2011 International Conference on Information and Network Technology
IPCSIT vol.4 (2011) © (2011) IACSIT Press, Singapore

Deploying Modified Hash Based Message Authentication Code HMAC in MATLAB Using GUI Controls

B.Sridevi[1], Dr.S.Rajaram[2]

[1]Assistant Professor, Department of ECE, Velammal College of Engineering and Technology, Madurai Tamilnadu, India
[2]Associate Professor, Department of ECE, Thiagaraja College of Engineering, Madurai Tamilnadu, India

Abstract: User Authentication is defined as 'Provision of Assurance that the message is originated from authorized user'. Message Authentication is defined as 'Provision of assurance that the message is not altered'. Former is provided by Message Authentication codes (MAC) and later is provided by Hash functions. When MAC uses Hash function to generate authentication code it is called Hash based MAC (HMAC). In this paper a new algorithm is proposed for HMAC which uses MD5 algorithm as hash function and based on parallel processing. The algorithm is implemented using GUI based controls in MATLAB toolbox called GUIDE which enables the algorithm to be friendly with the user.

Keywords: HMAC, MD5, XOR, GUI, MAC.

1. Introduction

Due to the advancement in the field of data communication through computer networks, problem arises with the security of data. In the context of communications across a network, the following attacks can be identified: 1. Disclosure: 2. Traffic analysis: 3. Masquerade: 4. Content modification: 5. Sequence modification**:** 6. Timing modification: 7. Repudiation.

Three ways to implement MAC as shown in Figure 1

1. Encrypt plaintext using CBC mode. Use the final block as the MAC
2. Use a different key for MAC and encryption if a confidential message is sent
3. Hash the message and encrypt the digest. Hash the message along with a shared key MAC generated using hashing is known as an HMAC.

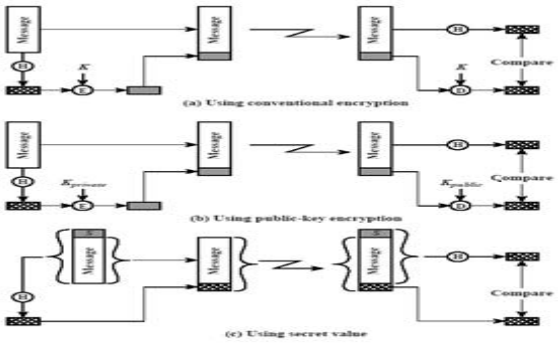


Fig 1: Types of Implementing HMAC MAC

2. 2. The HMAC Algorithm

Figure 1 shows the types of implementing MAC. Figure 1.c shows the HMAC type which is widely used to get secret code.

- **Hash** (·) be a cryptographic hash function.
- K be a secret key,K_0 be K padded to the right with extra zeros to the block size of the hash function
- text be the message to be authenticated
- ‖ denote concatenation, ⊕ denote exclusive or (XOR)
- opad be the outer padding (0x5c5c5c…5c5c, one-block-long hexadecimal constant)
- ipad be the inner padding (0x363636…3636, one-block-long hexadecimal constant)

First author TEL: 0452 2372561
Email address: aisveriya@yahoo.com

2.1 To Determine K_0

1. K=Length of Secret Key
2. B=Block size needed for hash function.
3. D=Length of Digest i.e. output of hash function

2.2 Step-By-Step Description

Step 1: If the length of K = B: set K0 = K. Go to step 4.

Step 2: If the length of K > B: hash K to obtain an L byte string, then append (B-L) zeros to create a B-byte string K0 (i.e., K0 = H (K) || 00...00). Go to step 4. (From Figure 2)

Step 3: If the length of K < B: append zeros to the end of K to create a B-byte string K0 (e.g., if K is 20 bytes in length and B = 64, then K will be appended with 44 zero bytes 0x00).

Step 4: Exclusive-Or K0 with ipad to produce a B-byte string: **K0⊕ ipad**.

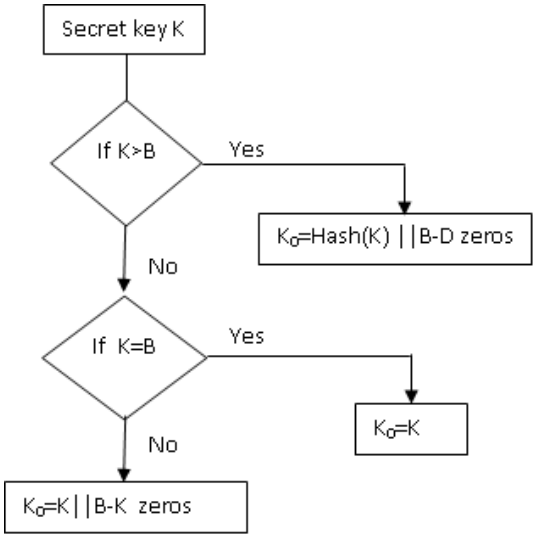


Fig 2: Determination of Ko

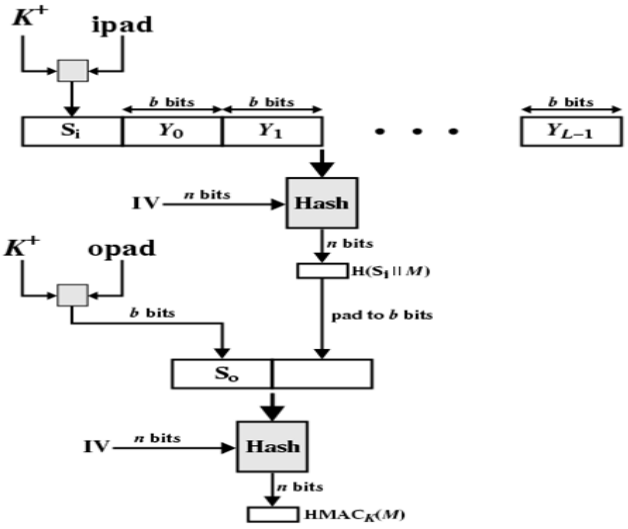


Fig 3: HMAC Algorithm

Step 5: Append the stream of data 'text' to the string resulting from step 4 :**(K0 ⊕ipad) || text.**

Step 6: Apply H to the stream generated in step5: **H ((K0 ⊕ipad) || text).**

Step 7: Exclusive-Or K0 with opad: **K0 ⊕opad.**

Step 8: Append the result from step 6 to step 7:**(K0 ⊕opad) || H((K0 ⊕ipad) || text).**

Step 9: Apply H to the result from step 8:**H((K0 ⊕opad)|| H((K0⊕ipad) || text)).**

Step 10: Select the leftmost t bytes of the result of step 9 as the MAC[4].

3. Role Of Hash Algorithm

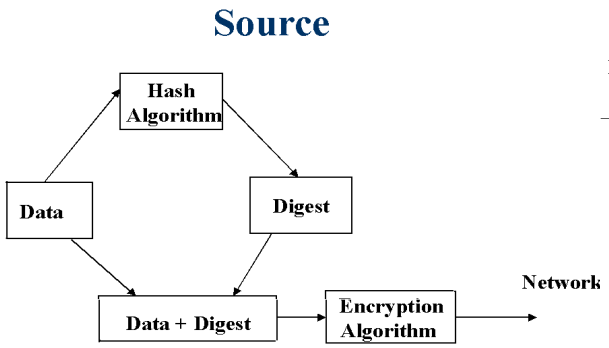


Fig 4: Role of Hash Algorithm in Source

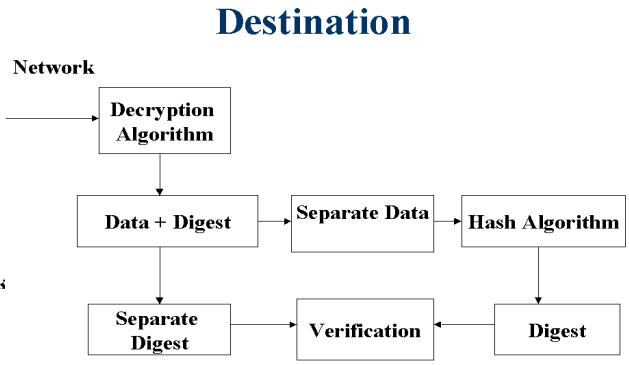
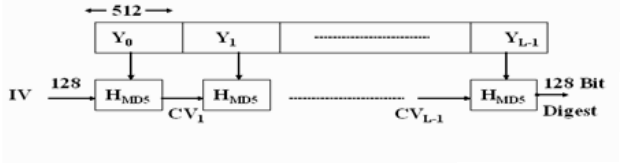


Fig 5: Role of Hash Algorithm in Destination

A hash function meeting the criteria may still have undesirable properties. For instance, most popular hash functions are vulnerable to length-extension attacks: given h (m) and len(m) but not m, by choosing a suitable m' an attacker can calculate h(m || m'), where || denotes concatenation. This property can be used to break naive authentication schemes based on hash functions. Functions intended for cryptographic hashing, such as MD5, are commonly used as stock hash functions [5].



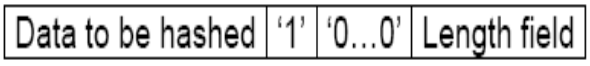


Fig 6: Hash Function

The Secure Hash Functions perform padding by doing the following:

1. Append a binary '1' pad bit.
2. Append binary zero pad bits up to the block length minus the size of the message length field. For SHA-256 and less the block length is 512 bits and the message length field is 64 bits. For SHA-384 and greater the block length is 1024 bits and the message length field is 128 bits.
3. Append the message length field containing a value consisting of the bit length of the input data.

4.MD5 (Message Digest) Algorithm

4.1 Computational Steps:

1. This algorithm takes as input a message of arbitrary length $<2^{64}$as in Fig 3. Message is padded so that its **Length = Multiple of 512 -64** as referred in [1], [2]
2. A 64 bit representation of length of original message is appended to the result of step1 .Thus the length will be **multiple of 512**.Now the expanded message is represented as blocks of 512 bits.
3. A buffer of 128 bit is initialized with four 32bit values

A=67 45 23 01, **B**=EF CD AB 89, **C**= 98 BA DC FE, **D**=10 32 54 76

In little endian format (LSB comes first).

A=01 23 45 67, B=89 AB CD EF, C=FE DC BA 98, D=76 54 32 10.

4. This step is named as HMD5 and this is the heart of the algorithm. This is a compression function that consists of four rounds as in Fig 4. Each round is having 16 steps. Four rounds have similar structure but each uses different primitive logical function F, G, H and I [2],[3], [6].
5. After all L 512 bit blocks have been processed, the output from the Lth sage is the 128-bit message digest.

$$CV_0 = IV \quad (1)$$

$$CV_{q+1} = SUM_{32}\left(CV_q,\ RF_I\left[Y_q, RF_G\left[Y_q, RF_G\left[Y_q, RF_F\left[Y_q, CV_q\right]\right]\right]\right]\right) \quad (2)$$

$$MD = CV_L \quad (3)$$

4.2 MD5 Compression Function:

Some formulae used are

1. **T[i]** is the Integer part of 2^{32} * abs (sin (i)) where i is in rad.

2. **X[i]** is M [q*16+i] --> ith 32 bit word of qth block.
3. **ρ2i** is (1+5i) mod 16.

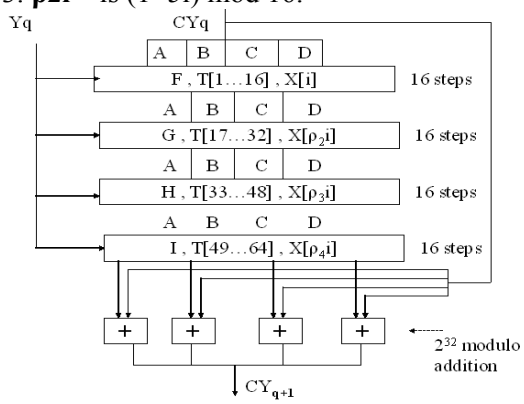


Fig 7:MD5 Compression

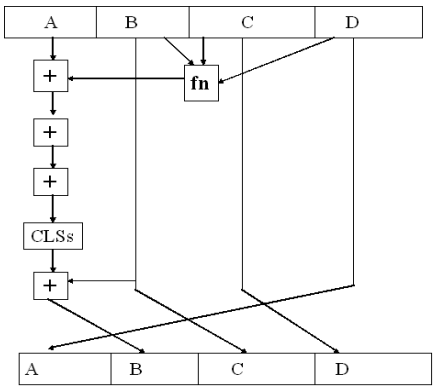


Fig 8:MD5 Single step operation

4. **ρ3i** is (5+3i) mod 16.
5. **ρ4i** is 7i mod 16.
6. **CLSs =** **C**ircular left shift by s bits

4.3 Single Step Operation:

The new Values of A, B, C, D will be updated by following equations as in Fig 8.

$$B = B + (A + fn(B,C,D) + X[k] + T[i] <<< S) \quad - \quad (4)$$

$$C = B \quad - \quad (5)$$

$$D = C \quad - \quad (6)$$

$$A = D \quad - \quad (7)$$

5. Proposed Method :

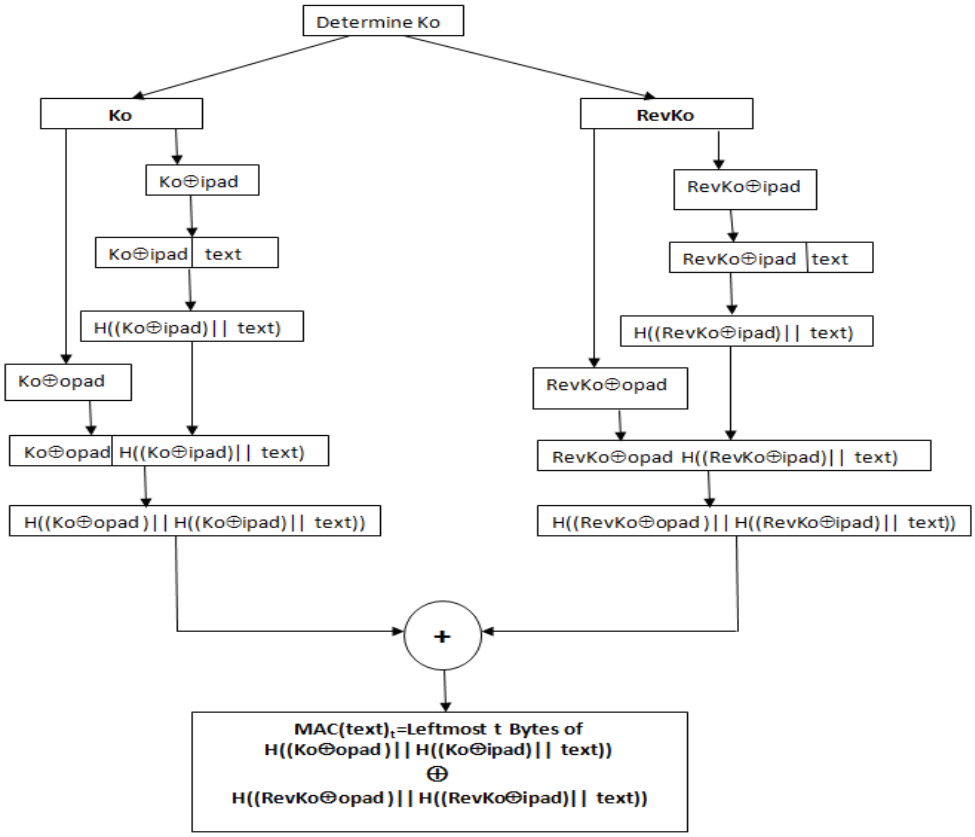


Fig 9: Modified HMAC

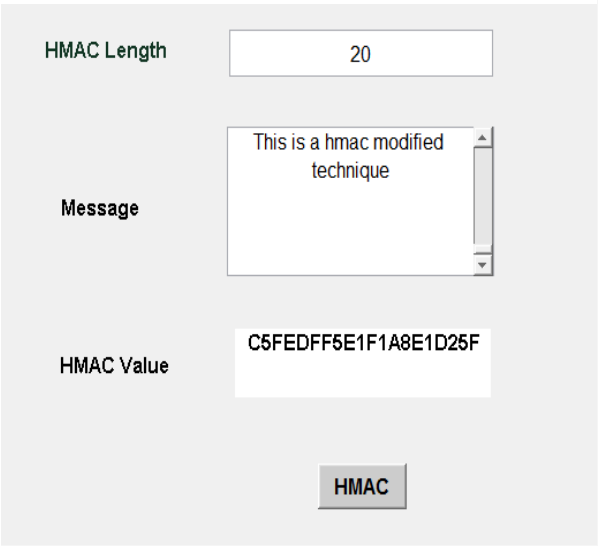
In this proposed work Ko is found similar to the conventional HMAC technique. From Ko, RevKo is derived which is the reverse of Ko i.e. reversing the binary equivalent of Ko. HMAC technique is done separately with Ko and RevKo. The result of both sides is XORed to get the MAC value. In this work the conventional HMAC using MD5 hash algorithm is implemented [2],[3]. Then the proposed algorithm has been implemented.

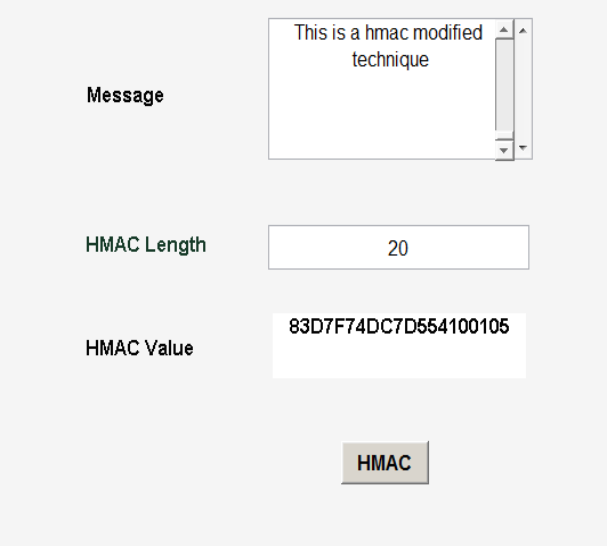
6. Simulation Results

This work has been implemented in MATLAB package. The MD5 is implemented in MATLAB separately which is called by a function in HMAC program. Before implementing the proposed algorithm the conventional HMAC is implemented with MD5 Hash function. Then the proposed HMAC is implemented as a separate program. To be friendly with the users, the proposed HMAC and conventional are implemented with GUI controls of MATLAB called GUIDE.

6.1 Conventional HMAC

Enter the Text: **'This is a hmac modified technique'**
Enter the length of HMAC: **20**
C5FEDFF5E1F1A8E1D25F





GUI based Conventional HMAC **GUI based Proposed HMAC**

6.2 Proposed HMAC

Enter the Text: **'This is a hmac modified technique'**
Enter the length of HMAC: **20**
83D7F74DC7D554100105

7. Limitation of MAC Algorithms

This algorithm takes more time compared to conventional algorithm but it improves the security by confusing man in the middle attack. The successful verification of a MAC does not completely guarantee that the accompanying message is authentic: there is a chance that a source with no knowledge of the key can present a purported MAC on the plaintext message that will pass the verification procedure. For example, an arbitrary purported MAC of t bits on an arbitrary plaintext message may be successfully verified with an expected probability of (1/2) t. This limitation is inherent in any MAC algorithm [6].

8. Conclusion

Thus to conclude, this paper introduces a modified algorithm for HMAC. This work is implemented as a MATLAB program as well as GUI based tool so that this work can be utilized directly for IP Security as well as in communication projects which needs authentication and integrity [1]. Applications like Mobile

Communication, WIMAX and Optical Communication can use this work for key sharing during handshaking.

9. References

[1]. Marc Fischlin, Anja Lehmann, and Daniel Wagner Hash Function Combiners in TLS and SSL, J. Pieprzyk (Ed.) LNCS 5985, pp. 268–283, 2010.,*Springer-Verlag Berlin Heidelberg 2010*

[2]. E.Biham and R.Chen "Near Collisions of SHA-0", *Advances in Cryptology CRYPTO 2004*, LNCS 3152, Spriger-Verlag, pp 290-305, 2004.

[3]. R.L.Rivest, "The MD5 Message Digest Algorithm" -*IETF Request for Comments*.

[4]. "The Keyed-Hash Message Authentication Code (HMAC)" *in Federal Information Processing Standards Publication*

[5]. Mihir Bellare, Ran Canettiy, Hugo Krawczykz, *"Keying Hash Functions for Message Authentication*" in *Advances in Cryptology -Crypto 96 Proceedings*, Lecture Notes in Computer Science Vol. 1109

[6]. Debrup Chakraborty and Palash Sarkar, HCH: A New Tweakable Enciphering Scheme Using the Hash-Counter-Hash Approach, *IEEE Transactions On Information Theory, Vol. 54, No. 4, April 2008*