

Configuring Authentication for Microsoft Windows

In this chapter:

Storing and Transmitting Credentials	69
Storing Secrets in Windows	83
Best Practices	86
Additional Information	87

Authentication attempts to answer the question, “User: How do I know that it is really you?” Ideally, the authenticating database would be able to identify the actual person attempting to access a given account, but this problem has proved frustratingly complex to solve when cost is a consideration, even with the most sophisticated biometrics. Until the technical and economic barriers, not to mention serious privacy issues, can be resolved, authentication revolves around identifying a user by that user’s knowledge of a shared secret (for example, a password or PIN) and possibly of something that only that user would possess (for example, a smart card or ID token). This chapter details how network authentication works in Microsoft Windows Server 2003, Windows 2000, and Windows XP.

Storing and Transmitting Credentials

The operating system is responsible for securely storing and transmitting credentials (user names and passwords) for accounts. Windows 2000 and Windows XP support a variety of protocols to transmit credentials across the network to authenticate accounts, including user accounts, computer accounts, and service accounts. The operating system also stores credentials in a variety of formats.

When a user logs on to Windows Server 2003, Windows 2000, or Windows XP by using the Windows Logon dialog box, several components work together to authenticate her credentials securely. Figure 4-1 shows the information flow of authentication.

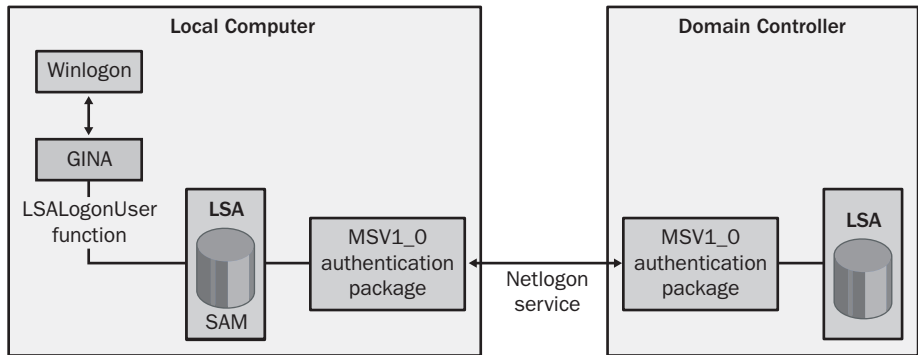


Figure 4-1 Authentication in Windows Server 2003, Windows 2000, and Windows XP

Windows Server 2003, Windows 2000, and Windows XP use Kerberos v5 as the default authentication protocol for domain authentication and also support authentication using the LAN Manager (LM), NT LAN Manager (NTLM), and NT LAN Manager version 2 (NTLMv2) protocols. Because few networks use a single operating system version or applications that use the same authentication methods, Windows Server 2003, Windows 2000, and Windows XP support current and legacy authentication methods to preserve compatibility with downlevel operating systems and applications. As a network administrator, you can, however, tune authentication in the Windows operating system based on your technical requirements.

LAN Manager

LAN Manager authentication is supported in Windows Server 2003, Windows 2000, and Windows XP to support legacy applications as well as some very specific implementations, such as internode authentication within a cluster. LM passwords are limited to 14 characters. For LM authentication, passwords are not stored by the operating system. Instead, passwords are encrypted with the LAN Manager one-way function (OWF), which is formed by converting the password to uppercase characters, adding padding for passwords with fewer than 14 characters, dividing the 14-character password into 7-character halves, and encrypting a constant value with the 7-character halves by using the Data Encryption Standard (DES) encryption algorithm. This process is illustrated in Figure 4-2.

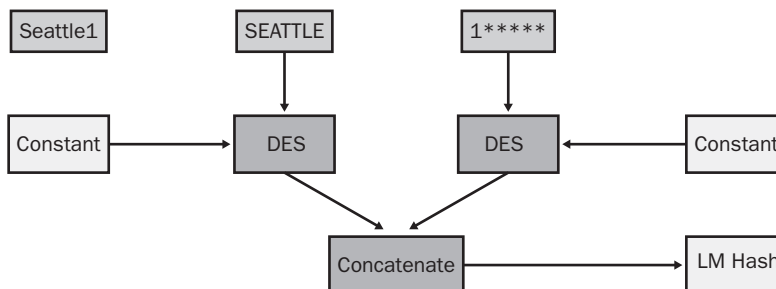


Figure 4-2 LM password storage

When a user authenticates a password by using the LM authentication protocol, the authentication is done with a simple challenge/response interaction from the authenticating domain controller or computer. Figure 4-3 shows the process of authenticating an account by using LM.

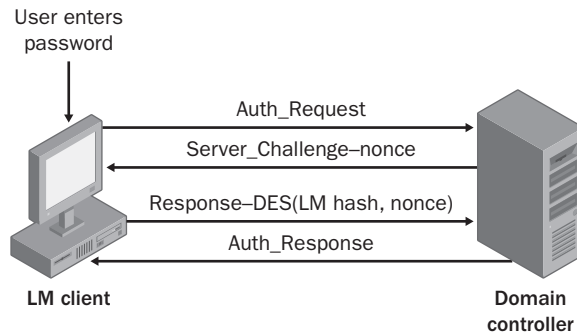


Figure 4-3 LM authentication

In Figure 4-3, the client sends an authentication request to the logon server. The server returns a challenge, which comprises a random number, or nonce. The client then uses the LM password hash to encrypt the nonce by using the DES encryption algorithm. Next, the server decrypts the encrypted nonce from the client by using the LM password it has stored in the accounts database. If the nonce matches the nonce sent to the client, the client's credentials are validated. The NTLM authentication sequence works the same way, albeit with larger nonces. If an attacker is able to capture all packets in a challenge/response authentication sequence, he could attempt brute force attacks on the packets to determine the password; this requires multiple cryptographic operations that, although time-consuming, can reveal weak passwords. Challenge/response sequences are not vulnerable to precomputation attacks because the nonces are random.

By reducing the character set for letters to include only uppercase letters, limiting passwords to 14 characters, and separating passwords into 7-character pieces, LM password hashes are especially vulnerable to brute force and dictionary attacks. Many tools available on the Internet can easily crack most LM password hashes given enough computer power and time. Although the idea had been around for some time, Project Rainbow Crack, made public in 2003, enables individuals to precalculate LM password hashes and store them in a series of tables. The tables need to be generated only once; thus, once an attacker or administrator has obtained the password hash, all that is left is a simple lookup in the table to reveal the plaintext password. All password hashes are vulnerable to precomputation attacks, just as all are vulnerable to brute force attacks within the limits of time and computational power. The precomputation attack on LM authentication works because of the confluence of two important factors:

- **Passwords are not salted.** A salt value, which is a pseudorandom string, is appended to passwords before they are stored. The value is different on a different system. For example, the plaintext password “Swordfish123” would have different password hashes stored on different systems even though the password is the same. Because LM authentication does not use a salt, the password hash for “Swordfish123” is the same on all systems running the Windows operating system worldwide. This is the primary reason that the Rainbow tables need be generated only one time.
- **A small total number of passwords are available.** The keyspace for LM authentication is very, very limited. In fact, all LM password hashes, stored in Rainbow tables, can be stored on as few as two DVDs.



More Info See the Project Rainbow Crack Web site for detailed information on precomputing password hashes at <http://www.antsight.com/zsl/rainbowcrack/>.

Windows Server 2003, Windows 2000, and Windows XP, like Windows NT 4.0, store the LM hash for local accounts in the Security Accounts Manager (SAM) database, which is stored persistently in encrypted partitions of the registry. Windows Server 2003 and Windows 2000 store the LM hash for domain accounts in the Active Directory directory service. You can, however, prevent the Windows operating system from generating and storing LM hashes. Of course, this will also prevent LM authentication from working, but many, if not most, networks running operating systems earlier than Windows 2000 do not rely on LM authentication. Before preventing LM authentication on your network globally, be sure to test critical resources fully for compatibility. The easiest way to do this is to create passwords longer than 14 characters for test accounts and use those accounts to test services and applications. The Windows operating system will not create LM hashes when passwords longer than 14 characters are created. Windows 2000 Service Pack 2 adds functionality that enables you to remove LM password hashes from Active Directory or the local computer SAM databases. Windows Server 2003 and Windows XP also support this. The following table shows the registry key and value to add to the registry to remove LM password hashes. You can also add a custom registry modification to Windows 2000 security templates and deploy the change by using Group Policy.

Location	HKLM\SYSTEM\CurrentControlSet\Control\Lsa
Type	DWORD
Key	NoLMHash

Windows XP and Windows Server 2003 both have a setting in the Security Options section of security templates to do this: “Network Security: Do Not Store LAN Manager Hash Value On Next Password Change.” To confuse the matter, Windows XP and Windows Server 2003 look for a registry value, not a key as Windows 2000 does.

The following table shows the key in Windows 2000.

Location	HKLM\SYSTEM\CurrentControlSet\Control\Lsa
Type	Key
Name	NoLMHash

The following table shows the value in Windows XP and Windows Server 2003.

Location	HKLM\SYSTEM\CurrentControlSet\Control\Lsa
Type	DWORD
Value	NoLMHash
Set to	1

You must implement this registry change on all domain controllers in the domain to fully prevent the creation of LM password hashes for domain accounts. Additionally, this change will not take place until users change their passwords the first time after the registry has been changed. You can also prevent the LM authentication protocol from being used by computers running Windows Server 2003, Windows 2000, or Windows XP by setting the LM compatibility to a level greater than 0. We discuss LM compatibility levels in further detail momentarily when we examine NTLMv2.



Tip Although removing LM hash values is a good step to take to reduce the risk of passwords being revealed, the real issue here is preventing the attacker from accessing the password hashes to begin with. Anyone with physical access to a computer, including the administrators of the system, can extract the password hashes. An attacker with physical access to the network (either wired or wireless) can also sniff entire challenge/response sequences to obtain material to attack through brute force if network communication is not protected.

NTLM

NT LAN Manager, also known as NTLM, first was included in Microsoft Windows NT and is an improvement over the LM authentication protocol. Unlike LM passwords, which use an ASCII character set, NTLM passwords are based on the Unicode character set, are case-sensitive, and can be up to 128 characters long. As with LM, the operating system does not actually store the password; rather, it stores a representation of the password by using the NTLM OWF. The NTLM OWF is computed by using the MD4 hash function, which computes a 16-byte hash, or *digest*, of a variable-length string of text, which in this case is the user's password.



Note A hash function takes a variable-length binary input and produces a fixed-length binary output that is irreversible. Because the binary output has a relatively short fixed length, theoretically many binary inputs produce the same binary output. However, it is nearly impossible practically to find two different inputs that result in the same hash. This is called a collision. Hash functions that produce predictable collisions, as is the case with SHA0, should not be used. At the time of the writing of this book, research is showing that both MD5 and SHA1 might be vulnerable to predictable collisions, similar to SHA0. Common hash functions include MD4, MD5, and SHA1.

Another difference between NTLM and LM is that NTLM passwords are not broken into smaller pieces before having their hash algorithm computed. NTLM uses the same challenge/response process for authentication as LM does. NTLM is the default authentication provider in Windows NT and Windows 2000 and later when not a member of an Active Directory domain, and even then is still used for many functions. To preserve backward compatibility, the LM hash is always sent with the NT hash. If you have removed LM hashes from the account database by creating a password greater than 14 characters or configuring the NoLMHash registry option, the Windows operating system will create a null value for the LM hash. Although this value cannot be used to authenticate the user, the null value will still be transmitted with the NTLM authentication sequence.

NTLMv2

NTLMv2, the second version of the NTLM protocol, was first available in Windows NT 4.0 Service Pack 4 and is included in Windows 2000 and later. NTLMv2 passwords follow the same rules as NTLM passwords; however, NTLMv2 uses a slightly different process for authentication. NTLMv2 also requires that the clocks of clients and servers be within 30 minutes of each other.

If both the client and the server support NTLMv2, enhanced session security is negotiated. This enhanced security provides separate keys for message integrity and confidentiality, provides client input into the challenge to prevent chosen plaintext attacks, and uses the Hash-Based Message Authentication Code (HMAC)-MD5 algorithm for message integrity checking. Because the datagram variant of NTLM does not have a negotiation step, use of otherwise-negotiated options (such as NTLMv2 session security and 128-bit encryption for message confidentiality) must be configured. Table 4-1 lists the registry key for setting NTLM negotiation options.

Table 4-1 Setting NTLM Negotiation Options

Location	HKLM\SYSTEM\CurrentControlSet\Control\Lsa\MSV1_0\
Type	DWORD
Value	NtlmMinServerSec
Default value	0x00000000

Table 4-2 lists the registry key for configuring the levels of NTLMv2 support.

Table 4-2 Configuring the Levels of NTLMv2 Support

Location	HKLM\SYSTEM\CurrentControlSet\Control\Lsa\MSV1_0\
Type	DWORD
Value	NtlmMinClientSec
Default value	0x00000000

Table 4-3 lists the values for the client negotiation options just described.

Table 4-3 Values for Setting NTLMv2 Client Negotiation Options

Value	Description
0x00000010	Message integrity
0x00000020	Message confidentiality
0x00080000	NTLMv2 session security
0x20000000	128-bit encryption



Tip You can combine multiple NTLMv2 options by performing a logical OR on the settings in Table 4-3.

Enable NTLMv2 by setting the LM compatibility level using Group Policy. The six LM compatibility settings are described in Tables 4-4 and 4-5. Table 4-4 describes the impact on the authenticating client. Table 4-5 describes the impact on the authenticating server. As you can see, levels 0 through 3 have no impact on which authentication protocols are accepted by the computer, whereas levels 4 and 5 do not impact which protocols are sent.

Table 4-4 Authenticating Client Impact of LAN Manager Compatibility Levels

Level	Sends	Accepts	Prohibits Sending
0	LM, NTLM,	LM, NTLM, NTLMv2	NTLMv2, Session security
1	LM, NTLM, Session security	LM, NTLM, NTLMv2	NTLMv2
2	NTLM, Session security	LM, NTLM, NTLMv2	LM and NTLMv2
3	NTLMv2, Session security	LM, NTLM, NTLMv2	LM and NTLM

Table 4-5 Authenticating Server Impact of LAN Manager Compatibility Levels

Level	Sends	Accepts	Prohibits Accepting
4	NTLMv2, Session security	NTLM, NTLMv2	LM
5	NTLMv2, Session security	NTLMv2	LM and NTLM

Setting LM compatibility to value 2 on all of your Windows-based computers will prevent LM authentication from being sent, but will allow all computers to accept it. Setting all client computers to value 2 and all server computers to value 4 will ensure that no computer sends LM and only client computers accept it. As you can see, setting LM compatibility is not exactly straightforward. Rather than making a global change in LM compatibility, you should consider rolling out the change in stages to simplify troubleshooting if you should run into application compatibility problems. You can do this both by slowly increasing the LM compatibility on computers and by deploying higher compatibility levels only to certain machines by using Group Policy. Windows-based computers do not negotiate LM protocols; if the authentication protocol the client uses to start the sequence is not supported by the server, the authentication fails. The client computer will not attempt to fall back on another authentication protocol. The registry key for setting LM compatibility levels is shown in Table 4-6. You can also set this option using Group Policy in the Security Options portion of the Security Templates.

Table 4-6 Setting LM Compatibility Levels

Location	HKLM\SYSTEM\CurrentControlSet\Control\Lsa
Type	DWORD
Key	LMCompatibilityLevel
Default value	0 in Windows 2000 and Windows XP, 2 in Windows Server 2003

Kerberos

Kerberos v5 is the default authentication protocol for computers running Windows Server 2003, Windows 2000, and Windows XP that are members of Active Directory. Windows operating systems earlier than Windows 2000 do not support the use of Kerberos and will use one of the LM authentication methods. The implementation of Kerberos in the Windows operating system is compliant with RFC 1510 and is interoperable with other Kerberos v5 realms that are RFC 1510 compliant with some minor configuration. Kerberos provides the following benefits:

- **Mutual authentication** Kerberos enables the client to verify a server's identity, one server to verify the identity of another, and the client to verify its identity to the server.
- **Secure transmission over the wire** Kerberos messages are encrypted with a variety of encryption keys to ensure no one can tamper with the data in a Kerberos message. Furthermore, the actual password is not sent across the network when using Kerberos. This does not mean, however, that an attacker cannot carry out a brute force attack against an intercepted authentication sequence—in fact, RFC 1510 specifically mentions this limitation: “The Kerberos protocols generally assume that the encryption used is secure from cryptanalysis; however, in some cases, the order of fields in the encrypted portions of messages are arranged to

minimize the effects of poorly chosen keys. It is still important to choose good keys. If keys are derived from user-typed passwords, those passwords need to be well chosen to make brute force attacks more difficult. Poorly chosen keys still make easy targets for intruders.” Because passwords are used as the base key material, users still must choose strong passwords.

- **Prevention of replay of authentication packets** Kerberos minimizes the possibility of someone obtaining and reusing a Kerberos authentication packet by using timestamps as an authenticator. By default in Windows 2000 and later, all system clocks are synchronized with the domain controller that authenticated them through the Network Time Protocol (NTP). For Kerberos tickets to remain valid, the system clocks must be synchronized to within 5 minutes of each other.
- **Delegated authentication** Windows services impersonate clients when accessing resources on clients’ behalf. Kerberos includes a proxy mechanism that enables a service to impersonate its client when connecting to other services. Among other things, this allows for n-tier applications to use user credentials on back-end systems without relying on duplicated accounts or passthrough authentication as would be required with LM authentication protocols. Windows Server 2003 provides a safer mechanism to use delegation through constrained delegation and supports other authentication protocols through protocol translation.

The following four components allow Kerberos v5 authentication between client computers and users that use Kerberos to domain controllers that run Windows Server 2003 or Windows 2000:

- **Key distribution center (KDC)** The network service that supplies both ticket-granting tickets (TGTs) and service tickets to users and computers on the network. The KDC manages the exchange of shared secrets between a user and a server when they authenticate with each other. The KDC contains two services: the Authentication Service and the Ticket Granting Service. The Authentication Service provides the initial authentication of the user on the network and provides the user with a TGT. Whenever users request access to a network service, they supply their TGT to the Ticket Granting Service. The Ticket Granting Service then provides the user with a service ticket for authentication with the target network service. In an Active Directory network, the KDC service runs on all domain controllers.
- **Ticket-granting ticket (TGT)** Provided to users the first time they authenticate with the KDC. The TGT is a service ticket for the KDC. Whenever the user needs to request a service ticket for a network service, she presents the TGT to the KDC to validate that she has already authenticated with the network. For additional security, the Windows operating system verifies, by default, that the user account is still active every time a TGT is presented to the KDC. In other words,

the KDC verifies that the account has not been disabled. If the account has been disabled, the KDC will not issue any new service tickets to the user. In Windows 2000 when the user's TGT is received, the server that has been trusted for delegation can request service tickets for the user to any other service on the network. In Windows Server 2003, the server can directly request a session ticket through protocol translation. Also different in Windows Server 2003, by using their Service Principal Name (SPN) you can determine which services the computer can delegate to. This is a huge security improvement over delegation in Windows 2000. Figure 4-4 shows the Delegation tab of a computer account in Windows Server 2003.

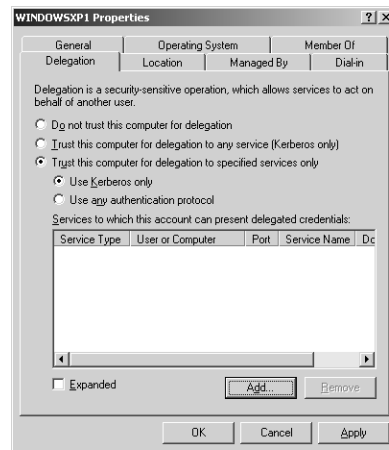


Figure 4-4 Delegation options in Windows Server 2003

- Service ticket** Provided by a user whenever he connects to a service on the network. The user acquires the service ticket by presenting the TGT to the KDC and requesting a service ticket for the target network service. The service ticket contains the target server's copy of a session key and contains information about the user who is connecting. This information is used to verify that the user is authorized to access the desired network service by comparing the authentication information—namely, the user's security identifier (SID) and his group SIDs—against the discretionary access control list (DACL) for the object that he is attempting to access. The service ticket is encrypted using the key that is shared between the KDC and the target server. This ensures that the target server is authenticated because only the target server can decrypt the session key.

- **Referral ticket** Issued any time a user attempts to connect to a target server that is a member of a different domain. The referral ticket is actually a TGT to the domain where the resource is located. The referral ticket is encrypted using an interdomain key between the initial domain and the target domain that is exchanged as part of the establishment of transitive trust relationships.

All Kerberos authentication transactions will be composed of some combination of these three message exchanges:

- **Authentication Service Exchange** Used by the KDC to provide a user with a logon session key and a TGT for future service ticket requests. The Authentication Service Exchange comprises a Kerberos Authentication Service Request (*KRB_AS_REQ*) sent from the user to the KDC and a Kerberos Authentication Service Reply (*KRB_AS_REP*) returned by the KDC to the user.
- **Ticket-Granting Service Exchange** Used by the KDC to distribute service session keys and service tickets. The service ticket that is returned is encrypted using the master key shared by the KDC and the target server so that only the target server can decrypt the service ticket. A Ticket-Granting Service Exchange comprises a Kerberos Ticket-Granting Service Request (*KRB_TGS_REQ*) sent from the user to the KDC and a Kerberos Ticket-Granting Service Reply (*KRB_TGS_REP*) returned by the KDC to the user.
- **Client/Server Authentication Exchange** Used by a user when presenting a service ticket to a target service on the network. The message exchange comprises a Kerberos Application Request (*KRB_AP_REQ*) sent from the user to the server and a Kerberos Application Response (*KRB_AP_REP*) returned by the target server to the user.

To use the Client/Server Authentication Exchange, the user enters her login name, password, and domain in the Windows Logon dialog box. The client computer then locates a KDC by querying the Domain Name System (DNS) server. Next, the user's computer sends a Kerberos Authentication Service Request (*KRB_AS_REQ*) to the domain controller. The user's account information and the current computer time are encoded by using the long-term key shared between the user's account and the KDC. Finally, the authentication service at the KDC authenticates the user, generates a TGT for her, and sends the TGT to her in a Kerberos Authentication Service Response (*KRB_AS_REP*) message.

When a user authenticates herself by using Kerberos, a series of packets is exchanged to complete the validation of her credentials. Figure 4-5 illustrates this process. The user sends a Ticket-Granting Service Exchange Request (*KRB_TGS_REQ*) to the KDC

to acquire a service ticket for her computer. The *KRB_TGS_REQ* contains an authenticator and the TGT that was issued to the user. The Ticket Granting Service of the KDC checks the TGT and the authenticator. If both are valid, the Ticket Granting Service generates a service ticket and sends it back to the user as a Ticket-Granting Service Response (*KRB_TGS_REP*). At the client computer, the service ticket is presented to the Local Security Authority, which will create an access token for the user. (We discuss the Local Security Authority in a moment.) From then on, any process acting on behalf of the user can access the local machine's resources.

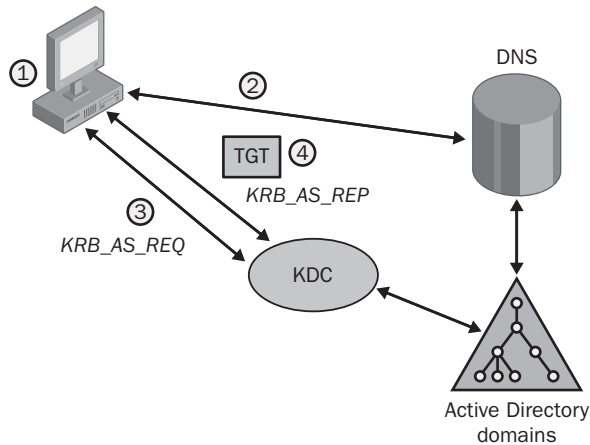


Figure 4-5 Initial Kerberos authentication

This sequence provides the user with the proper TGT. Now the user must acquire a service ticket for that computer. The following steps and Figure 4-6 explain how the service ticket for the computer is acquired.

1. The user sends a Ticket-Granting Service Request (*KRB_TGS_REQ*) to the KDC to acquire a service ticket for the target computer. The *KRB_TGS_REQ* includes the TGT and an authenticator.
2. The Ticket Granting Service of the KDC checks the authenticator and the TGT, generates a new service ticket, and sends it back to the user as a Kerberos Ticket-Granting Service Response (*KRB_TGS_REP*). The service ticket is encrypted by using the target service's long-term key, which is known only by the KDC and the target service.

3. The user sends the service ticket and an authenticator to the target server by using a Kerberos Application Request (*KRB_AP_REQ*).
4. The target server verifies the ticket with the authenticator, decrypts the session key by using the master key that is shared with the KDC, and sends back an authenticator to the user in a Kerberos Application Response (*KRB_AP_REP*). This provides mutual authentication of the user and server.

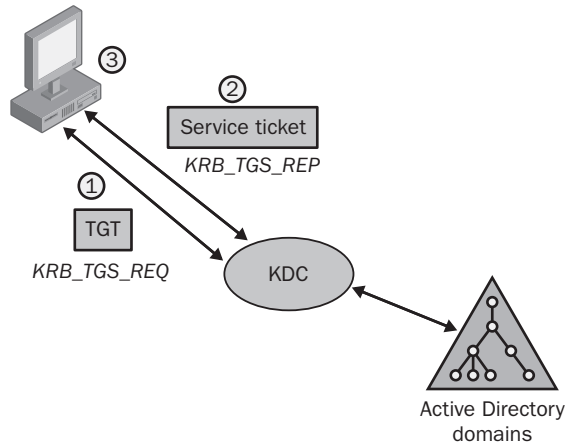


Figure 4-6 Obtaining a Kerberos session ticket

You can view the Kerberos tickets that have been issued to your user and computer accounts by using either Klist.exe, a command-line tool, or Kerbtray.exe, a GUI tool. Both of these utilities not only display the tickets, but also all their properties and expiration dates.

After initially authenticating with the network, the user must authenticate with other computers as she accesses resources on them. Every time the user connects to a resource or service on a remote computer, she has to perform a network authentication. Figure 4-7 shows how Kerberos clients, KDCs, and services communicate.

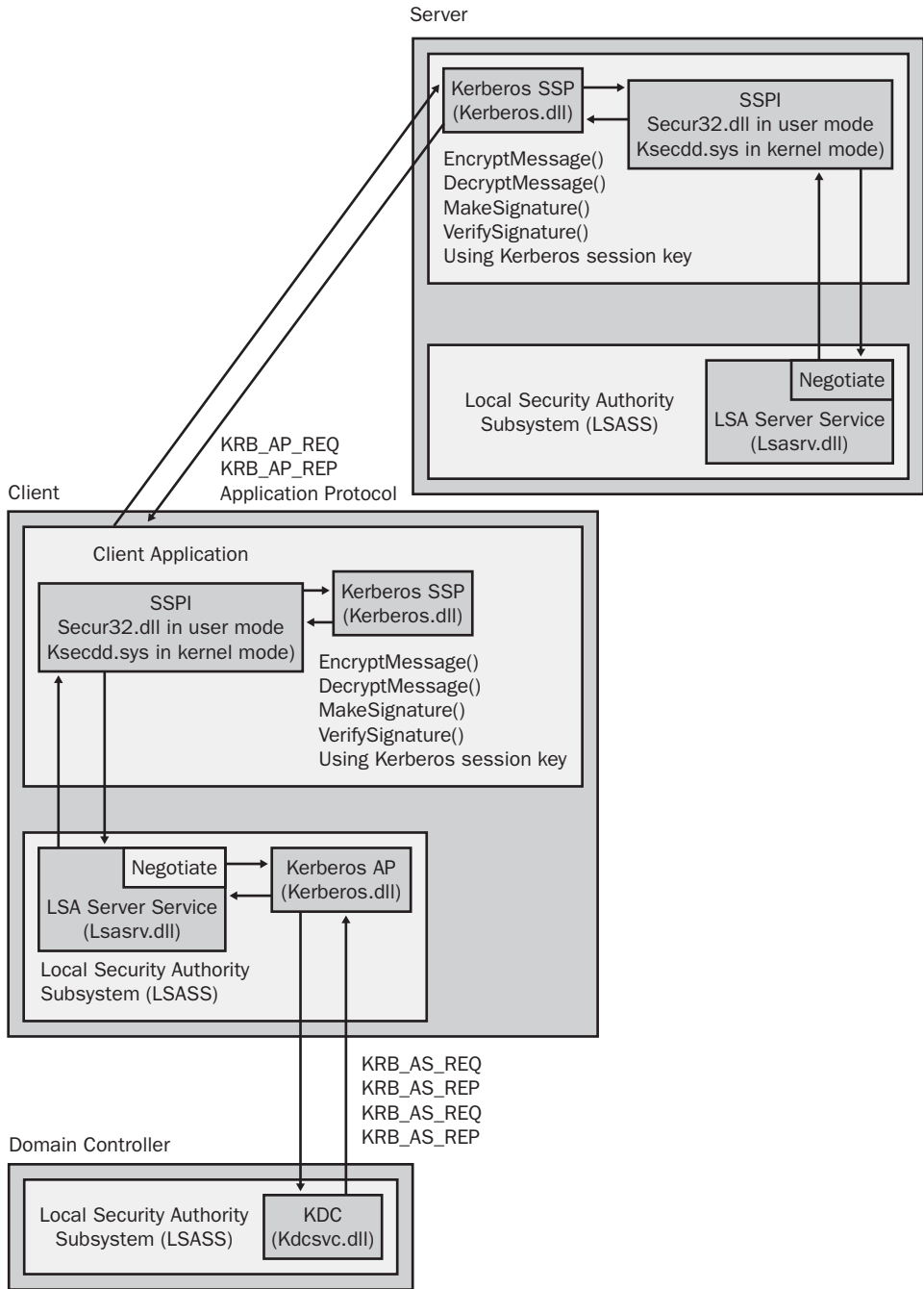


Figure 4-7 Communication flow of Kerberos in Windows 2000 and Windows Server 2003

Storing Secrets in Windows

In addition to storing passwords in Active Directory or SAM databases, Windows Server 2003, Windows 2000, and Windows XP store passwords and other secrets in other locations for a variety of purposes.

LSA Secrets

The Local Security Authority (LSA) maintains information about all aspects of local operating system security. The LSA performs the following tasks:

- Authenticates users
- Manages local security policy
- Manages audit policy and settings
- Generates access tokens

In addition, the LSA stores information used by the operating system, known as *LSA secrets*. LSA secrets include items such as persistently stored Remote Access Service (RAS) information; trust relationship passwords; and user names, passwords, and account names. Perhaps most important, account names and passwords for services that run under a user account context are stored as LSA secrets. LSA secrets can be revealed locally by accounts with the Debug Programs user right. Consequently, you should be careful about the information applications have stored in LSA secrets. Attackers who physically compromise the computer or become system administrators can easily gain access to the information stored as LSA secrets if no other precautions are taken, such as using the System Key (Syskey.exe).



More Info *Writing Secure Code, Second Edition*, by Michael Howard and David LeBlanc (Microsoft Press, 2003) contains detailed information and code samples of how to retrieve LSA secrets. Additionally, you can view most LSA secrets by using LSADUMP2.exe from BindView. The tool is available at <http://razor.bindview.com/tools/>.

Using System Key (Syskey.exe)

The System Key utility was first available in Windows NT 4.0 Service Pack 2 and is enabled by default in Windows Server 2003, Windows 2000, and Windows XP. You can configure the System Key by typing **syskey** at the command prompt if you are an administrator on the computer. The System Key is the master key used to protect the password encryption key and the computer account password; therefore, protection of the System Key is a critical system security operation. You have three options for managing the System Key:

- **Level 1** Uses a machine-generated random key as the System Key and stores the key on the local system. Because the key is stored on the operating system, it allows for unattended system restart. By default, System Key Level 1 is enabled on all computers running Windows 2000 and later operating systems.
- **Level 2** Uses a machine-generated random key and stores the key on a floppy disk. The floppy disk with the System Key is required for the system to start before the system is available for users to log on. Because the System Key is stored on the floppy disk, the operating system is vulnerable to destruction of the floppy disk, which would render the operating system unable to boot.
- **Level 3** Uses a password chosen by the administrator to derive the System Key. The operating system prompts for the System Key password when the system begins the initial startup sequence, before the system is available for users to log on. The System Key password is not stored anywhere on the system; instead, an MD5 hash of the password is used as the master key to protect the password encryption key. If the password is forgotten, the operating system will be rendered unable to boot.

Setting the System Key to Level 2 or Level 3 greatly increases the security of the operating system and the secrets it contains (such as contents of the SAM database and LSA secrets) in the event an attacker physically compromises the computer. However, the Level 2 and Level 3 settings can be difficult to manage because you cannot recover forgotten or lost keys. If a key is lost, the computer will not be able to boot. You should develop a secure method of archiving System Keys if you decide to implement System Key Level 2 or Level 3 on your network. In most corporate environments, using Syskey in mode 2 and 3 is not feasible because of the potential loss of data or availability interruption. The bottom line is that you must provide risk-adequate physical security for computers that store critical information, including credentials for other accounts.

Data Protection API

The Data Protection API (DPAPI) enables secrets to be stored securely by applications by using a key derived from the user's password. The encryption of secrets can only be done locally, unless roaming profiles are used. If the user's password is reset, the key used for DPAPI will be lost unless it has been previously archived. DPAPI is available only in Windows Server 2003, Windows 2000, and Windows XP. By using DPAPI, applications can store the encrypted information in any manner, for example, in the registry or a database. DPAPI is used by Windows XP and Windows Server 2003 to protect encrypting file system (EFS) keys in Workgroup scenarios.

Cached Credentials

By default, Windows Server 2003, Windows 2000, and Windows XP cache the credentials of domain accounts used to log on to the network at the local computer. The credentials include the user's user name, password, and domain. Rather than storing the actual credential information, the information is stored in an irreversibly encrypted form and on the local computer. After a user has successfully logged on to the network from the computer once, he can use his domain credentials, even if the computer is not attached to the network or if no domain controllers are available. This functionality is critical to laptop users and users in branch offices without local domain controllers. You can control the number of credentials stored on a computer at any time by setting the registry key shown in Table 4-7.

Table 4-7 Controlling the Number of Stored Credentials on a Computer

Location	HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT \Current Version\Winlogon\
Type	REG_SZ
Key	CachedLogonsCount
Default value	10
Recommended value	0–50, depending on your security needs

In high-security networks, you might want to set the number of cached credentials to 0. This setting requires all users of the computer to have their domain account credentials validated by a domain controller. This prevents a user who has been terminated from disconnecting her computer from the network, logging on by using cached credentials, and destroying information. When you implement this setting, laptop users can log on only when connected to the network, greatly limiting their mobility. Although the cached credential verifier is irreversibly encrypted, it is not invulnerable to brute force attack; hence, the cached credential is resistant to brute force attacks as much as the user has chosen a strong password.

Credential Manager

Windows XP introduces a new method of managing credentials as well as the credentials of a user who is logged on that is also implemented in Windows Server 2003. This functionality is provided by the Credential Manager, labeled Stored User Names And Passwords in Control Panel. The Credential Manager dynamically and manually creates credential sets (a user name and password) for resources that are available in the user interface and from the command line. You can manage the following types of credentials with the Credential Manager:

- User names and passwords
- X.509 certificates, including smart cards
- Passport accounts



Note The Credential Manager and DPAPI are long-term replacements for the Protected Storage service, which is no longer considered a secure method of storing secrets. The most significant Windows application that still uses the P-store is Microsoft Internet Explorer, which stores Auto-Complete information, including names and passwords used for forms-based authentication. See Chapter 12, “Managing Microsoft Internet Explorer Security and Privacy,” for more information on the Protected Storage service.

Best Practices

- **Remove LM hashes.** If LAN Manager authentication is not used on your network, remove the LM password hashes from all domain controllers and local computers.
- **Configure LM compatibility.** Set the LM compatibility to the highest level that applications on your network will support.
- **Upgrade to Windows Server 2003 if you are using delegation.** Because a computer that is trusted for delegation can use the credentials of any user who authenticates to it on any computer on the network, in Windows 2000 a security compromise of that computer could cause the compromise of the entire domain or forest. Constrained delegation greatly helps limit the potential damage if a computer that is trusted for delegation is compromised.

Additional Information

- The following Knowledge Base articles:
 - 147706: “How to Disable LM Authentication on Windows NT” (<http://support.microsoft.com/kb/147706>)
 - 299656: “How to prevent Windows from storing a LAN Manager hash of your password in Active Directory and local SAM databases” (<http://support.microsoft.com/kb/299656>)
 - 239869: “How to Enable NTLM 2 Authentication” (<http://support.microsoft.com/kb/239869>)
 - 102716: “User Authentication with Windows NT” (<http://support.microsoft.com/kb/102716>)
 - 175641: “LMCompatibilityLevel and Its Effects” (<http://support.microsoft.com/kb/175641>)
 - 266280: “Changing User Rights from a Batch File or Command Line” (<http://support.microsoft.com/kb/266280>)
- RFC 1510: “The Kerberos Network Authentication Service (v5)”
- RFC 3244: “Microsoft Windows 2000 Kerberos Change Password and Set Password Protocols”



Note The two articles above can be accessed through the Request for Comments repository. Go to <http://ietf.org/rfc.html> and enter the RFC number in the RFC Number text box.

- “Windows 2000 Kerberos Authentication” white paper (<http://www.microsoft.com/windows2000/techinfo/howitworks/security/kerberos.asp>)
- “Data Protection and Recovery in Windows XP” white paper (<http://www.microsoft.com/technet/prodtechnol/winxppro/support/DataProt.asp>)
- “Troubleshooting Kerberos Delegation” white paper (<http://www.microsoft.com/technet/prodtechnol/windowsserver2003/technologies/security/kerbdel.mspx>)
- “Kerberos Authentication in Windows Server 2003” white paper (<http://www.microsoft.com/windowsserver2003/technologies/security/kerberos/default.mspx>)
- “Microsoft Windows Server 2003: Kerberos Protocol Transition and Constrained Delegation” white paper (<http://www.microsoft.com/technet/prodtechnol/windowsserver2003/technologies/security/constdel.mspx>)

