Journal of Engineering and Development, Vol. 10, No. 3, September (2006) **ISSN 1813-7822**

Auth-Intg Security System for Communication Protocols

Lect. Kais A. Nafi Ridha
Computer & Software Eng. Dept., College of Eng.
Al-Mustansiriya University, Baghdad, Iraq

Asst. Lect. Anas Y. Hamza
Computer & Software Eng. Dept., College of Eng.
Al-Mustansiriya University, Baghdad, Iraq

Eng. Muna Dhia Sheet Khattab
Computer & Software Engineering Dept., College of Eng.
Al-Mustansiriya University, Baghdad, Iraq

Abstract

This work presents the development of a secure communication system that support protocols with built-in security capabilities, which are authentication and integrity capabilities. These capabilities can be achieved by applying a new proposed authentication protocol within the standard protocols.

The proposed authentication protocol is based on two different authentication services, which are Two-Way Challenge-Response authentication service, and One-Time Password authentication service. These two services are based on two of the most secure hashing functions, which are Message Digest version 5 (MD5), and Secure Hashing Algorithm (SHA-1).

The proposed communication system will be more secure than systems that depend only on cryptography algorithms, since the attacker should analyze and break two different security service algorithms.

الخلاصـــــة

هذه الورقة تقدم تطوير نظام اتصالات حصين والذي يجهز بروتوكولات الاتصالات بإمكانيات أمنية مبنية والتي هي خدمات الموثوقية والسلامة.

هذه الإمكانات تتم بتطبيق مقترح بروتوكول توثيق جديد ويُضَمَنُ في بروتوكولات اتصالات المنظومة. البروتوكول المقترح يعتمد على خدمتي توثيق هما الاختبار والاستجابة بكل الاتجاهين واستخدام كلمة السر المستخدمة لمرة واحدة.

هذه الخدمتين مبنية على أكثر الدوال التكرارية (Hash functions) أمناً وهما خوارزمية (MD5) و خوارزمية (SHA-1).

النظام المقترح سيكون أكثر أمناً من الأنظمة التي تعتمد على خوارزميات التشفير فقط، بسبب أن المهاجم يجب أن يحلل ويكسر خدمتين أمنيتين مختلفتين لغرض كسر النظام.

1. Introduction

When connecting to the Internet, it is essential to realize that the standard network protocols have security shortage. Messages sent over the Internet can be intercepted, redirected, or even fabricated by attackers. In particular, when a message is received, it is possible the apparent sender (e.g. the sender's name in an Email message) may not be the real sender; further, it may have been read and/or modified by attackers in transit [1]. There are many ways attackers may accomplish the above attacks. The most common techniques are known as [2]:

1. **Packet Sniffing:** due to network topology, the packets sent from a source to a specific destination can also be read by other nodes, which can then get hold of the payload (e.g., that may contain passwords or other private information) to subvert the logical organization of the network.
2. **IP Spoofing:** IP addresses can be very easily spoofed both to attack those services whose authentication is based on the sender's address.
3. **Connection Hijacking:** Whole TCP/IP packets can be forged to appear as legal packets coming from one of the two communicating parties, the goal of the attack being to insert his data in an existing channel.

All such attacks can be foiled with the large body of techniques known as "cryptography". These techniques are important to protect the sent message from threats to the confidentiality, integrity, authenticity, and availability as it transits through a potentially hostile environment like the Internet [3].

2. Security Technology

The lowest level of security technology is the security mechanisms, which is a mechanism that is designed to protect, prevent, or recover from a security attack. These mechanisms provide fundamental algorithms such as random numbers generators, encryption, and cryptographic hash functions [4]. The second layer of security technology is the security services which are independent of protocols that using them. The security service can be defined as a service that enhances the security features of the data processing systems and the information transfers of on the network. It is intend to counter security threats by using one or more security mechanisms to achieve its purpose [5]. At the top level of security technology is the framework that integrate the security services into security protocols [6]. **Table (1)** illustrates the structure of security technology.

Table (1) General layer stack of security technology

Security Protocol	Authentication protocol, Integrity protocol, …etc
Security Services	CRAM, OTP, …etc
Security Mechanisms	PRNG, Hash Function (MD5, SHA, ..etc), HMAC, and Encryption Techniques (Symmetric, Asymmetric)

2-1 Cryptographic Hash Function

Hash function defined as an algorithm that takes a block of input data and generates a shorter fixed length piece of it, based on specific rules. The purpose of a Hash function is to produce a "fingerprint" of that block of data. A special variant is a "Cryptographic Hash Function", which has specific requirements, making it suitable for certain uses in cryptography. In general a cryptographic Hash function "H" must have the following properties [6]:

1. H can be applied to a block of data of any size.
2. H produces a fixed-length output.
3. H(X) is relatively easy to be computed for any given data X, with respect to both hardware and software simplicity and practical implementation.
1. It is a one way function, that is computationally infeasible to find the source data "X" from the output data "h", where h =H (X).
2. Finally it should be strongly collision-free, i.e. each block has his unique output. For X and Y data blocks H (X) ≠H (Y) if the blocks X≠Y.

Hash functions are used in authentication, digital signatures, and data integrity. If any bit in the message is accidentally altered in transmitting, the message's fingerprint will be changed.

Most practically hash function H(M) divides the messages M into fixed-length blocks M_1, M_2, …etc, then padding the last block with zeros and message length. The inclusion of a length value makes more difficult a kind of attack known as a padding attack. The resultant last block (after all padding) is denoted by M_n. The last step is applying a collision free, one way function H to each of the blocks sequentially. **Figure (1)** illustrates this procedure graphically [7].

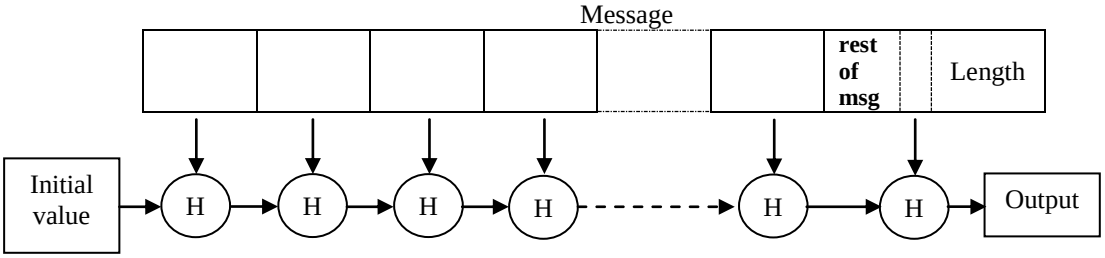

Figure (1) Mechanisms of the practical hash function

Three criteria are taken into account for choosing the cryptographic hash functions that are used in the proposed system. The first criterion is resistance against all known attacks, followed by speed and code compactness on a wide range of platforms, and finally the design simplicity. For these criteria, two Hash functions are chosen, Message Digest Algorithm version 5 (MD5) [8] and Secure Hash Algorithm (SHA-1) [9]. **Table (2)** shows a comparison between these two Hash functions.

Table (2) Comparison between MD5 and SHA algorithms

Difference point	MD5	SHA-1
Digest length	128 bits	160 bits
Basic unit of processing	512 bits	512 bits
Number of steps	64 (4 round of 16)	80 (4 round of 20)
Maximum message size	∞	2^{64-1} bit
Primitive logical function	4	4
Additive constants used	64	4

2-2 Hashed Massage Authentication Code (HMAC)

HMAC is a cryptographic Hash function mechanism for Message Authentication Code (MAC). This mechanism used with a cryptographic Hash function (e.g. MD5, SHA-1) in combination with a secret shared key. The cryptographic strength of HMAC depends on the properties of the underlying Hash function. **Figure (2)** illustrates the overall operation of HMAC [10].

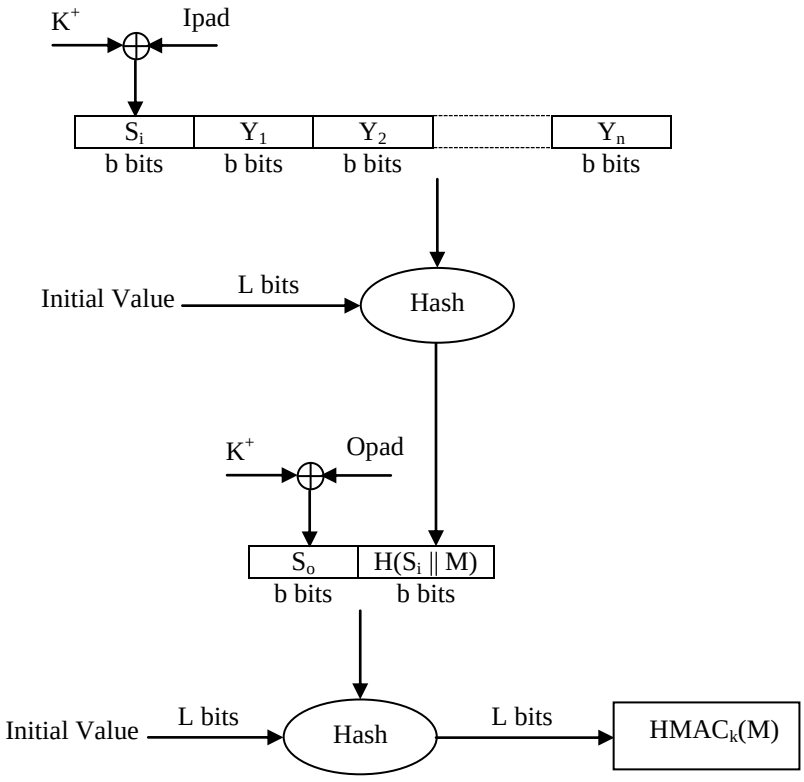

Figure (2) HMAC structure

The XOR of key with Ipad results in flipping one-half of bits of K. similarly, the XOR with Opad results in flipping one-half of the bits of K, but a different set of bits. In effect, by passing S_i and S_o through the Hash algorithm. The result is two keys generated pseudo-randomly from K [5]. The HMAC can be expressed mathematically as:

$$\mathbf{HMAC_K(M) = H[\ (K^+ \oplus opad) \ ||\ H[\ (K^+ \oplus ipad) \ ||\ M]] \ldots\ldots\ldots\ldots\ldots\ (1)}$$

where:

H:	Embedded hash function (MD5, SHA-1).
M:	Message input to HMAC.
n:	Number of blocks in M.
Y_i:	i-th block of M. $0 \le i \le B-1$.
b:	Number of bits in a block (for MD5 & SHA-1, b=512).
L:	Length of hash code produced by embedded hash function.
K:	Secret key.
K^+:	K padded with zeros on the left so that the result is b bit length.
Ipad:	00110110 (36 in hexadecimal) repeated b/8 times.
Opad:	01011100 (5C in hexadecimal) repeated b/8 times.
||:	String concatenation.

2-3 Challenge Response Authentication Mechanism (CRAM)

CRAM is an authentication service designed as an alternation to transmitting plain-text password over the network [6].

To authenticate, a client and server need a shared secret. The server issues a challenge to the client containing arbitrary string of random digits, timestamp, and the Fully Qualified Domain Name (FQDN) of the server. This information is formatted as a MSG-ID token.

The client responds to the challenge with the name of the user, followed by a space and a digest value. The digest value is computed by applying the Hashed for Message Authentication Code (HMAC), by using a shared secret as a key and MSG-ID as a text to HMAC functions.

When the server receives the response, it verifies the digest value. If the verification succeeds, the client is considered authenticated. **Figure (3)** illustrates CRAM service [11].

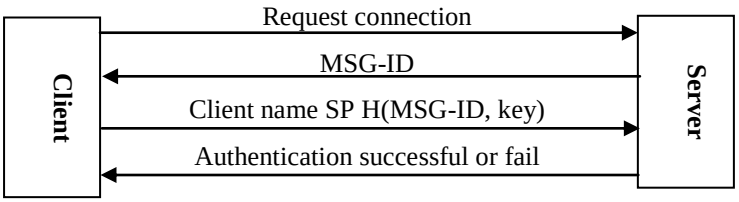

Figure (3) Mechanism of CRAM authentication service

2-4 One Time Password (OTP)

OTP provides a type of single-use passwords list for authentication purpose. The first RFC describing OTP was RFC1938 (A one Time Password system) [5]. In OTP, a secret pass-phrase is used to generate a list of single-use passwords. Each time a successful authentication is performed, the current password is expired. The next time the client needs to authenticate, it uses the next password in the list [6].

Since no secret needs to be provided through the terminal, OTP is well suited for environments that use public access terminals or for any environment where the security of input terminal cannot be completely trusted [12].

At the highest level, there are two elements involved in an OTP service. The client that generates the one-time passwords as needed is defined as generator, while the entity that the client is attempting to authenticate to it is defined as server. The mechanism of generating passwords can be simplified as follows:

1. A secret pass-phrase is used when generating passwords, and visible only to the generator. It should be at least 10 characters in length [13].
2. The OTP client concatenates the user's pass-phrase and the seed value sent from the server. The concatenated string is then passed through a cryptographic hash function and reduced to 64 bits.
3. The initial 64-bit value is processed through the hash function n times. For each subsequent password, the number of iterations is decrement by one. Thus to generate three one-time-passwords, the first hashed three times, the second is hashed twice, and the third is hashed once. The result is a series of 64-bit one-time passwords; the process is described in **Fig.(4)**.

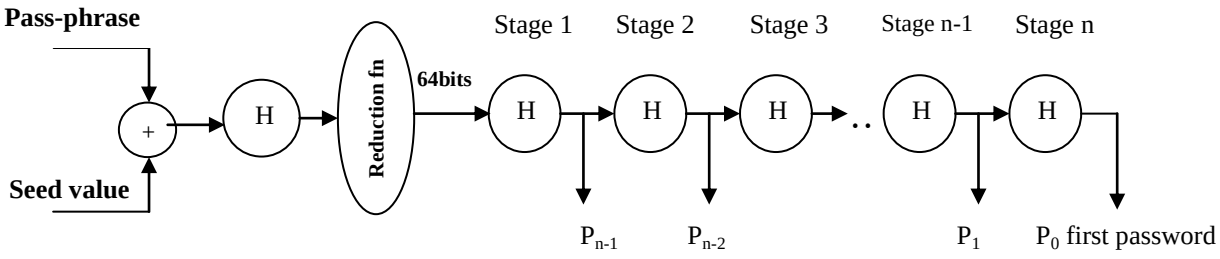

Figure (4) OTP password generation

The reduction function is very simple; the reduction value is obtained from the less significant 64-bit of the digest value [6]. The process of authentication is illustrated in **Fig.(5)**, with the following sequence:

1. The client sends its identity as a request for authentication.
2. The server sends a challenge to the client containing a sequence number of the expected password, this challenge is in a standard format so that automated generator can programmatically extract the necessary information.

3. The client uses the sequence number returned in the challenge to calculate the next OTP, this password should send.
4. The server verifies the validation of the received OTP.
5. The process for verifying an OTP is simple. The server maintains a database containing the OTP from the most recent successful authentication. The server passes the current passwords that come from the client through the Hash function once.

If the result matches the value of the previous password (stored in the database), the server stores the current password in the database and considers the authentication successful [13].

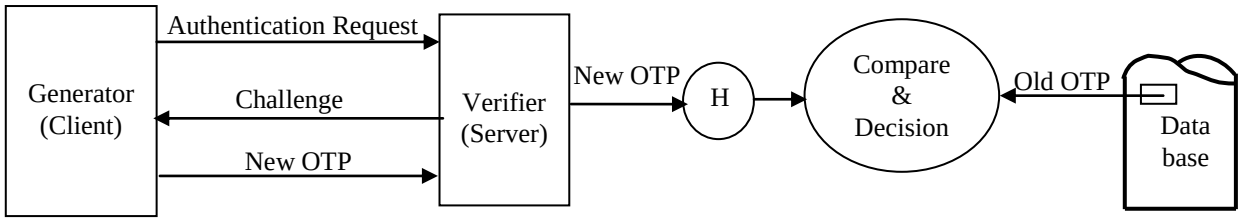

Figure (5) OTP authentication service

3. Security Protocol

A protocol is a multi-party algorithm, defined by a message format and a sequence of steps precisely specifying the actions required of two or more communication parties in order to achieve a specified objective [7].

Ordinary Network protocol enables the parts of a computer network to communicate with each other, but the security (cryptographic) protocol is a communication protocol that has been designed to operate in a potentially hostile environment [14].

Security protocols use cryptographic algorithms to enable the communicating parties to communicate in a secure way. On the other hand, the cryptographic protocol is a protocol that enables cryptographic mechanisms such as encryption and one-way hash functions to guarantee the confidentiality, authenticity, integrity, and/or availability.

Basically, an authentication protocol is a cryptographic protocol whose purpose is to create enough of evidence that one or more of the communicating parties may reasonably believe in something like identity, authorization…etc. In a client-server system, the client may want to convince the server that it has legitimate rights to access the service provided by the server, and the client may want to believe that the server is the right server. By exchanging cryptographic protected information, the communicating party may prove that it is really a legitimate one.

3-1 Key Establishment Protocols

Authentication protocol involves message exchanging, which require precise definition of both the message to be exchanged and the actions to be taken by each party.

The cryptography messages to be exchanged should contain or be processed by a secret key (password). These secret keys often establish by one of two types of establishment protocols [7]. These types of protocols are illustrated in **Fig.(6)**.

1. **Key establishment protocol:** is a process or protocol whereby a secret (key) becomes available to two or more parties, for subsequent cryptographic use.
2. **Authenticated key establishment protocol:** is a process or protocol to establish a secret (key) with a party whose identity has been (or can be) corroborated.

The authentication protocol usually uses one of these two protocols to establish the key required for authentication process. For the two comminuting parties, the key may be symmetric or asymmetric, the key either be static, which is fixed for all communication sessions between these two parties or dynamic, which changes in a manner with each session [11].

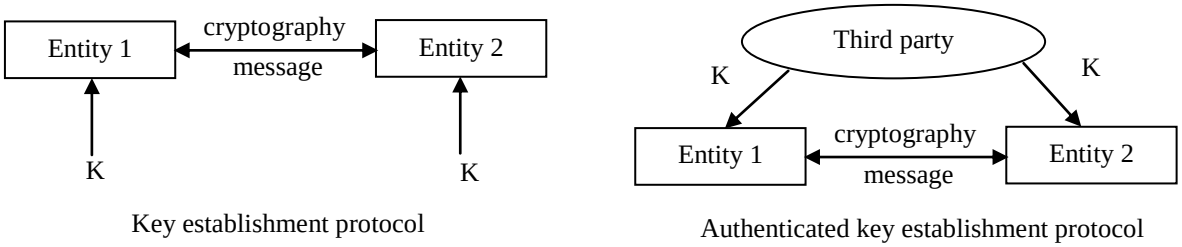

Figure (6) Key establishment protocols

4. The Suggested Authentication Protocol

The main aspect of this research is to design and implement an authentication protocol that can be added as an extension to essential network protocols. The extension protocol is designed to ensure operability and counter current authentication attack on the message as it travels from node to node through network system components. The proposed protocol has the following features:

1. Based on a Challenge-Response service.
2. Based on a shared secret key.
3. Ensuring Two-way authentication.
4. Its keys are One-Time-Password.
5. It is simple to add to many Network protocols like SMTP and POP3.
6. Having simple & efficient key establishment protocol.
7. Robust to current authentication attacks.

The proposed Authentication protocol is based on a principle found in many Authentication protocols; it uses two authentication services and combines between them to get more strong protocol. These services are CRAM and OTP. Each of these services has its advantageous feature that makes it strong to many security attacks.

4-1 Authentication Process and Message Sequence

The goal of an authentication protocol is achieved by exchanging cryptographic protected messages. Usually these messages involve information about the identity of the communicating parties, the current session as well as the time stamp.

The proposed authentication is done between the two communicating partners, so that the client authenticates to the server, and the server authenticates to the client, To authenticate, the client and the server need a shared secret (*Key*). This Key changes dynamically with each communication session. The changing is achieved by key update process. All exchange messages in authentication session are text oriented and composed from the US-ASCII character sets. The authentication process can be summarized as follows:

1. The client sends its identity to the server as a request for authentication.
2. The server then issues a challenge denoted as MSG-ID1 to the client. The MSG-ID1 token contains an arbitrary sequence of random digits, time stamp, FQDN of the server, and sequence number of the session.
3. The client responds to the challenge with the MAC value denoted as MSG-MAC1 followed by a space and another challenge denoted as MSG-ID2. The MSG-ID2 token that is sent to the server contains an arbitrary sequence of random digits, a time stamp, User identity or FQDN of the client, and the sequence number of the session. The MAC value is computed by applying a proposed MAC generator feeds with a shared secret as a key and MSG-ID1 as a text to the generator.
4. When the server receives the later response. It verifies the MAC value by regenerating it from the key and MSG-ID1 (since it knows the valid values), and compares the result with the MSG-MAC1 that received from the client.
5. Finally, the server responds to MSG-ID2 by the MAC value that is denoted as MSG-MAC2, and sends it back to the client. When the client receives the response, it verifies the MAC value. If two verifications succeed, the two parties are considered authenticated. **Figure (7)** illustrates the authentication process graphically.

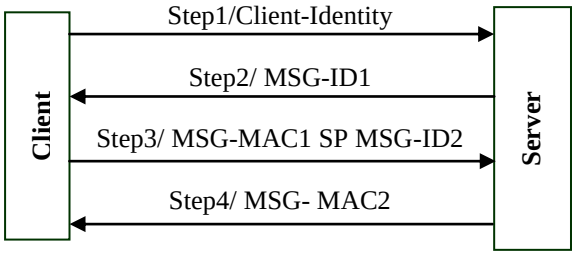

Figure (7) The proposed two-way challenge-response authentication protocol

The proposed authentication protocol defines the format and sizes of these exchanged messages as described in **Table (3)**.

Table (3) Messages format and length

Message	Format	Maximum length in characters
MSG-MAC	(16 or 20) Hex-digits depend on Hashing method	20
MSG-ID	[Random Digits] * [Time stamp] * [Account or FQDN] * [Sequence No.]	297
Random Digits	Decimal Random Number. It is 9 digits	9
Time stamp	DD/MM/YYYY SP HH:NN:SS a full description of the current time in Greenwich mean time	19
Sequence No.	A Number represents the session Number.	10
Account or FQDN	1-256 alphanumerical characters	256
Client-Identity	1-64 alphanumerical character	64

To ensure integrity, all messages that will be exchanged later should be digitally signed by the MAC value of them by using the current session key. There are two possible approaches. The first, the MAC value can be appended to the message and then the new created message is sent. The second, the MAC can be sent, and then the original message is sent separately. **Figure (8)** describe this process graphically.

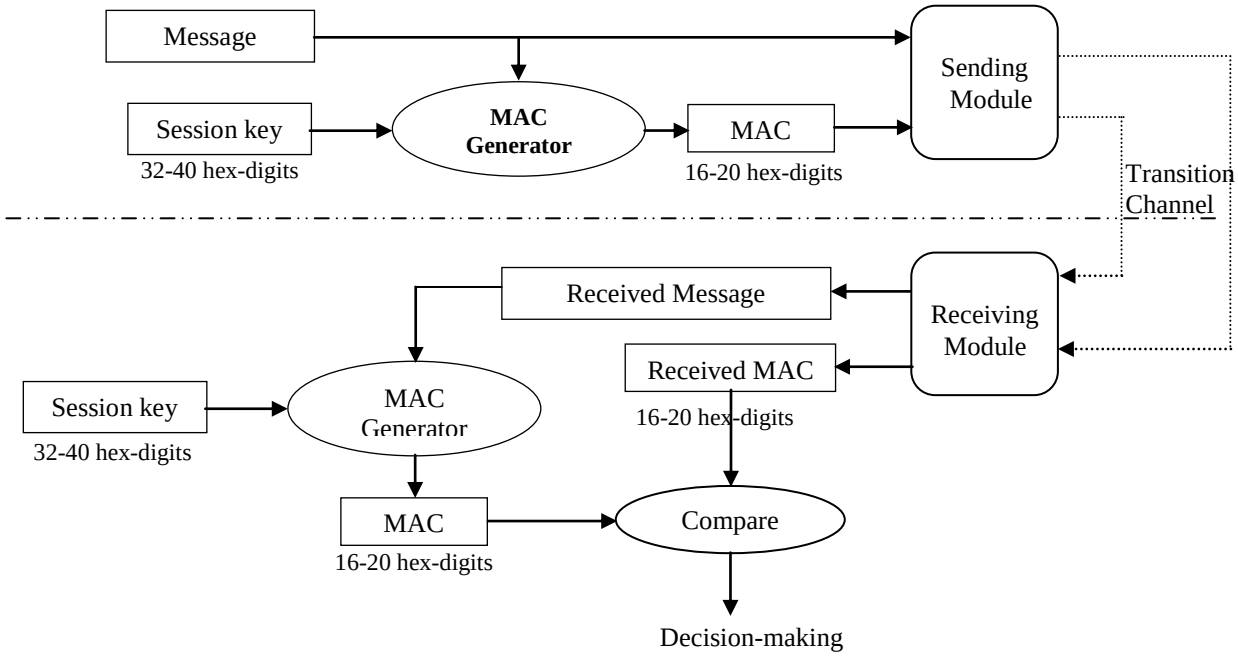

Figure (8) Message integrity using a message authentication code (MAC)

The keys are generated by using Key Establishment Process. The keys used are sequence of hex-digits of length equal to 32 or 40 characters. All the authentication and key generating processes are based on only one selected hash method (i.e. MD5, or SHA-1).

4-2 Massage Authentication Code Generator (MAC)

The integrity of information transmitted checking over an unreliable medium is a primer necessity in the world of open computing and communication. The mechanisms that provide such integrity check based on a secret key. These mechanisms are usually called *Message Authentication Codes* (MAC). Typically, MAC is used between two parties that share a secret key in order to validate information transmitted between them.

In order to generate the MAC value of message bits, the proposed generator uses the system of the structure shown in **Fig.(9)** which is based on HMAC function. It uses an additional truncating function. This truncating function was used twice, first for reduction the used Key to its half length, and the other used to truncate the outcome digest value from HMAC function to its half length.

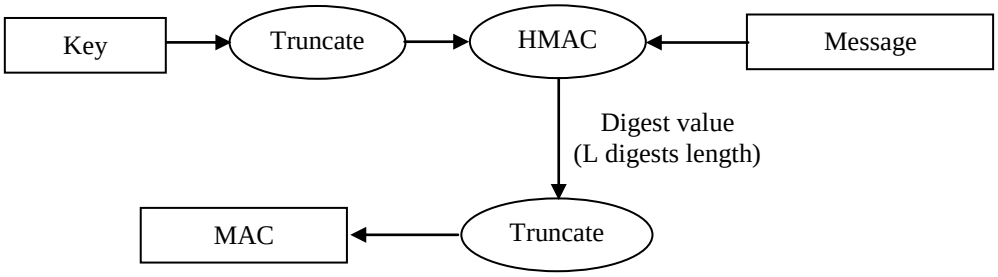

Figure (9) The structure of MAC generator

This truncating process function is to truncate the most significant L bytes from the total T input stream of bytes, where $T/2 \leq L \leq T$. The proposed system uses L equal to T/2 which gives the following security feature:

1. Preventing MAC bits from attacker since the proposed system transmits half generated MAC bits. This technique is against crypto-analysis attack
2. Disclosing the key used in HMAC function (T-key) by attacker, will not disclose the actual key used (session key).

4-3 Message Identifier (MSG-ID)

The MSG-ID token is a random unique identifier used between two communicating entities as a challenge in any challenge-response protocol. It consists of four parts separated by star (*) character, with maximum total length equal to 297 charters. These parts are:

1. Arbitrary sequence of random digits.
2. Time stamp of the current time in Greenwich Mean Time (GMT).
3. The issuer FQDN of the connection host or account name for users.
4. Session Sequence Number.

Figure (10) illustrate the proposed structure of MSG-ID token.

Random Digits (1-9 digits)	*	**Time Stamp (19 Characters)**	*	**FQDN or Account (1-256 characters)**	*	**Session Number (10 digits)**

Figure (10) The format of MSG-ID token

Since any MSG-ID is consider strong against security attacks as much as it random and unique, the proposed system use the above sophisticated MSG-ID format to prevent many type of authentication attacks (such as reply attack, reflection attack… etc).

4-3-1 Sequence of Random Digits

The sequence of random digits is generated by using any pseudo random number generator PRNG, the random number generated range is from 0 to 999,999,999. Since the MSG-ID was generated twice in each authentication session one from the client and the other from the server, the random digits for each entity are picked from different range.

To avoid reflection attacks, the client picks a random number from the range of 0 to 499,999,999, while the server picks a random number from the range of 500,000,000 to 999,999,999. The MSG-ID token will be rejected if the random digits received doesn't in the range allowed to the sender (client or server).

4-3-2 TIME STAMP

The existence of time stamp in MSG-ID is for the uniqueness and freshness assurance. With timeliness a party has all reason to believe that the purported other party is involved with the communication now and the authentication process is not a question of reply attack.

In the proposed system the time stamp subfield occupies 19 characters from the length of MSG-ID, with the following format:

Time Stamp = DD "/" MM "/" YYYY *SP* **HH ":" NN ":" SS**

where:

DD:	Day representation.
MM:	Month representation.
YYYY:	Year representation.
HH:	Hour representation.
NN:	Minute representation.
SS:	Second representation.

For compatibility and operability the included time must be determined in universal time. The problem with this approach is that machines' clocks are never exactly synchronized over a network, so there is some error interval during which a time stamp is valid. The proposed system suggests the following accepting inequality to constrain the freshness of the MSG-ID:

Current time- E_{max} < Time stamp + Timeout …… (2)

where:

E_{max}: is the maximum synchronization error between comminuting partner clocks, which will be assumed to be equal to 1min as maximum value.

Current time: is the time in GMT of the local machine.

Time stamp: is the time representation that back in MSG-ID token, which equal the time in GMT on the remote machine.

Timeout: is the maximum time for a TCP/IP packet consider a live in a Network, and it is equal to the maximum time required to transfer the challenge from remote machine to local machine.

4-3-3 Account or FQDN

This portion appended to inform the other party the identity of whom send the MSG-ID, it occupy as a maximum up to 256 characters, this length represent the standard of domain name length that defined by IETF standers. For user's machine, it contains the account name of the user identity and usually up to 64 characters while for public Domain machine, it contains the FQDN of host machine identity and reach up to 256 characters.

4-3-4 Session Sequence Number

This part of the MSG-ID includes a decimal number that reaches up to 10 digits. This number increases by one after each successful authentication session.

The session sequence number informs the other party the sequence number of the expected key (valid password). Since the keys change dynamically for each session, this number represents the agreement between the two authentication parties on a valid key number. Session sequence number also are used to trace any security breach occur.

4-4 Key Establishment Process

This section illustrates the design and implementation of key establishment process and related cryptographic techniques. For authentication purpose key establishment is a process or protocol whereby a shared secret becomes available to the two parties involved in authentication process, for subsequent cryptographic use. Key establishment broadly subdivided into key transport and key agreement.

A key transport protocol or key transport mechanism is a key establishment technique where one party creates (or obtains) a secret value. The creator securely transfers his key to other(s) that will communicate with them.

A key agreement protocol or mechanism is a key establishment technique in which a shared secret is derived by two communication parties as a function of key update.

The proposed system uses an encrypted key transport technique that implemented in many network systems. The key is updated after each successful authentication process in a manner like OTP service by using the proposed Hashing machine. Unlike simple systems

where keys remain fixed for all communication sessions, the proposed system updates the transport key periodically. Key updating is based on last session key through one of two hash functions (MD5, SHA-1), as illustrated in **Fig.(11)**. The key updating occurs after each successful authentication process.

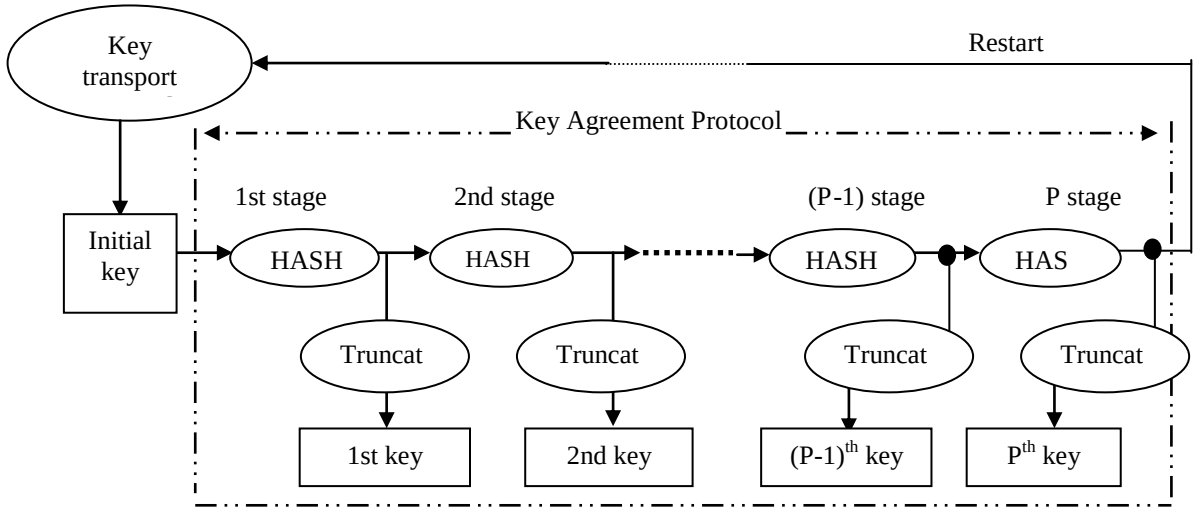

Figure (11) Key update process (Hashing Machine)

This process above is restarted after P stages and returns to the same initial key or another key, where P is given by key transport protocol. This restarted process required if the client finds its key unsecured or the session number unsynchronized. The period may by equal to infinite, i.e. the update process not restart, especially in the case of two server machines.

The proposed system allows using one of the two Hash functions MD5 or SHA-1. The connection partners agree on using one of these two Hash functions. This selected hashing method will not be changed all over the updating process.

5. Protocol Implementation

The proposed Authentication Protocol was implemented on an authentication server. The authentication server maintains an active directory (database file) of all users accounts registered to the server. Also the active directory contains all servers' accounts that the authentication server could establish authenticated session with them. This database file should be administrated continuously, and be encapsulated from unauthorized access.

5-1 Active Directory Database

An active directory database file contains several critical and confident fields as shown in **Table (4)**. This database file contains the following fields:

1. **Account Name or FQDN:** the account name is unique name, which refers to each user, it occupies 64 characters, while FQDN is unique string refers to each server in DNS, as standard DNS defined, it occupies 256-character space.
2. **Key:** It contains a last session key that the other end uses to succeed login to his service, this field reserves 40 character.
3. **Hash:** This field contains the name of hashing method that is used in authentication process as well as key update process with the other end; it contains one of two values "MD5", and "SHA".
4. **Seq. No:** This field contains the last session sequence number that the server establishes with other end; it occupies long integer space from record size.
4. **User Name or IP Address:** User name contains information about individual client registered to the server, and this field occupies 225 characters, while IP-Address contains the cached IP address of the Host specified in FQDN field, it occupies 15 characters from database record size, this information is optional.
5. **Initial key:** This field contains the initial password that is delivered to the server by the key transport mechanism. This field is used in key restart process.

Table (4) Portion of hosts database in servers

Account or FQDN	User Name	Seq. No.	Key	Initial Key	Hash
Abd.eng	Abd Alrahem	100001	45310FEBB710ADEFE429001AC5293710	Suheb_hh	MD5
Alcsoft	Anas Younis	0	Howareu	Howareu	SHA
Ali2003	Ali Husien	24	FA67431EBC6390765389253EAD248100	sql	MD5
Awsoof	Aws Younis	4000	6D43910AEC52491EB44811092374EC4DA8102921	Gourge	SHA
Vala_70	Valantina Ibra	330	7FEDCA5438001EC96DA657EAD2109876CFDE2986	English Woman	SHA
Mail.yahoo.com	196.164.107.20	1298376	EDA56DA71091EB4ADE23C55481109237B472CEC7	Bile gate	SHA
Smtp.hotmail.com	199.200.253.98	3542667	654FEAB8ADC2107843DE5462901FDE24ACDEEC3	Microsoft	SHA
Smtp1.uruke.net	196.231.87.66	231	F5319EC55201DD630EAD66627FEDCA100098E101	Iraq1	SHA
Smtp2.uruke.net	196.231.87.67	5368654	AE54BDF5D630EC554811092AD19003EA	Iraq2	MD5

The proposed system will retrieve the necessary data to achieve authentication process from the Active directory database. It is easily to implement the active directory database in the server with any database program like Microsoft Access Database and use of Active X data object that implemented in Visual Basic to deal with this database.

It is also easy to implement the client engine with Visual Basic, by use of WinSOCK and other objects that built by Visual Basic language.

Finally the system implemented and run in a hostile environment like Internet and the testing procedure prove the reliability, consistently, and robustness of the proposed authentication protocol against the authentication attacks.

6. Conclusion

The proposed system provides integrity and authenticity assurance by using CRAM and OTP algorithms together. This authentication protocol will be more secure than systems that depend only on traditional cryptography algorithms.

The system will provide authenticity and integrity assurance when the user wants to send a message to more than one recipient, while the traditional systems ensure between only the two communicating partners. The user dose not required obtaining a huge database of all users that he wants to communicate with them securely, he is only require to remember and keep his password with his Email provider.

The MAC generator is designed with some flexibility in terms of message length, and embedded standard hash function. This will increase the complexity of the attacking process. The security assumptions of the underlying standard hash function lead to well-understood cryptographic analysis of MAC strength mechanism. Also, there are no export restrictions from the United States or other countries for cryptographic hash functions, whereas conventional encryption algorithms, even when used for MACs, are restricted.

The initial key selection is very flexible, but it is very important to select key with size greater than the half of the digest message length. The keys greater than digest message length don't significantly increase the function strength.

Finally, structure of proposed authentication protocol is robust to many authentication attacks such as; replay attack, reflection attack, man in the middle attack, dictionary attack, brute force attack, and birthday attack.

7. References

1. Clear Water Bay, Kowloon, ***"Internet Security Handbook"***, Edition 2, Cyberapace Center, The Hang Kong University of Science and Technology, June 2001. **Site=[http://www.cyber.ust.hk]**

2. Madalina Baltatu, Antonio Lioy, ***"IP Security"***, Internet Paper, Italy, 2000. **Site [security.polito.it/doc/papers/ipsec.pdf]**

3. Symantec Publications, ***"Enterprise Solutions"***, Internet Paper, 2001. **Site[enterprisesecurity.Symantec.com/article.cfm?articleid = 1314 &EIDO]**

4. William Stallings, ***"NETWORK Security Essentials: Applications and Standards"***, Prentice Hall, New Jersey, 2000.

5. William Buchanan Macmillan, ***"Mastering Networks"***, First Edition, SYBOX, San-Francisco, April 1999.

6. Kevin Johnson, ***"Internet Email Protocols, Developer's, Guide"***, First Edition, Addison Wesley Longman, Hallow England, 2000.

7. Menzes, A., Van Oorsehot, P., and Vanstone, S., ***"Handbook of Applied Cryptography"***, CRC Press, Fifth Printing, August, 2001.
Site= [cacr.math.uwaterloo.ca/hac]

8. Rivest, R., ***"RFC1321: The MD5 Message-Digest Algorithm"***, MIT Laboratory for Computer Science and RSA Data Security Inc., April 1992.
Site = [www.fags.org/rfcs/rfc1321.html]

9. James, H. Burrows, ***"FIPS PUB 180-1: Secure Hash Standard"***, Federal Information Processing Standards Publication 180-1, U.S. Government Printing Office, April 1995.
Site = [http://csrc.nist.gov/publications/fips/fips180/fips-180-1.pdf]

10. William Mehuron, ***"The Keyed-Hash Message Authentication Code (HMAC)"***, Information Technology Laboratory, National Institute of Standards and Technology, Federal Information Processing Standards Publication FIPS #HMAC, 2001. **Site = [http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf]**

11. Andrew, S., Tanenbaum, ***"Computer Network"***, Third Edition, Prentice-Hall, Asimon & Suchuster Company, 1996.

12. Florian-Daniel Otel, ***"Securing you're a Network from Outsiders-in a Nutshell"***, Technical Report 00-25 Department of Computer Engineering Chalmers Technical University, Gothenbury, Sweden, December 2000.
Site [www.ce.chalmers.se/staff/otel/misc/lic-thesis/thesis2.PDF]

13. Haller, N., Bellcore, C., Metz, ***"RFC1938: A One-Time Password System"***, Kaman Sciences Corporation, May 1996. **Site = [www.fags.org/rfcs/rfc1938.html]**

14. Helsinki, J., ***"Modelling of Cryptographic Protocols: A Concurrency Perspective"***, Final Draft, Pekka Nikander, Final Internet Draft, 1997.
Site [www.tml.hut.fi/~pnr/publications/thesis97.pdf]