# Global Information Assurance Certification Paper

# GCNT Practical Assignment
Version 2.1b

# Kerberos Authentication in Windows 2000

August 16, 2001

Alfred Heiter

## Table of Contents

# 1 Introduction

Within a modern operating system which is used in a computer network consisting of many homogenous as well as inhomogeneous resources and where data with different security classifications is stored and processed, access control mechanisms which grant access to legitimate users and deny access to not legitimate users are required.

The first step in the process of access control is identification and authentication of users. If these two mechanisms do not work properly or may be compromised easily, access control will not be guaranteed.

In former versions of MS Windows (including NT 4.0 and 3.51, WfW, Windows 95 and 98) for authentication the lanmanager (LM) protocol is used. This protocol is based on a challenge response procedure, which uses some simple cryptographic algorithm that can be cracked easily using a dictionary or even brute force attack. In Windows NT 4.0 an improved version of LM authentication called NTLM (version 1 or 2) has been implemented which provides a more secure way for authentication.

For Windows 2000 Microsoft totally changed the authentication mechanisms of the operating system and by default uses the standard protocol Kerberos version 5. Kerberos is the preferred authentication method, LM/NTLM authentication is only used if the counterpart does not support Kerberos authentication, usually this is the case if Windows 2000 is used together with former versions of the MS Windows operating system (Windows NT 4.0, Windows 95/98 etc.).

# 2 The Kerberos Protocol

The Kerberos version 5 protocol is described in RFC 1510 [4], which was issued in September 1993. The implementation of Kerberos in Windows 2000 exceeds the standard as it adds some functionality (e.g. to avoid "denial of service" attacks) and uses some undefined fields (the authorization-data field in a Kerberos ticket) for OS-specific purposes.

The Kerberos protocol can be characterized and as following:

- Kerberos supports mutual authentication. This means, that not only the client has to prove its identity to the server but also the server has to prove its identity to the client. This is an improvement to other authentication protocols like NTLM where it is possible for an attacker to impersonate the server. [6]
- Kerberos supports secure authentication across an insecure channel like the Internet. Even if an attacker sniffs the network traffic or modifies the transmitted information during the authentication process he will not gain any secret logon information that enables him to impersonate any authorized user or compromise the logon mechanism. [4]
- Kerberos supports delegation of authorization. That means if a client accesses a service at a server and this service needs to access another service on a different computer (like in distributed applications) the first service can impersonate the client and act on its behalf. [6]
- Because Kerberos is an industry standard (RFC 1510 [4]) it is possible to integrate many different architectures with different operating systems. If all of these operating systems have implemented some sort of Kerberos authentication it is possible to authenticate each client on each server, independent of its architecture. This kind of interoperability exists in theory, in practice especially the Windows 2000 implementation of Kerberos contains some elements that make interoperability complicate.
- "Denial of service" attacks are not solved with Kerberos. It is possible for an attacker to disable either the Kerberos distribution center (which distributes the Kerberos tickets) or any other host within the network. Detection and solution of such attacks are left to the implementer or the administrator and user. [4] The implementation of Kerberos in Windows 2000 meets this problem partly in using some special pre-authentication. [6]
- The secret keys of the principals (users and services) must be kept in a secure place. If an attacker gains a principal's secret key, it will be possible that an attacker impersonates that principal. [4]
- "Password guessing" attacks are possible against messages that are encrypted with a key derived from the users password. If the password is weak a dictionary attack or even a brute force attack could crack the users password. Hence even with Kerberos authentication strong passwords that are changed regularly are mandatory for security. [4]
- The clocks on the hosts must be loosely synchronized, because timestamps are

used on the one hand for replay detection and on the other hand for the lifetime of Kerberos tickets. [4] Additionally if the clocks are synchronized over the network the clock synchronization protocol has to be secured.
- Kerberos was designed for single-user client systems regarding to the storage of client tickets. In a multi-user system the Kerberos scheme can be compromised by attacks of ticket stealing and ticket replaying. [13]
- The Keberos authentication model is vulnerable to brute force attacks against the key distribution center (KDC). If the system where the KDC is located is compromised, all authentication data of all users within the realm of the KDC is compromised. Hence the security of Kerberos authentication depends on the security of the KDC system. [13]

## *2.1 Client authentication*

Let's suppose user1 wants to send a message to user2. How can user2 be sure that the message really comes from user1? Using a shared secret that is only known by user1 and user2 can solve this problem. When user1 presents the secret to user2, user2 knows that the person who is sending the message must be user1 because only user1 and user2 know the secret. [4,6]

When communicating over a network the secret for authentication user1 against user2 cannot be sent over the network in plain text because the network is insecure by definition and anyone who is connected to the network could read and modify the secret during its transmission. To solve this problem the secret has to be hidden in that way user2 is able to read it but nobody else can read or modify the secret, at least a modification has to detected by user2. [4,6]

To put this in practice a secret key is used as secret. With the secret key some kind of information, is encrypted using an encryption algorithm like RC4 (as it is used by default by Windows 2000). The encrypted message is sent from the client to the server who decrypts it with its own secret key. If the decryption succeeds the server knows that the client is who it claims to be. To prevent an encrypted message to be resent a timestamp is included within the message. The server checks if the timestamp of the message represents a later time than that in the last message of the same client. Additionally the timestamp may not differ from the current time of the server's clock more than a distinct value (e.g. five minutes, the default value of Windows 2000 Kerberos). If this is true the server knows that the message originated from the client and has not been replayed. [4,6]

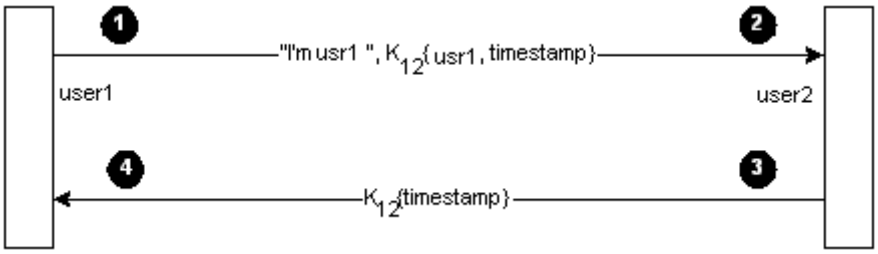

*Figure 1: client authentication and mutual authentication (Source: [6])*

## 2.2 Mutual authentication

To prove the client that the server is what it claims to be (or what it is supposed to be, respectively), the server encrypts a part of the message, which has been sent by the client with the shared secret key and sends it back to the client. The client decrypts the message and compares it with the contents of the message it had sent previously. If the contents match the client knows that the server is what it is supposed to be. To prevent replay attacks the server uses the timestamp information of the client to authenticate it. The server takes only a part of the message the client sent to prove its identity because if it returned the whole message it could just as well not decrypt the clients message an return it as it is (due to the fact that they are using symmetric key encryption). [4,6]

The information encrypted with the secret key that is sent by the client is called the authenticator. [4,6]

## 2.3 Key distribution (KDC and Kerberos Database)

The problem in authenticating each other is how the participants can exchange their secret key securely. Of course there is the possibility for people to meet each other and hand over the key, but this would not work if the two users are a client and a server program. [4,6]

To solve this problem the participants of a Kerberos authentication process rely on a trusted third party that issues a secret key for both parties to communicate. This third party is known as the key distribution center (KDC) and shares a secret key with every security principal within a Kerberos realm (a realm is the same as a domain in a Windows 2000 network), which is derived from the password of the user and is called the long-term key. [4,6]

Information about users (username, secret key etc.) is stored in a special database, the Kerberos database. Each record of the database contains information about one principal (the principal's identifier, the principal's secret key, the key version, the maximum lifetime for tickets and the maximum lifetime for renewable tickets) in the course of which one principal may have more than one entry. This may occur because

if a principal changes his password tickets based on the old password should remain valid until their specific lifetime. The Kerberos database may reside on another machine than the Kerberos service itself, folowing RFC 1510 this is not recommended because system management and threat analysis would become quite complex. [4,6]

If a client wants to access a server application (i.e. the client wants to authenticate against the server) he contacts the Kerberos authentication service (AS, in Windows 2000 the authentication server and the KDC are the same server, namely the primary domain controller) to get a session key (a secret key) for communication with the server. The request consists mainly of the username and a timestamp, which is encrypted with the user's secret key. The KDC decrypts the message, checks the validity of the timestamp and if the timestamp is valid the KDC sends a session key to the client and to the server. The method of sending an encrypted timestamp for verification is called preauthentication. [4,6]

To take precautions if the KDC cannot be reached for some reason, additional KDC(s) should be setup within a realm. One KDC acts as the master KDC; all other KDCs are the slave KDCs. Synchronization of the Kerberos database of all the KDCs must take place within appropriate time and securely. [4]

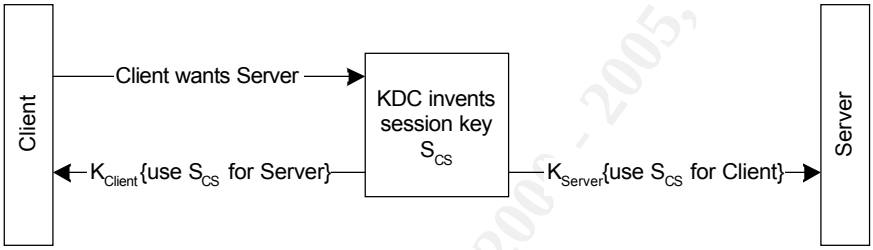


*Figure 2: Key Distribution Center (Source: [6])*

## *2.4 Session tickets*

Actually the KDC does not send the session key to both the client and the server but it sends the session key to the requesting client twice, first by encrypting it with the client's long-term key and second by encrypting it with the server's long-term key. The client is able to decrypt the session key as it was sent the first time but will not be able to decrypt the second one. When the client tries to open a session with the server it sends its own authenticator together with the encrypted session key it got from the KDC. The server decrypts first the session key with its own long-term key, receives the session key in that way and afterwards decrypts the authenticator with the session key to authenticate the server. Because the session key has been encrypted with the servers long-term key the server knows that it must have been encrypted by the KDC because only the server and the KDC know the long-term key of the server. [6]

The encrypted session key for the server is called session ticket. The session ticket not only contains the session key but also information about ticket lifetime and

authorization. The authorization data is not standardized by the Kerberos protocol as described in RFC 1510 but may be implemented individually by the vendor. In Windows 2000 Microsoft uses the authorization field to include the SID of the user as well as the SIDs of all groups the user belongs to. [4,6]

This technique (the usage of session tickets instead of sending the session key both to the client and the server) improves performance of the server in that way, that the server does not need to keep the session key stored until the client tries to connect to the server.

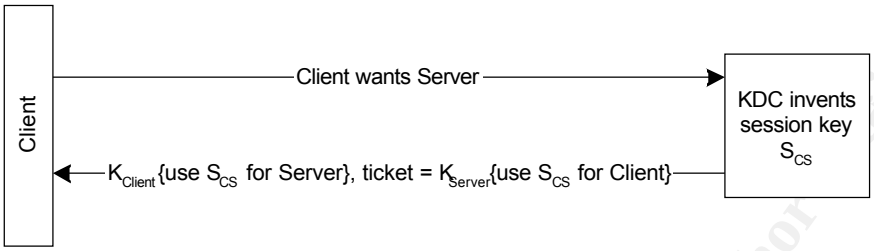


*Figure 3: Session tickets (Source: [6])*

## 2.5 Ticket granting tickets

The authentication process takes an amount of resources at the KDC, because every time the client wants to access a server he has to get a session key from the KDC and therefore he has to authenticate to the KDC. [4,6]

To circumvent this repeated authentication the client authenticates just one time (i.e. when the user logs on to the realm) and receives a ticket granting ticket (TGT) from the KDC. The TGT looks like a session ticket but is used for authentication when the client tries to obtain a session ticket from the KDC. [4,6]

When the user logs on he sends a nonce (a random number) encrypted with his long-term key to the KDC who sends back a TGT encrypted with the KDC's long-term key. When the user tries to obtain a session ticket to access a server within the realm of the KDC he sends the request for the session ticket together with the TGT to the KDC. The KDC checks the validity of the TGT and sends back the session ticket for accessing the corresponding server. [4,6]

## 2.6 Data structure of tickets

Every ticket (session tickets and TGTs) consists of an encrypted an a plaintext part. The encrypted part is encrypted with the server's secret key.

The unencrypted part includes the following fields: [4]

| | |
|---|---|
| Tkt-vno: | The version number of the ticket format (5 for Kerberos version 5) |
| Realm: | The name of the realm that issued the ticket |
| Sname: | The name of the server |

The encrypted part includes the following fields: [4]

| | |
|---|---|
| Flags: | Ticket options |
| Key: | The session key |
| Crealm: | The name of the client's realm |
| Cname: | The client's name |
| Transited: | A list of the Kerberos realms that took part in authenticating the client to whom the ticket was issued |
| Authtime: | The time of the initial authentication. In this field the KDC places a timestamp when he issues a TGT. When is issues a ticket based on the TGT the authtime is copied in the session ticket. |
| Starttime: | The time after which the ticket is valid |
| Endtime: | The time before which the ticket is valid (expiration time) |
| Renew-till: | If the ticket is renewable, the time before a ticket may be renewed |
| Caddr: | Addresses from which the ticket can be used. This field is optional |
| Authorization-data: | Data representing the authorization level of the client. This field is not specified by the Kerberos standard. |

The flags-field indicates which of the various options were used or requested when the ticket was issued. It contains 32 bits where a set bit (1) indicates that the corresponding option is selected and a reset bit (0) indicates that this option is not selected. Following the Kerberos standard in RFC 1510 11 of the 32 bits are defined, in Windows 2000 only the following six of these options are supported: [4,6]

| | |
|---|---|
| FORWARDABLE: | This option tells the KDC that a new TGT with a different network address based on the present TGT may be issued. (TGT only) |
| FORWARDED: | Inidcates that a TGT has been forwarded or a ticket was issued from a forwarded TGT. |
| PROXIABLE: | Tells the KDC that tickets with a different network address than the TGT may be issued. (TGT only) |
| PROXY: | Indicates that the network address in the ticket is different that one in the TGT that has been used to obtain the ticket. |
| RENEWABLE: | Indicates that the ticket may be renewed (in conjunction with the renew-till field) |
| INITIAL: | Indicates that the ticket is a TGT (TGT only) |

## *2.7 Timestamps, time to live and renewable tickets*

To reduce the risk that a ticket is compromised, every ticket is valid only for a distinct amount of time. The timestamp in the field starttime in the ticket defines, that a ticket is valid only after this time. The timestamp in the field endtime in the ticket indicates the time after that the ticket has expired (time to live). If a server receives an expired ticket from a client he rejects the ticket and sends an error message to the client. The client has to contact the KDC and obtain a new session ticket or to renew the expired session ticket. [4]

When requesting a ticket the client can specify an endtime for the ticket. The KDC sets the endtime to the minimum of the following: [4]

- The requested expiration time.
- The ticket's start time plus the maximum allowable lifetime associated with the client principal.
- The ticket's start time plus the maximum allowable lifetime associated with the server principal.
- The ticket's start time plus the maximum allowable lifetime set by the policy of the local realm.

If the RENEWABLE flag is set the ticket may be renewed. When a client asks the KDC to renew a session ticket, the KDC checks if the time in the field renew-till has not arrived yet. If this is true the session ticket is renewed in that way that a new session key is generated and inserted into the ticket and the fields starttime and endtime are updated. The value in the field renew-till remains unchanged because this value indicates the time when it is no longer possible to renew the ticket. After this the whole ticket has to be obtained including the authentication phase (either using the long-term key for a TGT or using the TGT for a session ticket). In Windows 2000 only TGTs are renewable. [4,6]

If an attacker gains a ticket and tries to crack it to obtain the session key, he will take so much time that the ticket already has expired when he gets the key and the session key has become invalid.

If a ticket expires while the session is still alive the session will not be interrupted due to the ticket expiration. The ticket must be renewed or obtained again just before a new connection is to be established. [6]

## *2.8 Delegation of authentication*

In the situation a client tries to access a service which to perform its tasks has to access another service, additional authentication and authorization to the second (back end) service is required. Using Kerberos authentication this could be done by the client requesting a session ticket for the back end server and passing this ticket to the back end server for authentication. [4,6]

This way of authentication would extinguish the advantages of the Kerberos protocol an the technique of using session tickets. To solve this problem two additional types of tickets have been invented, proxy tickets and forwardable tickets. [4,6]

If the Kerberos policy allows proxy tickets, the KDC sets the PROXIABLE flag when it issues a new TGT to a client. If required the client presents the TGT with the PROXIABLE flag set to the KDC and requests a proxy ticket (i.e. a ticket with the PROXY flag being set) to pass this ticket to the front-end server. The front-end server itself uses the proxy ticket to impersonate the client, to authenticate the client to the back-end server and to access the back-end server using the client's access rights. [4]

The more flexible way would be to delegate the requesting of session tickets to the front-end server. Acting that way would release the client from caring about which back-end servers are used and which tickets have to be obtained. The Kerberos protocol realizes this task in issuing forwardable TGTs. Before accessing the front-end server the client requests a forwardable TGT from the KDC. If the Kerberos policy permits forwarding of TGTs, the KDC issues a TGT with the FORWARDABLE flag set and the client passes this TGT to the front-end server. [4]

The front-end server requests a session ticket for accessing the back-end server in the name of the client from the KDC by presenting the TGT of the client with the FORWARDABLE flag being set. The KDC checks if the FORWARDABLE flag is set (because the TGT comes from another network address than the client ones) and if this is true it issues a session ticket with the FORWARDED flag set which can be used by the front-end server to access the back-end server. [4]

## *2.9 Cross-realm authentication*

The Kerberos protocol also supports authorization for clients against servers located in other realms. If the realm is trusted, the KDCs of both realms share an inter-domain key. Once this trust has been accomplished, each KDC is registered as a security principal to the other's realm, and treat each other just as another service. [4,6]

If a client requests a session ticket for a server located in a different, trusted realm, the KDC replies by sending the client a referral ticket. A referral ticket is a session ticket for accessing a KDC (the ticket granting service) in a trusted realm. The client requests the session ticket for the target server from the KDC in the trusted realm by using the referral ticket. The KDC in the trusted realm accepts the request of the client because of the valid referral ticket and replies with the session ticket for the requested server. [4,6]

If there are more than two realms, the process of accessing a server in a remote realm becomes more complicated. It would be possible to establish a trust between every two realms, but for example with five realms this would mean ten trust relationships. [4,6]

To solve this problem, the ticket granting service of a remote realm can be accessed using one or more intermediate ticket granting services. Realms typically should be organized hierarchically, where each realm shares an inter-realm key with its parent and a different key with each of its children. Using this technique a client that wants to access a remote realm using one intermediate KDC receives a referral ticket for the KDC in the intermediate realm where he can get a referral ticket for the KDC in the target realm. This is called transitive trust relationship. If distinct inter-realm access is used very frequently it is possible to bypass the hierarchical structure and establish a direct trust relationship by exchanging an inter-realm key with the KDC of the opposite realm. [4,6]

It may be possible that the server or KDC in the target realm needs to know the original realm of the client as well as all intermediate realms. To deliver this information the name of every realm that took part in a Kerberos ticket granting process is included in the session ticket that grants access to the target server. [4,6]

## *2.10 The subprotocols of Kerberos*

The Kerberos protocol is divided in three subprotocols that cover all the functions described above: [4]

First there is the authentication service exchange (AS), which covers the initial authentication against the KDC and the reception of a ticket granting ticket or a session ticket.

The second subprotocol is the ticket-granting service exchange (TGS). This protocol is used when the client wishes to obtain a session ticket or a proxy ticket or renew a ticket from a KDC. To get one of theses tickets (i.e. performing a TGS exchange) the client needs a valid TGT for the KDC, which has been obtained performing an AS exchange. The authentication service and the ticket granting service usually are situated on one server although this is not required.

Third there is the client/server authentication exchange (CS), which is used to access a distinct server. Before a CS exchange can take place, the client has to obtain a session ticket for this particular server from the KDC via a TGS exchange.

## *2.11 Encryption*

Tickets as well as the authenticator contain as described above an encrypted part of information. The Kerberos protocol assumes that the encryption algorithm used is strong enough that an encrypted message cannot be cracked within an appropriate amount of time. While in Kerberos version 4 the encryption algorithm was restricted to DES, in Kerberos version 5 the sort of encryption algorithm used is left to the

implementor.

RFC 1510 specifies encryption systems that consist of an encryption algorithm, block chaining methods and checksum methods. Additionally the use of a confounder is recommended and the method how to derive the user's secret key from it's password is defined.

Although any encryption system can be used the following are defined in RFC 1510: [4]

- NULL encryption system, which means that no encryption is used at all.
- Data encryption standard (DES) in cipher block chaining (CBC) mode with CRC-32 checksum (des-cbc-crc), which is not recommended because CRC-32 is not collisionproof.
- DES in CBC mode with MD4 checksum (des-cbc-md4)
- DES in CBC mode with MD5 checksum (des-cbc-md5)

Due to the low security of DES regarding to toady's technology (DES uses a 56 bit key which can be guessed by a brute force attack within not far too much time – assumed there is enough CPU power accessible), the Kerberos implementation of the Massachusetts Institute of Technology (MIT) in version 1.2 supports triple-DES for all keys.

Windows 2000 uses a proprietary encryption algorithm based on MD4 and RC4 but also supports des-cbc-crc and des-cbc-md5.

## *2.12 Transport protocols*

Although Kerberos may be implemented upon any transport protocol, the internet protocol (IP) would be used mostly in practice. When using IP transport, RFC 1510 defines to use the user datagram protocol (UDP, Port 88) at least for communication with the KDC. Windows 2000 deviates from that norm and uses the transmission control protocol (TCP), because in Windows 2000 the authorization data field in the tickets, which is not defined by RFC 1510, is filled with the user's SID as well as with the SIDs of every group the user is a member of. For that amount of information a UDP packet is too small, therefore TCP is used. When interacting with Non-Windows 2000 computers, always UDP is used. [4,6]

## *2.13 Security considerations*

The Kerberos protocol provides a secure mechanism for authentication but nevertheless there are some security issues that should be observed when implementing Keberos.

- Although Kerberos uses secret keys and session keys for encryption, the secret

keys are derived from user passwords and therefore are only just as strong as the passwords. Choosing a weak password will undermine all other advantages of encrypted authentication and the benefit of using Kerberos will decrease.

- The Kerberos authentication process takes much amount of computer power, just as for legitimate users who will get a TGT as well as for illegitimate users who won't be granted access to any resource within the realm. Therefore an attacker could perform a Denial of Service attack by permanently sending wrong security credentials that will be processed by the KDC and rejected. Legitimate users will not be able to access the KDC services and subsequently will not be able to access the resources on the network. [4]
- Kerberos uses timestamps for several purposes. They are used for preauthentication as well as for defining ticket lifetimes. Therefore it is necessary that all security principals of a Kerberos realm synchronize their clocks. If a time service is used for synchronization, this service is critical and should be protected against failure.
- It could be useful not to define the network address of the KDC server at every client but to use DNS with srv records to locate the KDC service within a domain. If the DNS server is compromised or impersonated by an attacker, the srv record could point to a fake KDC. To prevent this the DNS server should be protected sufficiently.
- Performing the three subprotocols of Kerberos (AS, TGS, CS) the user's password is never sent over the network, neither in plain text nor encrypted. But if the user wants (or has) to change his password the new password somehow must be sent over the network to the KDC. When implementing Kerberos a sufficient secure mechanism must be installed to perform password changes as well as replication of the user database between the KDCs of a realm.

# 3 Implementation of Kerberos in Windows 2000

Within Windows 2000 Kerberos version 5 is used as the default authentication method. Only if the opposite does not support Kerberos, another authentication method (LM/NTLM) is used. Especially if Windows 2000 Workstations are used within a Windows NT 4.0 domain, the use of Kerberos authentication would be omitted.

## *3.1 Components of Kerberos in Windows 2000*

**KDC and realm**

A Kerberos realm equals a domain in Windows 2000. Every domain controller within a Windows 2000 domain acts as a KDC. There is no distinction between a master KDC and a slave KDC, except the assignment of special domain controllers for flexible single master operations (FSMOs), which may also affect the KDC service.

**Kerberos database**

The KDC uses the domain controller's Active Directory as its account database and

provides the two services authentication service (AS) and ticket granting service (TGS). A copy of the account database resides on every domain controller within a domain, changes (password, user names etc.) can be made on every replica of the database, it will be propagated to all other domain controllers. Windows 2000 does not implement the Kerberos replication protocol but uses a proprietary multi-master replication protocol.

**The krbtgt account**

The KDC service on the domain controller uses the krbtgt account to take part in the authentication and ticket granting process. When the KDC issues a TGT to a client it encrypts the TGT with its own secret key derived from the password of the krbtgt account, therefore the KDC's security principal name is krbtgt. All KDCs within a domain use the domain account krbtgt. This account cannot be deleted or modified (except password changing). [6]

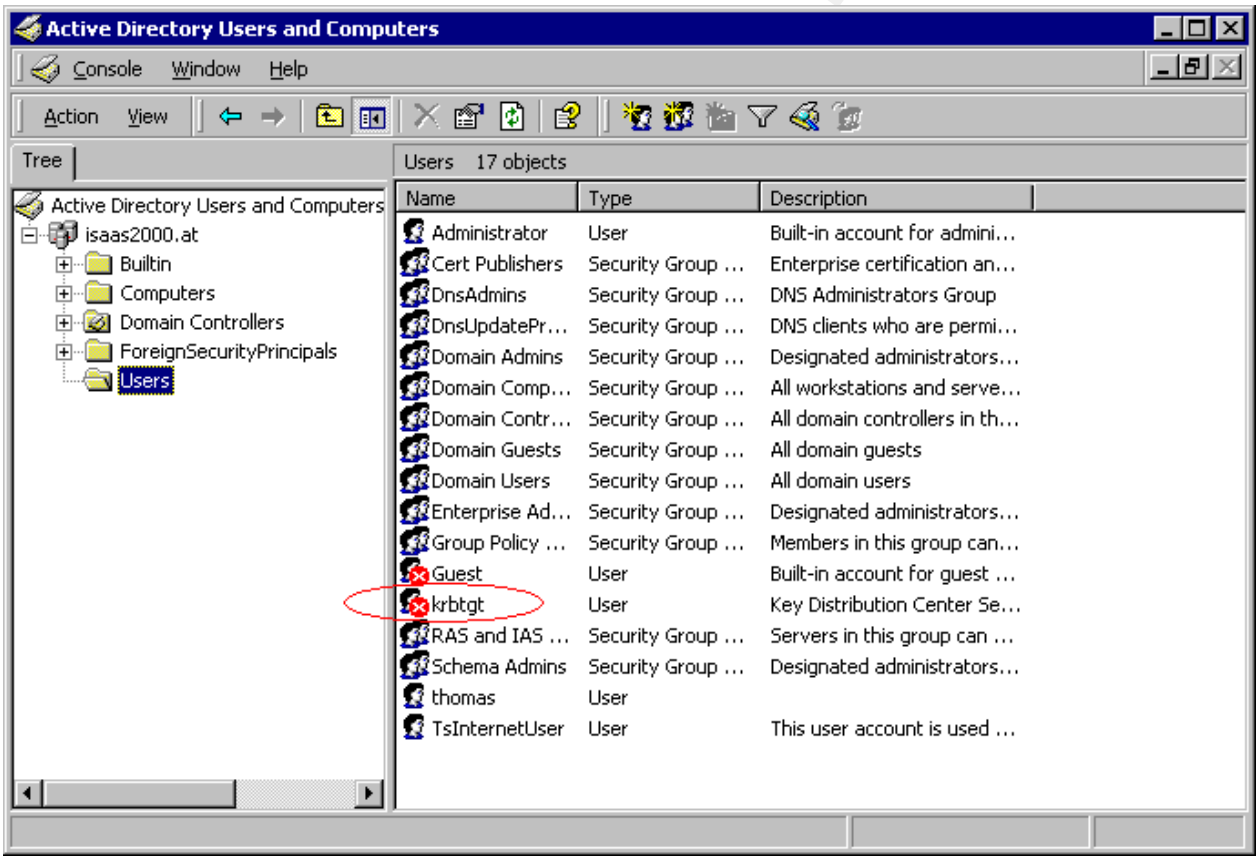


*Figure 4: The krbtgt account*

**Credentials cache**

Secret keys as well as session keys are not only stored on the KDC (Active Directory, only the secret keys are stored) but also at the client in a special place in memory, the credentials cache. The credential cache contains the user's secret key (derived from the logon password) as well as the secret keys for the client/server exchange. The credential cache is initialized when the user logs on to the domain and destroyed when

he logs off again. [6]

**Authorization**
As described above the Kerberos protocol is used just for authentication, not for authorization. Nevertheless within a ticket there is one data field reserved for authorization data but the structure of this field is not specified. Windows 2000 makes extensive use of this field. When issuing a TGT, the KDC inserts credential data of the user, especially the user's SID as well as the SIDs of all domain groups and – in a multi-domain environment – the SIDs of the universal groups the user belongs to. When the user requests a session ticket, the KDC of the server's domain copies the authorization data from the TGT to the session ticket. Additionally it looks up the Active Directory of the server's domain and adds the SIDs of any domain groups the user is a member of. [6]

As mentioned above the Kerberos protocol is defined to use UDP. If used on an Ethernet network a UDP datagram should not exceed the length of 1500 bytes, which is the maximum transmission unit (MTU) for an Ethernet frame. The authorization data in Windows 2000 can easily total more than 1500 bytes, thus the Transmission Control Protocol (TCP) is used. That's not true, if the Kerberos message is smaller than 2000 bytes. For Kerberos messages smaller or equal than 2000 bytes UDP is used (if the message is greater than 1471 bytes, the message is fragmented). Especially if used in conjunction with other Kerberos implementations, only UDP is used for communication. [6]

**Cross realm authentication**
Cross realm authentication is supported by Windows 2000 Kerberos. As already known in Windows NT 4.0 trust relationships may be established between two domains. In Windows 2000 all domains within a tree trust each other by default (by using a hierarchical transitive trust relationship), additional trusts to other domains or trees may be established. By establishing a trust relation ship, the KDCs of the trusting domains exchange their secret keys and thus cross realm authentication can be performed. [6]

**Smart card login**
Windows 2000 implements a public key extension to the Kerberos protocol. In standard Kerberos logons, users prove their identity to the KDC by encrypting preauthentication data with their secret key that is derived from their password. The same key is used by the KDC to decrypt the data, therefore this encryption mechanism is a symmetric one.

When using smart cards to logon, instead of the preauthentication data the user's public key certificate, which is stored on the smart card as an X.509 v3 certificate together with the user's private key is used for preauthentication. The KDC validates the certificate, extracts the public key and encrypts a logon session key, which is sent together with a TGT to the client. [6]

## *3.2 Configuring Kerberos in Windows 2000*

Kerberos is installed automatically in Windows 2000. By installing Active Directory on a domain controller it is set up as a KDC. There are not many settings that can be applied and the default settings usually are good. [9]

**Group policy**
In the group policy the following settings are available in Computer Configuration/Windows Settings/Security Settings/Account Policies/Kerberos Policy: [9]

- *Enforce user login restrictions*: This option forces the KDC to check if a user requesting a session ticket for a server has the right "Logon locally" or "Access this computer from the network". This should prevent disabled accounts to obtain new session tickets. The TGT expires after the amount of time set in the "lifetime for user ticket" option (10 hours by default) and this option should close the window of opportunity. The default value for this option is "Enabled".
- *Maximum lifetime for service ticket*: Determines the time (in minutes) a Kerberos session ticket is valid. The default value is 600 minutes.
- *Maximum lifetime for user ticket*: Determines the time (in hours) a TGT is valid. The default value is 10 hours.
- *Maximum lifetime for user ticket renewal*: Determines the time (in days) within which a user's TGT can be renewed. The default value is 7 days.
- *Maximum tolerance for computer clock synchronization*: Determines the maximum difference between the KDC's and the client's clock time. The default value is 5 minutes.

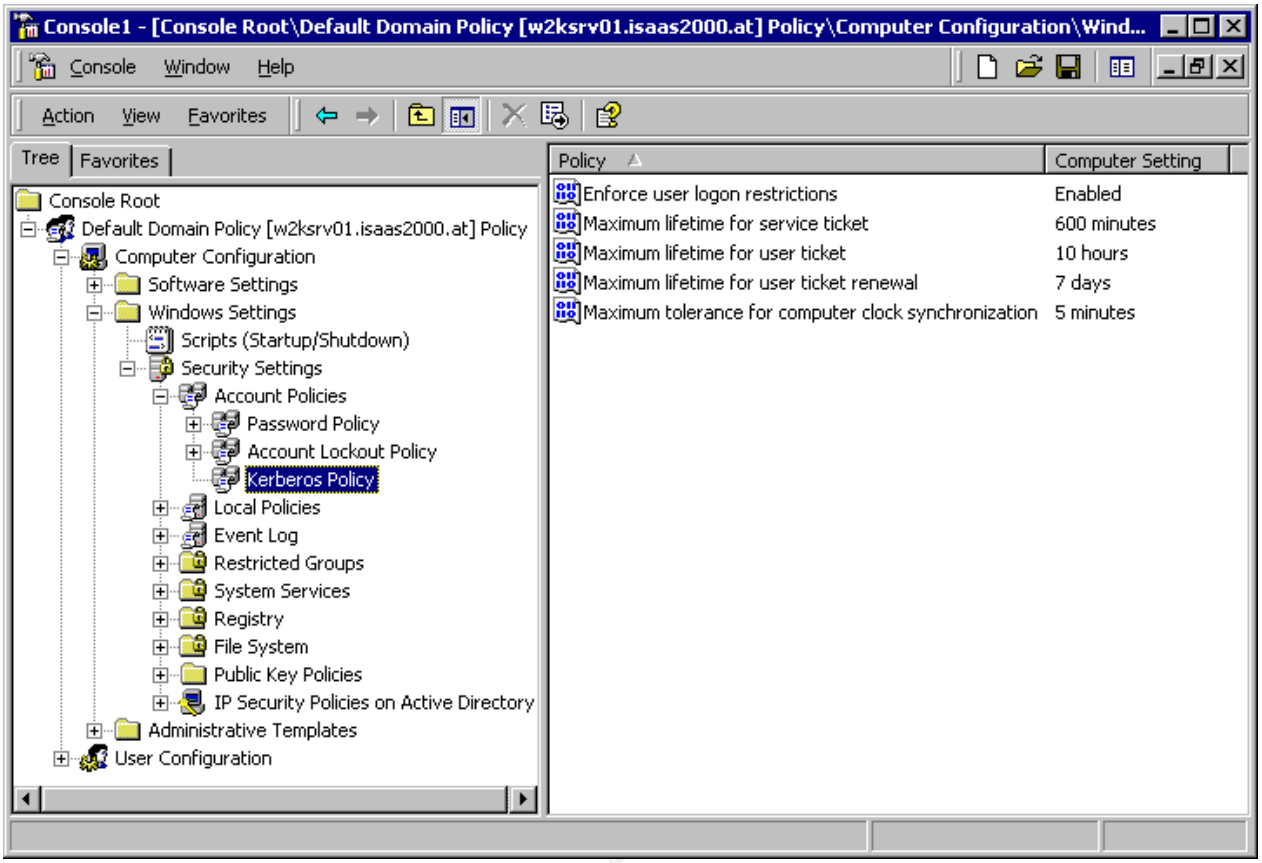

*Figure 5: Kerberos settings in group policy*

The default settings match the recommendations in RFC 1510 about KDC settings and are also recommended by the NSA as described in [9].

**User properties**

In the properties of an Active Directory user some options affecting Kerberos authentication are available (shown in the "Account" tab): [9]

- *Smart Card is required for interactive logon*: This option requires the user to use a smart card for logon.
- *Account is trusted for delegation*: If a service runs under the user's account, it is enabled to forward the user's Kerberos tickets. This option should only turned on for accounts that run services which need this feature.
- *Account is sensitive and cannot be delegated*: This option entails that a user's credentials cannot be forwarded.
- *Use DES encryption for this account*: DES encryption instead of RC4 encryption is used for this account. This option is used for interoperability requirements of some older implementations of Unix Kerberos. Since DES only uses 56 bit keys (instead of 128 bit like RC4) this option does not provide secure encryption and is not recommended.
- *Do not require Kerberos pre-authentication*: If preauthentication is used (which is the default in Windows 2000 Kerberos) the KDC only sends a response if the

preauthentication data is valid. An attacker who wants to guess passwords has to sniff the logon data for each user to collect data for password guessing. Without preauthentication the attacker could send a message to the KDC pretending he is a legitimate user and use the response message to guess the user's password.

All of these options are disabled by default.

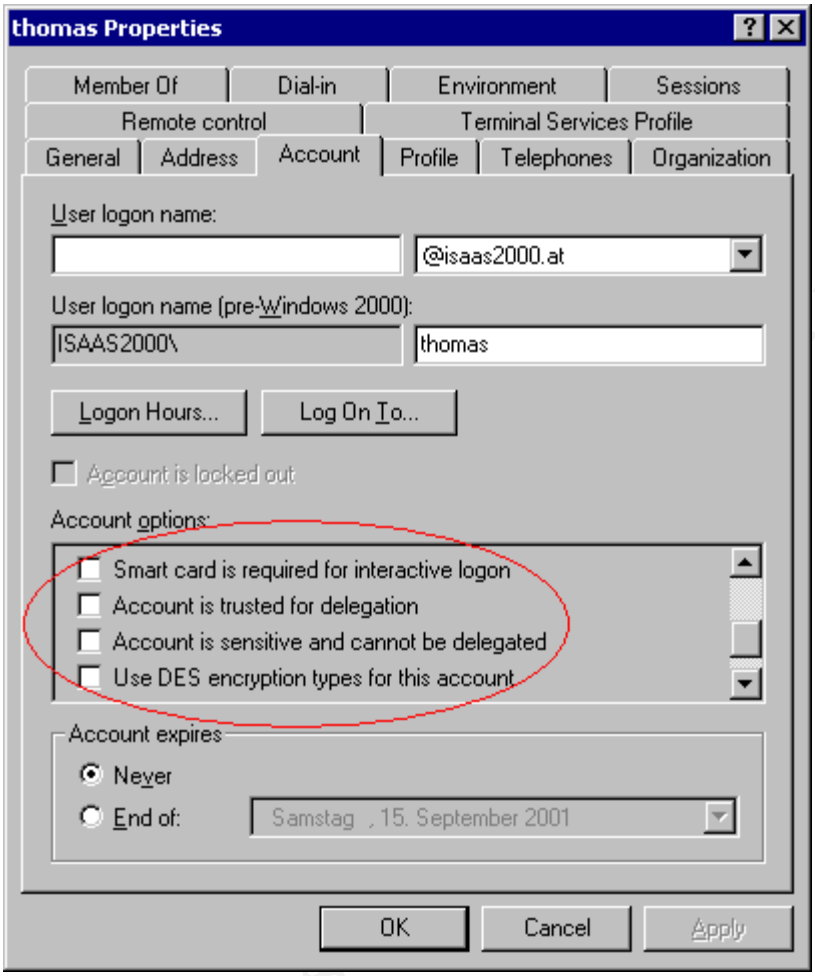


*Figure 6: Kerberos user properties*

To accept user-IDs from other (Non-Windows 2000) realms, usernames can be mapped to specific accounts of a Windows 2000 Domain. When right clicking on a user in Active Directory, choose "Name Mappings…" to map a username of a Non-Windows 2000 Domain to a Windows 2000 account.

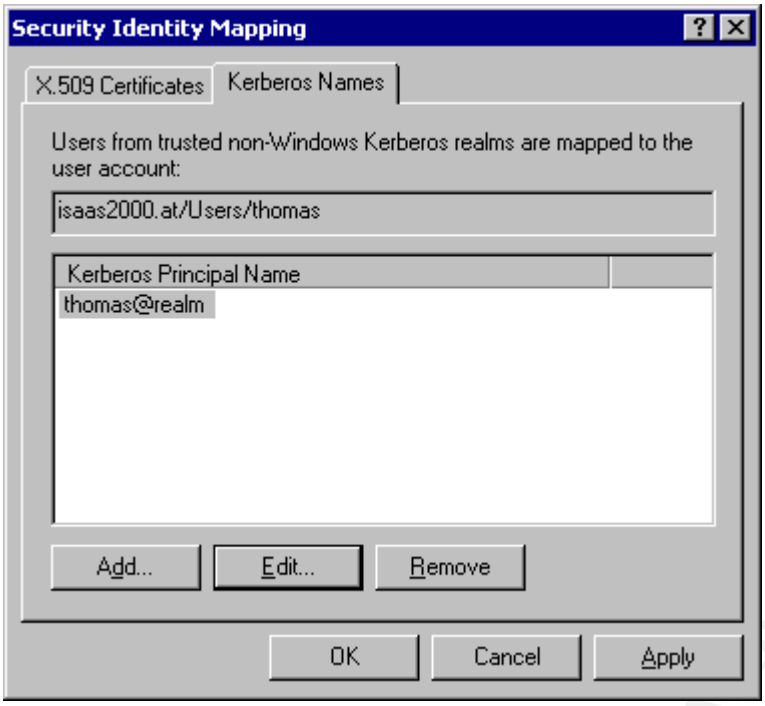

*Figure 7: Account name mappings*

**Computer properties**

Many services run as the local SYSTEM account of a computer instead of the account of a user. To prevent such services to request services from other computers impersonating the local SYSTEM account there is an option in the properties for a computer in Active Directory ("General" tab), which is called "Trust computer for delegation". If this option is disabled (which is the default value) services running as local SYSTEM may not request services on other computers. [9]

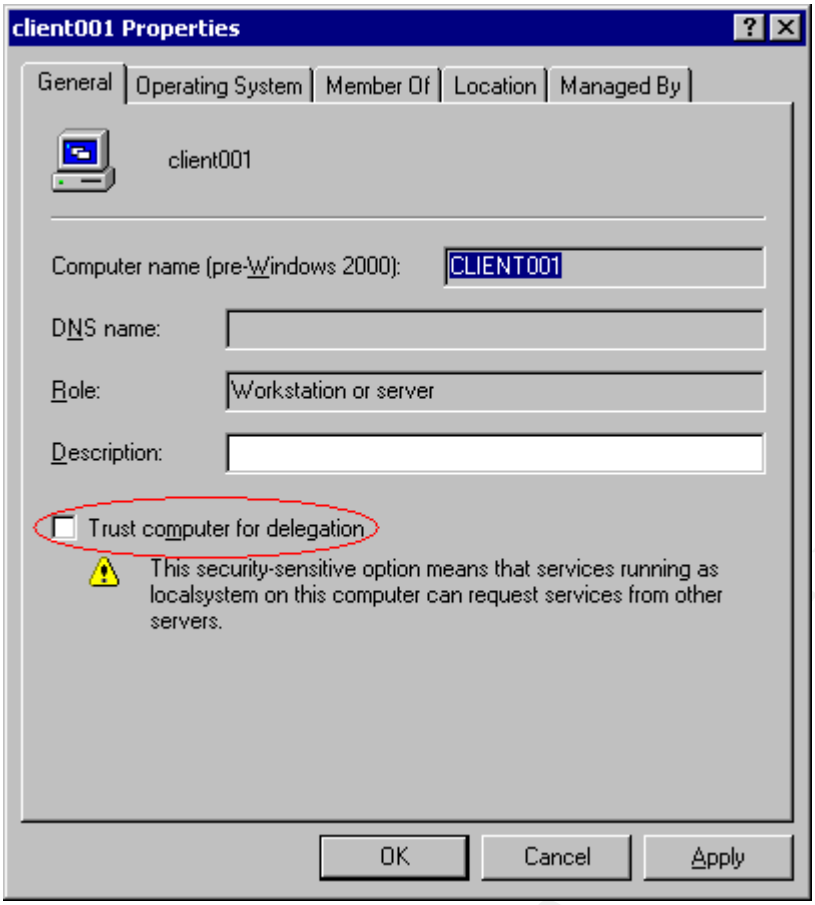

*Figure 8: Computer properties*

**Enabling event logging**

All Kerberos events on a specific computer can be logged by editing the registry. This feature can be for any troubleshooting actions regarding Kerberos in Windows 2000.

At this place it should be noted, that editing the registry incorrectly may cause serious problems that may require reinstall the operating system. Therefore the following steps only may be used on your own risk.

The following value must be added to the registry to enable Kerberos event logging:

Key:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Kerberos\Parameters
Registry Value: LogLevel
Value Type: REG_DWORD
Value Data: 0x1

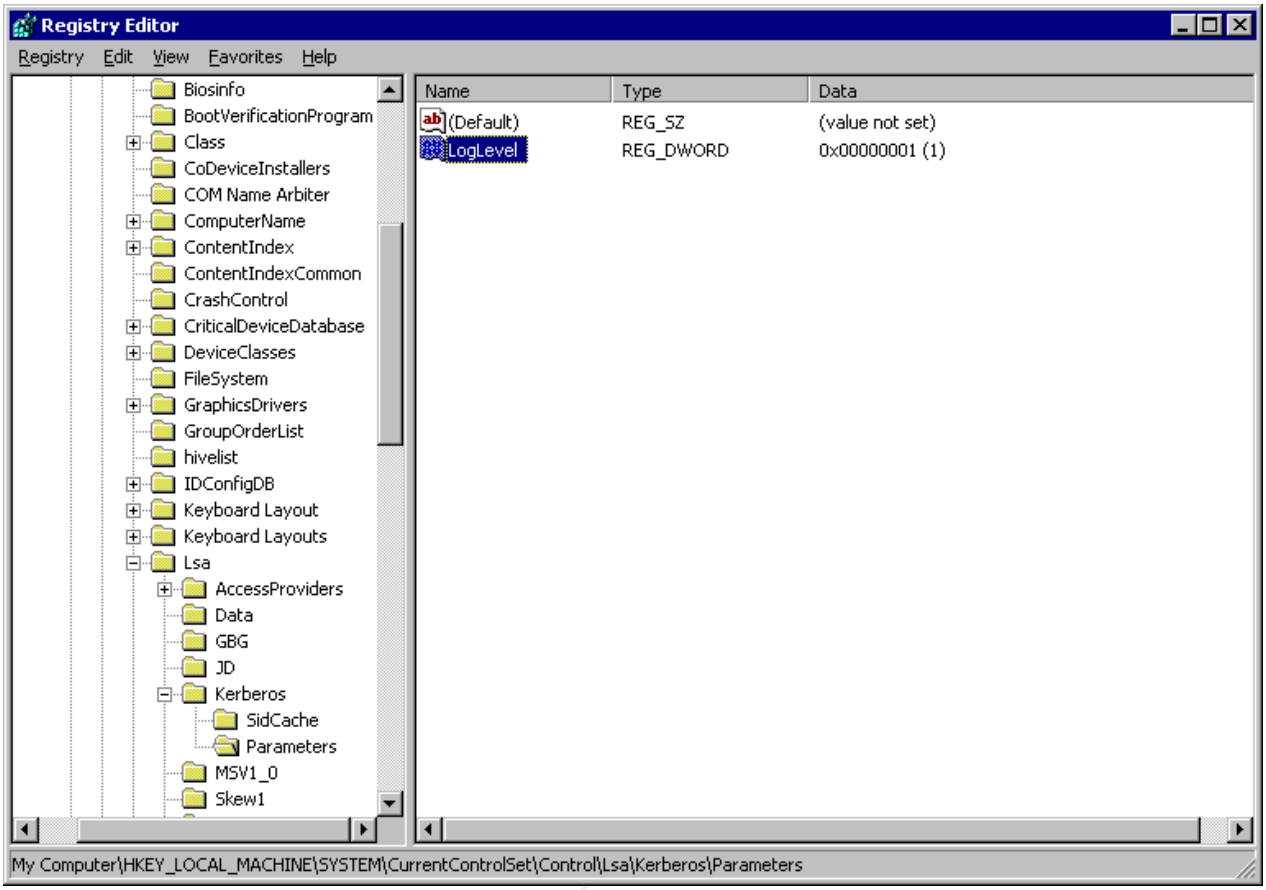


*Figure 9: Enabling Kerberos event logging*

This is also described in Microsoft Knowledge Base article Q262177. [18]

## *3.3 Advantages of Kerberos authentication over former authentication mechanisms of Windows*

There are mainly two advantages of the Kerberos authentication protocol over the former authentication methods Lanmanager (LM) and NT Lanmanager (NTLM).

### 3.3.1 Security

First Kerberos is much more secure than LM or NTLM. While LM/NTLM are sending (encrypted) passwords over the network, Kerberos never does so (except for password changing, which does not occur that much as other network connections are established). Additionally Kerberos uses stronger encryption algorithms, especially as technology evolves, additional (stronger) algorithms can be added. And at least not only the client proves its identity to the server but also the server proves its identity to the client (mutual authentication).

### 3.1.2 Speed

Kerberos authentication is much faster than LM/NTLM authentication. The KDC is only contacted at logon time of the user and for the first logon to a service during a logon session (additional contacting of the KDC is required if a ticket's lifetime ends). Using LM/NTLM the domain controller is contacted every time a network resource is accessed, which increases the network traffic significantly.

### 3.1.3 Additional advantages

An additional advantage is the interoperability of Kerberos. Because Kerberos is an industry standard, it is possible that Windows 2000 domains interoperate with Non-Windows 2000 realms. Due to the fact that establishing this interoperability requires a lot of configuration of the Windows 2000 resources as well as of the Non-Windows 2000 resources, the value of this advantage decreases. With LM/NTLM authentication it was already possible to use Non-Windows computers together with Windows computers in some way (e.g. the Samba server for Unix systems), therefore the advantage of interoperability faces the disadvantage of configuration work.

## 4 Interoperability of Windows 2000 Kerberos

As Kerberos is an industry standard Kerberos authentication in Windows 2000 can be combined with other operating systems that implement Kerberos authentication. Especially interoperation with the free implementation offered by MIT [21], which is used on Unix systems, is an interesting issue. The Microsoft implementation of Kerberos follows the RFC 1510 standard, unfortunately the standard defines various options as well as it leaves some issues unspecified.

In particular the following items are special to the Windows 2000 Kerberos implementation and differ from other implementations:

- A Windows 2000 client sends preauthentication data during logon. This is an option defined by RFC 1510 but commonly not implemented. [1]
- Windows 2000 Kerberos uses RC4 by default for encryption instead of DES which is commonly used (but although less secure). Nevertheless DES-CBC-CRC as well as DES-CBC-CRC encryption is supported by Windows 2000 Kerberos. [1]
- The scheme Windows 2000 Kerberos derives keys from user passwords does not exactly match the scheme MIT does it. [1]
- As described above, Windows 2000 makes extensive use of the authorization data field in the tickets. Because RFC 1510 does not specify the data format within this field, Microsoft uses it's own specific format, which is totally incompatible with other Kerberos implementations. [1]
- Transitive trust is only supported within a Windows 2000 domain-tree. For cross-

platform trusts hierarchical realm support is not supported. [8]
- Windows 2000 KDCs do not support post-dated tickets. [8]

Using Kerberos in mixed environment, following scenarios are possible: [7]

| Client | KDC | Resource to access |
|---|---|---|
| Windows 2000 | Windows 2000 | Non-Windows 2000 |
| Windows 2000 | Non-Windows 2000 | Windows 2000 |
| Windows 2000 | Non-Windows 2000 | Non-Windows 2000 |
| Non-Windows 2000 | Windows 2000 | Non-Windows 2000 |
| Non-Windows 2000 | Non-Windows 2000 | Windows 2000 |

There are detailed descriptions by Microsoft (review [7], [8], [12]) as well as by other Kerberos implementations (e.g. Heimdal [22], MIT [21]). Therefore configuration details will not be discussed here but only the support tools for Windows 2000 should be introduced briefly: [8]

- Ksetup: This tool is used to configure Kerberos realms, KDCs and Kpasswd (Kerberos password) servers
- Ktpass: This tool sets the password, account name mappings and keytab generation for Kerberos services.

These tools are found in the Windows 2000 resource kit.

# 5 Known bugs and vulnerabilities in Windows 2000 Kerberos

There are already some bugs and vulnerabilities within the Windows 2000 Kerberos implementation, some of them are listed below:

**Logged-On users may not be authenticated to services after krbtgt password change**
When the password of the KDC's service user krbtgt is changed, TGTs that have been issued to users cannot be decrypted any more because the KDC does not know the password (and the derived secret key) that has been used to encrypt the TGT. Applying Windows 2000 service pack 2 should fix this problem (i.e. implementing a password history for krbtgt). This problem is described in the Microsoft Knowledge Base article Q295083. [20]

**Administrator account is not usable by non-Windows 2000 Kerberos clients**
User accounts that have been created in a Windows NT 4.0 domain and the domain has been upgraded as well as the built-in Administrator account of the domain controller are only equipped with an NTLM password hash which is used by the default encryption type of Windows 2000 Kerberos RC4-HMAC-NT. Non-Windows 2000

Kerberos implementations usually use one of the DES encryption types (DES-CBC-CRC or DES-CBC-MD5) defined in RFC 1510 [4]. Windows 2000 only creates keys for that encryption type when a user is created within a Windows 2000 domain or the password is changed. This issue is described in the Microsoft Knowledge Base article Q248808. [15]

**Cannot use Kerberos trust relationships between two forests in Windows 2000**
Windows 2000 supports trust relationships using the Kerberos protocol only between domains within one forest. Trusts between parents and children are established by default; trusts between trees may be established. If a trust relationship is setup with a domain outside the forest, NTLM authentication is used for that trust relationship. This issue is described in the Microsoft Knowledge Base article Q274438. [19]

**IPSec does not secure Keberos traffic between domain controllers**
Even if IPSec is enabled, there is still some IP traffic that is not protected by IPSec by design. Besides other protocols, Kerberos is such an exemption. In the Windows 2000 Service Pack 1 protection of Kerberos by IPSec is implemented. Therefore it is recommended to apply the latest service pack to protect Kerberos traffic. This problem is described in the Microsoft Knowledge Base article Q254728. [17]

**Kerberos change password does not work when account password expires**
When the password of an account has been expired, it is no longer possible to change it using the Kerberos password change mechanism. Applying the Windows 2000 service pack 1 should fix this problem. Microsoft suggests a workaround, that the administrator should change expired passwords. Of course this workaround is not sufficient from a security point of view, therefore application of the latest service pack is recommended. This problem is described in the Microsoft Knowledge Base article Q253532. [16]

# 6 Conclusion

The Kerberos v5 protocol is an enhancement of authentication security in Windows 2000 compared to earlier methods of authentication (Lanman, NTLM). It uses more secure encryption mechanisms (including the fact that the password is not transmitted over the network during authentication) and adds some features like mutual authentication and delegation of authentication.

Although Kerberos is an open standard and Microsoft pretends that interoperability with other operating systems is supported, it is not easy to establish a Kerberos realm containing Windows 2000 computers and computers with other operating systems. The standard provides many vendor specific options and leaves distinct data structures undefined; the Microsoft implementation of Kerberos makes extensive use of these options and additional definitions. Presumable most people using Windows 2000 will try to avoid the effort of installing and maintaining mixed realms.

The main security issue when using Kerberos is to secure the system where the KDC is hosted. Additionally the computer running the time service and the DNS must be secured sufficiently. At least it is still necessary to force the users to use strong passwords or to use smart cards.

# 7 References


[1] Chappell, David. “Microsoft and the Kerberos Standard” ent online. 03 November 1999. URL: http://www.entmag.com/displayarticle.asp?ID=1149933809PM

[2] De Clercq, Jan. “Kerberos in Win2K” Windows 2000 Magazine. October 1999. URL: http://www.win2000mag.com/Articles/Index.cfm?ArticleID=7193

[3] Fontana, John. “Microsoft finally publishes secret Kerberos format” InfoWorld. 28 April 2000. URL: http://www.infoworld.com/articles/en/xml/00/04/28/000428enkerpub.xml

[4] Kohl J., Neumann C. RFC 1510: The Kerberos Network Authentiocation Service (V5). September 1993. URL: http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1510.html

[5] Microsoft Corporation. Windows 2000 Security Technical Overview. 2000. URL: http://www.microsoft.com/technet/prodtechnol/windows2000serv/deploy/confeat/sectech.asp

[6] Microsoft Corporation. Windows 2000 Kerberos Authentication. 1999. URL: http://www.microsoft.com/technet/prodtechnol/windows2000serv/deploy/confeat/kerberos.asp

[7] Microsoft Corporation. Windows 2000 Kerberos Interoperability. 2000. URL: http://www.microsoft.com/technet/prodtechnol/windows2000serv/maintain/featusability/kerbinop.asp

[8] Microsoft Corporation. Step-by-Step Guide to Kerberos 5 (krb5 1.0) Interoperability. 2000. URL: http://www.microsoft.com/windows2000/techinfo/planning/security/kerbsteps.asp

[9] Opitz, Dave (NSA). Guide to Windows 2000 Kerberos Settings. 12 April 2001. URL: http://nsa1.www.conxion.com/win2k/r1/windows_2000_kerberos_settings.pdf

[10] Schmidt, Jeff. Windows 2000 security. Munich: Markt-und-Technik-Verlag, 2001

[11] Shinder, Thomas; Shinder, Debora; White, Lynn. Windows 2000 Server Security. Bonn: Mitp-Verlag, 2000

[12] Taylor, Paul. “Kerberos Interoperability in Windows 2000” Windows NT Systems. April 1999. URL: http://www.ntsystems.com/db_area/archive/1999/9904/304s1.shtml

[13] N.N. Kerberos: Strenghts and Weaknesses. URL: http://www.oit.duke.edu/~rob/kerberos/kerbasnds.html

**Related Microsoft Knowledge Base Articles (http://search.support.microsoft.com/kb/c.asp?fr=0&SD=GN&LN=EN-US):**

[14] Q217098: Basic Overview of Kerberos User Authentication Protocol in Windows 2000

[15] Q248808: Administrator Account Is Not Usable by Non-Windows 2000 Kerberos Clients

[16] Q253532: Kerberos Change Password Does Not Work When Account Password Expires

[17] Q254728: IPSec Does Not Secure Kerberos Traffic Between Domain Controllers

[18] Q262177: How to Enable Kerberos Event Logging

[19] Q274438: Cannot Use Kerberos Trust Relationships Between Two Forests in Windows 2000

[20] Q295083: Logged-On Users May Not Be Authenticated to Services After KRBTGT Password Change

**Kerberos implementations for Unix:**

[21] Kerberos of the Massachusetts Institute of Technology (MIT), version 1.2, URL: http://web.mit.edu/kerberos/www/

[22] Heimdal Kerberos, URL: http://www.pdc.kth.se/heimdal/