**Kerberos Protocol**

# Introduction

Kerberos borrows its name from the mythical beast Kerberos (Cerberus to most) a three headed dog with a snake tail, who guards the gates to the underworld Hades.

The Kerberos protocol was originally developed for MITs distributed computing environment in the 1980s under the code name Athena, the Goddess of wisdom and war. Kerberos is regarded as an elder technology by internet standards and came about at around the same time as the Domain Name Server protocol (DNS).

(Kerberos.org, 2008)

Simply put Kerberos uses tickets which allows nodes (user clients mostly) to communicate with each other over a non-secure, hostile network, and prove their identity to each other in a safe and secure manner through mutual authentication. This protocol user port 88 by default which uses the Unreliable Datagram Protocol (UDP).

(En.wikipedia.org, 2019)

# Cryptography Methodology

Symmetric key cryptography is used in this system which does mean that it requires a trusted third party, which in Kerberos’s case is the server that distributes tickets. These Key Distribution Centres (KDC) are what both Clients and Servers both rely on to verify credentials and authorise sessions, all involved need not have prior relationships with each other, only with the KDC.

Kerberos is not limited to symmetric key cryptography, it has the option to use public key cryptography at any phase of authentication where it is required within a given system. This generally happens for the initial authentication between client/servers and the KDC.

(En.wikipedia.org, 2019)

In cryptography old technologies and protocols are sought after as it shows successful consistency over a long space of time, particularly if the protocol saw continuous use, growth and improvements over the years, in this case Kerberos is a success. Another huge aspect of this cryptographic protocols success was that its requirements were made clear from the start both functional and non-functional. Knowing what the Kerberos must do and the environment within which it must operate was a key ingredient to obtaining success.

(Kerberos.org, 2008)

Within the Windows server 2003 environment the Kerberos Protocol has being upgraded to a version 5, this provides again authentication and mutual authentication between clients and servers, or servers and servers. It implements this protocol as a Security Support Provider (SSP), which can be accessed through an SSPI (where ‘I’ stands for Interface).

The tickets produced by the KDC is simply a data structure where all of the users credentials are packaged, this ticket is then encrypted and messages are used to transport the users details across a network. It’s important to note that Kerberos only authenticates the users identity within its KDCs, access to servers is only achieved after the users identity is verified, a systems Local Security Authority will either authorise or deny access. In short a ticket will only prove that the user is who they say who they are.

There is a multitude of keys used to achieve the level of encryption and privacy that Kerberos offers:

* First is the Long-term key, which is only known by a target server and the KDC, this key is used to encrypt a clients ticket.
* Second is the client/server session key, which is a short-term key that is used to encrypt client to server and vice-versa messages, after identity and authorisation have been comfirmed.
* Third is the KDC/user session key, this key is shared between users and the KDC and is used to encrypt a message to a client which contains a session key.

As mentioned before Kerberos uses both Symmetric and Asymmetric encryption, this is strongly evident in how it handles tickets in their KDCs.

Additionally the Kerberos authentication system creates and securely delivers an authenticator that is packaged alongside the clients ticket, which is normally based on timestamps. This authenticator is unique to this requested session and is single use, this limited use minimises that chance of someone intercepting and reusing a client ticket, which would be a malicious attempt to steal a users identity and possibly pose as them in the network environment.

(Docs.microsoft.com, 2009)

# Kerberos Weaknesses

The original Kerberos 4 uses the Data Encryption Standard (DES) algorithm, which by todays standards is very outdated, a mediocre computing system could easily launch a brute-force attack and crack the DES encryption in a short space of time. This is a minor issue and only applies to older systems as more modern systems have Kerberos 5 implemented with uses the 3DES algorithm.

Thanks to the mutual authentication model, both client and server machines have to be designed with Kerberos in mind, again this isn’t a major issue as most systems and applications provide support for Kerberos, but some legacy systems and applications were not designed with any third-party authentication mechanisms in mind, these would have to be re-written to support Kerberos authentication.

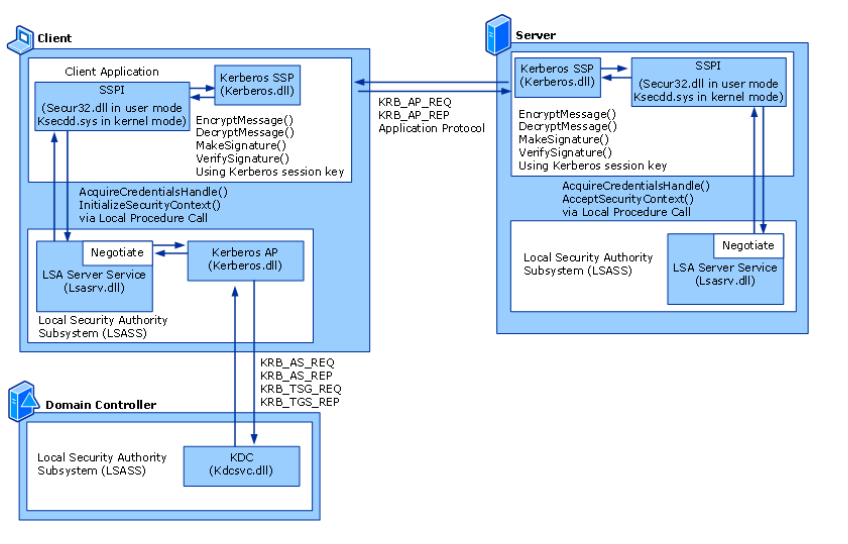
Finally an definitely the most important vulnerability is the KDC itself, the KDC is responsible for all ticket distribution which is vital for authentication, this places the KDC as a single point of failure for the entire system should it be compromised, all principals (clients and servers are singularly referred to as principals) connected to a compromised KDC will have all their personal data exposed to whoever the offender may be. An important point that recurs regarding Kerberos is

“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”.

Simply put, a lot of breaches in systems generally comes down to human error and not a flaw in the system itself.

(People.duke.edu, 2019)

# Kerberos Architecture



The above diagram shows how Kerberos authentication fits with other technologies in Windows Server 2003, depending on whether client and/or server applications are user-mode or kernel-mode applications, they will either use Secur32.dll or Ksecdd.sys respectively, by means od Secure Support Provider Interface calls to communicate with the Local Security Authority Subsystem (LSASS).

(Docs.microsoft.com, 2009)

# Implementation

Based on the above research I have decided to implement the Key Distribution Centre in the Kerberos protocol, this section will chronicle my attempt at implementation and whether I was successful or not.

After a large amount of research, it started to become very evident that implementing the KDC as a singular entity is well beyond my capabilities at this point, it requires database connection and a large amount of java libraries that I can’t get functioning in Java as I know it. Additionally I require two principals, ideally a server and client which at this point isn’t feasible for me to accomplish, that being said if I was given a full semester with no distractions I would very much like to implement this system. I have code that I have sourced from online, which is open source, but I lack the capabilities to run it and I feel using it doesn’t help with my understanding of the underlying implementation.

Next I’m going to narrow down and look exclusively at Data Encryption Standard (DES) and see if I can implement it in Java. During my KDC research I noticed that it comes up a lot seeing that it is the base of the Kerberos encryption, I have also chosen DES from version 4 over 3DES from version 5 because it should be easier to implement.

Having researched this through sources online, I was successfully able to encrypt and decrypt a message within a text file. This was achieved by having a plain text file that contained readable words, next the cipher will encrypt the text and write the encryption to an encrypted text file to show the encrypted values and finally the cipher will be decrypted and written to a new plain text file. I decided to use file input and output in place of a database where these encryptions would be stored. In a fully implemented Kerberos protocol this would resemble the encrypted tickets that would be issued to principals to verify their identity.

# Bibliography

Kerberos.org. (2008). [online] Available at: https://www.kerberos.org/software/whykerberos.pdf [Accessed 17 Dec. 2019].

En.wikipedia.org. (2019). *Kerberos (protocol)*. [online] Available at: https://en.wikipedia.org/wiki/Kerberos\_(protocol) [Accessed 17 Dec. 2019].

People.duke.edu. (2019). *Kerberos: Advantages and Weaknesses*. [online] Available at: http://people.duke.edu/~rob/kerberos/kerbasnds.html [Accessed 17 Dec. 2019].

Docs.microsoft.com. (2009). *What Is Kerberos Authentication?: Logon and Authentication*. [online] Available at: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc780469(v=ws.10)?redirectedfrom=MSDN [Accessed 17 Dec. 2019].

Docs.microsoft.com. (2019). *Using Kerberos integrated authentication to connect to SQL Server - SQL Server*. [online] Available at: https://docs.microsoft.com/en-us/sql/connect/jdbc/using-kerberos-integrated-authentication-to-connect-to-sql-server?redirectedfrom=MSDN&view=sql-server-ver15 [Accessed 17 Dec. 2019].

Docs.oracle.com. (2019). *Kerberos Requirements*. [online] Available at: https://docs.oracle.com/javase/8/docs/technotes/guides/security/jgss/tutorials/KerberosReq.html#SetProps [Accessed 17 Dec. 2019].

Docs.oracle.com. (2019). *Kerberos Requirements*. [online] Available at: https://docs.oracle.com/javase/8/docs/technotes/guides/security/jgss/tutorials/KerberosReq.html [Accessed 17 Dec. 2019].

Docs.oracle.com. (2019). *kinit - Kerberos tool*. [online] Available at: https://docs.oracle.com/javase/7/docs/technotes/tools/windows/kinit.html [Accessed 17 Dec. 2019].

Docs.oracle.com. (2019). *ktab - Kerberos tool*. [online] Available at: https://docs.oracle.com/javase/7/docs/technotes/tools/windows/ktab.html [Accessed 17 Dec. 2019].

Ibm.com. (2019). *IBM Knowledge Center Error*. [online] Available at: https://www.ibm.com/support/knowledgecenter/SSYGQH\_4.5.0/admin/secure/t\_install\_kerb\_create\_service\_account.html [Accessed 17 Dec. 2019].

Java?, S., White, A., White, A., Zwinck, J., Smith, M., Sanyal, A., Roussel, D. and Sanyal, A. (2019). *Simple Kerberos client in Java?*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/5804314/simple-kerberos-client-in-java [Accessed 17 Dec. 2019].

Kerberos.org. (2019). *MIT Kerberos Distribution Page*. [online] Available at: https://kerberos.org/dist/index.html#krb5-1.17 [Accessed 17 Dec. 2019].

Kontext.tech. (2019). *Java Kerberos Authentication Configuration Sample &amp; SQL Server Connection Practice - Java Programming - Kontext - Cloud, Data and AI Community*. [online] Available at: https://kontext.tech/column/javaprogramming/217/java-kerberos-authentication-configuration-sample-amp-sql-server-connection-practice [Accessed 17 Dec. 2019].

Programcreek.com. (2019). *Java Code Examples org.apache.directory.server.kerberos.kdc.KdcServer*. [online] Available at: https://www.programcreek.com/java-api-examples/?api=org.apache.directory.server.kerberos.kdc.KdcServer [Accessed 17 Dec. 2019].

Support.microsoft.com. (2019). [online] Available at: https://support.microsoft.com/en-us/help/816042/how-to-configure-an-authoritative-time-server-in-windows-server [Accessed 17 Dec. 2019].

Web.mit.edu. (2019). *Kerberos 5 Release 1.17.1*. [online] Available at: http://web.mit.edu/kerberos/www/krb5-1.17/ [Accessed 17 Dec. 2019].

Web.mit.edu. (2019). *Kerberos: The Network Authentication Protocol*. [online] Available at: http://web.mit.edu/kerberos/www/index.html [Accessed 17 Dec. 2019].

Web.mit.edu. (2019). *krb5.conf — MIT Kerberos Documentation*. [online] Available at: http://web.mit.edu/kerberos/krb5-1.13/doc/admin/conf\_files/krb5\_conf.html#libdefaults [Accessed 17 Dec. 2019].

Docs.oracle.com. (2019). *Cipher (Java Platform SE 7 )*. [online] Available at: https://docs.oracle.com/javase/7/docs/api/javax/crypto/Cipher.html [Accessed 17 Dec. 2019].

Docs.oracle.com. (2019). *Cipher (Java SE 13 & JDK 13 )*. [online] Available at: https://docs.oracle.com/en/java/javase/13/docs/api/java.base/javax/crypto/Cipher.html [Accessed 17 Dec. 2019].

JournalDev. (2019). *Java DES Algorithm Program - JournalDev*. [online] Available at: https://www.journaldev.com/1309/java-des-algorithm-program [Accessed 17 Dec. 2019].