AUTHMAC_DH: A New Protocol for Authentication and Key Distribution

Heba K. Aslan

The Electronics Research Institute, El-Tahrir St., Dokki, Cairo, Egypt
hebaaslan@yahoo.com

Abstract. In the present paper, a new protocol for authentication and key distribution is proposed. The new protocol has the aim to achieve a comparable performance with the Kerberos protocol and overcome its drawbacks. For authentication of the exchanged messages during authentication and key distribution, the new protocol uses the Message Authentication Codes (MAC) to exchange the Diffie-Hellman components. On the other hand, the new protocol uses nonces to ensure the freshness of the exchanged messages. Subsequently, there is no need for clock synchronization which will simplify the system requirements. The new protocol is analyzed using queuing model, the performance analysis of the new protocol shows that the new protocol has a comparable performance with the Kerberos protocol for short messages and outperforms it for large messages.

1 Introduction

In recent years, many applications that require the exchange of sensitive information over public networks increase considerably. This introduces the need of two requirements: firstly, the need for providing authenticity of the communicating parties and to ensure the freshness of the exchanged messages. Secondly, the need to ensure the security of exchanged information. In literature, many protocols for authentication and key distribution were given. While some of them use symmetric key techniques, the others use public key techniques to distribute a symmetric key. Examples of the first method are: the Kerberos protocol [1, 2], and the Kryptoknight protocol [3]. On the other hand, examples of the second method are: the SPX protocol [4, 5], and the authenticated Diffie-Hellman protocol [6]. Among the above protocols, the Kerberos protocol achieves a widespread use especially for UNIX environment. This is due to the fact that it uses symmetric techniques; therefore, it leads to a better performance over the other protocols.

In the present paper, a new protocol for authentication and key distribution is proposed. The new protocol, which is named AUTHMAC_DH, is based on work done in [7, 8] and has the aim to achieve a comparable performance with the Kerberos protocol and overcome its drawbacks [9]. The paper is organized as follows: in Section 2, a description of the new protocol is detailed. In Section 3, expressions for the performance analysis of both the Kerberos protocol and the new protocol were derived. In Section 4, numerical results of the new protocol are discussed. Finally, the paper concludes in Section 5.

A. Lioy and D. Mazzocchi (Eds.): CMS 2003, LNCS 2828, pp. 14–26, 2003.

2 AUTHMAC_DH: A New Protocol for Authentication and Key Distribution

The Kerberos protocol is based on the exchange of symmetric key between the communicating parties using symmetric key encryption. Therefore, in case of compromise of messages exchanged during authentication and key distribution, all subsequent communication will be susceptible to disclosure. Besides the above feature, to ensure the freshness of the exchanged messages, timestamps are used. Consequently, the need for synchronizing all clocks of the system, which is a difficult problem, becomes a vital requirement. To overcome the abovementioned drawbacks, the new protocol uses the Message Authentication Codes (MAC) as in KryptoKnight protocol to exchange the Diffie-Hellman components as in the authenticated Diffie-Hellman protocol. Since, the compromise of messages exchanged during authentication and key distribution will lead to the disclosure of the Diffie-Hellman components. Therefore, the attacker cannot calculate the symmetric key used between the communicating parties (the difficulty of computing discrete logarithm for a large modulo number is a well-known problem in number theory). The use of MAC will fasten the proposed protocol. On the other hand, the new protocol uses nonces to ensure the freshness of the exchanged messages. Subsequently, there is no need for clock synchronization which will simplify the system requirements.

In order to achieve the goals of authentication and key distribution, a third-party called the Security Manager (SM) is incorporated into the system. The SM stores all the Diffie-Hellman components of all registered users and it shares a symmetric key with all users. Each station needs to store its Diffie-Hellman secret and the symmetric key shared between it and the SM. Fig. 1 shows the steps required to perform authentication and key distribution. The steps of the protocol are described in the following paragraphs.

Step 1: When client A wants to communicate with server B, it sends to the SM a message consisting of: A's identity 'A', B's identity 'B', and a nonce N_a.

Step 2: After receiving A's request, the SM calculates the MAC of the following messages:

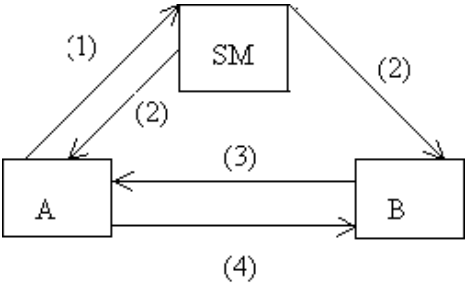
- The symmetric key shared between SM and A '$K_{SM,A}$', A, B, N_a, the Diffie-Hellman component of A 'a^xmodp', and the Diffie-Hellman component of B 'a^y-modp'.
- The symmetric key shared between SM and B '$K_{SM,B}$', A, B, N_a, a^xmodp, and a^y-modp. Then, it sends to both A and B a message consisting of: A, B, N_a, a^xmodp, a^ymodp, MAC($K_{SM,A}$, A, B, N_a, a^xmodp, a^ymodp), and MAC($K_{SM,b}$, A, B, N_a, a^xmodp, a^ymodp).

Step 3: Upon receiving the reply of the SM, both A and B authenticate the receiving message using the MAC appended in the reply. A ensures the freshness of the received message using N_a. Then, A and B calculate the symmetric key a^{xy}modp which will be used to encrypt subsequent communication between them. Next, B generates a nonce N_b and sends to A a message consisting of N_a and N_b both encrypted using a^{xy}modp. After receiving B's reply, A decrypts the message of B, and compares between the nonce included in the message and the nonce generated by it 'N_a'. If they are equal, this implies that B can calculate a^{xy}modp. Therefore, A authenticates B.

Step 4: A sends to B a message consisting of N_b encrypted using a^{xy}modp. Upon receiving the reply of A, B authenticates A by decrypting the reply and comparing between the nonce included in the message and the nonce generated by it 'N_b'. After completion of the abovementioned steps, both A and B authenticate each other, and share a symmetric key a^{xy}modp. The proposed protocol has the following advantages over the Kerberos protocol:

- It does not rely on timestamps which simplify the system requirements.
- Since, the Diffie-Hellman components are exchanged during authentication and key distribution and not the symmetric key itself as in the Kerberos protocol. Therefore, the compromise of messages exchanged during authentication and key distribution does not lead to the disclosure of the symmetric key used between A and B.

In the next section, expressions for the performance analysis of both the AUTHMAC_DH and the Kerberos protocols will be derived.



Step 1: A sends to the SM a request to communicate with B
Transmitted message of Step1: [A, B, N_a]
Step 2: The SM broadcasts its reply to both A and B
Transmitted message of Step2: [A, B, N_a, a^xmodp, a^ymodp,
MAC($K_{SM,A}$, A, B, N_a, a^xmodp, a^ymodp),
MAC($K_{SM,B}$, A, B, N_a, a^xmodp, a^ymodp)]
x : Diffie-Hellman secret of A
y : Diffie-Hellman secret of B
a, p : Diffie-Hellman parameters
Step 3: A authenticates B
Transmitted message of Step3: [N_a, N_b]a^{xy}modp
Step 4: B authenticates A
Transmitted message of Step4: [N_b]a^{xy}modp

Fig. 1. Steps required during authentication and key distribution for the AUTHMAC_DH protocol

3 Performance Analysis of the AUTHMAC_DH and the Kerberos Protocol

In order to derive the performance expressions, the following assumptions are made: number of clients = m and the number of servers = n. All clients are identical and

have the same statistics. The same is true for the servers. The client requests a service from the server at rate λ requests/sec. The rate of messages exchange between the client and the server is v messages/sec. Messages exchanged between clients and servers are of exponential distribution with mean length equals to L bits. Symmetric encryption is done using the Advanced Encryption Standard (AES) [10] with rate TAES bits/sec. Calculation of the Diffie-Hellman components is done using the Chinese remainder theorem [11] with rate TDH bits/sec. Message authentication codes are calculated using the UMAC algorithm [12] with rate TMAC bits/sec. The network capacity = C bits/sec. In the following subsections, expressions for the performance analysis of the AUTHMAC_DH and the Kerbros protocols will be derived.

3.1 Performance Analysis of the AUTHMAC_DH Protocol

Fig. 2 illustrates the time sequence diagram of the AUTHMAC_DH protocol. The steps of the protocol are given in Section 2. According to Fig. 2, the mean message time T_{mes} for transmitting a message from a client to a server is given by:

$$T_{mes}=\frac{1}{r}\{E[W_{c1}]+S_{c1}+W_{n1}+t_{n1}+E[W_{sm}]+S_{sm}+W_{n2}+t_{n2}+E[W_{s1}]+S_{s1}+W_{n3}+ t_{n3}+E[W_{c3}]+S_{c3}+W_{n4}+t_{n4}+E[W_{s2}] +S_{s2}\}+ E[W_{c4}]+S_{c4}+ W_{n5}+t_{n5}+ E[W_{s3}]+S_{s3} \quad (1)$$

where,

r is the number of transmitted messages between the client and the server using the key calculated in the key distribution phase, W_{c1} is the waiting time in the client queue (A) before processing the request for the communication with the server (B), S_{c1} is the time required for processing the request of A to communicate with B, W_{n1} is the waiting time in the network queue, t_{n1} is the time required for transmitting the request of A to the SM, W_{sm} is the waiting time in the Security Manager (SM) queue before processing the request of A, S_{sm} is the time required to calculate the UMAC of ($K_{SM,A}$, A, B, N_a, a^xmodp, a^ymodp), and the UMAC of ($K_{SM,B}$, A, B, N_a, a^xmodp, a^ymodp)], W_{n2} is the waiting time in the network queue, t_{n2} is the time required for transmitting the SM reply to both A and B, W_{s1} is the waiting time in the server's queue before processing the reply of the SM, S_{s1} is the time required to calculate the UMAC of ($K_{SM,B}$, A, B, N_a, a^xmodp, a^ymodp), to calculate a^{xy}modp, and to encrypt (N_a, N_b) using the AES cipher, W_{n3} is the waiting time in the network queue, t_{n3} is the time required for transmitting the server's reply to A, W_{c3} is the waiting time in the client queue before processing B's reply, S_{c3} is the time required to decrypt(N_a, N_b), and to encrypt (N_b) using the AES cipher, W_{n4} is the waiting time in the network queue, t_{n4} is the time required for transmitting A's reply to B, W_{s2} is the waiting time in the server's queue before processing the reply of A, S_{s2} is the time required to decrypt (N_b), W_{c4} is the waiting time in the client queue before encryption of the message transmitted to B, S_{c4} is the time required to encrypt a message of length L using the AES cipher, W_{n5} is the waiting time in the network queue, t_{n5} is the time required for transmitting the message from A to B, W_{s3} is the waiting time in the server's queue before processing the message of A, and S_{s3} is the time required to decrypt A's message.

In order to calculate the mean message time, the following assumptions are made: A's identity = B's identity = 8 bits, $N_a = N_b = 16$ bits, a^xmodp = a^ymodp = a^{xy}modp = 512 bits, $K_{SM,A} = K_{SM,B}$ = 128 bits, the encryption block of the AES = 128 bits, the output block of UMAC algorithm = 32 bits, and TAES = 20TDH = TMAC/5. According to the previous assumptions, the following parameters are calculated: $S_{c1} = 0$ (since it involves no encryption), S_{sm} = 476/TAES, S_{c2} = 10478/TAES, S_{s1} = 10510/TAES, S_{c3} = 48/TAES, S_{s2} = 16/TAES, and $S_{c4} = S_{s3}$ = L/TAES.For Simplicity the following assumptions are made: $W_{c1} = W_{c2} = W_{c3} = W_{c4} = E[W_c]$, $W_{s1} = W_{s2} = W_{s3} = E[W_s]$, and $W_{n1} = W_{n2} = W_{n3} = W_{n4} = W_{n5} = E[W_n]$

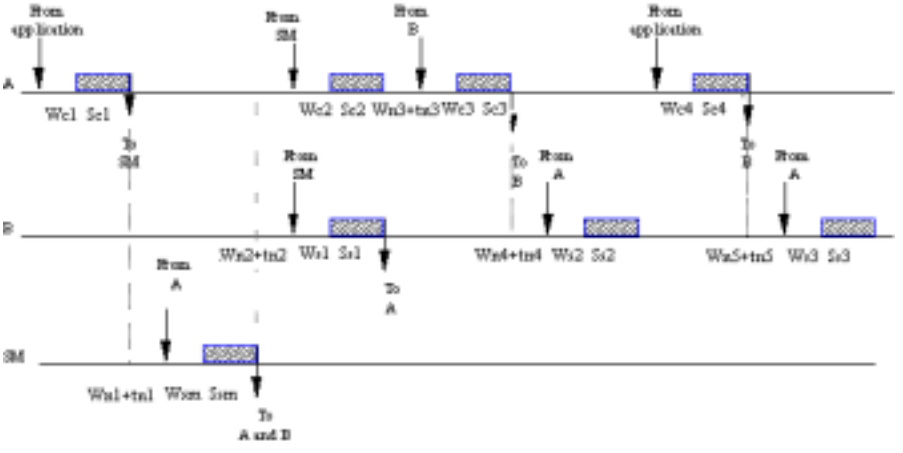


Fig. 2. The time sequence diagram for the AUTHMAC_DH protocol

For the client queue: The client is modeled as an M/G/1 queue. It has to be noted that the key calculated in the authentication and key distribution phase 'a^{xy}modp' has a length of 512 bits. Therefore, four keys each of 128 bits could be extracted from these 512 bits. Each key could be used r' times between the client and the server (i.e. r = 4r'). The client queue has the following parameters: the arrival rate λ_{client} = (number of arrivals per ticket/lifetime of the ticket)*number of servers = $(3+4r')n\lambda$ = $\frac{(3+4r')nv}{4r'}$, the mean service time $T_{client} = \frac{1}{4r'+3}(S_{c1}+S_{c2}+S_{c3}) + \frac{4r'}{4r'+3}S_{c4} = \frac{10497+4r'L}{(4r'+3)TAES}$, and the traffic intensity $\rho_{client} = \frac{nv(2624+r'L)}{r'TAES}$. For M/G/1 queues, the waiting time is given by [13, Eq. (2.65)]:

$$E[W] = \frac{\lambda E[\tau^2]}{2(1-\rho)} \tag{2}$$

where ρ is the traffic intensity, λ is the mean arrival rate, and $E[\tau^2]$ is the second moment of the service time and is equal to:

$$E[\tau^2] = \frac{109790788+8r'L^2}{(4r'+3)TAES^2}$$

Substituting into Eq. (2), the average waiting time $E[W_c]$ is equal to:

$$E[W_c] = \frac{nv(13723849 + r'L^2)}{r'TAES^2(1 - \frac{nv(2624 + r'L)}{r'TAES})} \quad (3)$$

For the server queue: The server queue is modeled as an M/G/1 queue with the following parameters:

$\lambda_{server} = \frac{mv(1 + 2r')}{2r'}$, $T_{server} = \frac{1}{4r'+2}(S_{S1} + S_{S2}) + \frac{4r'}{4r'+2}S_{S3} = \frac{5263 + 2r'L}{(2r'+1)TAES}$, $\rho_{server} = \frac{mv(2632 + r'L)}{r'TAES}$, and $E[\tau^2] = \frac{55230178 + 2r'L^2}{(2r'+1)TAES^2}$. Substituting into Eq. (2), the average waiting time $E[W_s]$ is equal to:

$$E[W_s] = \frac{mv(27615089 + r'L^2)}{2r'TAES^2(1 - \frac{mv(2632 + r'L)}{r'TAES})} \quad (4)$$

For the Security Manager queue: The SM queue is modeled as an M/D/1 queue with the following parameters: $\lambda_{sm} = \frac{mnv}{4r'}$, $T_{sm} = \frac{476}{TAES}$, and $\rho_{sm} = \frac{120mnv}{r'TAES}$. For M/D/1 queue the average waiting and service time is given by [13, Eq. (2.63)]:

$$E[w+s] = \frac{1 - \rho/2}{\mu(1 - \rho)} \quad (5)$$

where ρ is the traffic intensity to the queue, and is its service rate. Substituting into Eq. (5), the average waiting and service time $E[W_{SM} + S_{SM}]$ is equal to:

$$E[W_{sm} + S_{sm}] = \frac{\frac{476}{TAES}(1 - \frac{60mnv}{r'TAES})}{1 - \frac{120mnv}{r'TAES}} \quad (6)$$

For the network queue: The network queue is modeled as an M/G/1 queue with the following parameters: $\lambda_{network} = \frac{mnv(r'+1)}{r'}$, the mean packet length $= \frac{300 + r'L}{r'+1}$, $T_{network} = \frac{300 + r'L}{C(r'+1)}$, $\rho_{network} = \frac{mnv(300 + r'L)}{r'C}$, and $E[\tau^2] = \frac{314176 + 2r'L^2}{(r'+1)C^2}$. Substituting into Eq. (2), the average waiting time $E[W_n]$ is equal to:

$$E[W_n] = \frac{mnv(157088 + r'L^2)}{r'C^2(1 - \frac{mnv(300 + r'L)}{r'C})} \quad (7)$$

The mean message time T_{mes} given in Eq. (1) can be now calculated using Eqs. (3, 4, 6, and 7):

$$T_{mes} = \frac{2644 + 2r'L}{r'TAES} + \frac{300 + r'L}{r'C} + \frac{\frac{120}{r'TAES}(1 - \frac{60mnv}{r'TAES})}{1 - \frac{120mnv}{r'TAES}}$$
$$+ \frac{mv(2r'+1)(27615089 + r'L^2)}{4r'^2 TAES^2(1 - \frac{mv(2632 + r'L)}{r'TAES})} + \frac{nv(2r'+1)(13723849 + r'L^2)}{2r'^2 TAES^2(1 - \frac{nv(2624 + r'L)}{r'TAES})}$$
$$+ \frac{mnv(r'+1)(157088 + r'L^2)}{r'^2 C^2(1 - \frac{mnv(300 + r'L)}{r'C})} \quad (8)$$

3.2 Performance Analysis of the Kerberos Protocol

Fig. 3 shows the time sequence diagram of the Kerberos protocol. For more information about the Kerberos protocol, the reader could refer to [1]. The steps of the protocol are summarized below:

Step 1: A sends to the AS a request to communicate with B
Transmitted message of Step1: [A, B]
A : A's identity B : B's identity
Step 2: The AS sends its reply to A
Transmitted message of Step2:
$[T_{as}, L_1, K_{a,b}, B]K_{a,as}, [T_{as}, L_1, K_{a,b}, A]K_{b,as}]$
T_{as} : timestamp generated by AS
$K_{a,b}$: symmetric key between A and B
L_1 : lifetime of $K_{a,b}$
$K_{a,as}$: the symmetric key between A and AS
$K_{b,as}$: the symmetric key between B and AS
Step 3: A sends to B the AS's reply
Transmitted message of Step3: $[A, [T_{as}, L_1, K_{a,b}, A]K_{b,as}, [A, T_a]\ K_{a,b}$
T_a : timestamp generated by A
Step 4: A authenticates B
Transmitted message of Step4: $[A, T_a+1]\ K_{a,b}$

According to Fig. 3, the mean message time T_{mes} for transmitting a message from a client to a server is given by:

$$T_{mes} = \frac{1}{r'}\{E[W_{c1}]+S_{c1}+ W_{n1}+t_{n1}+ E[W_{as}]+S_{as}+ W_{n2}+t_{n2}+ E[W_{c2}]+S_{c2} +$$
$$W_{n3} + t_{n3} + E[W_{s1}] + S_{s1} + W_{n4} + t_{n4} + E[W_{c3}] + S_{c3}\}$$
$$+E[W_{c4}]+S_{c4}+ W_{n5}+t_{n5}+ E[W_{s2}]+S_{s2} \quad (9)$$

where,

r' is the number of transmitted messages between the client and the server using the key transmitted in the key distribution phase, W_{c1} is the waiting time in the client queue (A) before processing the request for the communication with the server (B),

S_{c1} is the time required for processing the request of A to communicate with B, W_{n1} is the waiting time in the network queue, t_{n1} is the time required for transmitting the request of A to the AS, W_{as} is the waiting time in the Authentication Server (AS) queue before processing the request of A, S_{as} is the time required to encrypt (T_{as}, L_1, $K_{a,b}$, A), and (T_{as}, L_1, $K_{a,b}$, B) using the AES cipher, W_{n2} is the waiting time in the network queue, t_{n2} is the time required for transmitting the AS reply to A, W_{c2} is the waiting time in A's queue before processing the reply from the AS, S_{c1} is the time required to decrypt (T_{as}, L_1, $K_{a,b}$, B), and to encrypt (A, T_a) using the AES cipher, W_{n3} is the waiting time in the network queue, t_{n3} is the time required for transmitting A's reply to B, W_{s1} is the waiting time in the server's queue before processing the reply of A, $S_{s1}$is the time required to decrypt (T_{as}, L_1, $K_{a,b}$, A), to decrypt (A, T_a), and to encrypt (B, T_a+1) using the AES cipher, W_{n4} is the waiting time in the network queue, t_{n4} is the time required for transmitting B's reply to A, W_{c3} is the waiting time in the client queue before processing B's reply, S_{c3} is the time required to decrypt(B, T_a+1) using the AES cipher, W_{c4} is the waiting time in the client queue before encryption of the message transmitted to B, S_{c4} is the time required to encrypt a message of length L using the AES cipher, W_{n5} is the waiting time in the network queue, t_{n5} is the time required for transmitting the message from A to B, W_{s2} is the waiting time in the server's queue before processing the message of A, and S_{s2} is the time required to decrypt A's message.

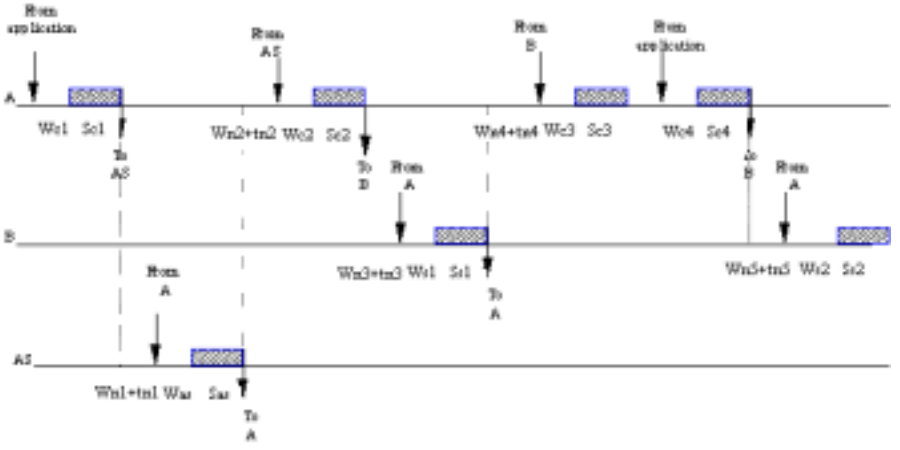


Fig. 3. The time sequence diagram for the Kerberos protocol

In order to calculate the mean message time, the following assumptions are made: A's identity = B's identity = 8 bits, T_{as} = 24 bits, L_1 = 8 bits, $K_{a,b}$ = 128 bits, and the encryption block of the AES = 128 bits. According to the previous assumptions, the following parameters are calculated: S_{c1} = 0 (since it involves no encryption), S_{as} = 336/TAES, S_{c2} = 200/TAES, S_{s1} = 232/TAES, S_{c3} = 32/TAES, and S_{c4} = S_{s2} = L/TAES. For Simplicity the following assumptions are made: W_{c1}= W_{c2} = W_{c3} = W_{c4} =E[W_c], W_{s1}= W_{s2} = E[W_s], and W_{n1}= W_{n2} = W_{n3} = W_{n4} = W_{n5} =E[W_n].

For the client queue: The client queue is modeled as an M/G/1 queue with the following parameters: $\lambda_{client} = \frac{(3+r')nv}{r'}$, $T_{client} = \frac{232+r'L}{(r'+3)TAES}$, $\rho_{client} = \frac{nv(232+r'L)}{r'TAES}$, and $E[\tau^2] = \frac{41024+2r'L^2}{(r'+3)TAES^2}$. Substituting into Eq. (2), the average waiting time $E[W_c]$ is equal to:

$$E[W_c] = \frac{nv(20512+r'L^2)}{r'TAES^2(1-\frac{nv(232+r'L)}{r'TAES})} \tag{10}$$

For the server queue: The server queue is modeled as an M/G/1 queue with the following parameters: $\lambda_{server} = \frac{mv(1+r')}{r'}$, $T_{server} = \frac{232+r'L}{(r'+1)TAES}$, $\rho_{server} = \frac{mv(232+r'L)}{r'TAES}$, and $E[\tau^2] = \frac{53824+2r'L^2}{(r'+1)TAES^2}$. Substituting into Eq. (2), the average waiting time $E[W_s]$ is equal to:

$$E[W_s] = \frac{mv(26912+r'L^2)}{r'TAES^2(1-\frac{mv(232+r'L)}{r'TAES})} \tag{11}$$

For the Authentication Server queue: The AS queue is modeled as an M/D/1 queue with the following parameters: $\lambda_{as} = \frac{mnv}{r'}$, $T_{as} = \frac{336}{TAES}$, and $\rho_{as} = \frac{336mnv}{r'TAES}$. Substituting into Eq. (5), the average waiting and service time $E[W_{SM}+ S_{SM}]$ is equal to:

$$E[W_{as}+ S_{as}] = \frac{\frac{336}{TAES}(1-\frac{168mnv}{r'TAES})}{1-\frac{336mnv}{r'TAES}} \tag{12}$$

For the network queue: The network queue is modeled as an M/G/1 queue with the following parameters: $\lambda_{network} = \frac{mnv(r'+4)}{r'}$, the mean packet length $= \frac{584+r'L}{r'+4}$, $T_{network} = \frac{584+r'L}{C(r'+4)}$, $\rho_{network} = \frac{mnv(584+r'L)}{r'C}$, and $E[\tau^2] = \frac{154176+2r'L^2}{(r'+4)C^2}$. Substituting into Eq. (2), the average waiting time $E[W_n]$ is equal to:

$$E[W_n] = \frac{mnv(77088+r'L^2)}{r'C^2(1-\frac{mnv(584+r'L)}{r'C})} \tag{13}$$

The mean message time T_{mes} given in Eq. (9) can be now calculated using Eqs. (10–13):

$$T_{mes} = \frac{464 + 2r'L}{r'TAES} + \frac{584 + r'L}{r'C} + \frac{\frac{336}{r'TAES}(1 - \frac{168mnv}{r'TAES})}{1 - \frac{336mnv}{r'TAES}}$$
$$+ \frac{nv(r'+3)(20512 + r'L^2)}{r'^2 TAES^2(1 - \frac{nv(232 + r'L)}{r'TAES})} + \frac{mv(r'+1)(26912 + r'L^2)}{r'^2 TAES^2(1 - \frac{mv(232 + r'L)}{r'TAES})}$$
$$+ \frac{mnv(r'+4)(77088 + r'L^2)}{r'^2 C^2(1 - \frac{mnv(584 + r'L)}{r'C})} \quad (14)$$

4 Numerical Results and Discussions

In the previous section, formulas for the mean message time for both the AUTHMAC_DH and the Kerberos protocols were derived. In order to obtain performance curves, the following assumptions are made: the number of clients m = 150, the number of servers n = 15, and the number of transmitted messages between the client and the server r' = 10. In the present paper, the performance is analyzed for the following case:

- Two message lengths are assumed: one for short messages where L = 1000 bits and the latter for large messages where L = 1 Mbits.
- Two encryption speeds are assumed: one for low encryption speed where TAES = 1 Mbps and the latter for high encryption speed where TAES = 1 Gbps.
- Two network rates are assumed: one for low network rate where C = 1 Gbps and the latter for high network rate where C = 10 Gbps.

In the following paragraphs, performance comparison between the proposed protocol and the Kerberos protocol will be given.

Figs 4–7 depict T_{mes} versus v for Kerberos, and the proposed protocol. Figs 4 and 5 are plotted for L = 1000 bits. Figs. 4.a and 4.b are plotted for TAES = 1 Gbps, while Fig. 5 is plotted for TAES = 1 Mbps. Figs 6 and 7 are plotted for L = 1 Mbits. Fig. 6 is plotted for TAES = 1 Gbps, while Fig. 7 is plotted for TAES = 1 Mbps. All (a) figures are plotted for C = 10 Gbps, while all (b) figures are plotted for C = 1 Gbps. From the figures, the following remarks can be deduced.:

1. For short messages(L = 1000 bits) and high encryption speed (TAES = 1 Gbps), two cases are examined: one for a transmission rate C = 10 Gbps and the latter for C = 1 Gbps. The following remarks could be deduced:
   - For C = 10 Gbps, the Kerberos protocol has a better performance than the new protocol. This is shown in Fig. 4.a. For short messages and high encryption speed, it could be concluded from all the queues of both the Kerberos protocol and the new protocol that the Kerberos protocol has a better performance over the new protocol.

- For C = 1 Gbps, the new protocol outperforms the Kerberos protocol as illustrated in Fig. 4.b. This is due to the fact that, as the network speed decreases, the performance becomes network dependent. Moreover, it could be concluded from the network queues (Eqs. 7 and 13) that the network time of the new protocol is less than that of the Kerberos protocol.

2. For short messages and low encryption speed (TAES = 1 Mbps), the Kerberos protocol has a better performance than the new protocol as illustrated in Fig. 5. This results from the fact that for low encryption rates, the performance becomes dependent on the server's queue. From the server queues (Eqs. 4 and 11), it is clear that for short messages, the server's waiting time in the Kerberos protocol is less than that of the new protocol.

3. For large messages (L = 1 Mbits), the new protocol outperforms the Kerberos protocol (Figs 6–7). This results since, for large messages, both the network and the server queues of the new protocol have a better performance over the Kerberos protocol for both low encryption speed and high encryption speed. The same conclusion could be applied for both high network speed and low network speed.

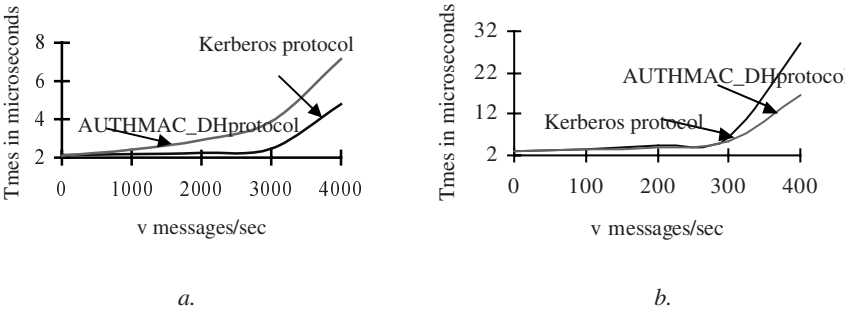


Fig. 4. T_{mes} versus v for L = 1000 bits and TAES = 1 Gbps:
a. C = 10 Gbps b. C = 1 Gbps

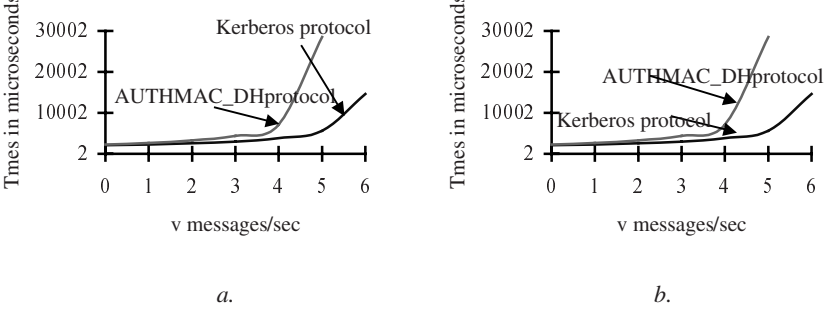


Fig. 5. T_{mes} versus v for L=1000 bits and TAES=1 Mbps:
a. C = 10 Gbps b. C = 1 Gbps

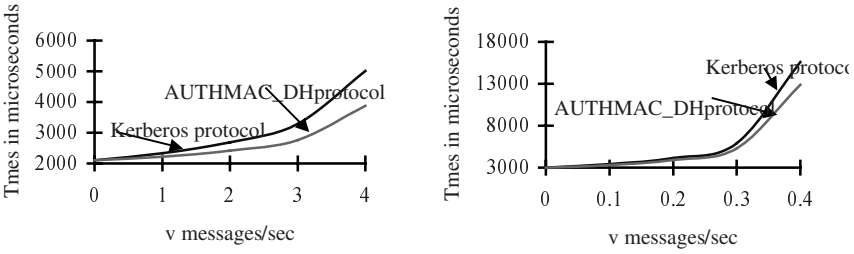


Fig. 6. T_{mes} versus v for L = 1 Mbits and TAES = 1 Gbps:
a. C = 10 Gbps b. C = 1 Gbps

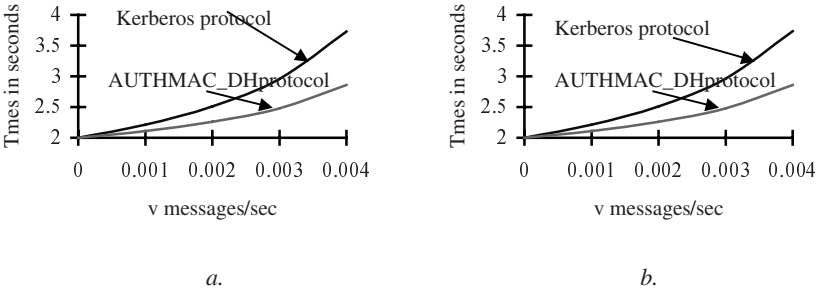


Fig. 7. T_{mes} versus v for L = 1 Mbits and TAES = 1 Mbps:
a. C = 10 Gbps b. C = 1 Gbps

5 Conclusions

In the present paper, a new protocol for authentication and key distribution called the AUTHMAC_DH is proposed. In order to provide authentication of the exchanged messages during authentication and key distribution, the AUTHMAC_DH protocol uses the Message Authentication Codes (MAC) to exchange the Diffie-Hellman components. Since, the compromise of messages exchanged during authentication and key distribution will lead to the disclosure of the Diffie-Hellman components and not the symmetric key itself. Therefore, the attacker cannot calculate the symmetric key used between the communicating parties. This feature is considered as an advantage over the Kerberos protocol in which the disclosure of messages exchanged during authentication and key distribution will lead to the disclosure of the symmetric key used between the communicating parties. It has to be noted that the use of MAC will fasten the proposed protocol. On the other hand, the AUTHMAC_DH protocol uses nonces to ensure the freshness of the exchanged messages. Subsequently, there is no need for clock synchronization which will simplify the system requirements which is considered as a second advantage over the Kerberos protocol. Performance expres-

sions for both the AUTHMAC_DH and the Kerberos protocols are derived using queuing model analysis. The performance is evaluated for several conditions: both short and large messages are examined, also the evaluation is considered for high and low encryption speeds, finally the analysis is undertaken for low and high network speeds. The performance analysis shows that the AUTHMAC_DH protocol has a comparable performance with the Kerberos protocol for short messages and outperforms it for large messages. In conclusion, besides that the AUTHMAC_DH protocol has a comparable performance with the Kerberos protocol, it overcomes its drawbacks.

References

1. Kohl, J. T., and Neuman, B. C.: The Kerberos Network Authentication Service (V5), RFC 1510, September 1993, (1993).
2. Kohl, J. T., Neuman, B. C., and Ts'o., T. Y.: The Evolution of the Kerberos Authentication Service, IEEE Computer Society Press Book (1994).
3. Molva, R., Tsudik, G., Herreweghen, E. V., and Zatti, S.: KryptoKnight Authentication and Key Distribution System, Proceedings of ESORICS 92, October 1992, (1992).
4. Tardo, J. J., and Alagappan, K.: SPX: Global Authentication Using Public Key Certificates, IEEE Privacy and Security Conference (1991).
5. Tardo, J. J., and Alagappan, K.: SPX Guide: A Prototype Public Key Authentication Service, Digital Equipment Corporation, May 1991, (1991).
6. Ford, W.: Computer Communications Security: Principles, Standard Protocols and Techniques, Prentice Hall (1994).
7. Aslan, H. K.: Logic-Based Analysis and Performance Evaluation of a New Protocol for Authentication and Key Distribution in Distributed Environments, Ph. D. Thesis, Electronics and Communications Dept., Faculty of Engineering, Cairo University, July 1998, (1998).
8. El-Hadidi, M. T., Hegazi, N. H., and Aslan, H. K.: Logic-Based Analysis of a New Hybrid Encryption Protocol for Authentication and Key Distribution, IFIP SEC98 conference (1998).
9. Bellovin, S. M., and Merrit, M.: Limitations of the Kerberos Authentication System, Computer Communication Review, October 1990, (1990).
10. Daemen J., and Rijmen, V.: AES Proposal: Rijndael, available at http://www.nist.gov/aes.
11. http://www.iaik.tugraz.at/aboutus/people/groszschaedl/papers/acsac2000.pdf
12. Black, J., Halevi, S., Krawczyk, H., Krovetz, T., and Rogaway, P.: UMAC: Fast and Secure Message Authentication, Advances in Cryptology, Crypto'99, Lecture Notes in Computer Science, Vol. 1666, Springer-Verlag (1999).
13. Schwartz, M.: Telecommunication Networks: Protocols, Modeling and Analysis, Addison–Wesley (1987).