Chapter 6

Configuring Servers

In this chapter:

In Chapter 4, "Hacking the Registry," you found a variety of registry hacks for customizing Microsoft Windows XP and Windows Server 2003. For example, you might want to customize file associations on a server, personalize the Start menu, or keep your servers' history lists clear.

This chapter focuses more on customizing Windows Server 2003 than on customizing Windows XP. It describes *server hacks* that you can use to customize servers. For example, you can optimize the Server service by editing the registry, you can configure Kerberos troubleshooting, and more.

Server Customizations

This section contains two useful server customizations. First, it describes how to add comments to server announcements, making it easier for users to discern the purpose of each server they see in My Network Places. Second, it describes how you can optimize the Server service to favor memory usage, network throughput, or a balance of both.

Adding Comments to Server Announcements

Windows broadcasts the contents of the `REG_SZ value srvcomment` in server announcements. This value is in `HKLM\SYSTEM\CurrentControlSet\Services\LanmanServer\Parameters`. If you set this value to a description of the server, such as "Marketing file server," users will see that description in Windows Explorer when they're browsing My Network Places.

Optimizing the Server Service

You can manually optimize the Server service, customizing it to favor memory usage, network throughput, or both. To do so, create the `REG_DWORD` value `Size`, if it doesn't already exist, in the key `HKLM\SYSTEM\CurrentControlSet\Services\lanmanserver`. Set this to one of the values described in the following list:

- **0x01.** Minimize memory usage
- **0x02.** Balance memory and network throughput
- **0x03.** Maximize network throughput

Authentication

The following sections contain customizations that help you troubleshoot and optimize authentication. The section "Configuring Kerberos" describes how to configure Kerberos for troubleshooting. The section "Disabling Global Catalog Requirement" describes how to remove the requirement of having a Global Catalog server at remote sites. Finally, the section "Enabling Verbose Winlogon Messages" describes how you can get more information from Winlogon for troubleshooting.

Configuring Kerberos

Kerberos is an authentication mechanism that is used to verify user or host identity. Kerberos is the preferred authentication method for services in Windows Server 2003. If you are running Windows Server 2003, you can modify Kerberos parameters to help troubleshoot Kerberos authentication issues or to test the Kerberos protocol. After you finish troubleshooting or testing the Kerberos protocol, remove any registry entries that you added. Otherwise, your computer's performance might be affected.

Table 6-1 describes the values you can configure in the key `HKLM\SYSTEM\CurrentControlSet\Control\Lsa\Kerberos\Parameters` and that are `REG_DWORD` values.

Table 6-1 `Control\Lsa\Kerberos\Parameters`

Name	Value	Description
`SkewTime`	5 (minutes)	This value is the maximum time difference that is permitted between the client computer and the server that accepts Kerberos authentication. In Windows Server 2003 checked build version, the default SkewTime value is two hours. A checked build version—also known as a debug version—of the Windows operating system is used in production and testing environments. This kind of build helps trace the cause of problems in system software by turning on many debugging checks in the operating system code and in the system drivers. These debugging checks help the checked build identify internal inconsistencies as soon as they occur. A checked build has many compiler optimizations turned off and is larger and runs more slowly than an end-user version of Windows does. An end-user version of Windows is also known as a free build version or a retail-build version. In a free build version, debugging information is removed, and Windows is built with full compiler optimizations. A free build version is also faster and uses less memory than a checked build version does.
`LogLevel`	0	This value indicates whether events are logged in the system event log. If this value is set to any non-zero value, all Kerberos-related events are logged in the system event log.
`MaxPacketSize`	1465 (bytes)	This value is the maximum User Datagram Protocol (UDP) packet size. If the packet size exceeds this value, TCP is used.
`StartupTime`	120 (seconds)	This value is the time that Windows waits for the Key Distribution Center (KDC) to start before Windows gives up.
`KdcWaitTime`	10 (seconds)	This value is the time Windows waits for a response from a KDC.
`KdcBackoffTime`	10 (seconds)	This value is the time between successive calls to the KDC if the previous call failed.

Table 6-1 Control\Lsa\Kerberos\Parameters

Name	Value	Description
KdcSendRetries	3	This value is the number of times that a client will try to contact a KDC.
DefaultEncryption-Type	23 (decimal) or 0x17 (hexadecimal)	This value indicates the default encryption type for pre-authentication.
FarKdcTimeout	10 (minutes)	This is the time-out value that is used to invalidate a domain controller from a different site in the domain controller cache.
NearKdcTimeout	30 (minutes)	This is the time-out value that is used to invalidate a domain controller in the same site in the domain controller cache.
StronglyEncrypt-Datagram	FALSE	This value contains a flag that indicates whether to use 128-bit encryption, as opposed to weaker encryption, for datagram packets.
MaxReferralCount	6	This value is the number of KDC referrals that a client pursues before the client gives up.
KerbDebugLevel	1 (for Windows Server 2003 checked build version), 0 (for Windows Server free build version)	This value indicates whether debug logging is on (1) or off (0).
MaxTokenSize	12000 (Decimal)	This value is the maximum value of the Kerberos token. Microsoft recommends that you set this value to less than 65535.
SpnCacheTimeout	15 (minutes)	This value is the lifetime of the Service Principal Names (SPN) cache entries. On domain controllers, the SPN cache is disabled.
S4UCacheTimeout	15 (minutes)	This value is the lifetime of the S4U negative cache entries that are used to restrict the number of S4U proxy requests from a particular computer.
S4UTicketLifetime	15 (minutes)	This value is the lifetime of tickets that are obtained by S4U proxy requests.
RetryPdc	0 (false) Possible values: 0 (false) or any non-zero value (true)	This value indicates whether the client will contact the primary domain controller for Authentication Service Requests (AS_REQ) if the client receives a password expiration error.

Table 6-1 `Control\Lsa\Kerberos\Parameters`

Name	Value	Description
`RequestOptions`	Any RFC 1510 value	This value indicates whether there are additional options that must be sent as KDC options in Ticket Granting Service requests (TGS_REQ).
`ClientIpAddress`	0 Possible values: 0 (false) or any non-zero value (true). (This setting is 0 because of Dynamic Host Configuration Protocol and network address translation issues.)	This value indicates whether a client IP address will be added in AS_REQ to force the Caddr field to contain IP addresses in all tickets.
`TgtRenewalTime`	600 (seconds)	This value is the time that Kerberos waits before it tries to renew a Ticket Granting Ticket (TGT) before the ticket expires.
`AllowTgtSessionKey`	0 Possible values: 0 (false) or any non-zero value (true)	This value indicates whether session keys are exported with initial or with cross realm TGT authentication. The default value is false for security reasons.

Table 6-2 describes the values that you can configure in the key `HKLM\SYSTEM\CurrentControlSet\Services\Kdc`. (Create the subkey `Kdc` if it doesn't exist.)

Table 6-2 `Services\Kdc`

Name	Value	Description
`KdcUseClientAddresses`	0 Possible values: 0 (false) or any non-zero value (true)	This value indicates whether IP addresses will be added in the Ticket Granting Service Reply (TGS_REP).
`KdcDontCheckAddresses`	0 Possible values: 0 (false) or any non-zero value (true)	This value indicates whether IP addresses for the TGS_REQ and the TGT Caddr field will be checked.
`NewConnectionTimeout`	50 (seconds)	This value is the time that an initial TCP endpoint connection will be kept open to receive data before it disconnects.

Table 6-2 `Services\Kdc`

Name	Value	Description
`MaxDatagramReplySize`	1465 (decimal, bytes)	This value is the maximum UDP packet size in TGS_REP and Authentication Service Reply (AS_REP) messages. If the packet size exceeds this value, the KDC returns a KRB_ERR_RESPONSE_TOO_BIG message that requests that the client switch to TCP.
`KdcExtraLogLevel`	2 Possible values: 1 (decimal) or 0x1 (hexadecimal): Audit SPN unknown errors. 2 (decimal) or 0x2 (hexadecimal): Log PKINIT errors. (PKINIT is an Internet Engineering Task Force [IETF] Internet draft for "Public Key Cryptography for Initial Authentication in Kerberos.") 4 (decimal) or 0x4 (hexadecimal): Log all KDC errors.	This value indicates what information the KDC will write to event logs and to audits.
`KdcDebugLevel`	1 (for checked build), 0 (for free build)	This value indicates whether debug logging is on (1) or off (0). If the value is set to 0x10000000 (hexadecimal) or 268435456 (decimal), specific file or line information will be returned in the data field of KERB_ERRORS as PKERB_EXT_ERROR errors during a KDC processing failure.

Disabling Global Catalog Requirement

Placement of Global Catalog servers in remote sites is usually desired to improve performance of user logon time, searches, and other actions requiring communication with Global Catalog servers, and to reduce wide area network (WAN) traffic. However, to reduce administrative intervention, hardware requirements, and other related overhead, you might not always want to locate a Global Catalog server at a remote site. This is especially relevant in environments that have a large number of sites that could experience substantially increased hardware costs when the size of the sites might not

justify that hardware and administration. The problem is that logons require the domain controller authenticating the user to contact a Global Catalog server to determine whether the user is a member of any universal groups. If the remote office does not have a Global Catalog server and a Global Catalog server cannot be contacted (for various reasons), then the user's logon request might fail (based on the rules stated earlier).

Windows Server 2003 offers an alternative to universal group caching. When this is enabled for a site, users who log on while a Global Catalog server is online can continue to do so if the Global Catalog server is inaccessible at the next logon.

To eliminate the need for a Global Catalog server at a site and to avoid potential denial of user logon requests, enable logons when a Global Catalog server is not available. You must configure this setting on the domain controller that performs the user authentication. To do that, add the `REG_DWORD` value `IgnoreGCFailures` to `HKLM\SYSTEM\CurrentControlSet\Control\Lsa`. Set this value to 0x01. After changing this value, you must restart the domain controller.

Caution The universal groups setting causes potential security vulnerabilities. Universal groups should not be used because if a user is a member of a universal group and the group is denied access to a resource, the key turns off enumeration of universal groups. The result is that the universal group SID is not added to the user's token, and the user could have access to the resource.

Enabling Verbose Winlogon Messages

You can configure Windows so that you receive verbose startup, shutdown, logon, and logoff status messages. Verbose status messages might be helpful when you are troubleshooting slow startup, shutdown, logon, or logoff behavior. To enable verbose status messages, create the `REG_DWORD` value `verbosestatus` in the key `HLKM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System`. Set this value to `0x01`. Note that Windows doesn't display status messages if the value `DisableStatusMessages` exists in the key `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System`.

Internet Information Services

The following sections describe how to customize Internet Information Services (IIS) 6.0. The section "Configuring Http.sys" describes settings for customizing Http.sys. The section "Using Incremental Site ID Numbers" describes customizing IIS to use incremental site identification numbers.

Configuring Http.sys

In Windows Server 2003, Http.sys is the kernel mode driver that handles HTTP requests. Several registry values can be configured according to specific requirements. Table 6-3 describes the settings that you can configure in the key `HKLM\System\CurrentControlSet\Services\HTTP\Parameters`. All of these values are `REG_DWORD` values.

Caution Microsoft considers customizing the values `MaxConnections`, `MaxEndpoints`, `MaxFieldLength`, `MaxRequestBytes`, `UrlSegmentMaxCount`, `UriMaxUriBytes`, `UriScavengerPeriod`, and `UrlSegmentMaxLength` to be very risky. Test your changes in a safe environment before introducing them into a production environment.

Table 6-3 Http.sys Parameters

Name	Value	Range	Description
`AllowRestrictedChars`	0	Boolean	If the value is non-zero, Http.sys accepts hex-escaped chars in request URLs that decode to U+0000–U+001F and U+007F–U+009F ranges.
`EnableNonUTF8`	1	Boolean	If the value is 0, Http.sys accepts only UTF-8-encoded URLs. If non-zero, Http.sys also accepts ANSI- or DBCS-encoded URLs in requests.
`FavorUTF8`	1	Boolean	If the value is non-zero, Http.sys always tries to decode a URL as UTF-8 first; if that conversion fails and EnableNonUTF8 is non-zero, Http.sys then tries to decode the URL as ANSI or DBCS. If the value is 0 (and EnableNonUTF8 is non-zero), Http.sys tries to decode the URL as ANSI or DBCS; if that is not successful, it tries a UTF-8 conversion.

Table 6-3 Http.sys Parameters

Name	Value	Range	Description
MaxConnections	MAX_ULONG	1024–2031616 (connections)	This value overrides the MaxConnections calculation in the driver. This is primarily a function of memory.
MaxEndpoints	0	0–1024	This value represents the maximum number of current end point objects that are permitted. The default value of zero implies that the maximum is computed from available memory.
MaxFieldLength	16384	64–65534 (bytes)	This value sets an upper limit for each header. See MaxRequestBytes.
MaxRequestBytes	16384	256–16777216 (bytes)	This value determines the upper limit for the total size of the Request line and the headers.
MaxFieldLength	16384	64–65534 (bytes)	This value sets an upper limit for each header.
PercentUAllowed	1	Boolean	If the value is non-zero, Http.sys accepts the %uNNNN notation in request URLs.
UrlSegment-MaxCount	255	0–16383 (segments)	This value represents the maximum number of URL path segments. If the value is 0, the count is bounded by the maximum value of a ULONG.
UriEnableCache	1	Boolean	If the value is non-zero, the Http.sys response and fragment cache is enabled.
UriMaxUriBytes	262144 (bytes)	4096–16777216 (bytes)	Any response that is greater than this value is not cached in the kernel response cache.

Table 6-3 Http.sys Parameters

Name	Value	Range	Description
UriScavenger-Period	120 (seconds)	10–0xFFFFFFFF (seconds)	This value determines the frequency of the cache scavenger. Any response or fragment that has not been accessed in the number of seconds equal to the value of UriScavengerPeriod is flushed.
UrlSegment-MaxLength	260	0–32766 (chars)	This value represents the maximum number of characters in a URL path segment (the area between the slashes in the URL). If zero, it is the length that is bounded by the maximum value of a ULONG.

Using Incremental Site ID Numbers

When you create a new Web site through the Internet Information Services (IIS) Manager in IIS 6.0, an identification number for that site is automatically generated and stored in the metabase. The identification number is based on the name of the Web site. In earlier versions of IIS, the Web site identification numbers were incremental. The new design ensures that all IIS 6.0 servers in a Web farm have a good chance of generating the same site identification numbers for sites with the same name. To use incremental site identification numbers, set the REG_DWORD value IncrementalSiteIDCreation to 0x01 in the key HKLM\SOFTWARE\Microsoft\InetMgr\Parameters. Create the value if it does not already exist.

Network Connections

This section contains two network customizations for Windows Server 2003. The section "Enabling IP Forwarding" shows you how to turn on IP forwarding in Windows Server 2003. The section "Changing MTU Settings" describes how you can optimize PPP and VPN connections by changing the MTU sizes in the registry.

Enabling IP Forwarding

By default, TCP/IP forwarding is disabled in Windows Server 2003. To enable IP forwarding, add the REG_DWORD value IPEnableRouter to the key HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters. Set this value to 0x01. This enables IP forwarding on all network adapters installed on the server.

Changing MTU Settings

This section describes how to change the default maximum transfer unit (MTU) size settings for Point-to-Point Protocol (PPP) connections or for virtual private network (VPN) connections. Windows uses a fixed MTU size of 1500 bytes for all PPP connections and a fixed MTU size of 1400 bytes for all VPN connections. These are the default settings for PPP clients, for VPN clients, for PPP servers, and for VPN servers that are running Routing and Remote Access.

PPP connections are connections such as modem connections and Integrated Services Digital Network (ISDN) connections, or direct cable connections over null serial cable or parallel cable. VPN connections are Point-to-Point Tunneling Protocol (PPTP) connections or Layer 2 Tunneling Protocol (L2TP) connections.

Note Use the methods discussed here to edit the registry to modify the MTU size settings. If you experience any problems or any performance-related issues after you modify the MTU size settings, remove the registry keys that you added.

To change the MTU settings for PPP and VPN connections, add the `REG_DWORD` values `ProtocolType`, `PPPProtocolType`, and `ProtocolMTU`, as well as `TunnelMTU`, to `HKLM\SYSTEM\CurrentControlSet\Services\Ndiswan\Parameters\Protocols\0`. (Create this subkey if it doesn't already exist.) Table 6-4 describes what settings to put in each value. `ProtocolType` and `PPPProtocolType` are required values. You can add either `ProtocolMTU`, `TunnelMTU`, or both.

Table 6-4 MTU Settings for PPP

Name	Setting	Description
`ProtocolType`	`0x800`	Set the protocol type.
`PPPProtocolType`	`0x21`	Set the tunnel protocol type.
`ProtocolMTU`	*Size*	Replace *Size* with the size of the MTU that you want to use for PPP connections.
`TunnelMTU`	*Size*	Replace *Size* with the size of the MTU that you want to use for VPN connections.

Shutdown Event Tracker

Shutdown Event Tracker is a feature of Windows Server 2003 that provides a way for IT professionals to track why users restart or shut down their computers. The feature captures the reasons users give for restarts and shutdowns to help create a comprehensive picture of an organization's system environment. It does not document why users choose other options, such as Log off or Hibernate. In Windows Server 2003,

Shutdown Event Tracker is enabled by default, and its tracking is a routine part of the computer shutdown process. This section describes the registry settings that you can use to configure the feature.

More Info For more information about the tools you can use with Shutdown Event Tracker, see *http://www.microsoft.com/resources/documentation/WindowsServ/2003/all/techref/en-us/Default.asp?url=/Resources/Documentation/windowsserv/2003/all/techref/en-us/w2k3tr_set_tools.asp*.

Shutdown Event Tracker interacts with the registry in the following ways:

- The expected shutdown dialog box reads custom shutdown reasons from the registry.
- Remote Shutdown (Shutdown.exe) reads custom shutdown reasons from the registry. It also writes bulk annotations to the registry and deletes keys from the registry.
- Custom Reason Editor (CustReasonEdit.exe) writes custom shutdown reasons to the registry.
- The unexpected shutdown dialog box reads from the registry to determine if the previous shutdown was unexpected.
- The Event Log service writes the Shutdown Event Tracker heartbeat to the registry and then deletes it just before a normal shutdown occurs. Upon restarting, it verifies whether the heartbeat is present and, if so, writes the DirtyShutdown key to the registry. Heartbeat is a time stamp interval, written once a minute, that indicates that Shutdown Event Tracker is still enabled.

The following list describes the values that Shutdown Event Tracker uses. (Unless otherwise noted, these values are in `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Reliability`.)

- **BugcheckString.** This value contains the bug check string information that is used to fill in the unexpected shutdown dialog box comment field (which appears at logon after an unexpected shutdown) if the previous shutdown was caused by a system failure (also know as a system crash).
- **DirtyShutdown.** This value is set during event log startup. It indicates whether a previous shutdown was expected.
- **LastAliveStamp.** This value is cleared during shutdown. It indicates the date and time of the previous unexpected shutdown if it is present during startup.

- **ReliabilityGUID.** This value enables a GUID to be written to the system state data file in order to uniquely identify the computer this file came from. It is not possible to physically identify the computer using this GUID, but it is possible to see how many different computers sent files and how many distinct reports were submitted by each computer. If the GUID is deleted from the registry, a new GUID is generated when a new system state data (*.xml*) file is created in the %SystemRoot%\System32\LogFiles\Shutdown\ directory at the time of an unplanned shutdown.
- **ShutdownIgnorePredefinedReasons.** This value prevents the predefined or built-in shutdown reasons from being displayed. If at least one custom reason is defined in the registry and this key is set to `0x01`, the built-in reasons are not displayed.
- **TimeStampInterval.** This value defines how often `LastAliveStamp` (or heartbeat) is written to the registry. By default, it is written every minute in Windows Server 2003.
- **UserDefined.** This subkey contains custom reasons stored as values. To add custom reasons, the user must define one value for each reason. Each reason has a major and minor code that uniquely identifies the reason.

More Info For more information about Custom Reason Editor, see the *Microsoft Windows Resource Kit* Tools Read Me.

- **ShutdownReasonUI.** Shutdown Event Tracker references the Group Policy key for this value first. If the Group Policy key is not present, then this key can be configured as `0x00` (off) or `0x01` (on). If the Group Policy key is not present and this key is invalid or missing, then Shutdown Event Tracker is off.

Note You can use Group Policy to manage Shutdown Event Tracker. Its Group Policy settings are in Computer Configuration\Administrative Templates\System.