Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Modified Client Authentication Responsibility in Kerberos Protocol

Ali M. Meligy, Walid A. Dabour, Alaa S. Farhat
Menoufya University,
Dept. of Mathematics,
Faculty of Science, Egypt

Abstract__ A variety of different external attacks were devised to threaten the distributed systems so they need a security mechanisms to authenticate and authorize the interactions between the nodes of the system. Kerberos is one of the authentication protocols which allow nodes to communicate over a non-secure network to prove their identity to one another in a secure manner. The aim of this paper is to develop an improved authentication protocol which allows constantly and successfully working in case of failure of Kerberos server.

Keywords __Kerberos, Authentication, Access Control, Client- Server, Security.

I. INTRODUCTION

Distributed systems involve the interaction between independent computers working towards a common goal. The authors in [7] identified the following advantages of using a distributed approach to system development: Resource sharing: A distributed system allows the sharing of hardware and software resources, Openness: Distributed systems are normally open system, which means that they are designed around standard protocols that allow equipment and software from different vendors to be combined, Concurrency: in Distributed system, several processes may operate at the same time on separate computers on the network, Scalability: in principle at least, distributed systems are scalable in that the capabilities of the system can be increased by adding new resources to cope with new demands on the system, and Fault tolerance: the availability of several computers and the potential for replicating information means that distributed system can be tolerant of some hardware and software failures. Distributed systems that are accessed over the internet are normally organized as client–server systems [7]. One advantage of using client-server is the ability to connect remote users (on a client computers) with remote resources to obtain some services that provided by servers. These servers achieved on the basis of certain access rights, policies or authentication protocol in an open and scalable way. However, this scalability and openness leads to insecurity [8].

A system that follows the client–server (as a type of distributed system) pattern is organized as a set of services and associated servers, and clients that access and use these services. In a client-server architecture, the functionality of the system is organized into services, with each service delivered from a separate server. Clients are users of these services and access servers to make use of them.

The principal advantage of this model is that servers can be distributed across a network. Therefore, general functionality can be available to all clients and does not need to be implemented by all services [7].

The major components of this model are explained figure 1.

a) A set of servers that offer services to other components.
b) A set of clients that call on the services offered by servers.
c) A network that allows the clients to access these services. Most client–server systems are implemented as distributed systems, connected using Internet protocols.

From a and b we are observing, clients may have to know the names of the available servers and the services that they provide. To securely ensure client-server model, servers need to know the identity of clients or how many clients are accessing their services. A client makes a request to a server and waits until it receives a reply. In a client–server architecture, an application is modeled as a set of services that are provided by servers. Clients may access these services and present results to end users. Clients need to be aware of the servers that are available but do not know of the existence of other clients.

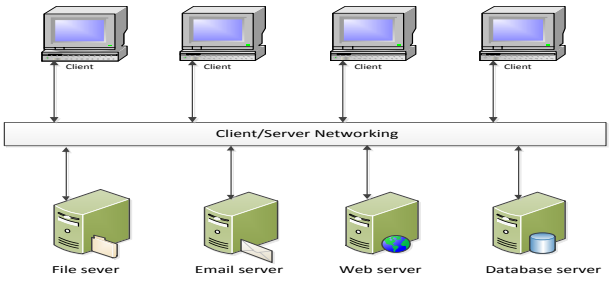

Figure 1. Client-Server Model.

From c, as more and more client-server models were connected to the internet, a variety of different external attacks were devised to threaten these systems. Security is one of the main challenges concerning distributed systems. Security in distributed systems is a complex issue which can produce several problems [4] such as disrupt the services offered by the system, steal confidential phishing or denial of service. The complexity of this issue relies on systems connected to the internet, a variety of different external attacks were devised to threaten systems.

In an unprotected network environment, any client can apply to any server for service. The obvious security risk is that of impersonation. An opponent can pretend to be another client and obtain unauthorized privileges on server machines. To counter this threat, servers must be able to confirm the identities of clients who request service [9]. Each server can be required to undertake this task for each client/server interaction, but in an open environment, this places a substantial burden on each server.

The Massachusetts Institute of Technology (MIT) developed Kerberos to protect network services provided by Project Athena [6]. Kerberos Is a computer network authentication protocol which works on the basis of 'tickets' to allow nodes communicating over a non-secure network to prove their identity to one another in a secure manner. Its designers aimed primarily at a client–server model. Kerberos protocol message are protected against eavesdropping and reply attacks. Not all services should be available to all clients, each server must identify the clients who can deal with them. Kerberos provides a centralized protection system, which to serves the network as a whole. Its job in basis is authenticate the client to the server and authenticate the server to the client. Security measures of servers should be proportional to value of the services and resources stored on that server.

The paper is organized as follows. In section II, Kerberos protocol is presented and its role in secure client server network. In section III previous work is surveyed in the area of how to secure client server network using Kerberos protocol. Finally in section IV, the proposed model is presented in order to achieve a secure authentication for Kerberos environment by separation between granting access rights and validate access.

II. KERBEROS PROTOCOL

Kerberos a network authentication protocol is now widely used in the distributed network applications. Its designed it primarily for a client–server model, it provides a mutual authentication between entities and high speed communication of authentication. based on reliable third-party authentication[5].

A *Thrust Third Party* is an entity which facilities interactions between two parties who both trust the third party, they use this trust to secure their own interactions. Kerberos makes use of a trust third party, termed a Key Distribution Center (KDC), which consists of two logically separate parts: an Authentication Server (AS) and a Ticket-Granting Server (TGS). The AS server gives a ticket-granting ticket (TGT) to the client. This ticket may be veiled for a day and may be used to obtain several service ticket (STs) for many different application server. The client can use TGT to access the TGS. The TGS server gives the service ticket (ST) to the client to can access Service Server (SS).

A single ST might be valid for a few minutes (although it, too, may be used repeatedly while it is still valid) [9]. This ticket used for a single application server. The client can use ST to access the request service.

Kerberos is used to ensure the security aspect when a client requests for certain services and it is works on the basis of " tickets" which serve to prove the identity of users.

The AS and TGS are responsible for creating and issuing tickets to the clients upon request [1] to be able to obtain the services provided by the different servers. Kerberos operates by encrypting data with a symmetric key. A *Symmetric Key* is a type of authentication where both the client and server agree to use a single encryption/decryption key for sending or receiving data.

A- How Kerberos Work

The user's goal is to be able to obtain on some services provided by various application servers (e.g., File Server, Email Server, Web Server, etc). A user operates on client, he enters a username and password on the client. The client performs a one-way hash on the entered password, and this becomes the secret key of the client. This key is considered a symmetric encryption key (single encryption / decryption key). The client sends a clear-text message of the user ID to the Authentication Server (AS) requesting services that provided by the server on behalf of the user. The AS generates the secret key by hashing the password of the user located in the database.

The AS looking for the client ID in its database. If it is , the AS sends respond to the client by two messages [6]:

Message1: Client/TGS session key encrypted using the secret key of the client.

Message 2: Ticket- Granting- Ticket which includes the (client ID, client network address, ticket validity period, and the client/TGS session key) encrypted using the secret key of the TGS. Once the client receives messages 1 and 2, it decrypts message 1 to obtain the client/TGS session key to can communicate with TGS . If the user entered password does not match the password in the AS database, the client's secret key will be different and thus unable to decrypt message

1. The client cannot decrypt message 2 because it was encrypted by TGS's secret Key. (The secret key that generated by AS should be match with the key was hashed by client on the user password because this key used to encrypts and decrypts message, if they are different, it cannot be authenticated to TGS). This sequence of steps are explained in figure 2.
2. At this point, the client has enough information to authenticate itself to the TGS.

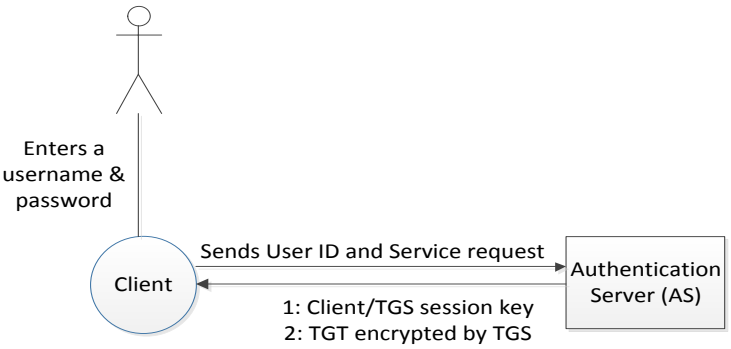

Figure 2. Client authentication.

Once the client receives the ticket TGT, the client sends the message 2 and ID of the request service to the following server. TGS receives this ticket then sends Service Ticket (ST) to the client which includes (the client ID, client network address, validity period and Client/Server Session Key) encrypted using the service's secret key and sends Client/Server Session Key encrypted with the Client/TGS session key encrypted with the client/TGS session key.

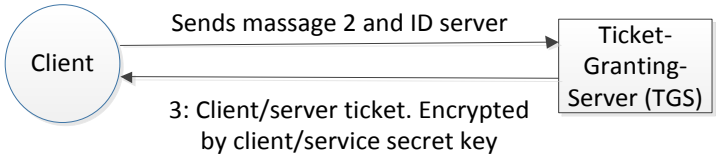

Figure 3. Client Service Authorization.

Upon client receiving Client/Server Session and Service Ticket, the client has enough information to authenticate itself to the Service Server (SS). The client connect to the SS and sends it the Service Ticket. The SS decrypts this ticket using its own secret key and confirm client true identity and willingness to serve the client, then the client can trust the server and can start issuing service request to the server. and the server can respond to it.

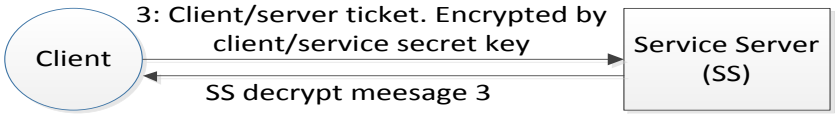

Figure 4. Client gets the service.

Finally, the user can to obtain on requested services provided by various application servers.

B- Kerberos drawbacks

The Kerberos protocol has some drawbacks such as *Password Guessing Attack*, Kerberos is not effective against password guessing attacks; if a user chooses a poor password, then an attacker guessing that password can impersonate the user [5]. Solving this problem by [9] through designing a user authentication protocol that is not susceptible to password guessing attacks. The main goal is to remove this password guessing attack.

Another problems in Kerberos protocol are *The Problem Of Timestamp:* Kerberos uses timestamp in order to prevent playback attack. But during the lifetime of the ticket, playback attack may still take effect [5]. *Network service accounting cannot be possible using Kerberos. Denial of service attacks* are not solved with Kerberos. There are places in these protocols where an intruder can prevent an application from participating in the proper authentication steps. Detection and solution of such attacks [9] is usually best left to the human administrators and users. Since all authentication is controlled by a centralized KDC, compromise of this authentication infrastructure will *allow an attacker to impersonate any user* [6].

Compromise of a verifier/server [9]: If the authentication mechanisms on the Authentication Server is compromised, all the services on that all servers in system is compromised. The attacker will be able to impersonate any client and connect to any to server to obtain services and can be able to decrypt any communication between the Authentication Server and a client. The security of the services running on a servers is dependent upon the security of the Authentication Server.

III. RELATED WORK

Establish a collaborative trust enhanced security model for distributed system in which a node either local or remote is trustworthy. Aruna et al [8] provides a promising solution with trust policies as authorization semantics by node registry and service level agreements are used to ensure the trust for a new client node. Developing authentication protocol which should defend most of the attacks on Kerberos known as replay attack, password guessing attack. The goal is to achieve a secure authentication for Kerberos environment with removal of weakness of existing Kerberos

protocol. Through proposed process in [9], the passwords that the hackers detected for each time will not be the same, so that the hackers are not able to validate those passwords. Traditional Kerberos authentication schemes do not meet distributed system requirements of authorization, failsafe operation, accounting of service usage and resilience to loss of connectivity. A^3 Kerberos protocol model proposed by Aditya et.al [5] meets the requirements of authentication, authorization and accounting.

Kerberos tickets used in KAMAN authentication scheme can be captured over the network are prone to replay attacks. Kashif Bashir and Mohammad Khalid Khan proposed that all of contents are encapsulated in an encrypted packet [2]. So the replay attacks become impossible. They were proposed methods can reduce the chances of reply attack in MANET using KAMAN as authentication protocol. Alan H. et. al [1] propose three components of the system architecture that designed to make attacks harder and to limit the damage done when attacks succeed.

The problem in the Kerberos protocol that we are going to be resolved is depend on the Kerberos server to authenticate the clients whose requests the services that provided by other application servers. When the Kerberos server is compromised, no any client can obtain some services. The goal is to achieve a secure authentication for Kerberos environment by separation between Granting access rights and Validate access.

IV. PROPOSED MODEL

In this paper, a new model is developed to ease the burden of client authentication on the server by making the client responsible for providing the credentials to Ticket Granting Server, then TGS check on this credential it is true or not. Authentication is generally accomplished in two ways: direct and indirect authentication. In direct authentication, two parties use pre-shared symmetric or asymmetric keys for verifying each other and the flow of data between them. In indirect authentication, a trusted third party, is made responsible for certifying one party to another party. In proposed model, the authentication through indirect authentication, that accomplished using Certificate Authority (CA).

A Certificate Authority (CA) is considered A *Thrust Third Party* is an entity which facilities interactions between two parties who both trust the third party. The CA is a trusted central administrative entity that can issue digital certificates to clients in network domain to make sure of its legitimate. The trust in the CA is the foundation of trust in the certificate as a valid credential. A CA uses its private key to create a digital signature on the certificate that it issues to validate the certificate origin. Others can use the CA certificate's public key to verify that authenticity of the certificates that the CA issues and signs. CA works with the base of the public key infrastructure . Which mean, each entity (client or server, etc.) have two keys, public key and private key. The public key is known to all entity in the system, but the private key is known to itself only.

When the client wishing to access the application servers would sends the it certificate to TGS. if it is true, the TGS allows the client to access the requested services from Services Server (SS).

Authentication using Certificate Authority has to go through three stages:

Stage1- Client Certificate Steps:

An organization that has determined it needs to provide certificate to clients or not. If CA is grant certificate to client, pass this stage, a number of steps:

In the first time,

1- The client sends the username and password to CA requesting granting certificate to itself to provides it to TGS to obtain the requested services.

2- The CA begins issuing a certificate to give it to the client. This certificate is contain:

i. Client ID.
ii. Client network address.
iii. Validity period. If the certificate expired, the client should update the certificate to allow him using the system.
iv. List of application servers that allowable to the client to access them.
v. Processes that the client can it work on the services. Say, "Read", "Modify" etc.
vi. Public key/ private key.
vii. Additional information.
viii. Digital signature of CA.

3- The CA sends the certificate to the client. At this point, the client has enough information to give it to the TGS to proof its right for access the server . See figure 5.

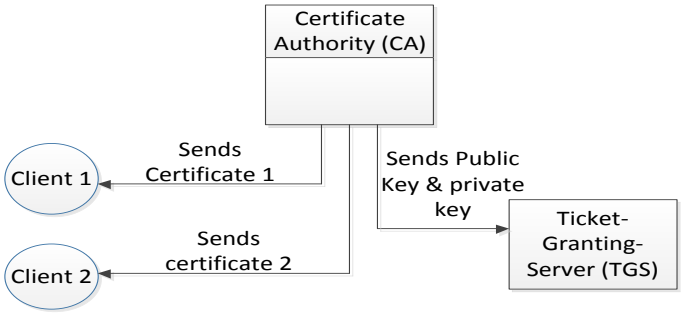

Figure 5. Clients have the Certificate.

Stage2- Submitting the Certificate to the TGS:

4- The client send this certificate exception its private key encrypted by the public key of the TGS.
5- The TGS decrypt this certificate using its private key.
6- The TGS has proof checker mechanisms to check on the certificate. If it is true, sends a message to the client to confirm its true identity and willingness to serve the client. The client can use this message until expiration data .
7- The TGS grant Service Ticket(STs) to the client even can access the application server. See figure 6.

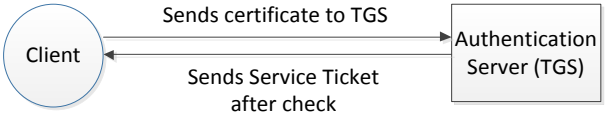

Figure 6. Client has the Certificate.

Stage3- Access to Required Services:

8- Finally, the client sends the ST in demand to SS encrypted by the public key of the SS.
9- The SS decrypt the ticket by its private key and it provide the service to the client.
In the Second time, Third time ….etc. the client provide the ST to SS direct without pass all previous steps until the end of the period of validity of the certificate . See figure 7.

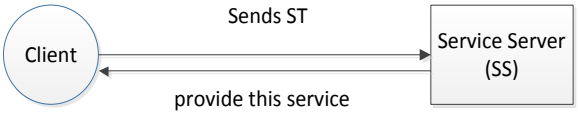

Figure 7. Client gets the service.

The summary of this phase is presented in figure 8. The CA is in charge of grant the client the its certificate, and public key/ private key for each entity in the system to communicate with them. The client presents this certificate to TGS to proof its true. Then sends it the ST to provide to application servers. The client can access the requested application server even expiration data.

This model separate between ownership of certificate that grant access rights (client) and who can validate this right (server).

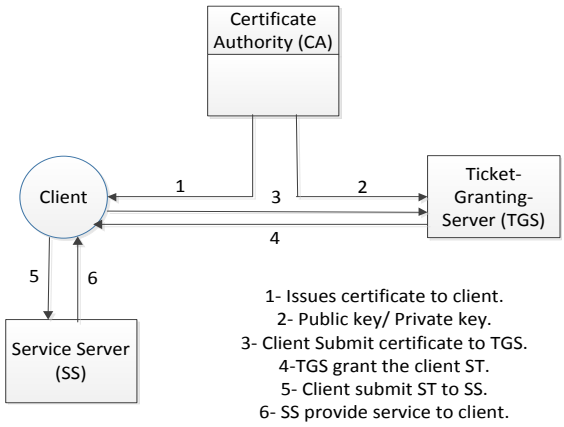

Figure 8. The proposed model.

V. *CONCLUSION*

This study revealed about new process model that aims to responsibility of the client to authenticate itself instead of the server is doing this task. This reduces the burden on the server and make the server is constantly. Making the its task is validate the credentials that owned by the client. The client is continues through working with the certificate until the expiration of the period of validity. Then reintroduced credentials back to the server and so. In future work, must search for model to authenticate clients to a server using another protocols.

REFERENCE

[1] Alan H. Karp and Smathers, K. Three Design Patterns for Secure Distributed Systems, *Intelligent Enterprise Technologies Laboratory HP Laboratories Palo Alto* HPL-2003-40 February 25th , 2003.
[2] Bashir, K and Khan, M Modification in Kerberos Assisted Authentication in Mobile Ad-Hoc Networks to Prevent Ticket Replay Attacks" *IACSIT International Journal of Engineering and Technology, Vol. 4, No. 3, June 2012*. 2012.
[3] Bauer, L; Pfenning, F and K. Reiter, M. Distributed System Security via Logical Frameworks, 2005.
[4] Garcıa, M; et. al. " Applying Dynamic Separation of Aspects to Distributed Systems Security", 2009.

[5] Harbola, A.; Negi, D and Harbola, D. A NEW A3 KERBEROS MODEL. *International Journal of Advanced Research in Computer Science and Software Engineering*, Volume 2, Issue 3, March 2012 ISSN: 2277 128X. 2012.
[6] http://en.wikipedia.org/wiki/Kerberos_(protocol). Accessed at 19/10/13.
[7] Ian, Sommerville *Software Engineering 9 th edition* Boston, New York, Addison-wesly (2011).
[8] Kumari, A; Mishra, S and Kushwaha. D.S. "A New Collaborative Trust Enhanced Security Model for Distributed System" *International Journal of Computer Applications (0975 - 8887) Volume 1 – No. 26. 2010.*
[9] Thakur, V and Hande, K.N. " Improving Kerberos Security Using Dynamic Password Based Authentication". 2, Vol.7 (November 2012), ISSN 2249-6149. 2012.

