

Enhanced a TCP security protocol by using optional fields in TCP header

Abdul Hadi Mohammed Alaidi

Wasit University Engineering College

Abstract

The Transfer Control Protocol (TCP) itself does not guarantee the security of data that it transmits. However, under some circumstances, the security of TCP communication is required, e.g. the client-server communication in banking systems. Nowadays, the source socket layer (SSL) protocol is widely used; however, SSL is based on RSA (a public-key cryptographic algorithm) algorithm, which would require more computational resource. Therefore, an alternative faster solution utilizing symmetrical algorithm (AES) are proposed to achieve these goals: bidirectional authentication, encrypted data transmission and Integrity check. Moreover, use the optional fields in TCP header for carrying related information and introduced a new device SAM (Secure access module) which provides security-related functionalities: encryption, decryption, key-diversification etc.

Keywords: SAM, bidirectional authentication, symmetrical algorithm, TCP header, optional field

تحسين امان بروتوكول (TCP) باستخدام حقول اختياري في راس (TCP)

ان بروتوكول (TCP) لا يضمن تشفير البيانات التي تنقل بواسطته . ومع ذلك، في بعض الظروف، يجب تأمين الاتصالات المرسله باستخدام TCP ، على سبيل المثال خدمة العملاء في القطاع المصرفي تحتاج الى تشفير البيانات بين العميل والسيرفر. في الوقت الحاضر، يتم استخدام بروتوكول طبقة المنافذ الآمنة (SSL)

على نطاق واسع. ومع ذلك، يستند SSL على خوارزمية RSA (خوارزمية تشفير المفتاح العام) والتي تتطلب المزيد من الموارد .

ان التقنية المقترحة في هذا البحث هي باستخدام خوارزمية متناظرة (AES) لتحقيق مجموعة من الأهداف: المصادقة ثنائية الاتجاه، نقل البيانات المشفرة والتحقق من سلامتها. كذلك استخدام الحقول الاختيارية في رأس TCP لنقل المعلومات ذات الصلة. أخيراً اقترح استخدام جهاز جديد SAM (وحدة الوصول الآمنة) لتحقيق وظائف ذات صلة بأمان المعلومات: التشفير، فك التشفير، ومفتاح التوزيع الخ

الكلمات المفتاحية: وحدة الوصول الآمنة ، المصادقة ثنائية الاتجاه ، خوارزمية متناظرة، رأس TCP، حقل اختياري

1-Introduction

Information that spread with in the Internet is under threat of prevalent and damaging types of attack may lead to denial of services, sniffing, and message modification or even delete part or entire message. Thus, security is needed to defend a network from nasty bad guys.

The transport control protocol/Internet protocol is adopted by the Internet. First step to start TCP connection is end point authentication where both sender and receiver have to know the identification of the other part in the connection. The TCP three handshakes have been used to transfer the authentication information and to set up the connection as well as stated in [1].

However, there are many security issues and weaknesses in the implementations of these widely used protocols. These vulnerabilities allow intruders to attack a network system that uses TCP protocol. Many efforts have been done to enhance

the security in the TCP protocol.

In this research, I tried to overcome the drawback of the TCP security by proposing a new method to make the communication between two end points more secure.

2-Research motivation

Security is a hot topic and data security in Transmission Control Protocol/Internet Protocol is one of that. The main motivation of this work was to find out how to enhance the security in TCP and proposed a secure method for client-server model.

3-Existing Techniques of Security in TCP

There are some mechanisms the have been proposed in the current studies based on the security in the TCP. These mechanisms will be discussed in the next sub-sections with their limitations and drawbacks for each mechanism regarding the security in the TCP.

3-1 MD5 Hashing using unused bit in TCP header

TCP is an insecure protocol and it might introduce some security-related risks to use it. Consequently, there are algorithms to protect the communication between client and server. MD5 Hashed Passwords is one of the strongest methods used to encrypt password and it takes advantage of the unused bits in TCP Header. The MD5 algorithm is irreversible and it is an extension of the MD4 message-digest algorithm. An encrypted message it is impossible to recover the original message. The algorithm takes a random message length as input and produces a 128-bit as output called "fingerprint" or "message digest". Moreover, where a large file must be "compressed" in a secure manner, the MD5 algorithm is intended for digital signature applications, before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA. However, MD5 is a strong algorithm, it can

be hacked during transmission and this attack is called hash. This hash can be hacked effectively by using two efficient techniques[2]. These are:

- a) Dictionary Attack this attack refers to the general technique of trying to predict secret by running through a list of possibilities, often a list of words from a dictionary.
- b) Rainbow Table new methods used for cryptanalysis hashed passwords.

The good thing about hashing is that the password can't be returned to plain text, but recently attacker finds methods to recover password from hashed using Dictionary Attack or Rainbow Table in seconds for weak passwords and hours for strong ones as stated in [3].

In this paper, new cryptosystem methods proposed to ensure that even the attacker find the password he can't use it because he needs Card Key and SAM Number which found only in client machine. Moreover, make the TCP communication more secure by exchanging the symmetric cryptographic and MAC key during the handshake procedure at the beginning of a session, each different clients have their receptive client keys, and the communication key changes in different sessions. Using this method will ensure that the TCP is secure.

3-2 SSL protocol

TCP protocol has some shortcoming in respect of security concerns as mentioned in [4]. Cryptography can enhance TCP with security services. This enhancement by using Cryptography is well known as source socket layer (SSL).

Netscape designed the SSL originally. Nowadays most popular browsers and web server such as yahoo, Amazon, eBay support SSL. To get clear idea how SSL work, let's go through internet commerce scenario. The client address and his/her payment card number have to be submitted to the server. Over ordinary connection this information would be the target for an intruder[1].

SSL solve these issue by enhance the TCP data integrity and client authentication. SSL takes place over HTTP. However, because SSL secures TCP, it could be placed by any application that runs over TCP.

In this paper, similar strategy as SSL adopting and using bidirectional authentication. Symmetric algorithms are being used to guarantee the performance efficiency and using 3 times messaging to authenticate instead of 4 steps. So that both clients ensuring that data will be send to the correct terminals. Encrypting the data will protect the message from being intercepting. Maintaining the integrity of data ensures there is no interpolation of the original data.

4- Existing Algorithms of Cryptography

This section is discussed the current algorithms of cryptography that is used to secure the data between server and client.

4-1 Symmetric-key Algorithms (Private-Key Algorithms)

Symmetric-key algorithms are algorithms for cryptography that use the identical cryptographic keys for both encryptions of plaintext and decryption of ciphered text. For examples, the DES/3DES, AES algorithms are widely used symmetric-key algorithms[5].

4-1.1 Characteristics of Symmetric-Key Algorithms

Compared with asymmetric-key algorithms (public-key algorithms) such as RSA and ElGamal algorithms, the symmetrical way has less amount of computation and is much faster and of high efficiency.

4-1.2 Model of Symmetric-Key Algorithm

Figure 1 shows the general procedure of symmetric-key algorithm. In symmetric encryption-decryption model, the sender and receiver agree on a private key for encryption and decryption. The sender encrypts the plain text to cipher text using the encrypt-algorithm as well as the key. During the transmission between sender and receiver, the message is represented in cipher text. At the receiver end, after the receiver gets the message in ciphered form, it would use the key to decrypt the cipher text back into plain text. These encrypt/decrypt methods are irreversible, that means that if one gets the plain text and cipher text, it is not able to work out the transmission key.

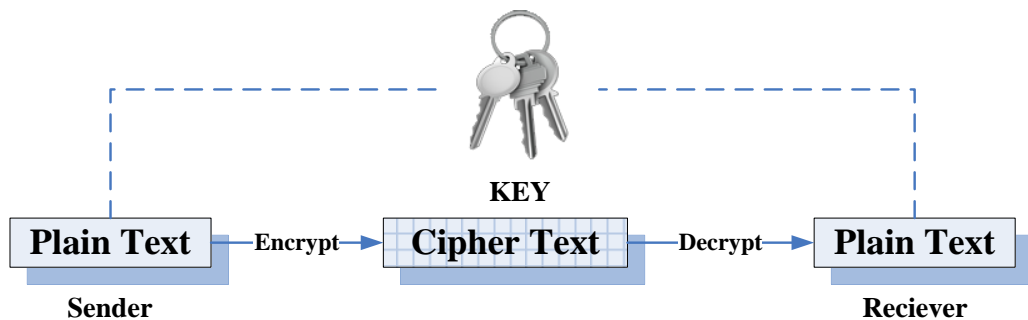


Figure (1) Model of Symmetric-key algorithm

4-2 AES/DES/3DES Algorithm

4-2.1 DES/3DES

DES (Data Encryption Standard) is developed in the early 1970s at IBM; it has brought great influence to modern cryptography in academic field. However, the traditional DES algorithm is cracked recently. Therefore, an enhanced version of DES algorithm was proposed which is called triple DES (3DES). 3DES applies the DES cipher algorithm 3 times to each data block in plain text. This kind of DES

algorithm is of high security and nowadays it is widely used in the financial and banking systems[6].

4-2.2 AES

AES stands for Advanced Encryption Standard; it is ratified by NIST in 2001. Since it is much faster than the 3DES algorithm, it becomes the most popular symmetric-key standard in these days[5].

4-2.3 Key-diversification in symmetric-key system

Key-diversification is a method often used to work with smart cards. For securing connections with a population of cards, a "key-generating key" is used with some data unique to a card to derive ("diversify") keys for use with that card. The data stored on the card is the card serial number or other number. Because of that the data is often public, it is very important to handle the key-generating key with a high degree of security so as not to interactions with the whole population of cards be placed in danger as mentioned in [1].

Generally speaking, key-diversification generates new sub key from the original master key with the diversifying data/factor (usually the card number/session number/randomly generated number). Also this procedure is irreversible, that means one could not work out the master key from the diversifying data/factor and the sub key. Figure 2 shows the algorithm of diversifying a master key by using 3DES algorithm.

1: Define Data: diversifying data, DataKey: new generated key, MyKey: original key 16 BYTES

7: Call MemoryCopy(TempKey, MyKey,16)

1: **for** (k = 0; k < iCounter; k++) **do**

```

2: for (i = 0; i < 8; i++) do
3:    $uszBuff[i] \leftarrow 0xFF - Data[k \ll 3 + 1]$ 
4: End for
5: Call DES3(Data+(k<<3),TempKey,DataKey)
6: Call DES3(uszBuff,TempKey,DataKey+8)
7: Call MemoryCopy(TempKey,DataKey,16)
8: End for

```

Figure (2) Key-diversification by 3DES Algorithm

4-3 MAC in Symmetric-Key System

MAC (*Message Authentication Code*) is an authentication technique using a secret key to generate a small block of data from the message and appending this block of data to the message[7]. The sender sends message as well as the MAC block to the receiver. As soon as the receiver gets the message and MAC, it re-generates MAC block of the received message using the key agreed by both sender and receiver. Then the receiver compares the received MAC block with the re-generated MAC block. If the two MAC blocks are equivalent, then the message is considered as integrated (that means the message was not modified by a third party during transmission). Figure 3 shows the MAC authentication.

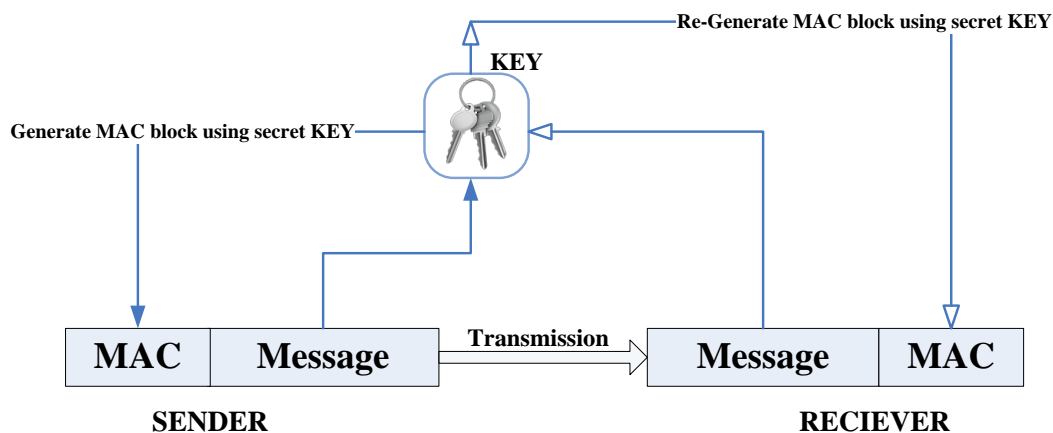


Figure (3) illustration of MAC authentication

Figure 4 illustrates the algorithm of MAC generating based 3DES algorithm.

```

void MyXor(unsigned char *puszStr1,
unsigned char *puszStr2, unsigned int
uiNum)
{
    unsigned int i;
    for(i=0; i< uiNum; i++)
        puszStr1[i] ^= puszStr2[i];
}
/*XOR the data according to ANSI99X
standard unsigned char *puszMsg [in]
message int iMsgLen [in] length of
message unsigned char *puszVal
[in,out] MAC data */
int Ansi99XorProc(unsigned char
*puszMsg,int iMsgLen, unsigned char
*puszVal)
{
    unsigned char uszTmp[8];
    int iFlg = 0;
    int iCount;
    int i;
    memset(uszTmp, 0,
sizeof(uszTmp));
    iFlg = iMsgLen % 8;
    if (iFlg)
        iCount = iMsgLen / 8 + 1;

    iCount = iMsgLen / 8;
    for (i=0; i<iCount-1; i++)
        MyXor(puszVal,
puszMsg+(i<<3), 8);
    if (iFlg == 0)
        MyXor(puszVal,
puszMsg+((iCount-1)<<3), 8);
    {
        memcpy(uszTmp,puszMsg+((iCount-1)
<<3), iFlg);
        MyXor(puszVal, uszTmp, 8);
    }
    return 0;
}
int Ansi99Xor3DesMac(unsigned char
*puszKey, unsigned char *puszMsg, int
iMsgLen, unsigned char *puszMac)
{
    unsigned char uszXorVal[8];
    memcpy(uszXorVal, puszMac, 8);
    Ansi99XorProc(puszMsg,
iMsgLen, uszXorVal);
    DES3(uszXorVal, puszKey,
puszMac);
    return 0;
}

```

Figure (4) Algorithm of 3DES MAC

4-4 SAM

SAM (*Security Access Module*) is a device with computational functions and storage ability. There are some maturely applied SAM devices, for examples: SAM Card (attached to client device through a special card reader), Financial Cryptography machine (usually it is used in the server end). There are some significant characteristics of SAM devices:

- The devices can provide cryptographic algorithms (*encrypt/decrypt/MAC/key-diversification*).
- The SAM devices have capability of generating random number and storing the latest random number in its register.
- The devices have capability of storing data (readable, e.g.: customer information/card number).
- The devices are able to store keys (*keys are unreadable but are usable*)
- SAM has to authenticate the device to which it is attached, failing in authentication might lead the SAM locked.

4-5 The Key Idea of the Proposed Technique

Since there are up to 40 octets of optional fields in TCP header, using these 40 octets can make the TCP based communication more secure. Unlike SSL protocol which is realized based on the service provided by TCP protocol, this design is making an extensional use of existing TCP protocol to make TCP protocol more secure. Regarding to the time performance as well as the limitation of the TCP itself, using the symmetrical-key methods is more appropriate.

4-5.1 Requirements of Securing Transmissions in TCP

Normally, the TCP protocol itself does not guarantee the security of data transmission. The data is transmitted in plain text without authentications. To make the TCP communication more secure in these aspects:

- Bidirectional authentication between the client and server.
- Exchanging the symmetric cryptographic and MAC key during the handshake procedure at the beginning of a session.
- Different clients have their receptive client keys.
- The communication key changes in different sessions.

The above points are achieved by exchanging information through the 40 BYTES (320Bits) space in the optional block of TCP header. Figure (5) shows the procedure that is used during the handshake in TCP connection.

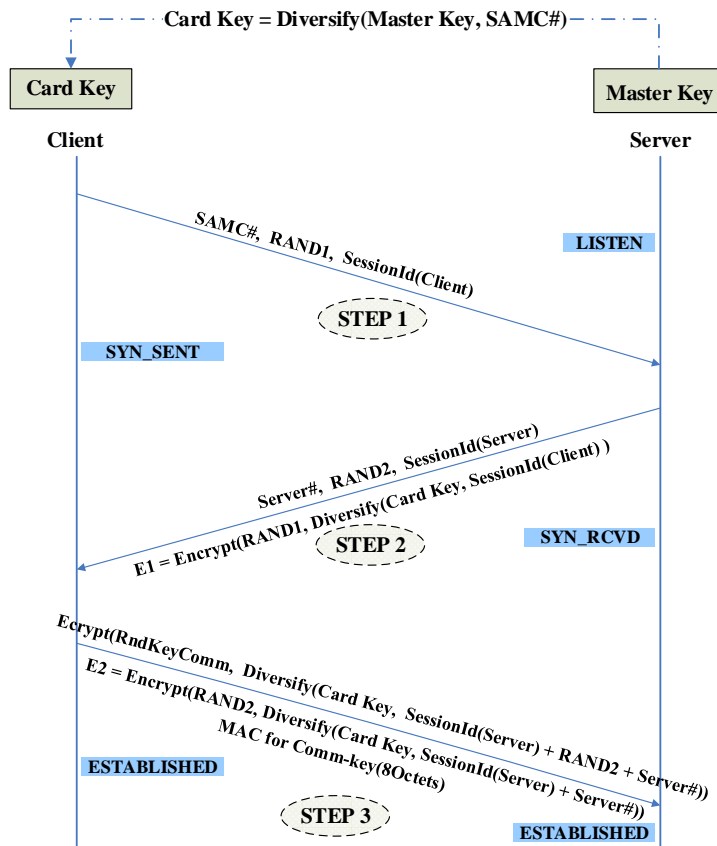


Figure (5) Procedure of TCP handshake

4-5.2 Scenarios of the Proposed Technique

In the proposed techniques, an extra device (SAM) are introducing, however, SAM device could not be adopted as widely as the common TCP protocol. The modified TCP protocol could be used for e-commerce/business systems, banking systems. The customers of these systems will get service from the server through TCP connection. The service providers are responsible to distribute the clients' SAM device to the customers (clients). Usually the clients' SAM device could be very small; they could be attached to the PC, or even could be embedded in the

handheld devices. The following features are added in the proposed technique in order to achieve more security in the transmissions in the proposed technique.

- **Client:** There's a SAM device (e.g. SAM Card) attached to the client. Each card has its unique ID (card number, SAMC#).
- **Server:** The server also connects to a security device, usually the cryptography machine. The security module of server also has a server ID (SAMS#).
- **Master Key:** Master key is the most significant key which is stored in the server's security device (unreadable, usable).
- **Card Key:** Card key is generated from the master key by key-diversification algorithm and distributed to each SAM card (NOT through the network). Since the master key is diversified by SAMC#, each SAM card has a different card key.
- **SAMC#:** SAM card ID.
- **Server#:** Server's ID (or the cryptography machine's ID).
- **RANDn:** The SAM devices (both the card and cryptography machine) have ability to generate random number. The generated random number will be stored in SAM device, and it would be covered by newly generated random number.
- **SessionId (Client):** This session ID is generated by the client, it is used to identify a session. It remains the same during one session. The format of session ID includes time stamp and a cycling self-increased sequential number.
- **SessionId (Server):** Similar to SessionId (Client), however, it is generated by the server.
- **New Key = Diversify (Root Key, Data):** The key-diversification algorithm. This function will generate a new key from the root key. The root key will be diversified into different keys through various diversification-data. This

algorithm is irreversible.

- **Cipher Text = Encrypt (Plain Text, Key):** Encrypt by using symmetric algorithm (we take AES algorithm in our design).
- **RndKeyComm:** Communication key for data transmission after the connection is established. This key is randomly generated by client in each session. This key would be used for data encryption/decryption and MAC checking in the process of data communication.
- **Step1:** The SAM Card generates a new random number (RAND1); the client puts the SAMC#, as well as the SessionId (Client) and random number into the optional space of TCP header and sends them to server. The server will verify the SAMC#, if the SAMC# is not valid, the server would reject the connection.
- **Step2:** The server will diversify the master key using the SAMC# and SessionId (Client) as diversifying data. The server will use this diversified key to encrypt RAND1 into E1. Also the server will generate Session Id (Server) and RAND2. The server then sends Server#, Session Id (Server), RAND2 and E1 to client.
- **Step3:** As soon as the client gets the information from server, it would first verify the Server#, if the server# is not valid; the client would terminate the session. Then the client will do the encryption of RAND1 and compare the result with E1. If the result is different from E1, the client would also terminate the session.
- **Sub Step1:** If the server is authenticated, the client would diversify the card key using the Server# and Session Id (Server) as key-diversification data. Client will use this newly generated key to encrypt RAND2 and get E2.
- **Sub Step2:** Client's SAM will randomly generate a communication key, and this key will be protected by a diversified key (card key diversified by RAND2, server# and SessionId (Server)). Then the encrypted

communication key and E1 will be sent to the server.

The server will do authentication on E2. If the client is authenticated, the server will decipher out the communication key and temporarily store this key in the server's SAM for this session, and using this key as data-encryption key and MAC key during this session.

5- TCP Message Format of Modified TCP

There are also 4 reserved bits in the reserved field of TCP header; the last bit of reserved field could be taking as an indicator. If this bit is set, that means the TCP message is using this security-improved sub protocol. Figure 6, 7, and 8 are the option field in the modified TCP header in addition the Table 1 shows the format of each component.

SAMC# (8 Octets)	RAND1 (8 Octets)	SessionId(Client) (8 Octets)	Options Could be extra client information (16 Octets)
----------------------------	----------------------------	--	--

Figure (6) Option filed for step1 of handshake

Server# (8 Octets)	RAND2 (8 Octets)	SessionId(Server) (8 Octets)	E1 (16 Octets)
------------------------------	----------------------------	--	--------------------------

Figure (7) Option filed for step2 of handshake

Encrypted Communication Key (16 Octets)	E2 (16 Octets)	MAC for ciphered communication-key (8 Octets)
---	--------------------------	---

Figure (8) Option filed for step3 of handshake

Field name	Format	Description
SAMC#	BIN (8Octets)	The SAM card's ID for each client
RAND1	BIN (8Octets)	Random number generated by client's SAM
SessionId(Client)	BCD (4Octets) + BIN (4Octets)	YYYYMMDD+ 4BYTES self-increased cycling
Server#	BIN (8Octets)	The Server's ID
RAND2	BIN (8Octets)	Random number generated by server's SAM
SessionId(Server)	BCD (4Octets) + BIN (4Octets)	YYYYMMDD+ 4BYTES self-increased cycling
E1	BIN (16Octets)	Cipher text for RAND1
Encrypted Communication Key	BIN (16Octets)	Communication key (cipher text)
E2	BIN (16Octets)	Cipher text for RAND2
MAC for ciphered comm-key	BIN(8Octets)	MAC(8Bytes) for the ciphered comm-key

Table (1) Format of each component

When the connection is successfully established between the client and server, they start data communication. All the data will be transmitted in cipher text (encrypted by the communication key); also the MAC data (8 or 16 octets) will be put into the option-field of TCP header in the data-exchanging stage.

6- Conclusion

In this paper, an enhanced TCP protocol proposed as security-enhanced for TCP protocol by utilizing the optional field in TCP header and adopting symmetrical cryptographic algorithms. Bidirectional authentication between server and clients, uniqueness of terminal keys, short life-cycle of authentication keys, randomness of communication keys are some goals achieved using this method. In the future, it will be interesting to simulate the proposed method and see the TCP performance.

7- Reference

1. Kurose, J. and K. Ross, *Computer Networking: A Top Down Approach*, 4e. Vol. 1. 2012.
2. Chawdhury, M.D.A. *Security enhancement of MD5 hashed passwords by using the unused bits of TCP header*. in *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on*. 2008. IEEE.
3. Guha, B. and B. Mukherjee, *Network security via reverse engineering of TCP code: vulnerability analysis and proposed solutions*. Network, IEEE, 1997. **11**(4): p. 40-48.
4. Raymond, J.-F. *Traffic analysis: Protocols, attacks, design issues, and open problems*. in *Designing Privacy Enhancing Technologies*. 2001. Springer.
5. Forouzan, B.A., *Cryptography & Network Security*. 2007: McGraw-Hill, Inc.
6. Alanazi, H., et al., *New comparative study between DES, 3DES and AES within nine factors*. arXiv preprint arXiv:1003.4085, 2010.
7. Bellare, M., J. Kilian, and P. Rogaway, *The security of the cipher block chaining message authentication code*. Journal of Computer and System Sciences, 2000. **61**(3): p. 362-399.