RESEARCH ARTICLE OPEN ACCESS

An Authentication Protocol based on Kerberos

Jayati Ghosh Dastidar
Department of Computer Science, St. Xavier's College, Kolkata (Autonomous)

ABSTRACT
In this paper a single-sign-on authentication protocol has been proposed. The protocol is a derivative of the Kerberos protocol that uses one server for authentication purposes, except that it is simpler in its' implementation. Nonces and time-stamps are used to prevent replay attacks. The encryption schemes are all based on symmetric key cryptography. The protocol also is not susceptible to reflection attacks. The paper discusses the working of the protocol and analyses the strengths and weaknesses of the same.

Keywords: authentication, nonce, replay attack, single sign on, symmetric key

I. INTRODUCTION

Computer security authentication means verifying the identity of a user logging onto a network. Passwords, digital certificates, smart cards and biometrics can be used to prove the identity of the user to the network. Computer security authentication includes verifying message integrity, e-mail authentication and MAC (Message Authentication Code), checking the integrity of a transmitted message. There are human authentication, challenge-response authentication [1], password, digital signature, IP spoofing and biometrics.

II. BACKGROUND STUDY

Human authentication is the verification that a person initiated the transaction, not the computer. Challenge-response authentication is an authentication method used to prove the identity of a user logging onto the network [2]. When a user logs on, the network access server (NAS), wireless access point or authentication server creates a challenge, typically a random number sent to the client machine. The client software uses its password to encrypt the challenge through an encryption algorithm or a one-way hash function and sends the result back to the network. This is the response.

As authentication is increasingly becoming important due to the exponential increase of network services different authentication methods have been put to practice. The general model that all authentication protocols use is this. Alice starts out by sending a message either to Bob or to a trusted KDC (Key Distribution Center), which is expected to be honest. Several other message exchanges follow in various directions. As these messages are being sent Trudy may intercept, modify, or replay them in order to trick Alice and Bob or just to gum up the works.

Initially if it is assumed that Alice and Bob already share a secret key, K_{AB} then a protocol can be created based on a principle found in many authentication protocols [3]: one party sends a random number, R_B to the other, who then transforms it in a special way and then returns the result. Such protocols are called *challenge-response* protocols. Random numbers used just once in challenge-response protocols like this one are called *nonces* [1].

The essence of the protocol is that Alice asks Bob to send a **large random number** which when received is encrypted by Alice using K_{AB} and transmitted back to Bob. When Bob sees this message, he immediately knows that it came from Alice because Trudy does not know K_{AB} and thus could not have generated it. Furthermore, since R_B was chosen randomly from a large space (say, 128-bit random numbers), it is very unlikely that Trudy would have seen R_B and its response from an earlier session. It is equally unlikely that she could guess the correct response to any challenge. This process is repeated by Alice to confirm that she is talking to Bob. Hence in all five messages are sent. This protocol can be shortened by using three messages instead of five, where Alice initiates the challenge-response protocol instead of waiting for Bob to do it. Similarly, while he is responding to Alice's challenge, Bob sends his own.

Under certain circumstances, Trudy can defeat this protocol by using what is known as a *reflection attack*. In particular, Trudy can break it if it is possible to open multiple sessions with Bob at once. It starts out with Trudy claiming she is Alice and sending R_T. Bob responds, as usual, with his own challenge, R_B. But Trudy can effortlessly get around by entering into another session with Bob, supplying the R_B taken from her challenge. Bob calmly encrypts it and sends back K_{AB} (R_B). Now Trudy has the missing information, so she can complete the first session and abort the second one. Bob is now convinced that Trudy is Alice, so when she asks for her bank account balance, he gives it to her without question.

Kerberos (V4) is a very famous Network Authentication Protocol proposed and implemented by Massachusetts Institute of Technology [4]. The working has been shown in Fig. 1.

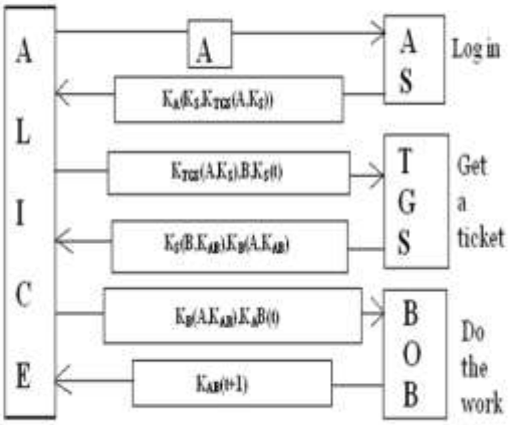
A
L
I
C
E
A
A
S
Log in
$K_A(K_S,K_{TGS}(A,K_S))$
$K_{TGS}(A,K_S),B,K_S(t)$
T
G
S
Get
a
ticket
$K_S(B,K_{AB}),K_B(A,K_{AB})$
$K_B(A,K_{AB}),K_{AB}(t)$
B
O
B
Do
the
work
$K_{AB}(t+1)$

Fig. 1. Kerberos (V4) Protocol

The advantages of the protocol are as follows:

- The passwords for authentication of both clients and servers are required.
- Client and server systems mutually authenticate - both the client and the server systems may be certain that they are communicating with their authentic counterparts.
- While the specific length of time for which a user's authentication remains valid after his initial ticket issued is implementation dependent, Kerberos systems typically use small enough ticket lifetimes to prevent brute-force and replay attacks. In general, no authentication ticket should have a lifetime longer than the expected time required to crack the encryption of the ticket.
- Authentications are reusable and durable. A user need only authenticate to the Kerberos system once (using his principal and password). For the lifetime of his authentication ticket, he may then authenticate to Kerberized services across the network without re-entering his personal information.
- As a side-effect of the dual-key encryption scheme employed in the Kerberos model, a service-session key is generated which constitutes a shared secret between a particular client system and a particular service. This shared secret may be used as a key for encrypting the conversation between the client and the target service, further enhancing the security of Kerberized transactions.
- Unlike many alternative authentication mechanisms, Kerberos is entirely based on open Internet standards. A number of well-tested and widely-understood reference implementations are available free of charge to the Internet community. Commercial implementations based on the accepted standards are also available.
- Unlike many of its proprietary counterparts, Kerberos has been scrutinized by many of the top programmers, cryptologists and security experts in the industry. This public scrutiny has ensured and continues to ensure that any new weaknesses discovered in the protocol or its underlying security model will be quickly analyzed and corrected.
- In Kerberos (V4) (the version of Kerberos used by AFS and Zephyr) all encryption is performed using the DES algorithm. While DES was considered "unbreakable" at the time of the release of Kerberos (V4), it is now believed that a sufficiently motivated miscreant could, with only modest computing resources, conceivably crack DES encryption in a relatively short period of time. Some researchers have, in fact, been able to do just that under certain specific circumstances. Since the trustability of Kerberos authentication depends entirely on unbreakability of the underlying encryption technology used by the system, this poses a threat to the security of Kerberos (V4). In the current release of Kerberos, Kerberos (V5), support is provided for "plug-in" symmetric encryption algorithms. Kerberos (V5) systems can use, for example, the much more secure triple-DES or IDEA encryption algorithms. The overall structure of Kerberos (V5) remains the same as that of Kerberos (V4) [5].
- Kerberos was designed for use with single-user client systems. In the more general case, where a client system may itself be a multi-user system, the Kerberos authentication scheme can fall prey to a variety of ticket-stealing and replay attacks. The overall security of multi-user Kerberos client systems (file system security, memory protection, etc.) is therefore a limiting factor in the security of Kerberos authentication. No amount of cleverness in the implementation of a Kerberos authentication system can replace good system administration practices on Kerberos client and server machines.
- Because Kerberos uses a mutual authentication model, it is necessary for both client machines and service providers (servers) to be designed with Kerberos authentication in mind. Many proprietary applications already provide support for Kerberos or will be providing Kerberos support in the near future. Some legacy systems and many locally-written and maintained packages, however, were not designed with any third-party authentication mechanism in mind,

and would have to be re-written (possibly extensively) to support Kerberos authentication.

- The Kerberos authentication model is vulnerable to brute-force attacks against the KDC (the initial ticketing service and the ticket-granting service). The entire authentication system depends on the trustability of the KDC(s), so anyone who can compromise system security on a KDC system can theoretically compromise the authentication of all users of systems depending on the KDC. Again, no amount of cleverness in the design of the Kerberos system can take the place of solid system administration practices employed in managing the Kerberos KDC(s) [6].

III. PROPOSED DESIGN

This authentication protocol that we are suggesting is intended to provide better all-round security across the insecure networks that connect the digital world.

The terminologies that would be used are as follows**:**

- Alice – The client workstation.
- Authentication Server(AS) – Verifies (authenticates) the user during log in.
- Ticket Granting Server(TGT) – Issues tickets to certify proof of identity.
- Bob – The server offering services such as network printing, file sharing or an application program.
- Trudy- A possible intruder i.e. a human or a machine.
- TICKET – A ticket is granted to a principal (typically a user) by the KDC. The principle can then use that ticket to authenticate itself to another principle (typically a service).
- TGT – A ticket granting ticket. A special ticket given to a principal after successful authentication, which allows that principle to request additional tickets.
- KDC – Key Distribution Center.
- NONCE – As the timestamp is a known entity, a nonce is used for added security. Nonce is basically used to be encrypted by session keys.

The Fig. 2 shows the model of the proposed model.

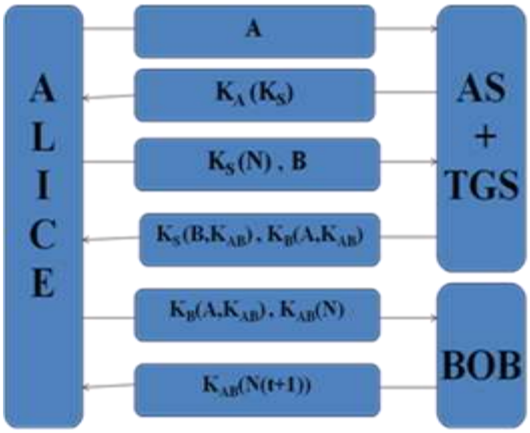
ALICE
A
$K_A(K_S)$
$K_S(N)$. B
$K_S(B,K_{AB})$. $K_B(A,K_{AB})$
$K_B(A,K_{AB})$. $K_{AB}(N)$
$K_{AB}(N(t+1))$
AS + TGS
BOB

Fig. 2. Proposed Protocol

The entire protocol can be divided into three phases:

A. Phase 1

1) Step 1

i) Alice sends its username to AS + TGS.

ii) AS + TGS sends back the password-generated secret key of Alice K_A with K_S, a session key (random number but mutually agreed format).

If Trudy intercepts either of the two messages as they travel across the insecure network it will not benefit because the username of Alice is known and the package K_A (K_S) can be opened only if the password of Alice is known.

2) Step 2

i) Alice's workstation asks for password, computes K_A, decrypts K_S if Alice's password is right and encrypts a nonce using K_S. Alice also requests for service from Bob (server). It sends back the package to AS + TGS.

ii) Now AS + TGS is sure this is Alice. It sends K_{AB}, a session key for Alice to use with Bob. Two versions of it are sent back. The first is encrypted with only K_S, so only Alice can read it. The second is encrypted with Bob's key, K_B, so that Bob only can read it.

3) Step 3

i) Now Alice sends the second package i.e. K_B (A, K_{AB}) to Bob which tells Bob that Alice wants to enter session with it using K_{AB}. Along with that K_{AB} is encrypted with a nonce.

ii) Bob acknowledges by sending back K_{AB} with an updated nonce.

B. Phase 2 (Accounting)

It refers to the methods to establish who, or what, performed a certain action, such as tracking user connection and logging system users. This information may be used for management, planning, billing, or other purposes.

Accounting simply records which clients accessed the network, what they were granted access to, and when they disconnected from the network.

This correlation can make other systems that are not user-aware more intelligent in the security decisions that they make.

C. Phase 3 (Auditing)

It refers to an evaluation of an organization, system, process, project or product. Audits are performed to ascertain the validity and reliability of information, and also provide an assessment of a system's internal control. Auditing verifies that processes are reasonable, appropriate for expected results.

Extension of Shared Session Key

The session key shared by Alice and Bob can be used by Alice to communicate with Bob for as many sessions possible within a limited frame of time, without going into the authentication process with AS+TGS [7]. The K_{AB} is saved in Alice's workstation encrypted using Alice's private key (K_A). Alice's workstation will authenticate Alice by asking her for her password and then generating K_A.

Nonce

As the timestamp is a known entity, a nonce can be used for added security [1]. In the Kerberos protocol the nonce is basically used to be encrypted by session keys. So it can be a combination of the timestamp and a random number, the range of which is large enough that even in a brute-force attack using the world's latest (fastest) technology it will take more than four days to try out all the combinations.

The protocol has the following advantages:

i) Instead of using two different servers we are using only one server. Also the client needs to authenticate once as AS and TGS are combined. The extended use of the shared session key (K_{AB}) saves the client and AS+TGS interaction (4 messages) in the beginning. Hence all these measures reduce overhead.

ii) Using nonce instead of timestamp we can ensure a higher level of security.

This protocol also suffers from the following disadvantages:

i) There is a single point of compromise but if an intruder can break into either AS or the TGS, then also the protocol is compromised.

ii) There is a possibility of a bottleneck as we are combining both the AS and TGS but this is also significantly reduced with the extension of shared session keys.

iii) The mapping between different AS and TGS cannot exist on merging the two servers into one but if we consider a small system with a limited number of servers where only one TGS suffices then it is possible to do so [8].

IV. CONCLUSION

The protocol that has been proposed here is a simplified form of the Kerberos. Instead of using a separate Authentication Server (AS) and a Ticket Granting Server (TGS) as in Kerberos, the proposed protocol needs to use just one that plays the role of both. The protocol can be further enhanced by incorporating the following features into it:

i) Introducing a mechanism that uses a different passcode each time authentication is required.

ii) Extending the Kerberos, which is currently supporting purely symmetric key distribution, to other key distribution systems like asymmetric key distribution (public key).

iii) Introducing the concept of realms and cross-realm interfacing including distant realms to enhance manageability and scalability.

iv) Making the authentication protocol compatible with cloud-based networking.

This protocol has been tested using Java Sockets, and have received promising results.

REFERENCES

[1] A. S. Tanenbaum, Computer Networks, Fourth Edition, Prentice Hall of India, 2002

[2] A. Kahate, Cryptography and Network Security, Second Edition, Tata McGraw Hill, 2003

[3] W. Stallings, Cryptography and Network Security - Principles and Practices, Sixth Edition, 2014

[4] B. C. Neuman and T. Ts'o, Kerberos: An Authentication Service for Computer Networks, IEEE Communications, 1994

[5] A. H. Harbitter, D. A. Menasce, Performance of Public-Key-Enabled Kerberos Authentication in Large Networks, Security and Privacy, Proceedings of IEEE Symposium, 2001

[6] P. Babu Tiwari, S. Ram Joshi, Single Sign-on with One Time Password, AH-ICI, IEEE, 2009

[7] M. A. Sirbu, J. Chung-I Chuang, Distributed Authentication in Kerberos Using Public Key Cryptography, Network and Distributed System Security, IEEE Symposium, 1997

[8] S. W. Jung, S. Jung, Secure Password Authentication for Distributed Computing, IEEE International Conference on Computational Intelligence and Security, 2006

International Journal of Engineering Research and Applications (IJERA) is UGC approved Journal with Sl. No. 4525, Journal no. 47088.

Jayati Ghosh Dastidar. "An Authentication Protocol based on Kerberos." International Journal of Engineering Research and Applications (IJERA) 7.7 (2017): 70-74.