A Network Authentication Protocol Based on Kerberos

Eman El-Emam †, Magdy Koutb ††, Hamdy Kelash †††, and Osama Farag Allah †††

† National Authority for Remote Sensing and Space Sciences, Cairo, Egypt
†† Department of Industrial Electronics & Control, Faculty of Electronic Engineering, Menouf, Egypt
††† Department of Computer Science & Engineering, Faculty of Electronic Engineering, Menouf, Egypt

Summary
We will focus on cryptographic protocols intended to achieve authentication over the networks. We aim to design a user authentication protocol that is not susceptible to password guessing attacks. We will present an authentication protocol based on the widely deployed Kerberos protocol with a little modification in the Kerberos database. The proposed protocol will be independent of the user password. The KDC will generate the realm principle secret key based on a saved profile in its database. The KDC will save a profile for every instance in the realm that it mange. This profile will be hashed and then, the output digest will be encrypted to generate the secret key. The lifetime of the secret key will be controlled using the system lifetime. By this way, we will overcome the weak passwords chosen by the network principal that are susceptible to password guessing attacks, the main drawback of the Kerberos protocol. In our implementation, we will use Triple-Des as an encryption algorithm, SHA-256 as a hashing algorithm, and Blum Blum Shub as a random number generator algorithm.

Key words:
Access control, authentication, authorization, computer network security, Kerberos, protocols.

1. Introduction

Over the centuries, an elaborate set of protocols and mechanisms have been created to deal with information security issues. The technical means to achieve information security in an electronic society are provided through cryptography. The cryptography is the study of mathematical techniques related to aspects of information security such as confidentiality, data integrity, access control, and authentication. Confidentiality is a service used to keep the contents of information from all but those authorized to have it. There are numerous approaches to provide confidentiality, e.g. the mathematical algorithms which render data unintelligible. Data integrity is a service that addresses the unauthorized alteration of data. To assure data integrity, one must have the ability to detect data manipulation by unauthorized parties. Data manipulation includes insertion, deletion, and substitution. Access control is the ability to limit the access to authorized users and applications. To achieve this, each entity trying to gain access must first be identified, or authenticated, so that access rights can be assigned to the individual. Authentication is a service related to identification. It is a fundamental building block for a secure networked environment. If a server knows the identity of a client, it can decide whether to provide the service, whether the user should be given special privileges, and so forth. In other words, authorization and accounting schemes can be built on top of authentication resulting in the required security to the computer network system.

Protocols play a major role in cryptography and are essential in meeting cryptographic goals. We need protocols to apply cryptographic algorithms and techniques among the communicating parties. Encryption schemes, hash functions, and random number generators are among the primitives which may be utilized to build a protocol. A cryptographic protocol is a distributed algorithm defined by a sequence of steps precisely specifying the actions required of two or more entities to achieve a specific security objective. The whole point of using cryptography in a protocol is to detect or prevent attacks.

The rest of the paper is organized as follows: we will begin with describing the motivation for the Kerberos approach and its environment in section 2. Then, we will outline a brief overview of the related work in section 3. After that, we will analyze Kerberos version 4, version 5, and the differences between them in section 4. While in section 5 we will discuss the Kerberos drawbacks. Then, we will examine the details used in our proposed authentication protocol, address its associated database, and present our testing environment in section 6. Finally, we will summarize the conclusions and the future work in section 7.

2. Motivation

Modern computer systems provide service to multiple users and require the ability to accurately identify the user

Manuscript received August 5, 2009
Manuscript revised August 20, 2009

making a request. The process of verifying the user's identity is called authentication. Today, more common in computer network architecture is a distributed architecture consisting of dedicated user workstations (clients) and distributed or centralized servers. In this environment, network connections to other machines are supported. Thus, we need to protect user information and resources housed at the server. The authentication service in these environments can be achieved by using Kerberos. It is one of the most widely used authentication protocols. It addresses an open distributed environment in which users at workstations wish to access services on servers distributed throughout the network. Kerberos employs one or more Kerberos servers (the KDC: Kerberos Distribution Center) to provide an authentication service. Kerberos requires the user to prove his or her identity for each service invoked. It also requires that servers prove their identity to clients. The overall scheme of Kerberos is that of a trusted third party that uses a protocol based on that proposed by Needham and Schroeder [1]. It is trusted in the sense that clients and servers trust Kerberos to mediate their mutual authentication. Assuming the Kerberos protocol is well designed, then the authentication service is secure if the Kerberos server itself is secure. Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users. Kerberos relies exclusively on symmetric encryption, making no use of public-key encryption. Most of the secure routing protocols rely on public key infrastructures (PKI) to authenticate communicating nodes. Although PKI is secure, it is based on asymmetric cryptography and hence requires excessive processing and communication resources [2]. This resource hungry feature makes PKI based systems more susceptible to Denial of Service attacks. In contrast, Kerberos [3] is a symmetric key based authentication mechanism.

3. Related Work

Massachusetts Institute of Technology (MIT) developed Kerberos to protect network services provided by Project Athena. Several versions of the protocol exist; versions 1–3 occurred only internally at MIT. Many members of Project Athena contributed to the design and implementation of Kerberos [4]. In [5] there is a dialogue that was written in 1988 to help its readers understand the fundamental reasons for why the Kerberos V4 protocol was the way it was. It was amazing how much this dialogue was still applicable for the Kerberos V5 protocol. Although many things were changed, the basic core ideas of the protocol have remained the same. Steve Miller and Clifford Neuman are the primary designers of Kerberos version 4 with contributions from Jerome Saltzer and Jeffrey Schiller [6]. They published that version in the late 1980s, although they had targeted it primarily for Project Athena. Version 5, designed by John Kohl and Clifford Neuman, appeared as RFC 1510 in 1993 [3] (made obsolete by RFC 4120 in 2005 [7]), with the intention of overcoming the limitations and security problems of version 4.

Security of Kerberos has been analyzed in many works, e.g. [8], [9], [10], [11], [12], [13] and [14]. Most commonly analyses identify certain limitations of Kerberos and sometimes propose fixes. This leads to the evolution of the protocol when a new version patches the known vulnerabilities of the previous versions. The current version Kerberos V5 is already being revised and extended [7], [15], and [16]. F. Butler, I. Cervesato, A. Jaggard, and A. Scedrov have analyzed portions of the current version of Kerberos and have formally verified that the design of Kerberos' current version meets the desired goals for the most parts [17]. A. Boldyreva and V. Kumar at 2007 take a close look at Kerberos' encryption and confirm that most of the options in the current version provably provide privacy and authenticity [18].

Kerberos is also used in wireless applications. M. Erdem proposed a high speed 2G wireless authentication systems based on kerberos [19]. He used DES, 3DES and AES as secret-key crypto algorithms. He also used SHA-1 message digest algorithm to hash the message blocks. Besides, A. Pirzada and Chris McDonald discuss how kerberos is used for authentication in mobile ad-hoc networks [20].

Kerberos is also introduced to be used in IPv6 networks. S. Sakane, N. Okabey, K. Kamadaz, and H. Esakix describe a method to establish secure communication using Kerberos in IPv6 networks [21]. They propose a mechanism to achieve access control using Kerberos and to deal with address resolution using Kerberos with modification.

Nitin et. al present an image based authentication system using the Kerberos protocol at 2008 [22]. That paper is a comprehensive study on the subject of using images as a password and the implementation of Jaypee University of Information Technology (JUIT) Image Based Authentication (IBA) system called as JUIT-IBA using Kerberos protocol.

In 2007, MIT formed the Kerberos Consortium along with some of the major vendors and users of Kerberos such as Sun Microsystems, Apple, Google, Microsoft, etc., to foster continued development. The MIT Kerberos Consortium was created to establish Kerberos as the universal authentication platform for the world's computer networks.

Kerberos has grown to become the most widely deployed system for authentication and authorization in modern computer networks. Kerberos is currently shipped with all major computer operating systems and is uniquely positioned to become a universal solution to the distributed

authentication and authorization problem of communicating parties [23].

4. Kerberos Messages Exchange

A simplified overview of the Kerberos actions is shown in Fig. 1. Exchange between the client and the Kerberos AS (Authentication Server) in messages 1 and 2 are used only when the user first logs in to the system. Exchange between the client and the Kerberos TGS (Ticket Granting Server) in messages 3 and 4 are used whenever a user authenticates to a new server. Message 5 is used each time the user authenticates itself to a server. And finally, message 6 is the mutual-authentication response by the server. The ticket plus the secret session key are the user credentials to be authenticated to a specific server.

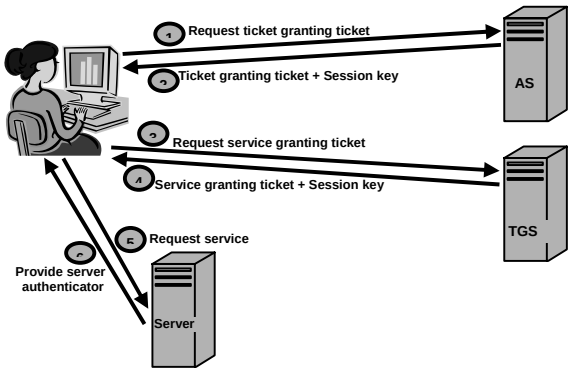

Fig. 1 Overview of the Kerberos actions

4.1 Kerberos 4 Authentication Dialogue

Kerberos Version 4 messages exchange is shown in Fig. 2. Fig. 2 (a) shows the technique for distributing the session key. The client sends a message to the AS requesting access to the TGS. The AS responds with a message, encrypted with a key derived from the user's password (K_C) that contains the TGS ticket ([24] describes the password to key transformation technique that is presented by the standard specification). The encrypted message also contains a copy of the session key, $K_{c,tgs}$, where the subscripts indicate that this is a session key for C and TGS. Because this session key is inside the message encrypted with K_C, only the client can read it. The same session key is included in the ticket, which can be read only by the TGS since it is encrypted by the TGS key K_{tgs}. Thus, the session key has been securely delivered to both the C and the TGS. Here, we will focus on some messages' elements (the details can be found in [24]). The keys $K_{c,tgs}$ and $K_{c,v}$ are the session keys; where the subscripts indicate the communicating parties. $Lifetime_2$ and $lifetime_4$ are the lifetime of the TGS ticket and the server ticket respectively. Finally, at the conclusion of this process, the client and server share a secret session key $K_{c,v}$.

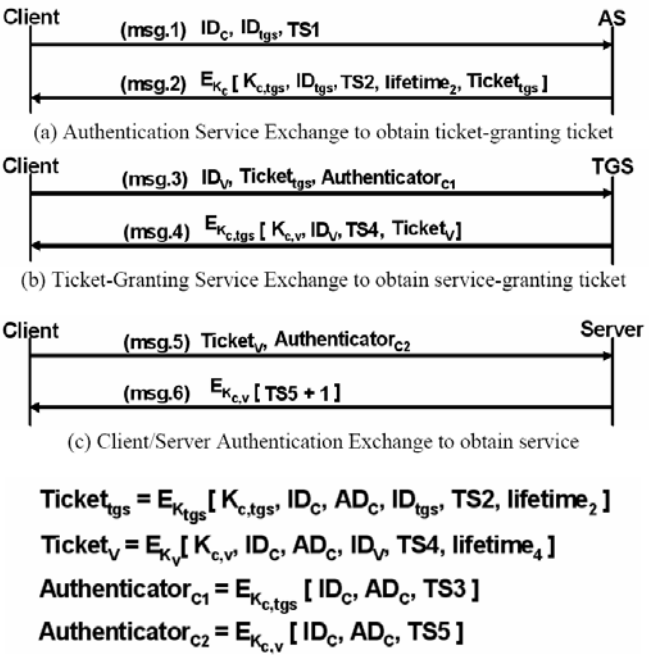

Fig. 2 Kerberos 4 messages exchange

4.2 Kerberos 5 Authentication Dialogue

Kerberos 5 messages exchange is shown in Fig. 3. This is best explained by comparison with version 4 (Fig. 2). In message (1), the following new elements are added:

- Realm: Indicates the realm of the client. Where the realm represents the nodes that are managed by a single KDC; i.e. share the same Kerberos database.
- Options: Used to request that certain flags be set in the returned ticket. These flags are an added feature in Kerberos 5.
- Times: Used by the client to request the following time settings in the ticket:
  - from: the desired start time for the requested ticket.
  - till: the requested expiration time for the requested ticket.
  - rtill: this field is only present in tickets that have the RENEWABLE flag set in the flags field. It indicates the maximum end-time that may be included in a renewal.

- Nonce: it is a random value to be repeated in message (2) to assure that the response is fresh and has not been replayed by an opponent.

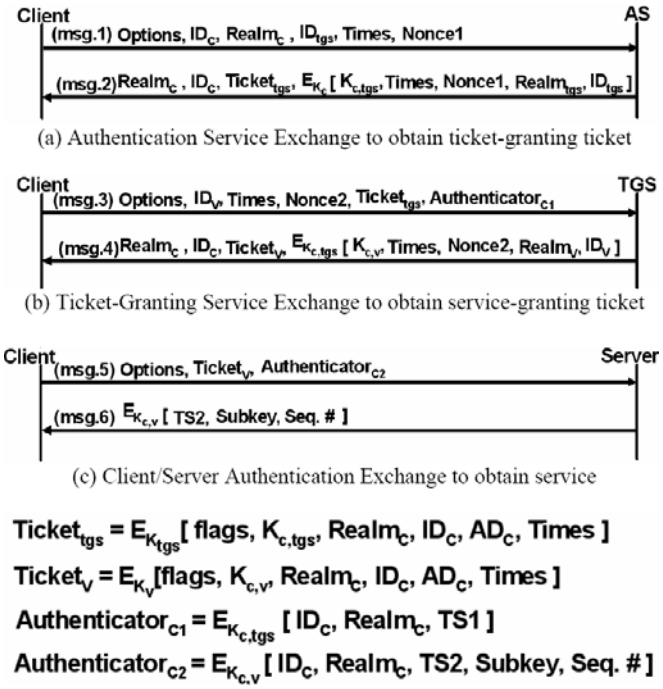

Fig. 3 Kerberos 5 messages exchange

Let us now compare the ticket-granting service exchange for versions 4 and 5. We see that message (3) in Fig. 3 includes requested times and options for the ticket and a nonce, all with functions similar to those of message (1).

Finally, for the client/server authentication exchange, several new features appear in version 5. In message (5), the client may request as an option that mutual authentication is required. The authenticator includes several new fields as follows:

- Subkey: The client's choice for an encryption key to be used to protect this specific application session. If this field is omitted, the session key from the ticket ($K_{c,v}$) is used. If the client selects a sub-session key, care must be taken to ensure the randomness of the selected key.
- Sequence number: An optional field that specifies the starting sequence number to be used by the server for messages sent to the client during this session (to detect replays).

After that, the server responds with message (6). This message includes the timestamp from the authenticator. The subkey field, if present, overrides the subkey field of message (5). The optional sequence number field specifies the starting sequence number to be used by the client.

4.3 Differences between Versions 4 and 5

Version 5 is intended to address the limitations of version 4. Let us briefly discuss the differences between the two versions:

1. Encryption system dependence: Version 4 requires the use of DES. In version 5, ciphertext is tagged with an encryption type identifier so that any encryption technique may be used.
2. Internet protocol dependence: Version 4 requires the use of Internet Protocol (IPv4) addresses. In version 5, network address is tagged with type and length. This allows any network address type to be used.
3. Ticket lifetime: Lifetime values in version 4 are encoded in an 8-bit quantity in units of five minutes. Thus, the maximum lifetime that can be expressed is 256 x 5 = 1280 minutes. In version 5, tickets include an explicit start and end times, allowing tickets with arbitrary lifetimes.
4. Authentication forwarding: Version 4 does not allow credentials issued to one client to be forwarded and used by some other clients. For example, a client issues a request to a print server that then accesses the client's file from a file server, using the client's credentials for access. Version 5 provides this capability.
5. Double encryption: Note in Fig. 2 that tickets provided to clients in messages (2) and (4) are encrypted twice, once with the secret key of the target server and then again with a secret key known to the client. The second encryption is not necessary and is computationally wasteful. It is avoided in version 5.
6. PCBC encryption: Encryption in version 4 makes use of a nonstandard mode of DES known as propagating cipher block chaining (PCBC) ([24] describes this mode of operation). Security problems have been demonstrated in that mode [11]. Version 5 makes use of the standard CBC mode for encryption.
7. Session keys: Each ticket includes a session key that is used by the client to encrypt the authenticator sent to the service associated with that ticket. In addition, the session key may subsequently be used by the client and the server to protect messages passed during that session. However, because the same ticket may be used repeatedly to gain service from a particular server, there is the risk that an opponent will replay messages from an old session to the client or the server. In version 5, it is possible for a client and server to negotiate a sub-session key, which is to be used only for that one connection. A new access by the client

would result in the use of a new sub-session key.

8. Password attacks: Both versions are vulnerable to a password guessing attack. The message from the AS to the client includes material encrypted with a key based on the client's password. An opponent can capture this message and attempt to decrypt it by trying various passwords. If the result of a test decryption is of the proper form, then the opponent has discovered the client's password and may subsequently use it to gain authentication credentials from Kerberos. Remember that when a user requests the ticket-granting ticket, the answer is returned encrypted with K_C, a key derived by a publicly-known algorithm from the user's password.

5. Kerberos Drawbacks

The protocol weaknesses can be summarized as follows:

1. Kerberos requires continuous availability of the KDC. When the Kerberos server is down, the system will be vulnerable to the single point of failure problem. This can be mitigated by using multiple Kerberos servers.
2. The system clocks of the hosts that are involved in the protocol should be synchronized. The tickets have a time availability period and if the host clock is not synchronized with the Kerberos server clock, the authentication will fail. In practice, Network Time Protocol daemons are usually used to keep the host clocks synchronized.
3. "Password guessing" attacks are not solved by Kerberos. If a user chooses a poor password, it is possible for an attacker to successfully mount an offline dictionary attack by repeatedly attempting to decrypt messages obtained which are encrypted under a key derived from the user's password.
4. There are no standards for the administration of the Kerberos protocol. This will differ between server implementations.

6. Contribution

It is obvious that Kerberos is vulnerable to password guessing attacks. We present an authentication protocol based on Kerberos with a little modification in the Kerberos database. It will be independent of the user password. Instead, the KDC will save a profile for every principal in the realm that it manages. The contents of the profile may be audio, video, image, or text data. The KDC database may have profiles of mixed data contents (some profiles may be audio, others may be images, and so on). The realm principal may be a client or a server instance that participates in the network communication. Every principle (user or server) has to register with the Kerberos database. The principal will register with the Kerberos server by the principal ID. Then, the KDC will map this ID to the principal profile. The Kerberos server will generate the principal secret key by applying a hashing algorithm to the principal profile. The input to the hashing algorithm will be the principal profile and the output will be encrypted to generate the principal secret key. The block diagram of Fig. 4 summarizes our proposed scheme to generate the principle secret key. It is also suggested to control the lifetime of that secret key. We introduce a simple idea for that. Since the system clocks of the hosts that are involved in the protocol should be synchronized (this can be maintained manually or assured by using Network Time Protocol daemons), we will append the current system timestamp to the principal profile every certain predefined period (this period is a design parameter; i.e. a site constant). Consequently, the input to the hashing algorithm will change, and thus the secret key will change too.

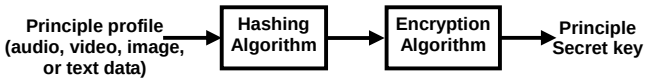

Fig. 4 Secret key generation block diagram

The machine which houses this database is called the master machine. It is extremely important that the master KDC will be installed on a carefully protected and physically secure machine. If possible, the machine should be dedicated to running the authentication server and the number of users with access should be limited. Also, there may be one more read-only copy of the Kerberos database on another machine called the slave. However, all changes to the database must be made on the master computer system. Changing or accessing the contents of a Kerberos database requires the Kerberos master password.

At the principle side (a client or a server), the secret key may be obtained by one of two ways depending on the network administrator choice. The first option will be chosen if the administrator decided to keep the type of the profile contents secret. Then the principles secret keys will be distributed using another secure method. This can be achieved using hardware equipments or by using a secure delivery system. The second option will be chosen if the administrator decided to announce the type of the profile contents. In that case, every principle may keep a copy of his or her profile and prompt to enter the path of that profile during the run of the Kerberos protocol.

6.1 Proposed Authentication Protocol

Our proposed protocol message dialogue is presented in Fig. 5. The elements of each message in the proposed protocol are summarized in Table 1. We introduce a comparison between Kerberos 4, Kerberos 5 and our proposed protocol in Table 2.

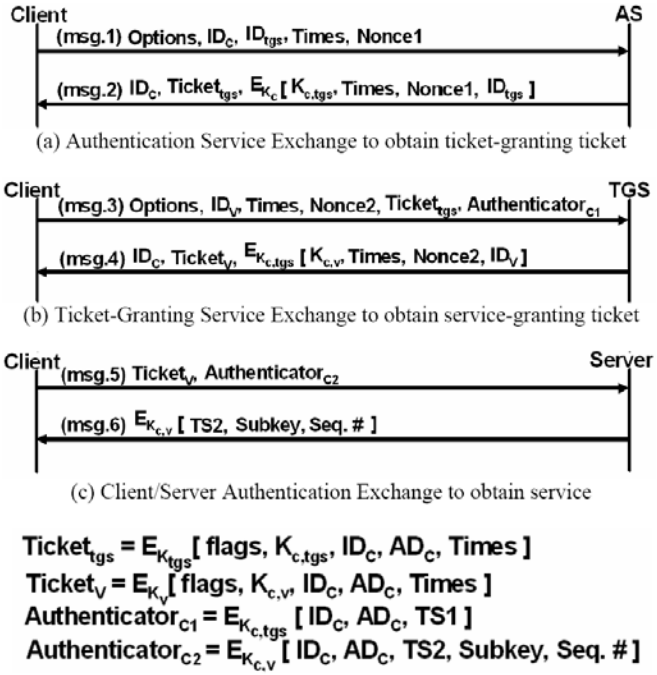

Fig. 5 Proposed authentication protocol message exchanges

Table 1. Summary for the elements of the proposed Protocol

Message (1)	Client requests ticket-granting ticket
Options	requests that certain flags be set in the returned TGS ticket
ID_C	Tells AS identity of user from this client
ID_{tgs}	Tells AS that user requests access to TGS
Times	Requests certain time settings in the returned TGS ticket (from, till, renew_till)
Nonce1	Random value to be repeated in message (2) to avoid replay attack
Message (2)	AS returns ticket-granting ticket
ID_C	The identity of user
K_C	Encryption by a key based on user's profile
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key
Times	The times settings of the returned TGS ticket
Nonce1	Repeat for the random value of message 1
ID_{tgs}	Confirms that this ticket is for the TGS
$Ticket_{tgs}$	Reusable ticket to be used by client to access TGS
K_{tgs}	Ticket is encrypted with key known only to AS and TGS, to prevent tampering
Flags	The flags of the returned TGS ticket
$K_{c,tgs}$	Copy of session key accessible to TGS used to decrypt authenticator, thereby authenticating ticket
ID_C	Indicates the rightful owner of this ticket
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket
Times	The times settings of the TGS ticket

(a) Authentication Service Exchange

Message (3)	Client requests service-granting ticket
Options	requests that certain flags be set in the returned server ticket
ID_V	Tells TGS that user requests access to server V
Times	Requests certain time settings in the returned server ticket (from, till, renew_till)
Nonce2	Random value to be repeated in message (4) to avoid replay attack
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS
$Authenticator_{C1}$	Generated by client to validate ticket. It assures TGS that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay
$K_{c,tgs}$	Authenticator is encrypted with key known only to client and TGS, to prevent tampering
ID_C	Must match ID in the TGS ticket to authenticate ticket
AD_C	Must match address in the TGS ticket to authenticate ticket
TS_1	Informs TGS of time this authenticator was generated
Message (4)	TGS returns service-granting ticket
ID_C	The identity of user
$K_{c,tgs}$	Key shared only by C and TGS
$K_{C,V}$	Copy of session key accessible to client created by TGS to permit secure exchange between client and server without requiring them to share a permanent key
Times	The times settings of the returned server

	ticket
Nonce2	Repeat for the random value of message 3
ID_V	Confirms that this ticket is for server V
$Ticket_V$	Reusable so that client does not need to request a new ticket from TGS for each access to the same server
K_V	Ticket is encrypted with key known only to TGS and server, to prevent tampering
Flags	The flags of the returned server ticket
$K_{C,V}$	Copy of session key accessible to client; used to decrypt authenticator, thereby authenticating ticket
ID_C	Indicates the rightful owner of this ticket
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket
Times	The times settings of the server ticket

(b) Ticket-Granting Service Exchange

Message (5)	Client requests service
$Ticket_V$	Assures server that this user has been authenticated by AS
$Authenticator_{C2}$	Generated by client to validate ticket. It assures server that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay
$K_{C,V}$	Authenticator is encrypted with key known only to client and server, to prevent tampering
ID_C	Must match ID in the server ticket to authenticate ticket
AD_C	Must match address in the server ticket to authenticate ticket
TS2	Informs server of time this authenticator was generated
Subkey	The client's choice for an encryption key to be used to protect this specific application session. If this field is omitted, the session key from the ticket $K_{c,v}$ is used
Seq. #	An optional field that specifies the starting sequence number to be used by the server for messages sent to the client during this session to detect replays
Message (6)	Optional authentication of server to client
$K_{C,V}$	Assures C that this message is from V
TS_2	Assures C that this is not a replay of an old reply
Subkey	The server's choice for an encryption key to be used to protect this specific application session. If this subkey present, it overrides the subkey field of message (5)
Seq. #	An optional field that specifies the starting sequence number to be used by the client for messages sent to the server during this session to detect replays

(c) Client/Server Authentication Exchange

Table 2. Comparison between Kerberos 4, Kerberos 5 and our proposed protocol

Comparison Item	Kerberos 4	Kerberos 5	Proposed Protocol
Times	No times	From, till, renew_till	From, till, renew_till
Encryption technique	DES	Encryption key is tagged with type & length	Triple-DES
DES mode of operation	PCBC (not standard)	The standard CBC mode	The standard CBC mode
Double encryption in message 2 & 4	Found	Not found	Not found
Session key	1/lifetime	Client & server may negotiate for subsession key (1/connection)	Client & server may negotiate for subsession key (1/connection)
Password guessing attack	Vulnerable	Vulnerable	Keys are independent of password
Network address	IPv4	Any (network address is tagged with type)	IPv4
Ticket lifetime	1280 minutes	Arbitrary (determined by start & end times)	Arbitrary (determined by start & end times)

6.2 Security properties of the proposed protocol

The security properties of the proposed protocol can be stated as follows:

- The realm principles long-term secret keys are independent of the password, thus the proposed protocol will be susceptible to the password guessing attack.
- Session key secrecy: For any client and any server, if the TGS generates a symmetric session key $K_{C,V}$ for a certain client and certain server, then the intruder does not learn that session key.
- Authentication of AS to client: If a client receives a valid AS response message (msg.2 in Fig. 5) and since the long term key of the client is secret, then this message was indeed generated by the KDC for this particular client and an adversary cannot learn the symmetric session key $K_{c,tgs}$ contained in this message.
- TGS authentication of its ticket (the TGT $Ticket_{tgs}$: the Ticket Granting Ticket): If a TGS receives a TGT and an authenticator $Authenticator_{C1}$ that contains a client identity ID_C and the authenticator is encrypted by the symmetric session key $K_{c,tgs}$ where the key $K_{c,tgs}$ and the client identity ID_C are contained in the TGT, then the TGT was generated by the KDC and the authenticator was created by that particular client whose identity is ID_C.
- Server authentication of the server ticket ($Ticket_V$): If a server receives a server ticket and an authenticator $Authenticator_{C2}$ that contains a client identity ID_C and the authenticator is encrypted by the symmetric session key $K_{c,v}$ where the key $K_{c,v}$ and the client identity ID_C are contained in the ST, then the server ticket $Ticket_V$ was generated by the TGS and the authenticator was created by that particular client whose identity is ID_C.

6.3 Testing Environment

Fig. 6 depicts our testing environment. The KDC is logically divided into the AS and the TGS. There exists a principal entry in the KDC database representing the TGS as a service. The AS (as well as the TGS) has access to the KDC's database and thus knows the long-term key associated with any user and any service registered or deployed in the realm. Besides, in our testing environment we have four client instances: client1, client2, client3, and client4. Finally, we got 2 servers: serverA, and serverB.

In our implementation, we used Triple-DES in CBC mode as an encryption algorithm, SHA-256 as a hashing algorithm, and Blum Blum Shub as a random number generator algorithm. In our design, the lifetime of the TGS ticket (the TGT) is 1 day, the lifetime of the server ticket is 8 hours, and the lifetime of the authenticator is 5 minutes.

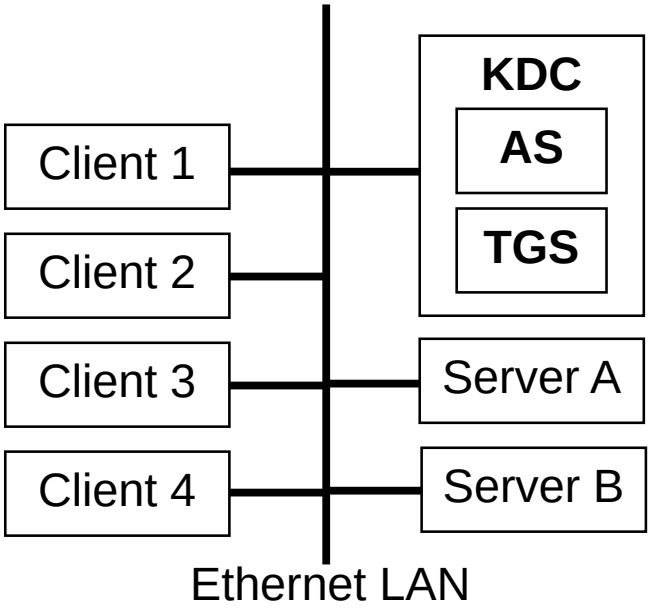

Fig. 6. A schematic for the testing LAN

7. Conclusions and Future Work

We introduced a LAN authentication protocol based on the widely deployed Kerberos authentication protocol with a little modification in the Kerberos database. It will be independent of the user password. The KDC will save a profile for every instance in the realm that it mange. This profile will be used to generate the principal secret key by applying a hashing algorithm to the profile. Then the output of the hashing algorithm will be encrypted to generate the principle secret key. The secret key lifetime will be controlled by appending the system lifetime to the instance profile. Thus, the secret key will be changed. By this way, we will overcome the weak passwords chosen by the network principal that are susceptible to password guessing attacks, the main drawback of the Kerberos protocol. We look forward to apply cross-realm authentication to our protocol in our future work.

References

[1] R. Needham, and M. Schroeder, "Using Encryption for Authentication in Large Networks of Computers". Communications of the ACM, December 1978.

[2] Y-C Hu, A. Perrig and D. B. Johnson, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless

Ad Hoc Networks", Proc of IEEE Workshop on Mobile Computing Systems and Applications, 2003.
[3] J. Kohl, and C. Neuman, "The Kerberos Network Authentication Service (V5)", RFC 1510. September 1993. Available at http://www.ietf.org/rfc/rfc3962.txt, 2005.
[4] C. Neuman and Ts'o. Theodore, "Kerberos: An Authentication Service for Computer Networks". IEEE Communications Magazine. September 1994.
[5] B. Bryant, "Designing an Authentication System: A dialogue in Four Scenes". Project Athena document (February 1988). Available at http://web.mit.edu/Kerberos/dialogue.html
[6] http://en.wikipedia.org/wiki/Kerberos_(protocol)
[7] C. Neuman, T. Yu, S. Hartman, and K. Raeburn, "The Kerberos network authentication service (V5)". Network Working Group. Request for Comments: 4120. Available at http: //www.ietf.org/rfc/rfc4120.txt, 2005.
[8] S. Bellovin & M. Merrit, "Limitations of the Kerberos Authentication System," SIGCOMM Comput. Commun. Rev., 20(5):119–132, 1990.
[9] G. Bella and E. Riccobene, "Formal analysis of the Kerberos authentication system". Journal of Universal Computer Science, 3(12):1337–1381, 1997.
[10] G. Bella and L. Paulson, "Kerberos version IV: Inductive analysis of the secrecy goals". In ESORICS '98. Springer, 1998.
[11] J. Kohl, "The use of encryption in Kerberos for network authentication". In CRYPTO '89. Springer, 1989.
[12] S. Stubblebine and V. Gligor. "On message integrity in cryptographic protocols". In Symposium on Security and Privacy '92. IEEE, 1992.
[13] T. D. Wu. "A real-world analysis of Kerberos password security". In NDSS '99. The Internet Society, 1999.
[14] T. Yu et al. "The perils of unauthenticated encryption: Kerberos version 4". In NDSS '04. The Internet Society, 2004.
[15] K. Raeburn. "Encryption and Checksum Specifications for Kerberos 5". Network Working Group. Request for Comments: 3961. Available at http://www.ietf.org/rfc/rfc3961.txt, 2005.
[16] K. Raeburn. "Advanced encryption standard (AES) encryption for Kerberos 5". Network Working Group. Request for Comments: 3962. Available at http://www.ietf.org/rfc/rfc3962.txt, 2005.
[17] F. Butler et al. "A Formal Analysis of Some Properties of Kerberos 5 Using MSR". In CSFW '02. IEEE, 2002.
[18] A. Boldyreva and V. Kumar, "Provable-Security Analysis of Authenticated Encryption in Kerberos". IEEE Symposium on Security and Privacy (SP'07). May 2007.
[19] M. Erdem, "High-speed ECC based Kerberos authentication protocol for wireless applications". Global Telecommunications Conference. GLOBECOM. IEEE Volume 3, (Dec 2003).
[20] A. Pirzada and C. McDonald. "Kerberos Assisted Authentication in Mobile Ad-hoc Networks". The 27th Australasian Computer Science Conference, conferences in Research and Practice in Information Technology. 2004.
[21] S. Sakane et al. "Applying Kerberos to the Communication Environment for Information Appliances", Symposium on Applications and the Internet Workshops (IEEE SAINT-w'03), 2003.
[22] Nitin et al. "Security Analysis and Implementation of JUIT—Image Based Authentication System Using Kerberos Protocol". Proceedings of the 7th IEEE/ACIS International Conference on Computer and Information Science, (ICIS 2008).
[23] http://www.kerberos.org/index.html
[24] William Stallings, "Cryptography and Network Security principles and practices", fourth edition. Pearson Prentice Hall, (2006). pp. 401-419, pp. 433-435.

Eman M. El-Emam received the BSc in Electronics & Communication engineering from Ain Shams University, Cairo, Egypt in 2000. She received a Diploma in Computer Networking from the ITI (Information Technology Institute), Giza, Egypt in 2001. She has been a communication engineer in the ESP (Egyptian Space Program) at NARSS (National Authority for Remote Sensing and Space Sciences), Cairo, Egypt since 2001 till now. Her research interest includes channel coding, network protocols, and network security.

Magdy A. Koutb received the Eng. Degree from the University of Menofiya, Egypt, in 1977 and M.Sc.degree in 1981 from the university of Al-Azhar, Egypt, and the Ph.D. degree from the University of Silesia, Poland in 1985. He has been a Professor since 1997 at the Faculty of Electronic Eng., Menoufiya University. In 2003 he was appointed as Vice-Dean of Post-Graduate Studies and Cultural affairs of the same faculty. He has extensive experience in Computer Engineering and Industrial Electronics. His main research interests include distributed systems, network security and intelligent control systems.

Hamdy M. Kelash received the Eng. Degree from the Institute of Electronic Engineering, Egypt in 1971, MSc degree from Faculty of Engineering Technology, Helwan University, Egypt, in 1979 and the PHD degree from Institute National Polytechnique (INP), France in 1984. He has been lecturer in 1984 at the Electronic Industry department, Faculty of Electronic Engineering, also a lecturer in 1987 at the Computer Sciences and Engineering department, and an Assistant Professor in 1993 and the Head of Computer Sciences and Engineering department, Faculty of Electronic Engineering, Menoufia University from 2001 to 2007. His main research interests include optical computing, artificial intelligence, network security, image processing, digital systems and parallel computing.

Osama S. Farag Allah received his BS in 1997, MSc in 2002, and PhD in 2007, all in computer science and engineering, from Menoufia University, Faculty of Electronic Engineering, Egypt. He was a demonstrator at the Department of Computer Science and Engineering, at Menoufia University, from 1997 to 2002, became an assistant lecturer in 2002, and was promoted to a lecturer in 2007. His research interests cover computer networks, network security, cryptography, Internet security, multimedia security, image encryption, watermarking, steganography, data hiding, and chaos theory.