**KERBEROS AUTHENTICATION PROTOCOL**

**James Bach, Jaymi Johnson, Stacey Sosa**

**ABSTRACT**

Kerberos is a widely used authentication protocol that is used when clients and resources want authenticated access control on unsecure networks. Information security is an important consideration of its design, innovations, and revisions. The ticket-based protocol provides confidentiality of information, mutual authentication, and data integrity. There are many advantages of the system and the disadvantages are often overcome by combining other effective access control features. The challenges of computer security are not exempt when it comes to this system; it is not invulnerable or fool-proof. However, Kerberos remains a reliable method of providing authentication, being used by many enterprises, institutions, and governments. Kerberos coupled with some other security protocols that make up for its disadvantages can provide a powerful authentication process. This technical report will discuss the history and usage, how it works, advantages and limitations, and the attacks with countermeasures used with the Kerberos protocol.

**INTRODUCTION**

Kerberos is a ticket-based network authentication protocol “for trusted hosts on untrusted networks”. [1] Kerberos is named after the three-headed dog Cerberus from Greek and Roman Mythology, which guarded the entrance to the underworld. The protocol was developed by MIT in the 1980’s and was intended to provide access control to Project Athena, a campus-based educational computing environment at MIT. Kerberos’ primary purpose was to protect system resources and minimize the frequency of common attacks. Since its creation, there have been several different versions of Kerberos but due to vulnerabilities and innovations, earlier versions have been deprecated. The current version, Kerberos 5, is the most secure, using triple DES and AES with 128 and 256-bit keys; it became an Internet Engineering Task Force Standard in 1993. At first Kerberos was designed for the UNIX operating system but it has since changed and it is at the disposal of major operating systems. [2] There are several advantages and a few disadvantages in the Kerberos authentication system, most of which revolves around the limitations, types of attacks, and the countermeasures the system can support. Kerberos helps provide authentication of users, verifying that users are who they say are they are. Additionally, information that is sent across a network is encrypted providing data integrity and confidentiality. The Kerberos system provides protection to its users against eavesdropping, phishing, and replay attacks. When considering the reliability of the system, it is important to consider the time that it has been out in the market and used in the real world. Kerberos has become a trusted system because it has existed for many years. A system that is used in the real world is vulnerable to many attacks, but it is extremely important in terms of computer security that the system manages to prevent, detect, and recover from future attacks in a timely manner. Throughout its history, Kerberos’ has proven its resilience and high reliability rate and a vast amount of enterprises, institutions, and even the United States government use this system. [3] Researchers and developers are frequently checking for vulnerabilities that may arise so patches with upgrades or modifications can be applied.

**DISCUSSION**

There are a few goals of using the protocol defined by the Kerberos IT Consortium that the system tries to accomplish: (1) the user’s password must not be sent on the network; (2) the user’s password must not be stored on their machine and must be discarded once the session has completed; (3) the user's password should not be stored in plaintext; (4) the user only has to provide a single point of entry to the session, a characteristic of single sign-on access control; (5) authentication server is centralized with all the necessary information to be self-contained and the resource server is not required for authentication; (6) the system provides mutual authentication; (7) the session with the resource is encrypted. [1]

There are several important definitions in understanding the Kerberos authentication protocol:

* **Principal** is the term used to describe how the authentication server database understands and refers to the entity requesting a ticket to use a specific system resource. A principal is associated with each service, host and user. [1]
* **Key Distribution Center (KDC**) is a single physical server but logically it can be split into the database, *Authentication Server (referred to as AS)*, *and Ticket Granting Server (referred to as TGS).* This unit distributes all the tickets and session keys that are needed for the whole authentication process. The database stores information about the principle such as encryption key, maximum duration of ticket, password expiration date, principal expiration date, or flags for the ticket if necessary.
* **Ticket Granting Ticket (TGT)** is an encrypted electronic container of information used in the Kerberos protocol that contains a principal/KDC session key, an expiration date, and the IP address of the principal (which prevents from man-in-the-middle attacks. [4]
* **Service Ticket** is another encrypted electronic container with principal information, resource requested for use, IP address, timestamp, ticket max lifetime/expiration, and a session key between the principal and the resource.
* **Session key** refers to a temporary key used for a session between the principal and the KDC and another session key exists between the principal and the resource
* **Resource** is the desired system resource of the principal. An example is an application server hosting Telnet or a print server.
* **Authenticator** is used by the principal to the KDC and the resource and contains the principal information and a timestamp.

[REQUEST/RESPONSE 1]

The Kerberos process begins with the identification step of the principal/user sending a request for a ticket to the KDC. The KDC’s logical unit, the authentication server, will handle this transaction. The authentication server will first determine if the user is authorized in this particular system. The AS will search the database and find the principal information and the secret key information it will use to effectively communicate with the user with confidentiality in mind. If the user is not found or authorized in the system, the communication is terminated. Assuming the user is found, the AS responds to the user with two pieces of information: the Ticket Granting Ticket and the session key between the user and the KDC. The Ticket Granting Ticket (or TGT) will be encrypted with an internal secret key that is only shared between the Authentication Server and the Ticket Granting Server. The session key that the AS is returning to the user is encrypted with the secret key it found in the database for that particular principal. If the principal is who they said they were and have the secret key/password they are expected to have, the principal will be able to decrypt the session key for the next step.

[REQUEST/RESPONSE 2]

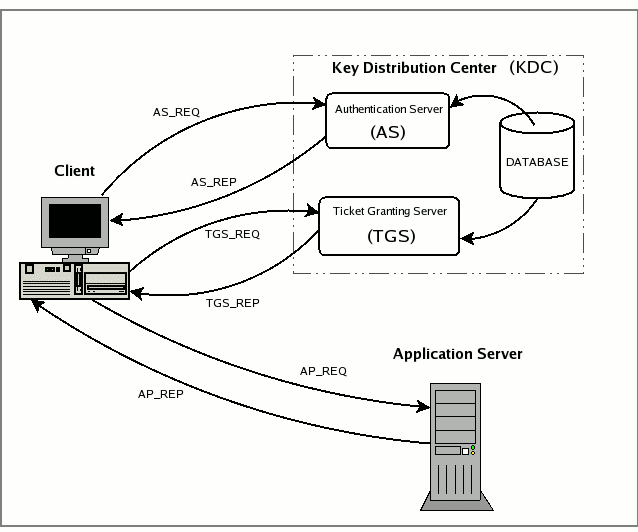
The principal now has the session key and an encrypted ticket granting ticket. The next request is sent to another KDC logical unit, the ticket granting server. The user will be sending two things: the encrypted TGT and an Authenticator. The Authenticator being sent will be encrypted with the session key that was previously granted by the authentication server and decrypted using the secret key associated with the correct password. The KDC will receive this request from the principal with the TGS and will be able to decrypt the ticket granting ticket with the shared secret key it shares with the authentication server. The information revealed by decrypting this ticket is the session key between the principal and the KDC, an expiration, and the IP address of the requesting user. The IP address of the ticket and the user sending the ticket must match but this alone is not enough to validate the principal. An attacker could have spoofed an address to ensure these match. The TGS will have to be able to decrypt the Authenticator with the session key contained in the previously decrypted ticket. The TGS will also be checking the timestamp in the Authenticator to ensure it has not been replayed. [1] If these previous actions check out, the TGS has effectively authenticated the user sending the responses. The AS will now be returning a valid Service Ticket to the user and a principal/resource session key. The TGS and the resource have a known shared key that the user will not know. This key is used to encrypt the Service Ticket. The session key for the user and the resource will be encrypted with the previous session key between the principal and the KDC. The user will receive the encrypted service ticket and the new session key.

[REQUEST/RESPONSE 3]

The user now has the Service Ticket it received from the Ticket Granting Server that can be used to access the resources or services. In order to begin usage of the resource the user sends the still encrypted service ticket to the resource and another Authenticator that has been encrypted with the new session key between the principal and the resource. The resource server will receive these two elements and decrypt the Service Ticket to get the session key between the principal and the resource, previously granted by the Key Distribution Center. The Resource Server must be able to decrypt the Authenticator using the session key between the principal and the resource it just extracted from the Service Ticket. Upon successful decryption, the Resource Server will then validate the time stamp and principal information in the Authenticator. This concludes the final authentication process and the verification step. The session will now be opened.

AS\_REQ/AS\_REP represents [Request/Response 1] where the principal is getting authenticated.

TGS\_REQ/TGS\_REP represents [Request/Response 2] is the process of granting a ticket to the client machine.

AP\_REQ/AP\_REP represents [Request/Response 3] is the submission of the valid ticket information the resource where the session can begin.

This diagram comes from [1] Kerberos Protocol Tutorial

There are some implementation features that exist with the Kerberos protocol. Renewable tickets, for example, can be resent to the KDC to be renewed. If the ticket is within the maximum renewal time and has not expired otherwise, the principal can renew the ticket without their password and continue to work within the session. In the event that the principal needs to login to another machine, once authenticated by the Kerberos protocol the principal can login to the new machine without re-entering their password utilizing another feature, forwardable tickets. Kerberos is more secure than rsh or public key authentication with ssh which would be needed to accomplish a similar set up. [1]

An advantage to Kerberos is that information is self-contained in the physical server that performs the authentication. There is a database that stores all principals, passwords associated with their respective encryption keys, the maximum valid duration a ticket can be issue for, the maximum renew time that the particular user can be issued, any specific privileges or behaviors attributed to their tickets, the user’s password expiration and the actual user’s expiration if intended to be a temporary user of the system. Kerberos does not authorize its users, this information has already been stored in a database. This design provides for higher security since passwords are not stored locally nor sent across the network, reducing the opportunity of an attack. [5] This can be seen as a limitation of the system as well. Since Kerberos only authorizes users and does not authorize them, there is a process of authorization that happened with the entry into the database that could have been compromised or erroneous.

An important advantage about Kerberos is that authentication is mutual between users and servers. “[Kerberos] enables mutual authentication for each and every communications session that a user establishes with services, but it manages these authentication procedures “under the covers” once a user initially authenticates successfully through the Kerberos system.”[6] This not only protects the server but also the client. The entities taking part in an exchange do not have to know each other or have had previous communication to be able to verify each other. Neither passwords nor secret keys are shared between the entities that are being verified, but because of the Kerberos design, the two entities can authenticate each other.

The most visible advantage is Kerberos provides its users with a single sign-on experience. [7] The ticket system allows the single sign-on property of access control. The user only has to perform the initial authentication with their credentials and receive their service ticket to gain access to all resources without re-authenticating. This applies as long as their service ticket has not expired and they retain the privileges to the resources they are accessing.

An advantage that carries a related minor disadvantage is that Kerberos is extremely time-sensitive. If the hosts do not have synchronized clocks, the authentication may fail even when both, user and server, are trusted parties. [6] Optimally, the system should be kept up to date and synchronized. Due to this important time requirement and the usage of timestamps, Kerberos is extremely resistant to replay attacks. If it is not updated then an attacker has an opportunity to exploit this discrepancy and manipulate the process, causing the protocol to fail in providing confidentiality, integrity and even availability. This strict time requirement leads to a low margin of error allowed for timestamps and because of this even legitimate users can be rejected

Another disadvantage associated with an advantage is a reciprocal of the single sign-on. Kerberos is considered to have a single point of failure so if the system is unavailable or denied availability then no user will be able to log in and no authentication, key distribution, or ticket granting can take place. Since Kerberos only authenticates, it is important that an entity uses other systems to complement Kerberos. This entity should integrate a system that has the capability of authorization and auditing functions. [8]

There are definitely limitations to the successful operation of the protocol. If the client system where the principal is entering the password information has already been compromised by malware such as a Trojan it may have already modified the entry point of the system. An attacker then could gain sufficient information to impersonate the user. The Kerberos protocol does not protect all messages sent and received from the two computers, it only provides protection for the messages from software that has been coded to utilize it. [9] Although Kerberos is a secure system it is not immune to certain types of attacks such as password guessing attacks. If the user chooses a poor password it has an opportunity of being compromised and because it uses symmetric encryption for authentication, the shared secret key is compromised. The most common way in which an unauthorized user can gain entry to an authorized user account is cracking the password. A computer generated account password is extremely unlikely to be cracked because it changes after it is used and will not be able to be used again. However, a user account password is more likely be cracked with password guessing techniques such as popular password attacks, rainbow tables, brute-force, or dictionary attacks. Kerberos has used DES, RC4-HMAC-EXP and other weak encryption algorithms in previous versions but has since been deprecated due to the brute force capabilities that have been discovered. [10]

The attacker must first compromise at least one system, which will be the initial point of attack. For example, in a Windows environment, once the attacker obtains this system then they can query the domain system controller to acquire access of the principal information. This allows the attacker additional opportunities to use as little or many tickets to attack a system [11]. In an example of an attack using Windows PowerShell, a task-based command-line shell and scripting language designed especially or system administration, the attacker can request a ticket for the web server offline without sending a single ticket to the server [12]. For the attacker to be recognized as authentic to the Kerberos server, they must first get a Ticket Granting Ticket. This ticket will only be used with the Key Distribution Center where the attacker must find accounts that can be cracked.

In a Microsoft Windows network using an older Kerberos version, it may use a NT LAN Manager (NTLM) hash for security protocols designed to prevent spoofing attacks. This can be manipulated once an attacker has the password for the user account it is able to request a ticket for specific or all accounts. The attacker can do this in one of two ways: (1) request a ticket for individual services, to blend in with the other users requesting tickets; (2) use a service to request mass quantities of tickets while not connected to the remote service which helps the attacker to stay stealthy and less likely to be detected. The first action is designed to counter intrusion detection protocols that might be implemented, by performing similar tasks functions as a user. When requesting a ticket the system resource does not have to be accessible, available, or even exist anymore, as long as the corresponding information has not been updated. The Key Distribution Center then grants the service tickets because the attacker’s ticket granting ticket is authentic and Kerberos is not authorizing entity [11]. This enables the attacker to utilize or attack the resource.

In previous versions of Kerberos within a Windows environment, an attacker could also attack and decrypt the ticket once it was received and modify all the information in the ticket including the privilege to gain unauthorized entry into an authorized users account or resource servers. The Service Ticket that the attacker received from the Ticket Granting Service contains the Privilege Attribute Certificate (PAC), which contains all the relevant user information, for example principal identifiers, principal IP address, or group membership for access control. Once inside the users account, the attacker can modify any of this information to access different groups and use different credentials [11]. This certificate contains enough information that the service the attacker wants to access can decide if the user/attacker is allowed. The successfully modified service ticket alone has enough power in the system that it does not need to ask the Windows Domain controller for additional information, minimizing opportunities of being caught.

A countermeasure to protecting the privileges is that it contains two hash-based authentication code signatures. The first signature that is always checked is the service account’s hash as secret key, which if the attacker is able to read the PAC then they are able to crack the signature because it is the same key for encrypting and signing. The second signature is the Kerberos Ticket Granting Ticket hash as a key, this is not practical to crack; however, the second signature is only checked sometimes. If the resource server or remote service runs a protocol for ‘Trusted Computer Based’ or TCB permission then the second signature will be asked to be verified [11]. This is opportunity to catch compromised certificates. During administration or creation of a Kerberos system the administrator can force the services to verify the PAC second signature; although this is time consuming and does not prevent cracking of the account it will prevent the attacker to rewrite any details about the user [11].

There are a few other countermeasures that can protect against attacks on Kerberos. For example, in a Microsoft Windows environment, during installation of Kerberos you are asked for accounts to use for each service, by doing this the user is setting up principal names. The SQL service will then register the principal names automatically if the account is a domain administrator. This is the easiest way to set up the account, but the more susceptible way to be attacked. Microsoft suggests using Active Directory Service Accounts to create service accounts, although it is much more time consuming it will help eliminate the risk of being attacked. The user is also responsible for choosing a really good password that is not likely to be guessed; using random password for this system is okay because it is only used a few times and it does not always need to be easily remembered. The user can also monitor a domain controller which is a server that can respond to security requests for authentication.

**CONCLUSION**

The widely accepted and commonly used Kerberos protocol is open for anyone to download and view the code. This might reassure users that Kerberos is a reliable system because, despite the availability of the code where attackers have an opportunity to evaluate potential attacks, the system maintains its reliability as a trusted authentication system. Organizations can decide whether to download it from MIT or opt for a vendor’s version. [13] Kerberos has become a system that can be easily implemented to any network organization and for client use. [2] The Kerberos system can reduce administration costs because of its independent operational capabilities and providing computer security. There are convincing advantages to using this particular protocol to providing authentication, data integrity, and confidentially and features such as single sign-on, resistance to replay attacks or eavesdropping, and mutual authentication. Most of the disadvantages of using the system can be overcome by using additional countermeasures or security protocols. Kerberos continues to remain a contender for providing authentication among networks and resources.

**REFERENCES**

[1] Fulvio Ricciardi (2007, Nov 11). “Kerberos Protocol Tutorial”. Kerberos IT Consortium [Online]. Available: http://www.kerberos.org/software/tutorial.html [Accessed: Oct. 27, 2014].

[2] SANS Institute (2004). “Understanding Kerberos v5 Authentication Protocol”. Global Information Assurance Certification [Online]. Available: http://www.giac.org/paper/gsec/3462/understanding-kerberos-v5-authentication-protocol/105672 [Accessed: Nov. 3, 2014].

[3] Ken Hornstein (2000, August 18). “Kerberos FAQ”. U.S. Navy Research Lab [Online]. Available: http://www.cmf.nrl.navy.mil/krb/kerberos-faq.html [Accessed: Nov. 17, 2014].

[4] Mark Walla “Kerberos Explained”. Microsoft Technet [Online]. Available: http://technet.microsoft.com/en-us/library/bb742516.aspx [Accessed: Oct. 27, 2014].

[5] V. Alex Brennen (2004). “An Overview of a Kerberos Infrastructure”. Linux Documentation Project [Online]. Available: http://www.tldp.org/HOWTO/Kerberos-Infrastructure-HOWTO/overview.html [Accessed: Nov. 3, 2014].

[6] MIT Kerberos Consortium (2008). “Why is Kerberos a credible security solution?” Kerberos IT Consortium [Online]. Available: http://www.kerberos.org/software/whykerberos.pdf [Accessed: Nov. 3, 2014].

[7] Kerberos Consortium (2008, July 23). “Best Practices for Integrating Kerberos into Your Application”. Kerberos IT Consortium [Online]. Available: http://www.kerberos.org/software/appskerberos.pdf [Accessed: Oct. 27, 2014].

[8] Tillman Hodgson. “14.5. Kerberos”. The FreeBSD Project [Online]. Available: https://www.freebsd.org/doc/handbook/kerberos5.html [Accessed: Nov. 3, 2014].

[9] B. Clifford Neuman and Theodore Ts’o (1994, September). “Kerberos: An Authentication Service for Computer Networks”. Institute of Electrical and Electronics Engineers [Online]. Available: http://gost.isi.edu/publications/kerberos-neuman-tso.html [Accessed: Oct. 27, 2014].

[10] L. Hornquist Astrand (2012 July) “Deprecate DES, RC4-HMAC-EXP, and Other Weak Cryptographic Algorithms in Kerberos.” Internet Engineering Task Force [Online]. Available: http://tools.ietf.org/html/rfc6649.html [Accessed: Nov 12, 2014].

[11] T. Medin. “Attacking Microsoft Kerberos: Kicking the Guard Dog of Hades”. Derbycon 2014. [Online]. Available: http://www.irongeek.com/i.php?page=videos/derbycon4/t120-attacking-microsoft-kerberos-kicking-the-guard-dog-of-hades-tim-medin [Accessed: 5 November 2014]

[12] Windows Server. “Active Directory Cmdlets in Windows PowerShell,” Microsoft Technet. [Online] Available: http://technet.microsoft.com/en-us/library/ee617195.aspx [Accessed: Nov. 5, 2014].

[13] “What is Kerberos?” Kerberos: The Network Authentication Protocol [Online]. Available: http://web.mit.edu/kerberos/#what\_is [Accessed: Nov. 3, 2014].

[14] Stanford. “Authentication: Kerberos – How Kerberos Works,” ITServices Stanford. Available: https://itservices.stanford.edu/service/kerberos/user\_guide/how [Accessed: Oct. 27, 2014].