Global Information Assurance Certification Paper

Microsoft Windows 2000
Kerberos version 5 Protocol

What does it do and how does it work?

Submitted by
Joseph A. Brown

GCNT Practical Assignment
Version 2.1b

Submitted
July 31, 2001

Table of Contents

Abstract 1
Introduction 1
Overview 2
How Kerberos Works 4
    Obtaining the Ticket Granting Ticket 4
    Obtaining a Ticket 6
    Connecting to the Resource or Service 6
What Else is Happening 7
    The Actual Ticket 7
    Ticket Lifetimes 8
    Other Types of Tickets 9
        Renewable Tickets 9
        Proxy Tickets 9
        Forwardable Tickets 10
Kerberos Policy Settings 10
Authentication 11
    Pre-authentication 11
    Authentication Delegation 12
DNS and Kerberos 12
The Credentials Cache 13
Authentication versus Authorization 13
Kerberos Strengths and Weaknesses 14
    NTLM or Kerberos 15
    Connection Authentication Lifetime 15
    Man in the Middle 16
    Forcing NTLM in Windows 2000 16
Microsoft versus RFC 1510 17
    DNS Functionality 17
    IP Packet Transport 18
Conclusion 18

Appendixes
    Windows NT Appendix
        NTLM Authentication (Figure 1) APP-NT 1

    Windows 2000 Appendix
        Windows 2000 Kerberos Authentication (Attachment 1) APP-2000 1
        Windows 2000 Kerberos Authentication (Figure 2) APP-2000 3

Abstract

The main feature of Windows 2000 touted by Microsoft is the improved security features of the operating system over prior versions of Windows. One of the primary enhancements being emphasized is the use of the Kerberos protocol for authentication purposes. This function is one of many steps in the security improvements of Windows 2000, but few users fully understand its true function. Kerberos is used only to validate that the users and machines accessing network resources are who they actually are based on their credentials. The process used to complete this function is complex and involves many steps from beginning to end. This paper provides a basic overview of the functionality of the Kerberos protocol in Windows 2000 and how it interacts with the various systems in a network environment.

Introduction

The world of computers is a constantly evolving environment, which continuously creates new challenges for the users. During the past decade the Internet has exploded not only in web pages, but also in the use of Internet connections to complete daily business transactions and communications. In the past, most network users worked on self-contained networks in a single location, or connected to the network by dedicated point to point communication lines. Within this type of environment an outsider would have to first infiltrate the network parameter before they could begin obtaining any type of information about the environment. The Internet has taken away this type of "protected environment" by the creation and use of the world wide web not to mention public networks.

Today, users connect to the Internet to submit their taxes, complete banking transactions, shop, and complete a wide variety of other tasks. This connectivity from the "outside" into the networks of businesses opened a huge hole in the previous defenses of company information. It was no longer necessary for the outsider to find their way into the network that had no direct connection to the outside world before they could obtain information, they simply needed to find their way through the Internet portal. This change in how company computers and networks were connected to each other and the Internet magnified the weaknesses in the previous Windows security features. One of these was the method which computers used to authenticate users.

Prior to Windows 2000 there were two basic authentication methods:

- Lan Manager or LanMan
- NT Lan Manager (NTLM)

When released, based on the environment they were created for, these methods worked well. The length of the password they could accept, how the hash was used in each, and the limitations of the uniqueness of the encryption used, limited the earliest

versions. Many of these shortcomings were overcome with later releases including no longer using the LanMan hash of the user password for authentication, use of a more advanced encryption system, and the addition of a timestamp to avoid the reuse of authentication information at a later time. There was also a utilitiy (SYSKEY.EXE) included with Windows NT Service Pack 3 which allowed the Security Accounts Manager (SAM) database to be encrypted on the domain controllers thus limiting the possibility of unauthorized access. However, even with these changes, access to network resources still required the authentication function be completed from beginning to end anytime the client accessed a different network resource.

During the development of Windows 2000, security was one of the main focus points. Kerberos v5.0 was integrated into the operating system and took the place of the prior authentication methods although NTLMv1 and NTLMv2 are still included with Windows 2000 to allow backward compatibility. Kerberos v5.0 has not fully removed the ability of attacks against networks, but through the use of mutual authentication it has greatly reduced the possibility of successful attacks being carried out.

Overview

In developing the authentication mechanism for Windows 2000 Microsoft deviated from their normal path and chose to use a non-proprietary protocol. Kerberos version 5 used by Microsoft is formatted according to RFC 1510 specifications. There are few deviations between the Microsoft version and the original version making integration with other operating systems much easier.

The Kerberos protocol uses many steps to assure valid authentication of users and minimize the chances of the information being compromised. In comparison, NTLM was not established to protect against outside attackers.

Based on information from the MSDN Library (Microsoft, NTLM) NTLM authentication in Windows NT consists of 7 basic steps:

1. A user accesses a client machine and provides a domain name, user name, and password. The client computes a cryptographic hash of the password and discards the actual password.

2. The client sends the user name to the server (in plaintext).

3. The server generates a 16 byte random number, called a *challenge* or *nonce*, and sends it to the client.

4. The client encrypts this challenge with the hash of the user's password and returns the result to the server. This is called the *response*.

5. The server sends the following three items to the domain controller: the user name,

the challenge sent to the client, and the response received from the client.

6. The domain controller uses the user name to retrieve the hash of the user's password from the Security Account Manager (SAM) database. It uses this password hash to encrypt the challenge.

7. The domain controller compares the encrypted challenge it computed (in step 6) to the response computed by the client (in step 4). If they are identical, authentication is successful.

Anytime a client access a different server (printing, file access, application, etc.) this entire process is complete from beginning to end. This can become a strain on the network, adds time necessary to access resources, and provides numerous opportunities where the authentication information could be obtained by another party and used for unauthorized access to the system. To review a graphical representation of the NTLM authentication steps above refer to Figure 1 in the Windows NT Appendix (APP-NT 1).

The authentication process used by Kerberos is much more complex. The entire theory behind the Kerberos protocol is "shared-secret authentication" which simply means the information needed to authenticate the client and the server is maintained as a secret by all parties to the transaction (Microsoft. Kerberos Authentication, 3). The secret information is used by the client and server to verify they are exactly who they say they are prior to allowing the completion of any connection between the two. Either party is capable of ending the communication if the authentication is not valid. This type of two-way authentication is an improvement over the prior NTLM authentication as there was no way for the client to authenticate the server was the correct connection. Only the server was able to authenticate the user was correct.

The Kerberos name was taken from Greek mythology (Microsoft. Kerberos Authentication, 5).

> "Kerberos (or Cerberus) was a figure in classical Greek mythology, a fierce, three-headed dog who guarded the gates of the Underworld. Like Kerberos the guard, Kerberos the protocol has three heads: a client, a server, and a trusted third party to mediate between them."

This description tells where the name comes from and also provides the three parties to Kerberos authentication. The client is the user attempting to access a network resource, the server is the resource, and the trusted third party is a service running on a domain controller known as the Key Distribution Center (KDC). The KDC is the location where all of the user's secret information is maintained so their identity can be authenticated. This secret information is known as the user's long-term key and is normally their current password after being hashed. The actual password is not sent across the wire (either in plain text nor encrypted) during the Kerberos transaction thus removing the chances of it being obtained for unauthorized use.

The user takes a piece of information, normally the system time, encrypts it using their long-term key and sends it to the KDC. The KDC pulls the user information and attempts to decrypt the information received. If successful the KDC will send this information back to the user encrypted, again with their long-term key, but will also include information encrypted with the KDC long-term key. At this point the user decrypts the information in the client section (obtaining the KDC section from it) and returns the KDC section to the KDC. This will show the KDC the user can access the encrypted information and should be authenticated to access the resource. The KDC then issues tickets to be used by the client to access the network resources needed. These tickets contain the information to not only allow the client to access the resource, but also to assist in the authentication process.

How Kerberos Works

This description of how Kerberos works assumes the environment is using the mutual authentication function between the client and server during all exchanges. This setting is a default during the installation of the Kerberos service and is one of the additional functions of the protocol to assure only authorized users are accessing the network resources.

The Kerberos protocol is made up of three sub-protocols, which complete the separate steps of authentication (Microsoft. Kerberos Authentication, 9).

1. Authentication Service (AS) - This service performs the first step of authentication when the client asks the KDC for a Ticket Granting Ticket (TGT) and the KDC provides it to the client.

2. Ticket Granting Service (TGS) - This is the second part of authentication when the client uses the TGT to request a ticket to the desired network resource and the KDC provides it to the client.

3. Client/Server (CS) - This is the third and final part of the authentication process where the client uses the TGS to contact the actual resource on the network and gain access based on the credentials it is able to present.

Obtaining the Ticket Granting Ticket

The first step in Kerberos is for the client to obtain a Ticket Granting Ticket (TGT) from the KDC. When the client logs onto their system and asks for a TGT, the Kerberos client will normally take the user's long-term key (derived from the hashing of the password entered) and encrypt the system time on the machine. It will take this information (user name and encrypted time) and send it to the KDC with a request to access a server. This request is named KRB_AS_REQ and is received and processed

by the AS service according to the following steps.

The KDC is a server which will also be one of the domain controllers for the network. Because of this it will have the Active Directory (AD) services installed on it which maintains all user account information. Because of the information stored on the KDC it must be in a physically secured location and local access extremely limited. When the KDC receives the request from the client for a TGT it will contact the AD to obtain the user password and obtain the hash from this information. At this point, the KDC will use the hash to decrypt the timestamp sent by the client. If the timestamp is valid (i.e., matches the network time within the specified parameters) the KDC will authenticate the client. At this point a ticket to obtain future tickets is sent to the client.

This initial ticket contains information which will allow the client to ask the KDC for future tickets to access other network resources without being required to go through the initial authentication process each time. It also contains information which the client and KDC will use for future communications during this network session and is split into two parts. The first part contains the information which will be used by the client and includes a logon session key to be used for all future transactions rather than the user's hashed password information. The second part contains the TGT information to be used by the KDC and also includes the logon session key. This second part is encrypted with the KDC long-term key (which is unknown to the client) and is embedded within the first part which is encrypted with the client long-term key.

When the client receives the initial ticket it decrypts the information using the previously hashed password and obtains the new logon session key. Since this will now be the key used for future communications, the client discards the previously hashed password information. In addition to the session key, this ticket also contains the TGT information the client needs to obtain future tickets. However, since the server encrypted it, the client does not have access to it and can only provide this portion of the ticket back to the server when needed. This type of information adds more security to the protocol showing the client was able to decrypt the ticket sent.

The TGT is considered nothing more than a ticket to the KDC. The KDC has no purpose other than the authentication and granting of tickets for other resources. The TGT is a basic ticket which is unique to the KDC and allows the client access throughout the current session to work with the KDC in obtaining future tickets. Notice throughout this process there has not been any information sent across the network lines which would allow someone monitoring the packets to steal the information and begin impersonating the user or KDC.

Once the client receives the TGT, they are now able to request the ticket to obtain access to the network resource they need.

Obtaining a Ticket

The TGT is used by the client to request a ticket to access the desired network resource or service. The TGT is sent to the KDC in a request named KRB_TGS_REQ which includes (Microsoft. Kerberos Authentication, 11):

- User name of requestor
- Authenticator encrypted with the current logon session key just issued (this is normally the workstation timestamp)
- TGT which will still be encrypted with the long term key for the KDC which only it can decrypt
- Name of the resource or service where access is required

When this request is received by the KDC it first decrypts the TGT with its own long term key, then uses the current logon session key to decrypt the authenticator information from the client. If the KDC can verify the client, it creates the ticket for the client to access the requested resource.

With this new ticket, several pieces of information are created. A new logon session key is created as the connection between the client and desired resource or service is wholly separate from communication with the KDC. At this point a ticket is created, again with two parts:

1. The logon session key for the session between the client and the desired resource or service
2. A ticket for the resource or service, which also includes the logon session, key for that session. This portion is encrypted with the resource long-term key to assure it is the only system which can access the information.

All of the information in 1 and 2 above is encrypted with the current logon session key information (the session between the client and the KDC) and sent to the client for its use.

Connecting to the Resource or Service

The client is now able to contact the resource or service which it is requesting access to by sending the ticket obtained. The server where the resource or service resides receives the ticket and works with the client workstation to completely authenticate the user prior to granting access.

The ticket used to access the server is known as the Kerberos Application Request (KRB_AP_REQ) and contains (Microsoft. Kerberos Authentication, 11):

- The authenticator (still normally the client timestamp)
- The session key for this new connection
- The ticket which was obtained in the TGS exchange with the KDC, and

- A flag indicating that mutual authentication is requested (which is an assumed setting for this description)

The server first decrypts the information contained in the KRB_AP_REQ using its own long-term key. Once the ticket is decrypted the server reviews the authenticator and compares the information to what is considered allowable parameters. If the authenticator is valid, the session key received is used to encrypt the authenticator and then it is returned to the client. At this point the server is assured the client is valid and is ready to initiate the desired connection. The client then decrypts this information, verifies the server is valid, and allows the connection between the two to commence.

There are numerous steps in the completion of Kerberos authentication, and more than twice as many as those completed during the NTLM authentication. To review a step by step listing along with a graphical representation of how the Kerberos authentication process works, refer to Attachment 1 (APP-2000 1) and Figure 2 (APP-2000 3) in the Windows 2000 Appendix.

What Else is Happening

There are several pieces of information being sent to different parties of the process in connecting to a resource or server. A primary ticket is created when the client requests the first ticket to access other resources, known as the TGT. The authentication process performed for the TGT must only take place once as long as the ticket remains valid. However, all of the subsequent tickets issued use a similar format for the information provided for the server and the client.

The Actual Ticket

In the sections above there have been numerous references to tickets and their uses in Kerberos. Only a very limited number of fields are being addressed, as these are the primary ones as they relate to the authentication process. However, the information contained in the entirety of the ticket contains all of the needed information to not only authenticate, but also to identify many other parts of information used by the protocol.

Tickets are broken into two basic sections; one for the information provided to the Client by the KDC and one to be used by the client to provide information to the resource or server the client is connecting to. The client portion of the ticket contains the information that they must have to connect to the desired resource. This includes the following fields, which are unique to the client: the Kerberos version, realm, and server or resource desired. There are also the following fields of the client portion where both the client and the server, during the authentication use the information: session key, flags, timestamp, start-time, end-time-renew-till (Microsoft. Kerberos Authentication, 13).

Figure 3 (APP-2000 4) shows a diagram of the structure of a TGS or AS Reply as presented in "Windows 2000: How it Works" and is the most comprehensive and comprehendible outline of the data locatable. The information in this diagram is slightly different than the narrative description provided in RFC1510 of the make up of the tickets used in the authentication which indicates there are separate tickets issued for each of the separate functions. The author feels that indicating all of the information available as though it was used in one single ticket is much easier to understand.

Ticket Lifetimes

Kerberos allows the lifetime of any ticket to be set as needed according to the environment. The setting for the TGT has a default lifetime of ten hours but can be changed as needed to the closest number of hours allowed (Microsoft. Kerberos Authentication, 17). If an installation uses the default setting for the TGT a user will normally only have to complete the initial phase of authentication once per day. However, when they log in the next day their system will have to re-authenticate with the KDC in order to be able to obtain future tickets.

During the time the TGT is valid for the user, their system maintains the information in a location known as the "credentials cache". This is a part of the volatile memory, which is erased when the power to the machine is interrupted. The credentials cache information is never stored on the disk in the machine or any other static memory location where it could be obtained and used without authorization.

The other tickets obtained during a logon session, those used to access other resources or services, follow similar guidelines. The environment establishes the lifetime of the ticket and allows the user to maintain that session information for the entire time it is valid. The session ticket default lifetime is ten hours (just as the lifetime of the TGT), but the time is set in minutes, not hours (Microsoft. Kerberos Authentication, 17). The maximum lifetime for a session ticket can be anything above ten minutes up to and including (but never more than) the maximum lifetime for the TGT.

The KDC has the final decision on the lifetime assigned to a ticket. This includes a start-time and an end-time, which is determined by the information provided by the client and the parameters set in the Kerberos policy for the environment. When the client requests a ticket they may (but are not required to) include a start-time. The request processed by the KDC must include an end-time for the ticket. If the client does not specify one, it will default to the result of the start-time plus the maximum lifetime allowed for a ticket.

The server tracks the allowable lifetime for a logon session rather than the user machine. This information is contained in fields which are part of the body of the ticket, but can be read by the client machine to allow it to manage its credentials cache. When a ticket is about to expire (whether a TGT or a session ticket) there is no

notification or message received by the user (Microsoft. Kerberos Authentication, 17). Only if the user machine presents a ticket which has expired, will an error message be received. At this point the client would then have to return to square one and complete the authentication process to obtain a new ticket in order to gain access to resource or service.

Other types of tickets

There are three types of tickets which are issued by the KDC when referring to the lifetime of the ticket. The basic ticket is described above. The other two types are Renewable tickets and Forwardable tickets.

Renewable Tickets

With the basic ticket it is only issued for a finite, one time period and cannot be reused or changed in any way. A renewable ticket is one issued with the intent of being able to change small parts of information on the ticket and reuse them for extended periods of time.

Renewable tickets are issued with the same information on them as the basic tickets, but one of the flags on the ticket informs the client it is renewable. When the ticket comes close to expiration (the ticket cannot be expired prior to this action) the client sends a request for renewal to the KDC. The KDC looks at the flags on the ticket to verify it is renewable as the first step. It then compares the end-time field on the ticket. This field is determined as described above where (if the client does not provide any other time span still within the acceptable parameters) the start-time for the ticket is added to the maximum allowable lifetime of a ticket to arrive at what the end-time for the ticket should be. The KDC then takes this information and compares it to the "renew-till" field in the ticket. This field indicates the cumulative life that a ticket can be reused prior to its complete expiration.

If the date of the request is prior to the end-time of the ticket and prior to the renew-till field, the KDC will reissue the ticket with a new session logon key.

The purpose of the renewable ticket is to reduce the risk of the original ticket information being obtained, compromised, and used by unauthorized individuals. During the transmission of the tickets across the network, it is possible that a user could intercept one and decipher the logon key being used by one party or the other. They could then use this key to impersonate that user and gain unauthorized access to the systems. By changing the session key at regular intervals, the session key would be continuously updated and amount to making the information obtained by the malicious users of no value to them.

Proxy Tickets

Proxy tickets are used in situations where a client is using a function where it needs to

connect to two separate servers to complete the required function, such as a system with an application and a SQL backend information table. The application may be run from the application server while the SQL table is maintained on a separate SQL specific server. A proxy ticket can be used by the client in order to complete this two-part connection.

The proxy ticket is utilized when the client wants server1 to access server2 impersonating the client. The client obtains the ticket from the KDC stating that server1 will be using the ticket received from the client to access server2. When the KDC sends the ticket to the client it has the proxy flag set in the flag section of the ticket so all of the systems involved understand the client requested the ticket and is simply allowing server1 to forward it to server2. The client then sends this ticket to server1 so that the entirety of the application (both application and SQL tables, etc.) can be accessed when needed using the client's access.

Forwardable Tickets

Forwardable tickets are very similar to proxy tickets in that they are utilized when the client needs to connect to two separate servers for some reason. The situation is the same as that described for proxy tickets, but the process is different. Forwardable tickets are utilized when the client wants server1 to impersonate them when communicating to the KDC (not to server2) in order to obtain a ticket granting access to server2.

The client sends a request to the KDC asking for a forwardable ticket to be sent to a specific server, in this case server1. When the KDC sends the ticket (a TGT) back to the client they in turn forward it to server1. The server will, in the future until the ticket expires, use the TGT to access other resources needed by the client. Server1 (impersonating the client) will send the TGT from the client to the KDC when asking for access to server2 so the client is truly the user accessing the systems.

Kerberos Policy Settings

The Kerberos protocol allows the administrator to establish a wide range of customized settings (policy) within the domain. All of the Kerberos policies are set through the KDC and apply throughout the domain. The main security function implemented in Windows 2000 was the implementation of the Active Directory which maintains all of the security settings for the applicable domain. The Active Directory (AD) is always maintained on a Domain Controller in the environment along with the KDC (Key Distribution Center). This allows the KDC to easily access the AD in order to obtain its settings and policy parameters. As with most of the settings within AD, the settings for the Kerberos policy can only be changed by members of the Domain Administrators group. This limitation is necessary to assure the protocol has the strength and integrity to perform its functions correctly.

According to the "Windows 2000 Kerberos Authentication" (17) white paper from Microsoft TechNet, the primary options in the policy are below along with their default settings:

- Maximum user ticket lifetime -- The settings are in hours -- the default is ten hours
- Maximum lifetime that a user ticket can be renewed -- The settings are in days -- the default is seven days
- Maximum service ticket lifetime -- Settings are in minutes -- the setting must be greater than ten minutes and less than the setting for *Maximum user ticket lifetime* -- the default is ten hours
- Maximum tolerance for synchronization of computer clocks -- The settings are in minutes -- default is five minutes

  NOTE: There are several other functions which require time synchronization to perform their functions correctly. Among these are the installation and functionality of the client components in Systems Management Server 2.0. A simple command in the user logon scripts will synchronize the clock on the workstation to the machine indicated in the command, provided the logged user has the appropriate rights to update the time.
- Enforce user logon restrictions -- when enabled the KDC validates every request for a session ticket by examining user rights policy on the target computer to verify that the user has the right either to *Log on locally* or to *Access this computer from network* -- the default setting is enabled, but the administrator may choose to disable to function to reduce network traffic and speed access to network services

Authentication

Unlike the policy settings for Kerberos, many of the authentication functions can be set to individual user accounts or even machines. The entire purpose of Kerberos is authentication, or to validate the user is actually who they are supposed to be by means of several different measures and checks.

Pre-authentication

The first of these checks is the pre-authentication performed by the server to verify the client is who they say they are based on information presented. The pre-authentication occurs during the initial exchange and is normally the time stamp of the client being sent to the server encrypted with the client's long term key. The server must decrypt the authenticator and verify it is within the acceptable parameters in order for the authentication process to continue.

In addition to being one of the first steps of authentication, the pre-authentication process is also one of the settings for the Kerberos protocol which can be established for users as necessary. The domain administrator can set pre-authentication to only apply to a few users, or to apply to most of the users and not apply to only a few of the

users.

Authentication Delegation

As stated previously, the entire purpose of Kerberos is to validate the user is actually who they are supposed to be. The only exception to Kerberos allowing the circumvention of only the true user accessing a resource is the delegation of authority. The use of forwardable tickets is an example of this exception. Kerberos will allow a client to obtain a ticket and provide it to a server to use in the name of, and appearing to be the client. This delegation setting is similar to the pre-authentication setting, as it can be as limited as needed. This particular function can be allowed for all users but a few, or vice versa.

DNS and Kerberos

According to the specification in RFC1510 (Section 8.2.1) the client should contact the KDC using only the IP address. This indicates the client must be able to resolve the name of the server to the actual IP address. The only manner which allows Kerberos to do this is the Domain Name Service (DNS). When a Windows 2000 client attempts to contact the KDC it will search for the IP address of the server in the DNS records. If the client is not able to locate the IP address for a KDC via DNS an error message is returned to the client indicating it cannot locate any such domain.

The Kerberos protocol in the Windows 2000 environment is set up to function with the KDC always being installed on one of the domain controllers. The domain controllers are automatically entered into the DNS records and registered in the DNS service locator records. In addition to being on the same server as the domain controller, this same server will also run the Lightweight Directory Access Protocol (LDAP) server which is also registered in the DNS service locator record (Microsoft. Kerberos Authentication, 19).

In realms other than Windows 2000, the functions of the server running the KDC are not as specific. In these instances, the clients may require having the DNS names for the KDC servers stored in the client registry. This will tell the Kerberos Security Support Provider (SSP) where to look for the DNS servers which will then resolve the name of the KDC from the client registry to the IP address used for communication (Microsoft. Kerberos Authentication, 19).

The configuration of a Windows 2000 client to attach to a network which is not using Windows 2000 for the domain, will require some specialized attention due to the way that the KDC relies on the DNS entries of the entire network in order to function.

Microsoft did realize this would be an issue and included additional information about the possible issues in ksetup.exe in the Support folder on the Windows 2000 CD.

The Credentials Cache

Throughout this paper the credentials cache has been mentioned in regards to where the user logon session key information is stored during any communication which takes place between two machines using Windows 2000 as the operating system utilizing Kerberos authentication. The credentials cache is where the client stores all of the information it receives from the KDC during the authentication procedure such as the tickets and session keys for various servers and connections.

The credentials cache is a section of the volatile memory where the information needed to authenticate the client and server connections is stored. This section is never written to a hard disk and anytime the machine is rebooted or the user logs off, the cache is erased by the machine. This avoids the possibility of another user being able to obtain the passwords for unauthorized use within the network.

The Local Security Authority (LSA) is the man behind the scenes in Windows 2000 in maintaining the services and programs are correctly started and running to complete each step of the protocol. This same description holds true for the credentials cache. The credentials cache is operated by the SSP which is run as a part of the LSA. The SSP is the dynamic link library (dll) which is used to operate the authentication protocol. One SSP is used for Kerberos and another for NTLM authentication, but both are always loaded when the machine is powered up. When a Windows 2000 machine is authenticating to another system using Windows 2000, the Kerberos authentication along with its SSP is used.

As the client obtains tickets and session keys, the LSA calls the SSP to manage the storage of this information in the credentials cache. As the tickets and or session keys expire, the SSP is used to renew or obtain new authentication information.

The LSA also maintains a copy of the user's hashed password in its memory. This allows the SSP to obtain the necessary information from the LSA when it is requesting tickets or renewals without requesting the client to re-enter the information. This password information is never stored to disk and is destroyed anytime the user logs off or the machine is rebooted. This is not the same as the hashed passwords used to access the computer or various services on the computer which are still stored in the registry in Windows 2000 as they were in Windows NT.

Authentication versus Authorization

The Kerberos protocol is only the authentication portion of the security features of Windows 2000. It does not complete any authorization services such as determining if a user has the proper rights assigned to access the file or resource for which they are requesting a ticket. In essence a client could obtain a ticket and go through all of the steps (obtaining the TGT, obtaining the ticket, forwarding the ticket, completing the authentication with the server) when the logged on user does not have the appropriate rights to access the resource, such as a network share. Kerberos does however, play a role in the determining the authorization for a user's access. In the ticket there is a space in the server portion where the KDC stores the information needed to determine if the user should be allowed access to the requested resource.

When the client requests the TGT the KDC queries the AD on the domain controller for the user's unique Security Identifier (SID). The SID is the number assigned to each user and will never be duplicated for any two users. Along with the SID, attributes are assigned to designate what groups the SID is a member of thus providing the list of the resources the SID is authorized to access. During the KDC query process, the user's SID along with all of the security attributes for the SID are returned and placed in the Authorization Data field portion of the TGT (Microsoft. Kerberos Authentication, 21). When the TGT is returned to the KDC requesting a ticket to a particular resource, the KDC is able to simply copy the information in this field to the ticket being sent back to the client. This alleviates the need of the KDC or the resource to perform the authorization query each time a new resource is accessed.

When the resource receives the ticket it not only uses the information in the ticket to authenticate the client is valid, but it also accesses the information in the authorization data field. The resource will compare this information to the access control listings for the resource and if a match is found (i.e., one of the client attributes matches the allowable attributes for the resource) the access to the resource is granted and the connection completed.

Kerberos Strengths and Weaknesses

The Kerberos version 5 protocol was developed in the mid-1990's to enhance the security of networks interconnected with the Internet. It was also developed to promote standardizing the authentication protocol used among the various operating systems in network environments such as Windows or Unix which are the leaders in network operating systems. The Kerberos protocol specification was originally outlined in RFC1510 in September 1993 and the Microsoft Kerberos implemented in Windows 2000 closely resembles the original specifications.

In Windows 2000, Kerberos is a vast improvement over the previously used NTLM authentication from Windows NT. There are numerous additional levels of authentication and the opportunity for a malicious user to be able to obtain session information and decipher it into a useful format while the session information is valid

have been greatly minimized. Even with these changes, there are still weaknesses in the protocol which have been identified, and others which may be viably used in the future to circumvent the authentication process.

NTLM or Kerberos

One of the greatest weaknesses is that the Kerberos protocol is only used when the client and desired resource are all using Windows 2000 (or other Kerberos compatible operating system such as Unix). If any of the systems are using an older version of Windows, such as Windows NT the authentication will revert to NTLM. Windows 2000 installs Kerberos, NTLM, and Schannel authentication protocol. When a client logs onto the domain, the SSP automatically negotiates with the client to arrive at the highest authentication protocol compatible with both parties to the communication.

This down coding of authentication makes the entire network only as strong as its weakest operating system. If a network uses all Windows 2000 operating systems but has a single Windows NT machine connected, a malicious user could target that machine obtaining the password hashes via the NTLM communications then begin impersonating the user. This exhibits that to fully realize the enhancements of the Windows 2000 Kerberos protocol the entire network would require upgrading to the Windows 2000 environment. This type of upgrade can be cost prohibitive based on the additional processor speed and memory requirements for Windows 2000, which could perpetuate the use of NTLM for a long period into the future.

Connection Authentication Lifetime

The Kerberos protocol performs an extensive process of authenticating that the client and resource are valid according to their true identities. Then the resource uses the authorization data in the ticket from the client to determine whether or not the client has the appropriate rights to access the resource prior to allowing the connection. In addition to these steps, the ticket granting the access to the resource is only provided with a finite lifetime to live. Once this lifetime has expired, the client would have to obtain a new ticket with a new session key in order to connect to the resource once again.

However, even with all of these precautions, there is one portion of the functionality of Windows 2000 which could create a potential risk to unauthorized connections to resources. After the client has been authenticated and authorized, the server allows the connection to be completed. Once completed this connection will remain constant until the client disconnects even if the ticket for the session expires (Microsoft. Kerberos Authentication, 14). The connection could (feasibly) be continuous based on this piece of information. If a malicious user were able to find this connection and somehow tap into the systems, they would be able to use the connection for an

undetermined period of time without any type of warning being visible.

This type of continuous connection could be the result of a user who must access an application which maintains a continuous connection to a table of information in a SQL database. The user would have to complete the authentication process to connect, but then the connection would be live for as long as they allowed it. In most environments the number of users who are not aware of the possible effects of walking away from a system currently connected to sensitive resources on the network are the majority. This lack of connection need verification could create issues in the future for exploitation.

Man in the Middle

The Kerberos protocol in Windows 2000 has implemented many enhancements in the security verification process to validate users and the appropriate access to be allowed. However, when all of the enhancements are reviewed and the possibilities for attacks explored, the main strength of the Kerberos protocol is to protect against a man-in-the-middle attack.

In a Windows NT environment a machine could be set up with a network interface card running in promiscuous mode and set so that all packets from certain locations could be viewed by this machine. The user could then use the information sent back and forth during the NTLM authentication process to impersonate the client to the server and the server to the client. By doing this they would obtain the same authentication information provided to the client by the server and be able to initiate a logon session with it.

In the Windows 2000 Kerberos protocol, this type of attack is not feasible. The machine could still be set up to intercept the packets that are being sent and the user could still review the contents of each packet. However, the current encryption key used by the client and the server is not sent across the network.. In addition to this the key to the session is changed multiple times with each using their own secret key to decipher the new information. Since the complete information needed to hijack a logon session is not sent across the wire at any time, the risk of these types of attacks are severely minimized.

Forcing NTLM in Windows 2000

As reviewed above the security of a Windows 2000 network is only as strong as its weakest operating system. However, there are theories that a malicious user could force a system to use a lower level of authentication thus opening the door to a variety of attacks. According to the specifications in RFC1510 (Section 8.2.1), the initial communication between the client and the KDC are to take place via port 88. This particular specification has also been followed by Microsoft in the Windows 2000 Kerberos protocol.

A SYN flood attack of false requests could be sent to the KDC against port 88 effectively closing the port. At this point the client would read the Kerberos protocol as not being available (as though the server were using Windows NT instead of 2000) and automatically downgrade the authentication used to NTLM. The malicious user would then have a host of options available to obtain the authentication needed to access the systems (Kurtz, et al. Hacking, 229).

Although the author was not able to find any documented cases of this type of attack against a network, the functionality of it exists in the layout of the protocol. As more networks are created using Windows 2000 and others migrate from Windows NT to Windows 2000 this may become one of the more prevalent attacks against the systems.

Microsoft versus RFC1510

The use of the Kerberos protocol is a massive departure for Microsoft. In the past the functionality of the operating systems, services, communications, and many other software aspects have not been known to closely resemble any industry standard to a great extent. During the early development of networks Microsoft, as well as all the other software developers at the time, were forced to create and recreate the wheel for each company product. Each one was trying to create the best functions and processes within the software so that the other manufacturers would lean toward adopting it as a standard. When Windows began its domination, Microsoft had "won the war" so to speak and could then simply create its own world.

This proprietary attitude in its operating system creation has led to many difficulties for environments where the operating systems range from Windows NT to Windows 2000 to Unix to Macintosh. With the implementation of the Kerberos protocol in Windows 2000 Microsoft has implemented a protocol which very closely resembles the exact specifications released in RFC1510. The intention was to allow the end-user to be able to easily integrate Windows 2000 servers into environments where there is a mixture of operating systems or to attach Windows 2000 workstations to network environments using different types of Kerberos protocol, such as Unix. The inter-operability between the Windows 2000 Kerberos and other operating system Kerberos has continued to be an issue in allowing this desired integration. However, Microsoft along with other manufacturers are continuing to develop software which will allow the systems to function together as intended (Fontana).

The most obvious deviations from the standard of RFC1510 are the result of how Windows 2000 stores and transfers information, and are not simply arbitrary changes.

DNS Functionality

RFC1510 (Section 8.2.1) specifies that the Kerberos protocol will send its transmission

using the IP addresses of the clients, servers, and resources. In Windows 2000 this is performed when the client initiates the request for the ticket, the first step would be to identify the IP address of the server to contact.

In a normal domain structure, the name resolution for the server would be stored in the DNS table. Since the client will be contacting the KDC this is a relatively simple process. In Windows 2000 the KDC is always installed on the same systems as the Active Directory services which are part of the functionality of the domain controllers. In addition to this all domain controllers in Windows 2000 are also considered to be Lightweight Directory Access Protocol (LDAP) servers. Both domain controllers and LDAP servers are automatically registered in the DNS servers. This allows the KDC to easily locate the address of the KDC from the name of the server (Microsoft. Kerberos Authentication, 19).

In a domain where a Windows 2000 client is accessing a network where the servers are using operating systems other than Windows 2000 the process may not be this simple. In these types of environments, the KDC may not be installed on the domain controller and may subsequently not be registered in the DNS entries. The only available option in these types of instances would be to enter the names for the KDC in each client's registry. This would allow the client to pull the information directly from their own memory and not require them to contact another server for the information.

IP Packet Transport

RFC1510 (Section 8.2.1) includes in its specifications how the client should connect to the KDC for communication purposes. It states the client should utilize User Datagram Protocol (UDP) and send the request to port 88 for the KDC to receive the information. The KDC is then to respond back to the originating port from the clients machine with its reply. UDP is a one way communication which precisely what is needed for the communications prior to the connection between the client and server.

However, UDP is limited in the amount of data it can transport in a single packet. If the data must be broken into multiple packets there is a high risk of loss or corruption due to the lack of any type of check system being built into the protocol. The maximum transmission unit, using UDP on an Ethernet frame is 1500 octets. This is assumed to be the normal maximum amount as most networks utilize the standard Ethernet frame.

The packets being transmitted in Windows 2000 Kerberos can easily exceed this maximum limit which indicated UDP would not be a viable option. In place of the UDP Microsoft instead chose to utilize the Transport Control Protocol (TCP) for its Kerberos protocol. The maximum limit for TCP is greater than that for UDP, which alleviates the packet size issues (Microsoft. Kerberos Authentication, 19).

Conclusion

When Microsoft began developing the new operating system, Windows 2000, they choose to enhance the security functions. This was a logical move based on the direction the network and computer worlds are moving. In the near future, it will be unusual to find anyone without some type of Internet access which could involve many different types of access needed to various interconnected networks around the world.

Rather than create a security system from scratch, as they had previously done, Microsoft chose to integrate a higher level of compatibility into this operating system. After reviewing various schemes for authentication they arrived at the decision to utilize the Kerberos version 5 protocol as outlined in RFC1510. The specifications provided in this RFC outlined the implementation of an authentication scheme which necessitated multiple levels of authentication prior to connecting to a resource and which periodically changed the information used to obtain the authentication. In addition to this the protocol eliminated the sending of "all the pieces of pie" across the network during the authentication process thus drastically reducing the possibility of a man-in-the-middle attack being successful.

The Kerberos version 5 protocol used in Windows 2000 is a vast improvement over the NTLM authentication protocol utilized in Windows NT. It is geared toward the new world where the separation between networks will become more blurred as the integration and use of the Internet continues to explode.

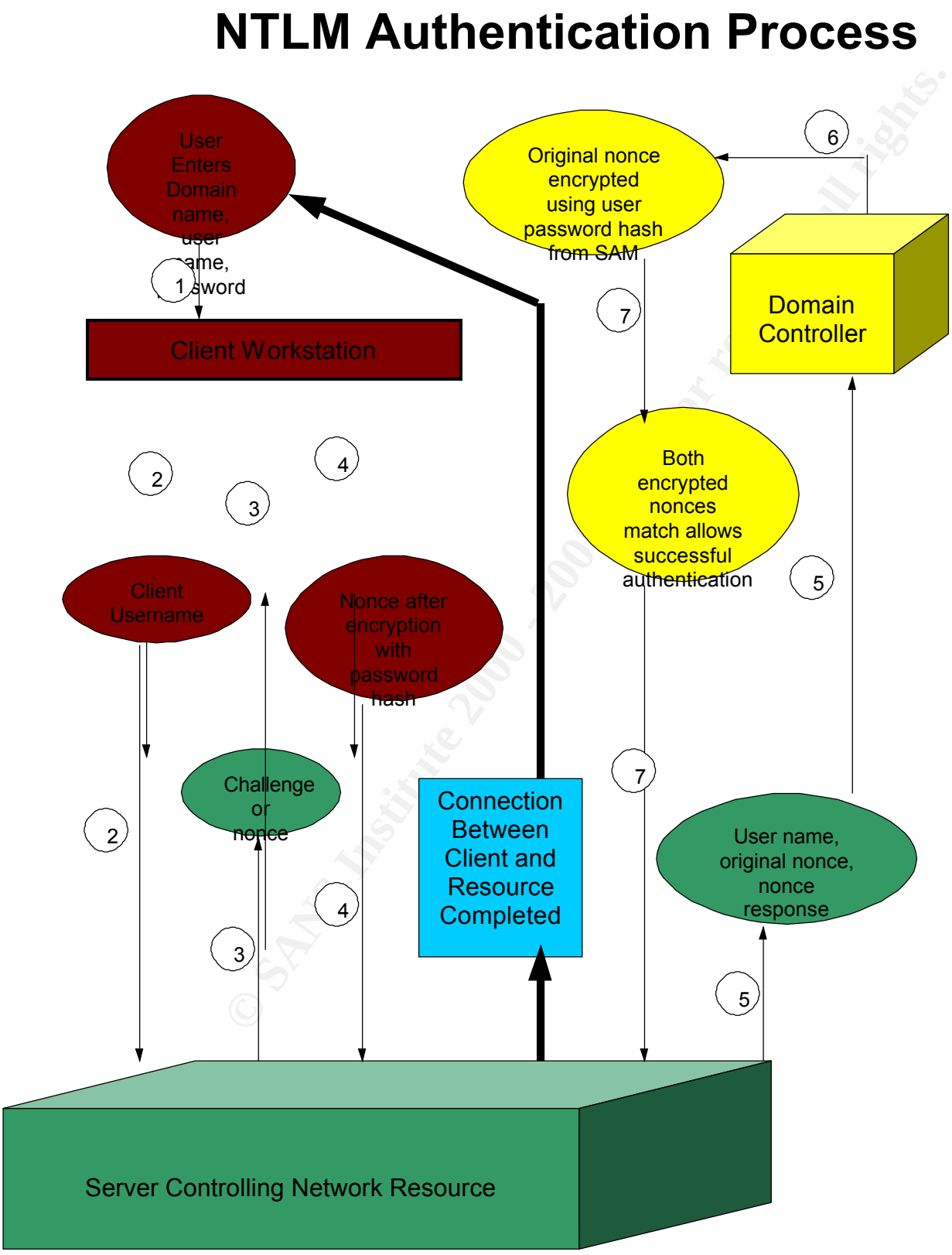
However, there are still issues to be addressed and considered. The Kerberos protocol is only utilized when all of the clients are using an operating system compatible with it. If all cannot utilize Kerberos, then the highest common denominator for authentication is utilized which could mislead many users into thinking their networks are safer than they truly are. Also, this is an indication that the only way to implement Kerberos protocol in a truly functional format (in a Windows network environment) would be to upgrade all systems to utilize Windows 2000 as the operating system. Although the upgrading of the operating systems would always be enjoyed, the cost of the software and hardware are very prohibitive to many organizations.

Perhaps as an alternative Microsoft could create patches and additional services or programs to be installed on other Windows systems (Windows95, Windows98, Windows NT, etc.) to allow them to update the authorization they use to the Kerberos protocol thus eliminating the need for the mass upgrade to realize the added security features.

Overall the Kerberos authentication is a vast improvement over prior modes of authentication and, for a Windows environment, its implementation should be considered as soon as it is truly feasible. Just as all things introduced, this authentication already has issues that could create security risks and as more users migrate to Windows 2000 more of these will be found.

Windows NT Appendix

Figure 1



Windows 2000 Appendix

Attachment 1

Windows 2000 Kerberos Authentication

This document contains a step by step description of the process completed in Windows 2000 Kerberos Authentication from the time a user logs onto the network until they are finally connected to the desired network resource.

1. Client sends a request encrypted using the user's long term key (KRB_AS_REQ) to the KDC asking for a TGT to request subsequent tickets.

2. The KDC receives the request and accesses the AD service to obtain the user password hashes then uses this information to decrypt the request.

3. Provided the information for the client is validated, the KDC creates:
   a) A new logon session key encrypted with the client's long term session key
   b) A TGT which contains information about the client. This section is encrypted using the KDC's long term key.

4. The KDC combines the information in 3a and 3b above together and returns this to the client (KRB_AS_REP).

5. The client uses its own long term key to decrypt the new logon session key, stores it in the credentials cache, and discards the long term key from the cache (this was previously used as the session key to the KDC).

6. The client stores the TGT received from the KDC in the credentials cache.

7. The client sends a request (the TGT) to the KDC asking for access to the network resource (this request is known as KRB_TGS_REQ). This is simply the client asking the KDC to issue a ticket to the client to gain access to the resource.

   This request contains the user name, authenticator (encrypted with the user's logon session key), the TGT, and the name of the resource the client wants to access.

8. The KDC decrypts the TGT using its own long term key and obtains the session key. The KDC then uses the session key to decrypt and validate the authentication information provided by the client.

9. The KDC creates a new session key to be used for the connection between the client and the desired resource.

10. The KDC creates a ticket for the client to present to the server. This ticket includes the session key for communication between the client and resource. The entire ticket is encrypted using the long term key belonging to the desired resource.

© SANS Institute 2000 - 2005 Author retains full rights.

11. The session key created in 9 above is encrypted using the current session key along with the authentication data.

12. The KDC combines the items created in 10 and 11 above (this is the KRB_TGS_REP) and sends them to the client.

13. When the client receives the information from 12, it decrypts it using the current session key to obtain the information in the reply. It stores the key for the new session in the credential cache along with the ticket for the desired resource.

14. The client sends a request (KRB_AP_REQ) to the desired resource.

    This request includes: the authenticator information (encrypted with the session key), the item created in 12, and the appropriate indicator for pre-authentication.

15. The desired resource receives the information from 14 and uses its long term key to separate the information parts.

16. The desired resource then uses the new logon session key to decrypt the authenticator information and validates the client.

17. If the client requests mutual authentication (which is assumed to be set in this environment), the original authentication data is encrypted with the current session key and returned to the client.

18. The client decrypts and compares the authentication data sent and received for the session. If the comparison validates the server is correctly identified, the connection is allowed to proceed.

Figure 2

Windows 2000 Kerberos Authentication Process

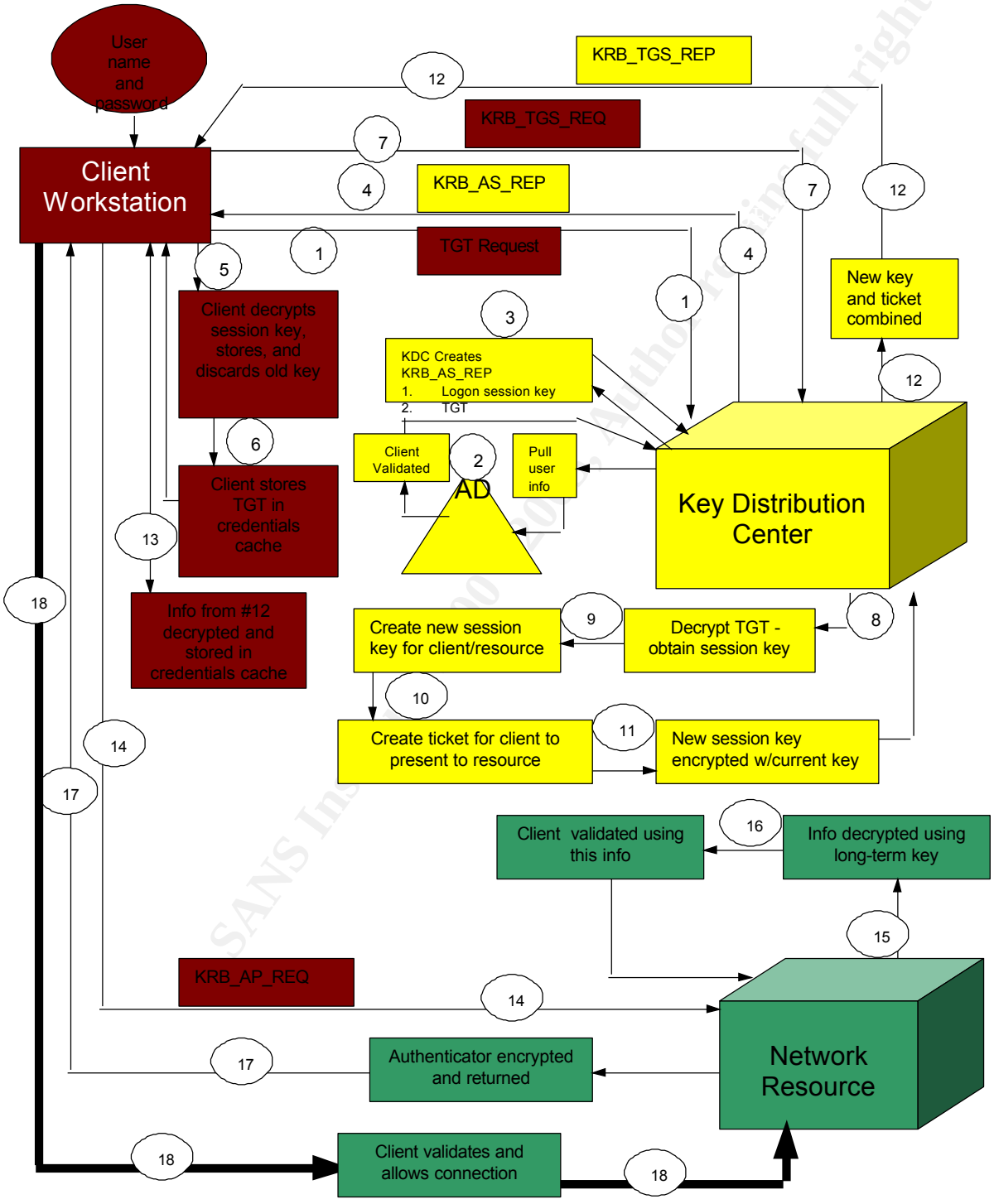


Figure 3

Structure of TGC or AS Reply

This Entire Reply Encrypted with Security Principal's PW Hash

<table>
<tr><th colspan="2">Client Information</th><th colspan="4">TGT or Ticket Information</th></tr>
<tr><td>Session Key</td><td rowspan="2">Ticket Version Number (V5 for W2K)</td><td>Session Key</td><td rowspan="2">Timestamp when ticket was issued</td><td rowspan="2">Authorization Data (used in W2K to hold SID of user/computer and group accounts)</td><td rowspan="2">Specific client address from which the ticket can be used (optional)</td></tr>
<tr><td>Flags</td><td>Client's Name</td></tr>
<tr><td>Timestamp</td><td rowspan="2">Realm (Domain) where ticket was issued</td><td rowspan="2">Client's Realm</td><td rowspan="2">Time after which ticket is valid (Start Time)</td><td rowspan="2">Flags
• Forwardable
• Forwarded
• Proxiable
• Proxy
• Renewable
• Initial (TGT)</td><td rowspan="2">Renew-Till (max interval during which ticket can be renewed) (optional)</td></tr>
<tr><td>StartTime</td></tr>
<tr><td>EndTime</td><td rowspan="2">Server for which the ticket is issued</td><td rowspan="2">Realms Transited</td><td rowspan="2">Time after which ticket is invalid (EndTime)</td></tr>
<tr><td>Renew-Till</td></tr>
</table>

- The TGT is encrypted with the password hash from the kerberos service account
- The Ticket is encrypted with the validating server's password hash
- The entire reply is encrypted with the password hash for the machine currently considered the client.

NOTE: This figure and information was obtained from Bill Boswell. Windows 2000: How it Works. May 2001. Baltimore, Maryland. SANS Institute.

References

Boswell, Bill. Windows 2000: How it Works. May 2001. Baltimore, Maryland. SANS Institute.

Fontana, John. "Microsoft, others target Kerberos interoperability". Network World February 7, 2000.
URL: http://nwfusion.com/archive/2000/86582_02-07-2000.html

Kohl, J., Neuman, C. RFC1510: The Kerberos Network Authentication Service (V5). Network Working Group. September 1993
URL: http://www.faqs.org/rfcs/rfc1510.html

Kurtz, George, McClure, Stuart, and Joel Scambray. Hacking Exposed, Second Edition Network Security Secrets and Solution . Berkeley. Osborne/McGraw-Hill. 2001

Microsoft Corporation. "Microsoft NTLM." December 5, 2000.
URL: http://msdn.microsoft.com/library/psdk/secpack/ntlmssp_0k19.htm

Microsoft Corporation. "Windows 2000 Kerberos Authentication White Paper." 1999.
URL:
http://www.microsoft.com/TechNet/prodtechnol/windows2000serv/deploy/confeat/kerberos.asp

Microsoft Corporation. "Windows 2000 Security Technical Overview White Paper." 2000.
URL:
http://www.microsoft.com/TechNet/prodtechnol/windows2000serv/deploy/confeat/sectech.asp

Walla, Mark and G Robert Williams. "Kerberos Explained." 2000.
URL:
http://www.microsoft.com/TechNet/prodtechnol/windows2000serv/maintain/kerberos.asp. (extracted from The Ultimate Windows 2000 System Administrator's Guide, Williams & Walla, Addison Wesley 2000)

Last Updated: July 28th, 2017

Upcoming Training

SANS Virginia Beach 2017	**Virginia Beach, VA**	**Aug 21, 2017 - Sep 01, 2017**	**Live Event**
SANS Network Security 2017	**Las Vegas, NV**	**Sep 10, 2017 - Sep 17, 2017**	**Live Event**
SANS vLive - SEC505: Securing Windows and PowerShell Automation	**SEC505 - 201709,**	**Sep 18, 2017 - Nov 13, 2017**	**vLive**
Secure DevOps Summit & Training	**Denver, CO**	**Oct 10, 2017 - Oct 17, 2017**	**Live Event**
SANS San Francisco Winter 2017	**San Francisco, CA**	**Nov 27, 2017 - Dec 02, 2017**	**Live Event**
San Francisco Winter 2017 - SEC505: Securing Windows and PowerShell Automation	**San Francisco, CA**	**Nov 27, 2017 - Dec 02, 2017**	**vLive**
SANS Cyber Defense Initiative 2017	**Washington, DC**	**Dec 12, 2017 - Dec 19, 2017**	**Live Event**
SANS Southern California- Anaheim 2018	**Anaheim, CA**	**Feb 12, 2018 - Feb 17, 2018**	**Live Event**
SANS OnDemand	**Online**	**Anytime**	**Self Paced**
SANS SelfStudy	**Books & MP3s Only**	**Anytime**	**Self Paced**