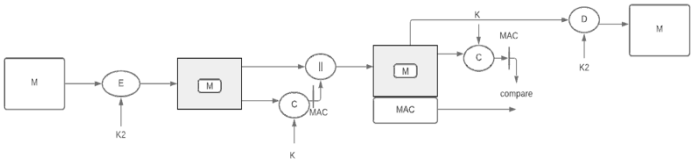
K2= Symmetric Key

||= appendment

2.3 Authentication tied to ciphertext

Here the plaintext is first encrypted using the symmetric key, and then message authentication protocols are applied to the encrypted data using a secret key.

The MAC is first regenerated at the receiving end using the message authentication protocols and the secret key. It is compared to the original MAC from the sender's side appended with the data. The encrypted data is decrypted using the symmetric key into plain text.



E= Encryption

D= Decryption

K2= Symmetric Key

||= appendment

3. Difference between MAC and cryptographic Hash functions

The significant difference between Hash functions and MAC is that MAC is generated using message authentication functions and a secret key. At the same time, hash code is generated by only utilizing the Hash functions.

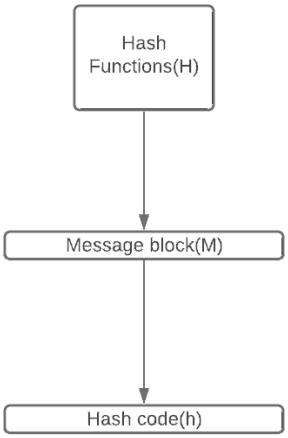
4. Cryptographic Hash Functions

Hash functions are used to compress a long message into a fixed-size message called a hash code. However, it satisfies any extra norms and is used for cryptographic applications.

Hash functions work on the principle formula:

H(M)=h

Where H= Hash function, M= Plain text, h= fixed-length hash code



Hash functions were not initially used in message authentication.

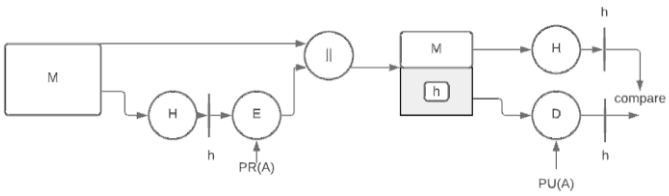
4.1 Hash Functions with public key message encryption or Asymmetric Key

Here the Hash functions are applied to the client text field and then encrypted using the private key of station A, and then the hash code is generated, which is appended to the client text field and is transmitted over the channel.

The same procedure is followed at the receiving end, i.e., the hash code is regenerated using the Hash functions and is compared to the one appended with the data.

However, there are certain drawbacks to this method

- The data is left as it is, i.e., in plain text form without encrypting it.
- The hash code is encrypted using the sender's private key, but it is decrypted at the side of the receiver using a public key of the sender's station, which is available to all the network users.



E= Encryption

D= Decryption

||= appendment

PR(A)= private key

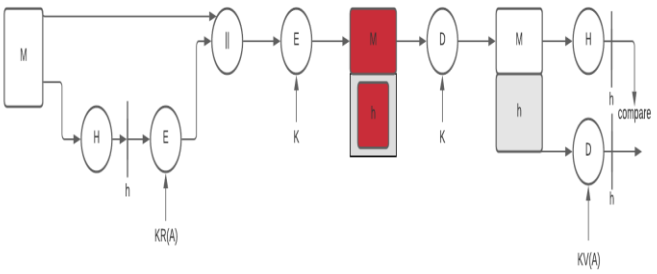
PU(A)= public key

4.2 Hash Function with the use of Symmetric and Asymmetric key

This procedure is used to overcome the drawbacks faced by the above method. Hash functions are applied to the text field, and then the hash code is generated, which is appended to the data using the sender's private key. Only the hash code is encrypted, so we use a secret key to encrypt the text field, encrypting the hash code again(double encryption) and is transmitted over the channel.

On the receiving side, the message is decrypted using the secret key first, and now the text field is decrypted totally and the hash code partially.

Now the hash code is regenerated using the Hash functions, which is then compared to the one appended to the data, decrypted using the public key of the sender's station, which is available to all the network users. In this method, symmetric and asymmetric encryption is done, i.e., asymmetric when using the private key and symmetric when using the secret key.



E= Encryption

D= Decryption

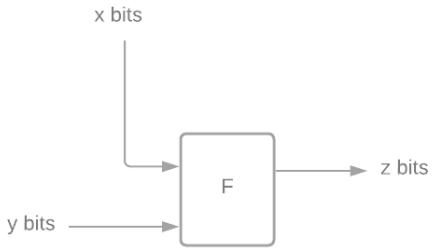
||= appendment

KR(A)= private key

KV(A)= public key

4.3 Hash functions with compression

A compression function takes two fixed-size inputs: a chaining value and a message, and returns a fixed-size value. So it's essentially a hash function with fixed input size.



Where x and y are two fixed-size inputs and z is the fixed size output after compression. [Ri, SHA]

4.4 Merkle Damgard Scheme(iterated Hash function construction)

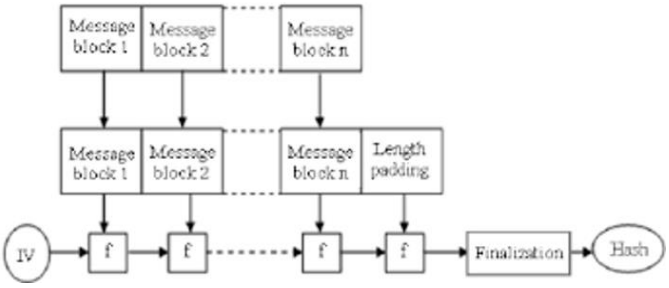
A particular methodology for constructing collision-resistant hash function has been proposed by Merkle [Me] (and later by Damg˚ard [Da].)

Merkle-Damgard is an iterated Hash function scheme. It is a domain extender that turns that compression function into a hash that supports arbitrarily long messages.

This scheme uses the following steps:

(i) The message length and padding are appended to the message to create an augmented message that can be evenly divided into blocks, each consisting of n bits, where n is the size of the block to be processed by the compression function.

(ii) Then the augmented message is divided into t blocks, each consisting of n bits, i.e., $m_1, m_2, \ldots m_t$, so it has titerations, and at each iteration, one message digest will be created, i.e., $h_1, h_2, \ldots h_t$



(iii) There are t iterations; the compression function at the i^{th} iteration operates on H_{i-1} and M_i to create a digest h_i, where h_{i-1} is produced by compression function in the previous iteration

$F(H_{i-1}, M_i) = h_i$

Where f is the compression function before starting the iteration.

The digest h_0 is set to a fixed value, usually called iv(initialization vector).

When the first iteration's compression function f operates on the digest h_0 and first block h_1, the message digest h_1 is created.

Similarly, the second iteration's compression function f operates on the digest h_1 and block m_2, and the message digest h_2 is created.

(iv) The digest H_2 created after the last(t^{th}) iteration is the message digest of the original message.

5. Keyed Hash Functions

This method aims to build a way to perform message authentication securely.

This first obstacle is that although secret keys are an essential part of message authentication, most cryptographic hash(SHA or MDA-5) do not use them. Hence a way to use Hash functions in combination with keys should be defined.

The most frequently used approach to key a Hash function is by applying a key to the data that has already been hashed using the Hash functions(hash code) as mentioned previously. [Ts]

This method aims to hash the Initialization Vector(IV) by replacing some standardly known and fixed IVs derived from the primary function. [VW]

This method also has some significant analytical advantages that help in modeling the Keyed Hash Functions as per the requirement of the security analysis.

This method can also be imitated with the use of padded keys appended to the data.

Keyed Hash functions can be defined as a family of functions by using the Keyed IV method.

Here, Keyed compression function f_k is determined by $f_k(m)= f(H,m)$

Then we associate an iterated Hash function like SHA-1 or MDA-5 as a family of keyed functions.

$\{F_k\}_k$ $m=m_1,m_2$-----$m_n\}$

$F_k(m)=H_{n+1}$ where

$H_i=f(H_{i-1},m_i)$ for i=1----n+1

$H_0=H$ and $m_{n+1}=|m|$

Now the space between the keys in the Keyed compression function and Keyed iterated Hash function is relatively identical.

The original iterated hash function is now obtained as a particular number of the Keyed family, namely $F_{IV.}$

6. Nested Construction(NMAC)

Function of NMAC

Let k_1 and k_2 be two keys of the MAC function F such that $k=(k_1,k_2)$

MAC function NMAC(m)

Where 'm' indicates the input of random length.

$NMAC_k(m) = f_{k1}(F_{k2}(m))$

The outer compression function (f_{k1}) acts on the output of the iteration function ($F_{k2}(m)$). Thus, it only involves one iteration of the compression function.

Here $F_{k2}(m)$ is padded to full block size.

This construction is efficient and straightforward. Although the underlying functions are keyed, the cost is the same as that of the keyless function. The only complex part is the application with additional cost, which again is involved only in one iteration.

This is a very strengthy construction compared to the underlying cryptographic Hash functions.

7. HMAC fixed IV(initialization vector)

As the library code for hash functions like MDA5[Ri] and SHA are widely available, building a MAC mechanism that used these functions is an advantage.

Hence the MAC can just be implemented by calling the function which is already existing.

The applicability to NMAC and HMAC can be shown with the additional premise of the compression function.

8. Functioning of HMAC

Function F is an unkeyed and an iterated hash function that is initialized with its regular IV.

HMAC works on a random length input m and uses an arbitrary string k of length l as its key.

$HMAC_k(m). = F(\oplus opad,F(\oplus ipad,x))$

Here, is made to a fullb= b-bit block size of the iterated hash function by adding 0's the commas representing the concatenation of the data, and '⊕' is the bitwise exclusive.

The ipad uses the byte x'5c', and the opad is created by repeating x'36' as needed to get a b-bit block.

9. Implementation consideration of HMAC

Some issues regarding HMAC are pointed here

(i) HMAC functions slower than NMAC because it requires two additional enumerations of the compression function. This is trivial when authenticating humongous data streams but is significant for small streams of data.

This can be avoided by accumulating(caching) the values of k_1 and k_2. This means that these values are enumerated only when the k is generated or shared the first time and stored as an actual key to NMAC.

The construction needs to initialize the IVs to use separate keys of the hash function before processing the data.

In this way, HMAC helps those constructions that require the iterated hash functions without modification.

(ii) To support the keys in HMAC, the inputs should be more than l-bits. Else the function would be weak, and also, the input much longer than l-bits does not strengthen the function but is safe compared to the one that is less than l-bits.

Lastly, it is essential to manage the keys in a secure way, crucial for security functions.

There should also be a fixed amount of time to refresh the keys. AS extended use of the keys may lead to privacy issues.

10. CONCLUSIONS

It has reached a surprising conclusion after performing experiments, research, testing, and evaluating the method and programme. Message/Data Authentication Using Hash Functions delves into the capabilities of various MACs and Cryptographic Hash Functions, as well as how these experiments may be used to improve their performance and security.

REFERENCES

[1] [At1] R. Atkinson, "Security Architecture for the Internet Protocol", IETF Network Working Group, RFC 1825, August 1995.

[2] [At2] R. Atkinson, "IP Authentication Header", IETF Network Working Group, RFC 1826, August 1995.

[3] [BCK1] M. Bellare, R. Canetti and H. Krawczyk, "Pseudorandom functions revisited: the cascade construction and its concrete security," Proceedings of the 37th Symposium on Foundations of Computer Science, IEEE, 1996.

[4] [BGR] M. Bellare, R. Gue ŕ in and P. Rogaway, "XOR MACs: New methods for message authentication using finite pseudorandom functions," Advances in Cryptology – Crypto 95 Proceedings, Lecture Notes in Computer Science Vol. 963, D. Coppersmith ed., Springer- Verlag, 1995.

[5] [BKR] M. Bellare, J. Kilian and P. Rogaway. "The security of cipher block chaining." Advances in Cryptology – Crypto 94 Proceedings, Lecture Notes in Computer Science Vol. 839, Y. Desmedt ed., Springer-Verlag, 1994.

[6] [BGV] A. Bosselaers, R. Govaerts, J. Vandewalle, "Fast hashing on the Pentium," Advances in Cryptology – Crypto 96Proceedings, Lecture Notes in Computer Science Vol. 1109, N. Koblitz ed., Springer-Verlag, 1996.

[7] Damg å rd, "A design principle for hash functions," Advances in Cryptology – Crypto 89 Proceedings, Lecture Notes in Computer Science Vol.435, G. Brassard ed., Springer-Verlag, 1989.

[8] H. Dobbertin, "Cryptanalysis of MD4," Fast Software Encryption Workshop, Lecture Notes in Computer Sciences, vol. 1039, Springer Verlag, 1996, pp. 53- 69.

[9] Vol. 2 No. 2, Summer 1996. http://www.rsa.com/rsalabs/pubs/cryptobytes.html National Institute for Standards and Technology, "Digital Signature Standard.

[10] [KBC] H. Krawczyk, M. Bellare and R. Canetti, Internet draft draft-ietf-ipsec-hmac-md5- txt.00, March 1996.

[11] [Me] R. Merkle, "One way hash functions and DES," Advances in Cryptology – Crypto 89 Proceedings, Lecture Notes in Computer Science Vol. 435, G. Brassard ed., Springer-Verlag, 1989. (Based on unpublished paper from 1979 and his Ph. D thesis, Stanford, 1979).

[12] [Ne] J. Nechvatal, "Public Key Cryptography," in Contemporary Cryptography, The Science of Information Integrity, G. Simmons ed., IEEE Press, 1992.

[13] [PV1] B. Preneel and P. van Oorschot, "MD-x MAC and building fast MACs from hash functions," Advances in Cryptology – Crypto 95 Proceedings, Lecture Notes in Computer Science Vol. 963, D. Coppersmith ed., Springer-Verlag, 1995.

[14] [PV2] B. Preneel and P. van Oorschot, "On the security of two MAC algorithms," Advances in Cryptology – Eurocrypt 96 Proceedings, Lecture Notes in Computer Science Vol. ??, U. Maurer ed., Springer-Verlag, 1996.

[15] [Ri] R. Rivest, "The MD5 message-digest algorithm," IETF Network Working Group, RFC 1321, April 1992.

[16] [SHA] FIPS 180-1. Secure Hash Standard. Federal Information Processing Standard (FIPS), Publication 180-1, National Institute of Standards and Technology, US Department of Commerce, Washington D.C., April 1995.

[17] [To] J. Touch, "Performance Analysis of MD5", Proceedings of Sigcomm '95, pp. 77-86. (See also RFC 1810).

[18] [Ts] G. Tsudik, "Message authentication with one-way hash functions," Proceedings of Info- com 92.

[19] [VW] P. van Oorschot and M. Wiener, "Parallel Collision Search with Applications to Hash Functions and Discrete Logarithms", Proceedings of the 2nd ACM Conf. Computer and Communications Security, Fairfax, VA, November 1994.

[20] [X9.9] ANSI X9.9, "American National Standard for Financial Institution Message Authentication (Wholesale)," American Bankers Association, 1981. Revised 1986.

BIOGRAPHIES

Akhil Chandra Gorakala,
B.Tech Final year,
Dept. of Computer Science and Engineering, GITAM Deemed to be University.

K V R A Vikram,
B.Tech Pre-Final year,
Dept. of Computer Science and Engineering, GITAM Deemed to be University.

Venkateswarlu K,
Assistant Professor,
Dept. of Computer Science and Engineering, GITAM Deemed to be University.