Global Information Assurance Certification Paper

Kerberos: Secure Authentication
Jose L. Marquez
Version 1.2D

KERBEROS: SECURE AUTHENTICATION

KERBEROS: At a Glance

Kerberos was developed at MIT in the late 1980's. It is open source code that is freely available to anyone. Kerberos is one of a few security initiatives implemented by Microsoft to harden their Windows 2000 Operating System. Kerberos has traditionally been used in colleges and universities. It has also found uses in financial institutions. With its current deployment as part of the Windows 2000 OS it promises to reach a much wider user audience. There are some issues with interoperability between Kerberos and the Unix Operating System as Microsoft has installed some proprietary components into the products.

Kerberos is an authentication protocol that allows clients and servers to reliably verify each others identity before connectivity can happen. It provides advantages such as mutual authentication and message integrity as well as data confidentiality. Kerberos must go through a process of establishing a secure authenticated network connection. This process is performed as client and server validate their respective identities to each other before performing any application functions. Both the client and the server must establish a "trust" before a network connection can be established. In practical terms this means that the service must be able to determine who the client is without asking the client and the client must be able to determine who the service is without asking the service.

The Kerberos authentication protocol provides a mechanism for mutual authentication between entities before a secure network connection is established. Both client and server can also be referred to as security principals.

Kerberos in itself is comprised of three sub-protocols:

(1) *Authentication Service Exchange (AS)*: In this sub-protocol, the Key Distribution Center or as it is more commonly known (KDC) gives the client a logon session key and a Ticket-Granting Ticket (TGT)

(2) *Ticket-Granting Ticket (TGT) Exchange*: This sub-protocol allows the KDC to distribute a service session key and a ticket for the service

(3) *Client/ Server (CS) Exchange*: In this sub-protocol, the client presents the ticket for admission to service

For more information on this, simply go to the links below.
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/netdir/ad/about_mutual_authentication_using_kerberos.asp

KERBEROS 101

Packets traveling on a network may be subject to viewing, interception, and tampering. Methods must be devised to not only protect the privacy and confidentiality of data but

its integrity also. Kerberos provides password and symmetric key encryption to authenticate users and thereby offer communications between two computers, or many computers a certain degree of security while being transparent to the users. The protocol supports “DES” and “TripleDES” for data encryption.

The Kerberos authentication protocol is based on the interaction of three parties. There are two devices, usually in the form of a client and a server, that wish to open up a channel for communication with each other. Each of these devices has their own unique long-term password (the password that is used at each login). The third party in the authentication process is the Key Distribution Center or as it is more commonly known, the KDC. It mediates between the client and server by providing a secret that they can share amongst each other. The KDC also knows the password of each and every client/ server that it mediates for. That information is maintained in an encrypted database.

To function Kerberos uses a process called secret key cryptography, or as it is better-known, symmetric key cryptography. Secret key cryptography takes a plaintext message and converts it into ciphertext (unintelligible data). After communication has been authenticated the data is converted back into plaintext using one key. Thus, the two devices will share a secret key to encrypt and decrypt their communication. This transaction forms the basis of the authentication method.

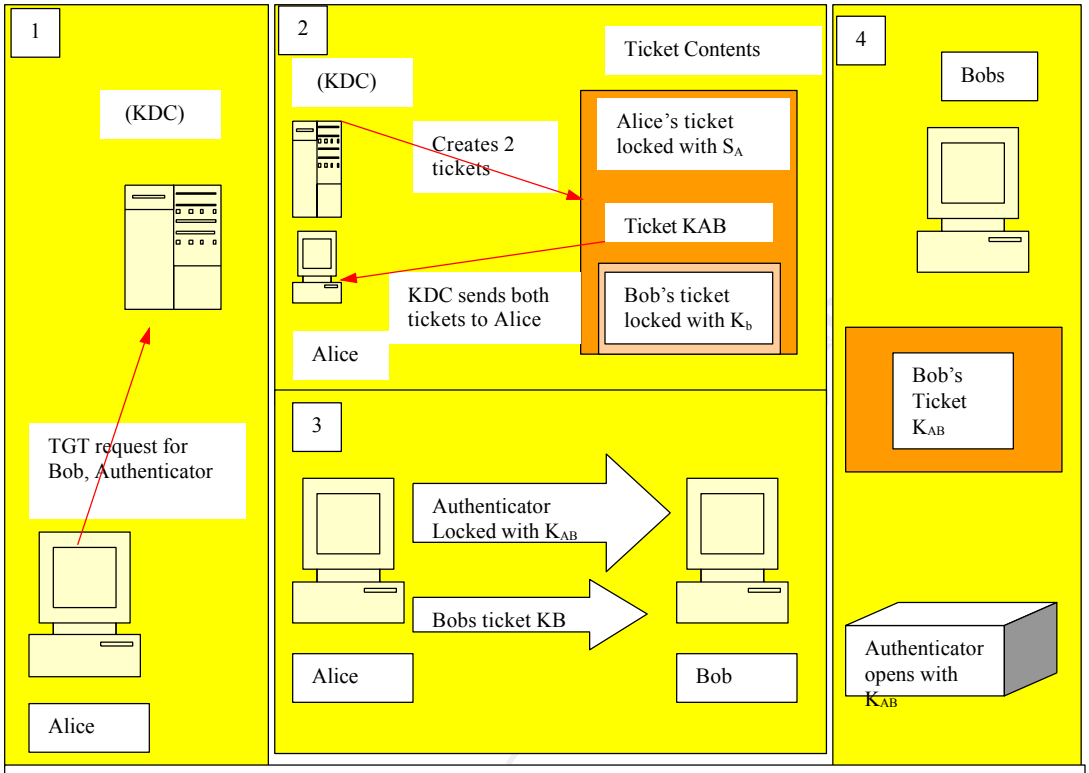
The Kerberos transaction consists of what are termed as session keys and tickets. The session key is a secret prepared specifically for the two devices to share authentication data and to also provide message integrity and confidentiality. The ticket itself is composed of data pertaining to the actual transaction and gives the ticket requester access to another machine. A client must receive a session key and a ticket for each machine it wants to establish access to.

The Kerberos ticket is comprised of data that includes but is not limited to a timestamp that states when the ticket was created and an expiration time for the ticket. The clients/ servers use this timestamp as a source for the authentication process so it is absolutely critical that their clocks are as closely synchronized as possible. The expiration time determines how long the ticket is valid for. This was put in place as a security measure to limit the possibility that an intruder could possibly hijack a ticket and cause some damage. The lifetime for a ticket is normally a standard workday.

Given the current trends, the normal workday is not the same for all computer users. The System Administrator can adjust the expiration time to be as short or as long they want. This is dependent of the security posture that the organization happens to be in. As a rule the expiration time should not be set for too short an interval. This causes bandwidth to be compromised on the network as different clients and services continually re-authenticate. Another by-product of this re-authentication would be that users would also have to continuously re-enter their passwords. On the other hand, expiration times should not be of an excessively long duration. You must create a policy that will fall in-line with the organizations normal trends and security posture and temper it with some good common sense.

Dividing your client/servers who authenticate to one KDC can create realms. These machines have a direct trust relationship with that particular KDC. Communications between these realms can occur through a process called cross-realm authentication. This enables you to have dedicated client/ servers authenticating to one KDC, which speeds up

the process, with the flexibility to communicate with client/ servers in other realms.



The Authenticator. When Alice wants to authenticate to Bob, she must first contact the Key Distribution Center (KDC). 1) Alice sends her Ticket-Granting Ticket (TGT) and a request to authenticate to Bob to the KDC. 2) The KDC creates two tickets and a new session key (K_{AB}). The KDC nests Bob's ticket inside Alice's ticket. Alice's ticket is encrypted with Alice's secret key (S_A), and Bob's ticket with his secret key (K_B). Both tickets include the new key (K_{AB}). The KDC returns both tickets to Alice. 3) Alice decrypts her own ticket, revealing the new session key (K_{AB}). Alice then encrypts an authenticator with K_{AB} and sends Bob's ticket and the encrypted authenticator to Bob. Bob decrypts his ticket with K_B, revealing the session key (K_{AB}), which he can use to reveal the authenticator from Alice.

THE PROCESS

In a Kerberos environment, the authentication process begins at login. Alice enters her user name and password into her client workstation. The information is relayed to the KDC, which contains a master database of unique, long-term keys for all the principals located within its realm. The KDC scans the database and based on Alice's password, locates her master key (K_A). After the KDC locates the master key it generates two items: a session key (S_A) to share with Alice, and also a Ticket-Granting Ticket ($_{TGT}$). The TGT contains a great deal of information. Housed within the TGT is a 2nd copy of the session key (S_A), Alice's user name and an expiration date. This ticket is encrypted by the using KDC with its own master key (K_{KDC}). This data is only known by the KDC.

All the information is then encrypted by the KDC and forwarded to Alice's workstation with her master key attached (K_A). The encrypted information is run through

a one way hashing function, converting her password into her master key (K_A). Using Alice's master key, the workstation now has a session key that it shares with the KDC and a TGT. It will now only use the session (S_A) and TGT for and subsequent communication with the KDC. The checks and balances that are put in place make Kerberos a formidable authentication method.

Alice's workstation contacts the KDC with a request to speak to Bob's workstation. The communication from Alice consists of the TGT, the request and an item called an authenticator. The authenticator is a critical piece of information. It is a timestamp. The workstations must be as closely synchronized as possible for the Kerberos authentication protocol to work properly. The authenticator is also encrypted using Alice's session key (S_A) that she shares with the KDC.

The TGT is then decrypted by the KDC using its own master key (K_{KDC}). The TGT contains Alice's user name and a copy of the shared session key (S_A). The KDC uses this information to decrypt the authenticator. Since only Alice and the KDC share this information, it's easily confirmed by the KDC.

The next step in the process is for the KDC to create a pair of tickets. One for Alice and another for Bob. The data contained within the ticket is essential for the process to proceed. This data is the name of the principals involved in the transaction, a timestamp verifying when the ticket was created, and an expiration showing how long the tickets will be valid for. Also contained within the ticket is a new (K_{AB}) that will be shared between Alice and Bob. Bob's ticket is now encrypted by the KDC using Bob's master key (K_B). The ticket is nested within Alice's ticket by the KDC, which also contains the new key (K_{AB}). All the pertinent data is then encrypted using the session key (S_A) that the KDC shares with Alice. The ticket is then sent to Alice.

This ticket is decrypted by Alice using the session key (S_A). The new session key is revealed (K_{AB}), and also the ticket for Bob. Alice's workstation still cannot read Bob's ticket. The authenticator (Timestamp) is encrypted using the new (K_{AB}) and is sent along with Bob's ticket to the workstation (Bob's). Once Bob receives these items, Bob first decrypts his own ticket using his master key (K_B). This permits access to the new session key (K_{AB}), which can then decrypt the authenticator from Alice.

Both Alice and Bob now possess the new key. Bob can be sure that Alice is who she claims to be because Alice used the (K_{AB}) to encrypt the authenticator. If it is necessary at this point for Bob to communicate with Alice, Bob will use the new (K_{AB}). Alice will know who Bob is who he claims to be, as he had to use his master key (K_B) to get the new key (K_{AB}). For each and every communication request, the same process must be adhered to. The KDC will create a unique session key for the principals that are involved each and every time.

WHAT IS IN A TICKET

The fields in the table denote the ticket structure .The exact data structures for tickets as well as messages can be found in RFC 1510.

Table 1: Fields of a Ticket

Field Name	Description

The first three fields in a ticket are not encrypted. The information is in plaintext so that the client can use it to manage tickets in its cache.	
tkt-vno	Version number of the ticket format. In Kerberos v.5 it is 5.
Realm	Name of the realm (domain) that issued the ticket. A KDC can issue tickets only for servers in its own realm, so this is also the name of the server's realm.
Sname	Name of the server.
The remaining fields are encrypted with the server's secret key.	
Flags	Ticket options.
Key	Session key.
Crealm	Name of the client's realm (domain).
Cname	Client's name.
Transited	Lists the Kerberos realms that took part in authenticating the client to whom the ticket was issued.
Authtime	Time of initial authentication by the client. The KDC places a timestamp in this field when it issues a TGT. When it issues tickets based on a TGT, the KDC copies the authtime of the TGT to the authtime of the ticket.
Starttime	Time after which the ticket is valid.
Endtime	Ticket's expiration time.
renew-till	(Optional) Maximum endtime that may be set in a ticket with a RENEWABLE flag.
Caddr	(Optional) One or more addresses from which the ticket can be used. If omitted, the ticket can be used from any address.
Authorization-data	(Optional) Privilege attributes for the client. Kerberos does not interpret the contents of this field. Interpretation is left up to the service.

The *flags* field is a bit-field in which options are set by turning a particular bit on (1) or off (0). Although the field is 32 bits long, only nine ticket flags are of interest to Kerberos administrators.

Table 2: Ticket Flags

Flag	Description
FORWARDABLE	(TGT only) Tells the ticket-granting service that it can issue a new TGT with a different network address based on the presented TGT.
FORWARDED	Indicates either that a TGT has been forwarded or that a ticket was issued from a forwarded TGT.

PROXIABLE	(TGT only) Tells the ticket-granting service that it can issue tickets with a different network address than the one in the TGT.
PROXY	Indicates that the network address in the ticket is different from the one in the TGT used to obtain the ticket.
RENEWABLE	Used in combination with the *endtime* and *renew-till* fields to cause tickets with long life spans to be renewed at the KDC periodically.
INITIAL	(TGT only) Indicates that this is a TGT.

Excerpt for tables 1 and 2 taken from:
Step-by-Step Guide to Kerberos 5 (krb5 1.0) Interoperability: White Paper Published by Microsoft Posted Feb 2000

INTEROPERABILITY CONCERNS

Microsoft's implementation of the Kerberos protocol is based on standards-track specifications recommended to the Internet Engineering Task Force (IETF). As a result, the implementation of the protocol in Windows 2000 lays a foundation for interoperability with other networks where Kerberos version 5 is used for authentication.

Windows KDC. Non-windows Kerberos implementations can authenticate to the KDC in a Windows 2000 domain. Non-windows Kerberos users and hosts can authenticate to a domain controller by using **kinit** and DES-CBC-MD5 or DES-CBC-CRC encryption.

Non-windows Kerberos KDC. Systems running Windows 2000 can authenticate to a host serving as the KDC of a Kerberos realm. In addition, a standalone Windows 2000 system can be configured so that local computer accounts map to Kerberos principals. This configuration allows users to log on simultaneously to both the computer and the Kerberos realm.

Windows Client, Non-Windows Kerberos Service. Client applications running on Windows 2000 can authenticate to non-Windows Kerberos services if the services support the Generic Security Service Application Program Interface (GSS-API) defined in RFC 1964.Windows 2000 does not provide the GSS-API. Applications written for the Windows 2000 operating system should use the SSPI to get support for Kerberos version 5 authentications. The two interfaces are compatible and similar.

Non-Windows Kerberos Client, Windows Service. Client applications running on non-Windows Kerberos systems can authenticate to services running on Windows 2000 if the client applications support the GSS-API as defined in RFC 1964.

Trust Relationships. Windows 2000 domains can be configured to trust non-Windows Kerberos realms. Non-Windows Kerberos realms can be configured to trust Windows domains. The trusts are not complete like domain trusts. For instance, user principals from the Kerberos realm do not have the authorization data that Windows 2000-

based services need for access control. In order for this authorization data to be added to the user's ticket, an account mapping mechanism is used. Selected domain user accounts are used to provide identity for Kerberos principals in the trusted realm. These mappings are kept on the altSecurityID property on the user account object. They can be managed through Active Directory Users and Computers.

SUMMARY

The security of personal computers, a few computers on a small network or a few thousand workstations requires that users be proactive. Users must not wait for an event to happen. Measures must be in place that will minimize the chances that anyone will exploit critical systems. The authentication process discussed in this reading is but one part of hardening a network. Intrusion detection systems must also be incorporated on the network. Systems need both a network based Intrusion detection (IDS) system to monitor the outside of the network and a host based IDS to monitor what's happening on the inside. An application based firewall should be configured and in place. Scans must be performed on networks periodically for vulnerability assessments. Content/mail scanners also must be in place as well as an anti-virus program running in the background with the most current virus definitions. Once this is in place operational status must be maintained through configuration management. Only through vigilance, awareness, and the use of the right tools can you hope to survive what has been termed as "Information Warfare".

References

Millman, Howard "Prepare for your greatest security risk: An inside attack" Aug 1, 2001 http://www.techrepublic.com/article.jhtml

Configuration utilities for configuring Windows 2000 for interoperability with UNIX: http://www.microsoft.com/WINDOWS2000/library/howitworks/security/kerbint.asp.

Step-by-Step Guide to Kerberos 5 (krb5 1.0) Interoperability: White Paper Published by Microsoft Posted Jan 10 2000 http://www.microsoft.com/windows2000/techinfo/planning/security/kerbsteps.asp.

About Mutual Authentication Using Kerberos Platform SDK Release: June 2001 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/netdir/ad/about_mutual_authentication_using_kerberos.asp

Publisher: CSWL, Inc. "Enforcing windows 2000 security " Published: March 21, 2001 http://click.techrepublic.com

Kerberos
TechRepublic: www.techrepublic.com
Many authors

The SANS Institute, "Windows NT Security Step by Step" (2000 - 2001)
Many Authors

© SANS Institute 2000 - 2005 Author retains full rights.