



realtimepublishers.com[™]

The How-To Guide[™] To

Windows Server 2003 Terminal Services



triCerati

Greyson Mitchem

Introduction to Realtimepublishers

by Sean Daily, Series Editor

The book you are about to enjoy represents an entirely new modality of publishing and a major first in the industry. The founding concept behind Realtimepublishers.com is the idea of providing readers with high-quality books about today's most critical technology topics—at no cost to the reader. Although this feat may sound difficult to achieve, it is made possible through the vision and generosity of a corporate sponsor who agrees to bear the book's production expenses and host the book on its Web site for the benefit of its Web site visitors.

It should be pointed out that the free nature of these publications does not in any way diminish their quality. Without reservation, I can tell you that the book that you're now reading is the equivalent of any similar printed book you might find at your local bookstore—with the notable exception that it won't cost you \$30 to \$80. The Realtimepublishers publishing model also provides other significant benefits. For example, the electronic nature of this book makes activities such as chapter updates and additions or the release of a new edition possible in a far shorter timeframe than is the case with conventional printed books. Because we publish our titles in “real-time”—that is, as chapters are written or revised by the author—you benefit from receiving the information immediately rather than having to wait months or years to receive a complete product.

Finally, I'd like to note that our books are by no means paid advertisements for the sponsor. Realtimepublishers is an independent publishing company and maintains, by written agreement with the sponsor, 100 percent editorial control over the content of our titles. It is my opinion that this system of content delivery not only is of immeasurable value to readers but also will hold a significant place in the future of publishing.

As the founder of Realtimepublishers, my *raison d'être* is to create “dream team” projects—that is, to locate and work only with the industry's leading authors and sponsors, and publish books that help readers do their everyday jobs. To that end, I encourage and welcome your feedback on this or any other book in the Realtimepublishers.com series. If you would like to submit a comment, question, or suggestion, please send an email to feedback@realtimepublishers.com, leave feedback on our Web site at <http://www.realtimepublishers.com>, or call us at 800-509-0532 ext. 110.

Thanks for reading, and enjoy!

Sean Daily
Founder & Series Editor
Realtimepublishers.com, Inc.

Introduction to Realtimepublishers..... i

Chapter 1: Installation and Configuration of Terminal Services1

Requirements for a Terminal Server Environment.....2

How to Install the Terminal Server Role3

 During an Unattended Installation of Windows4

 In a Sysprep Image.....5

 Via Remote Installation Services.....6

How to Install Terminal Server Licensing.....6

 Manually7

 During an Unattended Installation of Windows8

How to Activate a Terminal Server License Server8

 Types of Terminal Server CALs.....10

How to Add Terminal Server CALs to a License Server11

How to Configure Licensing Mode on a Terminal Server12

 Manually12

 Via Group Policy12

 Terminal Server License Server Discovery13

How to Override the TSLs Discovery Process.....14

 Via Registry Setting.....14

 Via GUI.....15

 Via Group Policy16

How to Restrict Access to Terminal Server License Servers17

Summary18

Chapter 2: Application Installation and Configuration.....19

Application Compatibility Subsystems.....19

 Registry Mapping.....20

 INI File Mapping20

 Root Drive.....21

How to Toggle Between Install and Execute Mode22

Windows Installer Service23

How to Install MSI Packages on Terminal Servers23

 Via IntelliMirror/Group Policy23

 Create a Share24

Create Administrative Installations.....	24
Add the Packages to a GPO	25
Filtering Applications	26
MSIEXEC Command-Line Reference	28
Application Compatibility Scripts	29
How to Add an Application Compatibility Script that Does Not Require a Root Drive...29	
How to Add an Application Compatibility Script that Requires a Root Drive	30
User Logon Process and Scripts	30
How to Invoke a Per-User Logon Script.....	32
How to Invoke a Logon Script via Group Policy	33
Summary	34
Chapter 3: User Session and Environment Configuration.....	35
Session Length Limits.....	35
Types of User Profiles	36
Client Device Redirection and Resource Mapping.....	36
How to Configure User Session Timeouts.....	37
On a Per-User Basis via Active Directory Users and Computers.....	37
On a Per-User Basis via Active Directory Scripting Interface	39
On a Per-User Basis via Group Policy.....	40
On a Per-Server Basis via GUI.....	41
On a Per-Server Basis via Group Policy.....	42
How to Configure Client Device Redirection and Resource Mapping.....	43
On a Per-User Basis via Active Directory Users and Computers.....	43
On a Per-User Basis via ADSI.....	44
On a Per-Server Basis via GUI.....	44
On a Per-Server Basis via Group Policy.....	47
How to Configure User Profiles	48
On a Per-User Basis via Active Directory Users and Computers.....	48
On a Per-User Basis via ADSI.....	49
Via Group Policy	49
Per Server (Local Machine Policy).....	51
Third-Party Products for Profile Management	51
User Profile Hive Cleanup Service	52

How to Install UPH Clean	52
How to Manage Printing in a Terminal Server Environment	52
Microsoft Fallback Printer Driver.....	52
Third-Party Products for Printer Driver Management	54
Command-Line Reference	55
Summary	55
Chapter 4: Management, Load Balancing, and Optimization.....	56
Requirements for Logging on to Terminal Server	56
How to Manage User Rights Assignments	57
How to Manage RDP Protocol Permissions	59
Via Group Policy	60
How to Manage the Remote Desktop Users Group.....	60
Manually	60
Via Script	61
Via Group Policy	62
How to Manage the <i>Deny this user permissions to log on to any Terminal Server Setting</i>	63
Manually	63
Via ADSI	64
Managing User Sessions	64
How to Configure Remote Control Options	65
On a Per-User Basis via ADSI.....	66
On a Per-Server Basis via GUI.....	66
Via Group Policy	67
Order of Precedence.....	68
How to Control a User Session.....	69
Session Directory	71
How to Join a Terminal Server to a Session Directory.....	73
Windows System Resource Manager	74
How to Install WSRM	74
How to Configure WSRM	75
Third-Party Products for Server Resource Management	76
Summary	78

Copyright Statement

© 2005 Realtimepublishers.com, Inc. All rights reserved. This site contains materials that have been created, developed, or commissioned by, and published with the permission of, Realtimepublishers.com, Inc. (the "Materials") and this site and any such Materials are protected by international copyright and trademark laws.

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtimepublishers.com, Inc or its web site sponsors. In no event shall Realtimepublishers.com, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials (including but not limited to the text, images, audio, and/or video) may not be copied, reproduced, republished, uploaded, posted, transmitted, or distributed in any way, in whole or in part, except that one copy may be downloaded for your personal, non-commercial use on a single computer. In connection with such use, you may not modify or obscure any copyright or other proprietary notice.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtimepublishers.com and the Realtimepublishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtimepublishers.com, please contact us via e-mail at info@realtimepublishers.com.

[**Editor's Note:** This eBook was downloaded from Realtime Nexus—The Digital Library. All leading technology guides from Realtimepublishers can be found at <http://nexus.realtimepublishers.com>.]

Chapter 1: Installation and Configuration of Terminal Services

Terminal servers can be a vital component of any IT Infrastructure. They can be used to replace hundreds of workstations in a homogeneous computing environment—a call center, for example—or they can be used to complement a workstation-based environment for remote access or disaster recovery. Terminal servers can even be used in smaller environments as a way of quickly setting up and managing workgroups of 5 to 10 users with similar computing needs.

Because of their versatility, and their increased usage in the workplace today, the technologies that make up terminal servers should be familiar to Windows administrators. In addition, administrators should know the steps needed to install and configure Terminal Services. This guide will explore these tasks—it is meant to be not only a manual for learning more about Terminal Services but also a handy reference guide for systems administrators. For a more detailed introduction to Terminal Services, refer to *The Definitive Guide to Windows Server 2003 Terminal Services* (Realtimepublishers.com).

Guide Conventions

The following list highlights conventions that will be used through the guide to refer to common tasks and locations within the Windows operating system (OS):

Registry Locations:

HKLM = HKEY_LOCAL_MACHINE

HKCU = HKEY_CURRENT_USER

HKCR = HKEY_CLASSES_ROOT

HKU = HKEY_Users

Windows Shell Locations:

Start menu—Drill downs within the Start menu will be separated by bars “|”. For example,

Start | All Programs | Administrative Tools | Terminal Server Licensing

Administrative Tools

Start | All Programs | Administrative Tools

Start | Control Panel > Administrative Tools

System Control Panel

Start | Control Panel > System

Right-click the My Computer icon and select Properties

As with any Windows technology, the first step to a successful deployment is a clean, repeatable installation process and a consistent and usable configuration of your servers and their roles. Both Microsoft and many third-party vendors offer tools to help automate your installation and configuration process; the right solution for you will depend on your environment, the size of the deployment, and the budget you have to work with.

This chapter will cover the native options available to you from Microsoft. These include manual installation and configuration, automated installation, and disk duplication. It will introduce you to the Terminal Services entries available in unattend.txt and sysprep.inf files. First, we will explore the basic required components for any Terminal Services implementation.

Requirements for a Terminal Server Environment

There are three basic requirements for any Terminal Services implementation:

- Terminal server
- License server
- Authentication server

In a small workgroup or small office/home office (SOHO) environment, these can all be on one physical server, but in most cases, they will be distributed across two or more servers.

- A **terminal server** is a Windows Server 2003 (WS2K3) system with the Terminal Server role installed. This server is the one that the users will actually be logging onto and will be providing their working environment (in most cases, a Windows Desktop Shell). The size and configuration of this server will be determined by the number of users that it needs to support. All user applications will be installed on this server.
- A **license server** is a WS2K3 system with Terminal Server Licensing Service (TSLS) installed. TSLS is an optional Windows component. In a workgroup environment, TSLS can be installed on any server, even the terminal server itself. In a domain environment, TSLS must be installed on a domain controller.
- An **authentication server** is a WS2K3 system used to authenticate users when they log on. In a workgroup environment, this server will be the local Security Account Manager (SAM) database on the terminal server. In a domain environment, this server will be the domain controller. All user accounts must be set up on the local SAM or in the domain before users can logon to the terminal server.

In a larger enterprise environment, it is common to have more than one terminal server to support a larger number of users. In this case, you will also need a load balancer—either Session Directory or a third-party product—to evenly distribute users across the terminal servers and to reconnect users to the server hosting their session in the event of a connection drop.

How to Install the Terminal Server Role

After the installation of WS2K3, use the Manage Your Server wizard to install the Terminal Server role. The Manage Your Server wizard launches automatically when an administrator logs on to the server, or it can be launched manually from the Start menu. Figure 1.1 shows the Manage Your Server wizard.

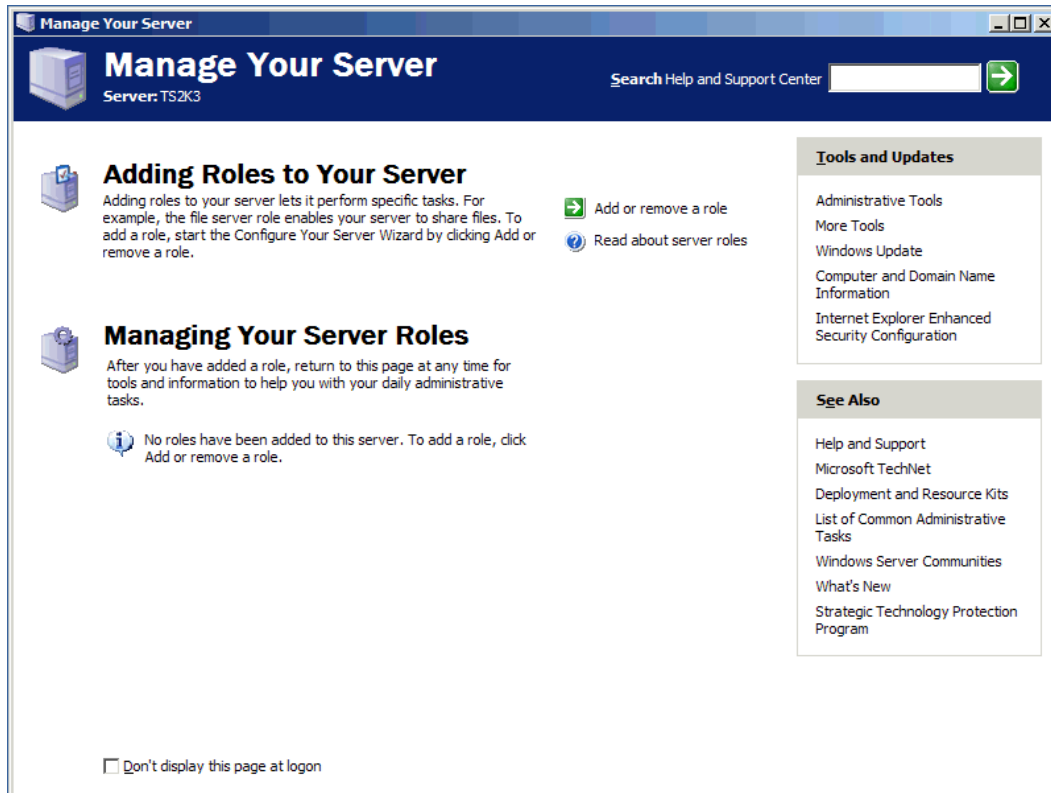


Figure 1.1: The Manage Your Server wizard.

From the window that Figure 1.1 shows, click *Add or remove a role*. You will be presented with a list of available roles (see Figure 1.2). Select *Terminal server*, and click Next.

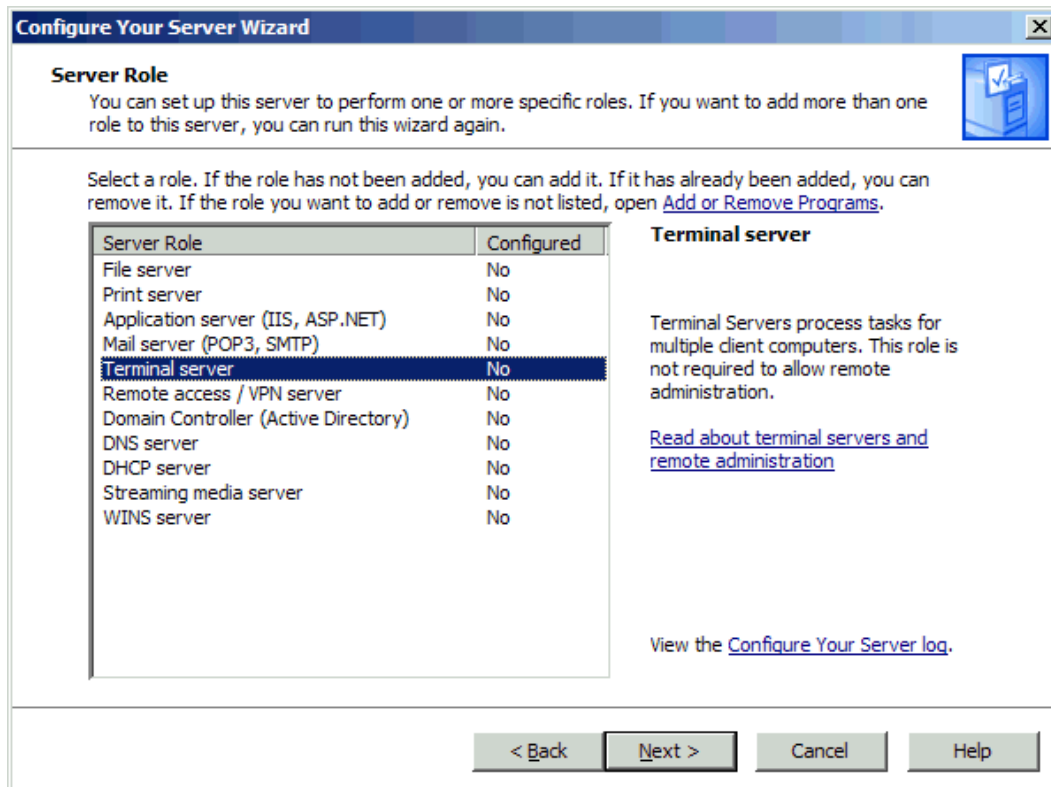



Figure 1.2: Default roles available in WS2K3 Standard Edition.


The wizard will warn you that the installation of Terminal Services will require a reboot. If you are ready to proceed, click Next—the Terminal Server role will be installed and the server will automatically reboot. You may be prompted for your WS2K3 source files during the installation.

 You will not have the option to delay the reboot when installing the Terminal Server role. Once you start the installation, the server will be rebooted automatically.

During an Unattended Installation of Windows

If you are automating your WS2K3 installation, you can automatically include the Terminal Server role in your base installation. To do so, modify the unattend.txt or winnt.sif file that you have created for your unattended installation by adding the following entries:


```
[Components]
TerminalServer=On
```

 The [Components] section may already exist in your unattend.txt file; if so, simply add the TerminalServer entry to it.


 For more information about unattended installations of Windows, see Microsoft's documentation included in the Deployment Tools.cab file (the WS2K3 Service Pack 1—SP1—version of the Deployment Tools.cab can be found at <http://support.microsoft.com/kb/892778>).

This addition will install the Terminal Server role with the default configuration for licensing mode and permissions settings. To override the default settings, add a new section to the file:

```
[TerminalServices]
AllowConnections=1
LicensingMode=PerDevice | PerUser
PermissionsSetting=0 | 1
```


 Later, this chapter will discuss the licensing modes.

The permissions setting determines how to harden your terminal server. Setting it to 0 applies the same permissions as would be used on a normal WS2K3 or Windows XP box. This setting is recommended because it protects critical areas of the file system from being modified by non-administrative users. Setting it to 1 applies the permissions that were used on Windows NT 4.0. You should only use the NT 4.0 permissions settings if you are planning to install a legacy application that requires these relaxed permissions.

 If you add [TerminalServices] AllowConnections=1 to your unattend.txt or winnt.sif file without adding Terminalserver=On to the components section, you will enable the Remote Desktop feature (known as Terminal Services in Remote Administration Mode under Windows 2000—Win2K).

In a Sysprep Image

If you are using Sysprep to prepare server images for disk duplication, simply install the Terminal Server role via either of the earlier methods before running the Sysprep utility.

 For more information about using Sysprep, see Microsoft's documentation at <http://support.microsoft.com/kb/892778>.


Via Remote Installation Services

If you are deploying a base installation of WS2K3 via Remote Installation Services (RIS), you can associate a custom .sif file to a RIS image. This SIF file will need the same entries as described earlier for an unattended installation of WS2K3. To associate a SIF file with a RIS image file, follow these steps:

1. Open Active Directory Users and Computers.
2. In the console tree, right-click the applicable RIS server.
3. Click Properties, then select the Remote Install tab.
4. Click Advanced Settings, then select the Images tab.
5. Click Add.
6. Click *Associate a new answer file to an existing image*, then click Next.
7. Click one of the following options:
 - Windows image sample files
 - Another remote installation server
 - An alternate location
8. Select the installation image to which the answer file will be associated, then click Next.
9. Select the answer file that you want to associate with the installation image, then click Next.
10. Enter the description and Help text, then click Next.
11. Review the settings summary, then click Finish to complete the process.

How to Install Terminal Server Licensing

WS2K3 will stop accepting Terminal Services connections 120 days after the first non-administrator logs on to the server via Terminal Services. To continue to use the terminal server after this 120-day trial period, the server must be able to locate a license server on the network.

 Even if you use Citrix MetaFrame to enhance your Terminal Services deployment, you will still need to purchase and maintain Microsoft Terminal Server Client Access Licenses (CALs) and confirm that your terminal servers can locate a license server on the network. This purchase is in addition to the MetaFrame Presentation Server licenses that you must purchase from Citrix.

Manually

To install TSLS manually, use the Add/Remove Programs control panel applet, then select Add/Remove Windows Components. From the list of available components, select the Terminal Server Licensing check box, and click Next (see Figure 1.3).

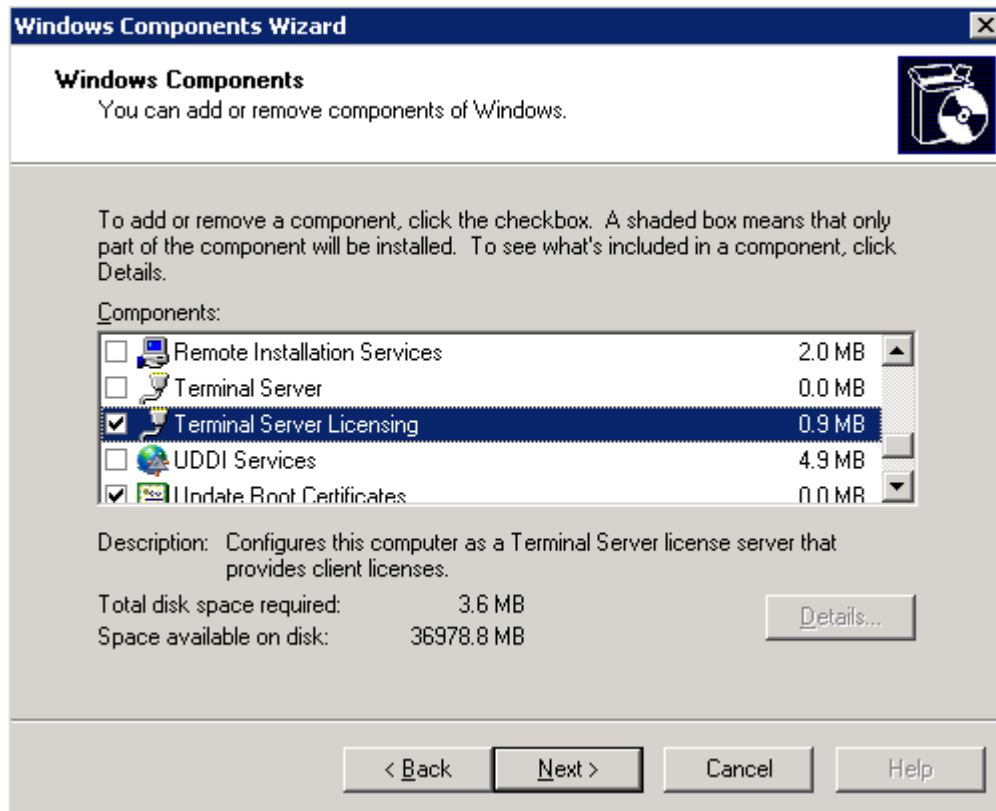


Figure 1.3: Installing Terminal Server Licensing.

If you are installing TSLS on a server on an Active Directory (AD) domain controller, you are presented with two options for the mode of the server: Domain/Workgroup and Enterprise. If you are in a workgroup or non-AD domain, the Enterprise option is not available.

- ❗ Choose your license server mode carefully. Enterprise license servers will be automatically discovered by all terminal servers in the same AD forest as long as they are in the same AD site as the domain controller. Domain license servers will be automatically discovered by all terminal servers in the same domain as the domain controller regardless of site affinity.

Select the mode in which you want the license server to run, and click Next. You may be prompted for your WS2K3 source files during the installation. No reboot is required after installing TSLS.

During an Unattended Installation of Windows

You can also automate the installation of TSLs via either the unattend.txt or sysprep.inf file during an automated installation of Windows. To do so, add the following entry to the [Components] section of the file:

```
[Components]
    Licenser=On
```

This entry is rarely used, as you still need to manually activate the license server, and typically even a large enterprise will have only a few TSLs servers.

How to Activate a Terminal Server License Server

After you install TSLs, the license server must be activated by contacting the Microsoft Clearinghouse. To do so, launch the TSLs administrative tool, right-click the server, and click Activate Server. The Terminal Server License Server Activation Wizard will launch, offering you three options for contacting Microsoft. Figure 1.4 shows the options in the wizard:

- **Automatic connection**—This method is the easiest way to activate the licensing server. This method requires that the server running TSLs has Internet connectivity on port 443 (Secure Sockets Layer—SSL). Simply fill in the company and contact information, and click Activate.
- **Web Browser**—If the automatic connection method fails, or if the server running TSLs does not have Internet connectivity, you can still activate the server over the Web from another computer. To do so, from a Web browser, go to <https://activate.microsoft.com>, and fill in the company and contact information as well as the unique TSLs ID number that the activate server wizard provides. The Web site will respond with the activation code that you can then enter into the licensing service.
- **Telephone**—If you do not have Internet connectivity, you can contact the Microsoft Clearinghouse by telephone. Select your country/region in the activate server wizard, and the correct phone number will be displayed. Provide the customer service person with your company name, contact information, and server ID code; they will provide you with the activation code. Be sure to either activate the server while still on the phone with the customer service representative or be very careful to record the activation code accurately.

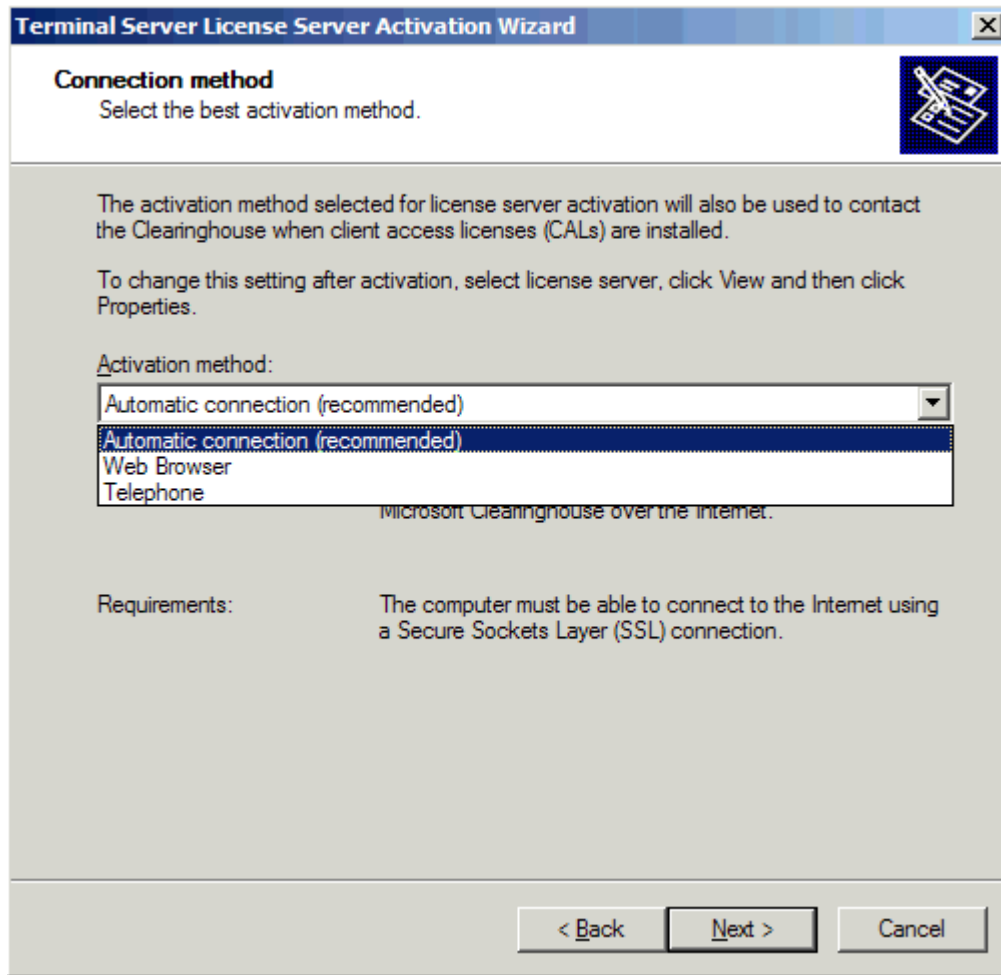


Figure 1.4: Activating a Terminal Services licensing server.

After the license server is activated, it will immediately begin to issue temporary Win2K and WS2K3 terminal server tokens. These temporary tokens give the administrator a 90-day period in which to install the appropriate permanent CALs on the license server so that it can issue permanent tokens.


☞ If you are upgrading a Win2K server that has TSLs installed to WS2K3, you might need to re-activate the licensing service. To do so, select Re-Activate Server from Advanced in the Actions menu in the TSLs administrative tool.

Types of Terminal Server CALs

Terminal Services CALs can be purchased through any of the Microsoft licensing programs—Open, Select, Enterprise, and so on. You can even purchase packs of licenses via retail or OEM channels. In addition to selecting the correct licensing program to purchase your Terminal Services CALs, you must also select the correct type of CALs to buy.

WS2K3 TSLs can issue Terminal Services CALs (also referred to as *tokens*) to both Win2K and WS2K3 terminal servers. Win2K supports only per-device licenses; WS2K3 supports both per-device and per-user licensing. You should choose the appropriate license type for your environment. The following list highlights the available license types:

- WS2K3 Terminal Server Device CALs—WS2K3 terminal servers that are in per device licensing mode will request these licenses from the TSLs server.
- WS2K3 Terminal Server User CALs—WS2K3 terminal servers that are in per user licensing mode will request these licenses.
- WS2K3 Terminal Server External Connector licenses—This license allows unlimited connections by external users to a terminal server running WS2K3.
- Win2K Terminal Services CALs—Terminal servers running Win2K will request these licenses from the licensing server for clients running OSs other than Win2K Professional or Windows XP. You need these licenses only if you have terminal servers running Win2K.
- Win2K Terminal Services Internet Connector license—This license allows as many as 200 simultaneous anonymous connections to a terminal server running Win2K by non-employees across the Internet.
- Win2K Built-In licenses—Clients that are running Win2K Professional or Windows XP are issued a token from the built-in pool of license tokens when connecting to a terminal server running Win2K.

 WS2K3 TSLs will issue “built-in” Terminal Services CALs to Win2K Professional and Windows XP clients that are connecting to Win2K terminal servers. These licenses are included with the purchase of the client OS. WS2K3 terminal server CALs are not built-in and must be purchased separately.

How to Add Terminal Server CALs to a License Server

After you select the licensing program and purchase the appropriate type of Terminal Services CALs for your deployment, the licenses must be added to your license server. The connection method you used to activate the license server will determine the connection method used to install CALs. Before you go to install the CALs, be sure you have either the license code from the retail package or your agreement number for you Select/Open/Enterprise account.


To add a license pack to the license server, launch the TSLs administrative tool, right-click the server to which you want to add the licenses, and select Install Licenses. If you are installing a retail license pack, the type of license will be automatically selected. If, however, you are installing licenses through a Select, Open, or other Microsoft license agreement, you will need to select which type of licenses you want to add. Figure 1.5 shows the Terminal Server CAL Installation Wizard for the automatic connection method.

Figure 1.5: Adding licenses to a TSLs server.

After the licenses are installed, the TSLs will list how many are available and how many have been issued. You can also drill down to see the names of users and devices that have been issued licenses.

How to Configure Licensing Mode on a Terminal Server

The type of CALs that a terminal server requests from the license server is determined by the licensing mode of the terminal server. If the terminal server is in per-device licensing mode, and you only have per-user tokens installed on your TSLS, then the connection will be refused. You can configure the licensing mode of a terminal server either manually or via Group Policy.

 If your terminal server is in per-user licensing mode, it will attempt to validate both per-user and per-device tokens presented by the client. In this mode, if a per-device CAL cannot be validated, the terminal server will request a new per-user token from the TSLS. If, however, the terminal server is in per-device mode, it will not accept per-user CALs presented by the client—even if the client presents a valid per-user CAL, the terminal server will request a new per-device token.

Manually

To manually configure licensing mode on a terminal server, launch the Terminal Services Configuration administrative tool, and select the Server Settings node. Double-click the licensing setting, and choose the appropriate mode. Figure 1.6 shows the options in the tool.

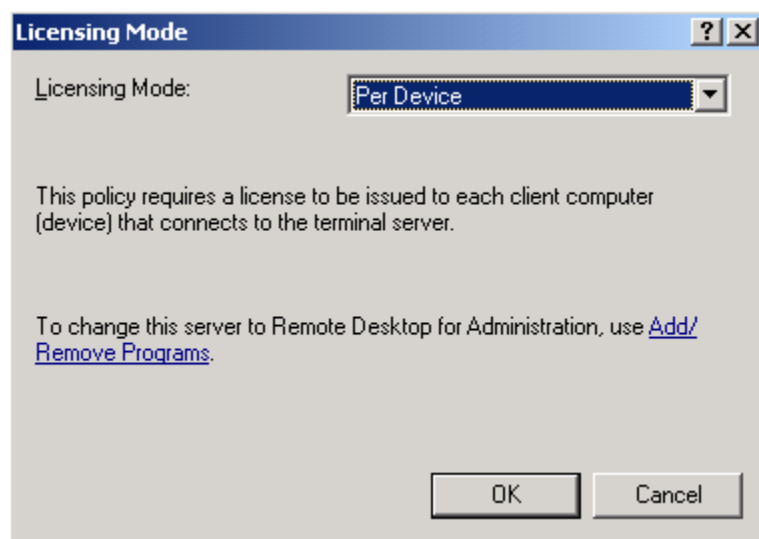


Figure 1.6: Manually specifying the terminal server licensing mode.

Via Group Policy

Under WS2K3 SP1, you can centrally configure licensing mode via Group Policy. To do so, use either the Group Policy Management Console (GPMC) or Active Directory Users and Computers to edit a Group Policy Object (GPO) that applies to your terminal servers. Within the Group Policy Editor, drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services, and double-click the *Set the Terminal Server licensing mode* setting. In the dialog box, set this setting to enabled, and choose the appropriate mode from the drop-down list (see Figure 1.7).

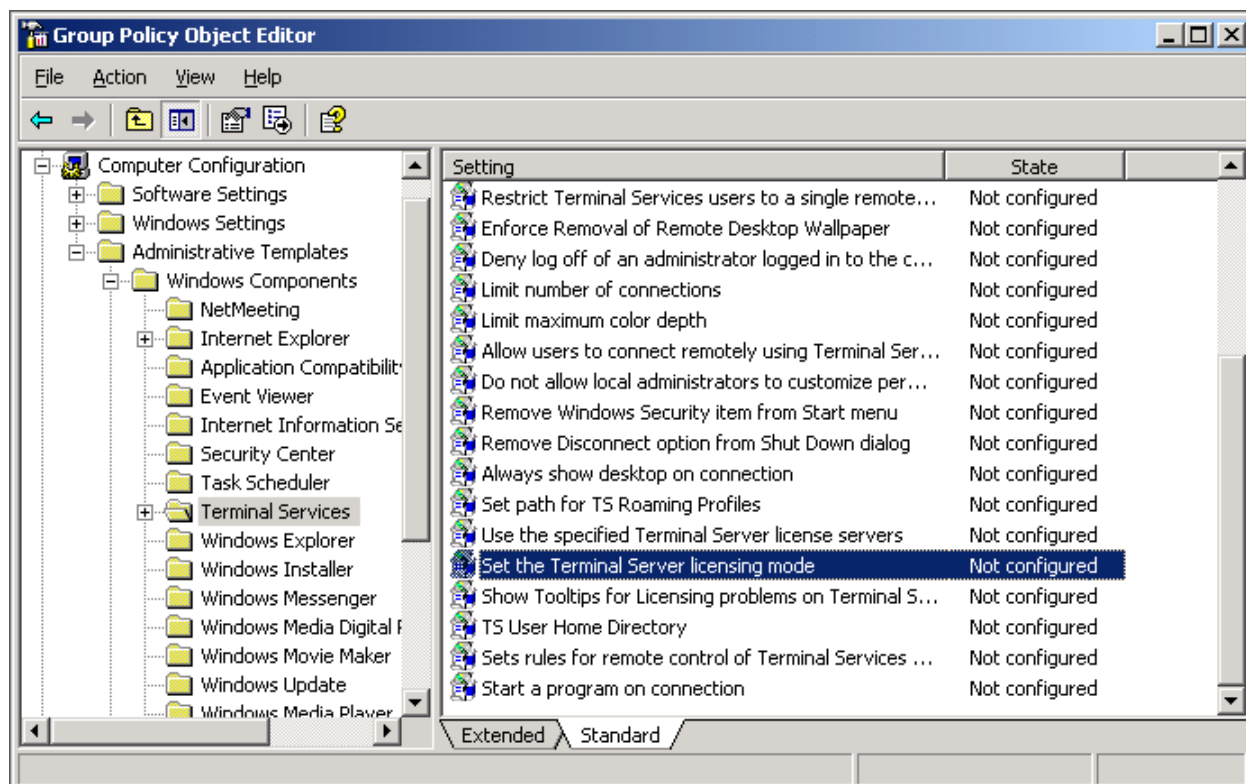


Figure 1.7: Configuring licensing mode via Group Policy.

Terminal Server License Server Discovery

During the boot process, WS2K3 will attempt to find TSLs servers on the network. This process is called *license server discovery*. The license server discovery process is determined by your environment. However, the preference of which license servers to use is the same. Terminal servers will always request CALs from TSLs servers in the following order:

1. Enterprise or domain license servers that are specified in Group Policy
2. Enterprise or domain license servers that are specified in the registry
3. Enterprise license servers in the same AD site and forest as the terminal server (discovered automatically)
4. Domain license servers in the same domain as the terminal server (discovered automatically)
5. Workgroup license servers on the same subnet as the terminal server (discovered automatically, but only used when the terminal server is not in a domain)

In a workgroup setting, the terminal server will first look to determine whether TSLs is installed on the terminal server (common in single-server environments). If TSLs is not installed, the server will then perform a mailslot broadcast to the local subnet and any servers running TSLs will respond. The terminal server will record the name of the license server in the registry for future reference and proceed to request CALs from the TSLs.

In a domain, TSLs must be installed on a domain controller. In fact, the installation wizard will not allow you to install TSLs on a member server. The terminal server will query each domain controller in the forest, starting with domain controllers in its own AD domain and site, using a mailslot request called TermServLicensing. All domain controllers with TSLs installed in domain mode will respond if they are in the same domain as the terminal server. However, if the domain controller has TSLs installed in enterprise mode, it will only respond if the terminal server is in the same AD site as the domain controller. Once the discovery process is complete, the terminal server will record the TSLs server names in its registry.

How to Override the TSLs Discovery Process

Regardless of whether you are in a domain or workgroup environment, you can override the discovery process by listing specific TSLs servers in either the terminal server's registry or in a GPO that applies to the server. In a workgroup setting, overriding the discovery process is required if your TSLs is on a different subnet than your terminal server. In a domain environment, the overrides can be used to force a terminal server to request licenses from a domain mode license server in a different domain or an enterprise license server in a different site.

Via Registry Setting

To manually specify license servers in the registry, follow these steps:

1. Click Start, Run, type regedit, then click OK.
2. Right-click the following key in the registry:
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TermService\Parameters.
3. In the context menu, point to New, then click Key, and type LicenseServers as the name of the new key.
4. Right-click the key you just created, and once again point to New, then click Key. Name this new sub-key one of the following:
 - The NetBIOS name of the server
 - The fully qualified domain name (FQDN) of the server
 - The IP address of the server

Figure 1.8 shows an example of manually configured TSLs servers. In the example, SVR1 and SVR2 are the NetBIOS names of the TSLs servers.

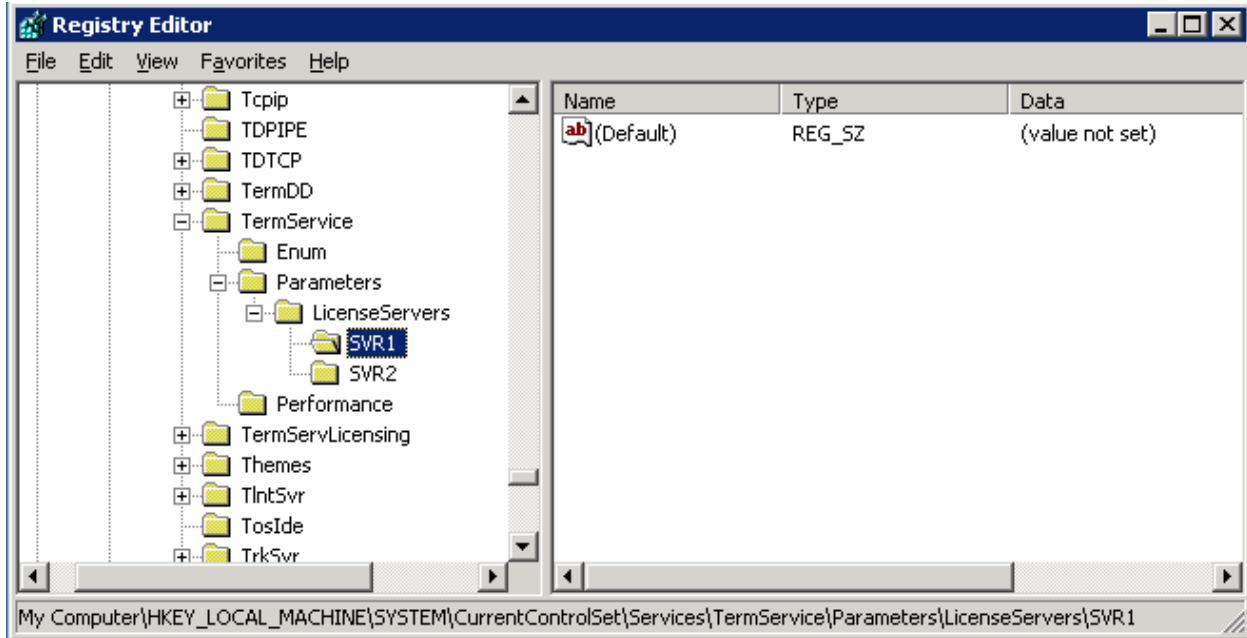


Figure 1.8: Overriding license server discovery by specifying license servers in the registry.

Via GUI

In WS2K3 SP1, Microsoft included a GUI for adding the necessary registry keys. This GUI is found in the Terminal Services Configuration administrative tool. To configure preferred license servers via this GUI, follow these steps:

1. Go to Administrative Tools, and launch Terminal Services Configuration.
2. Select Server Settings.
3. Double-click *License server discovery mode*.
4. Click *Use these license servers*, and enter the NetBIOS names, FQDN, or IP addresses of the license servers in the text box, separating them with commas.
5. Click Check names, make any necessary corrections, and click OK.

Figure 1.9 shows the GUI for configuring preferred license servers.

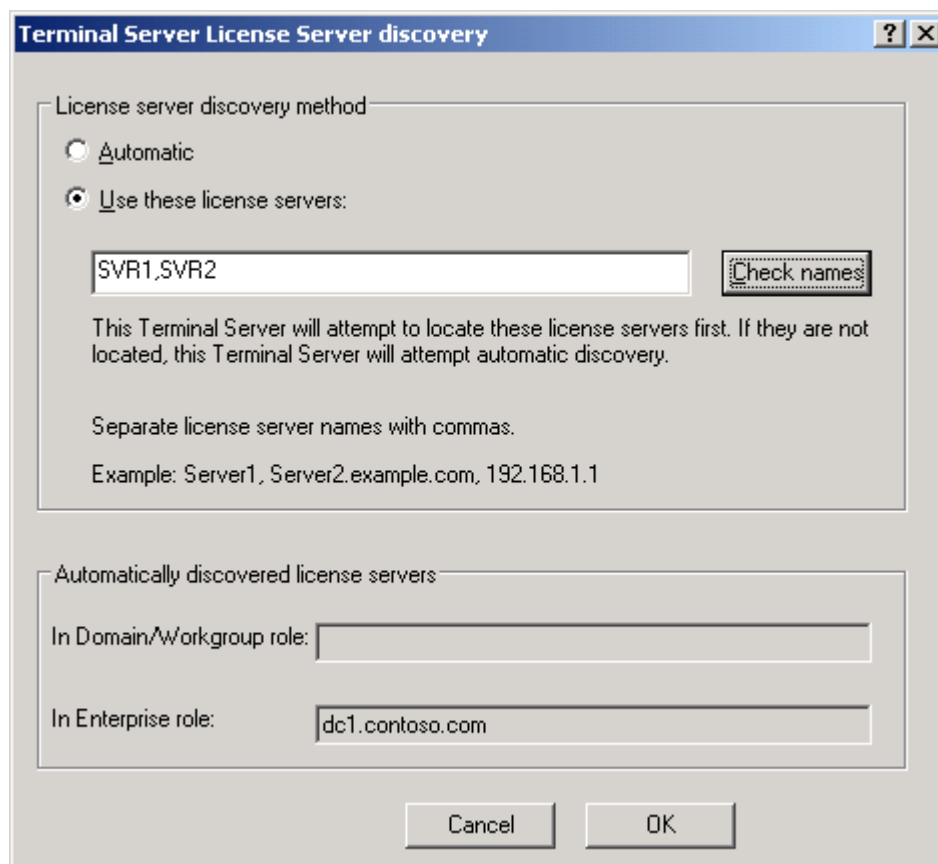



Figure 1.9: Configuring preferred license servers via GUI.

 Configuring preferred licenses servers via either the registry or GUI requires a reboot of the terminal server to take effect.

Via Group Policy

WS2K3 SP1 also includes the ability to centrally control preferred license servers via Group Policy. To do so, use either the GPMC or Active Directory Users and Computers to edit a GPO that applies to your terminal servers. Within the Group Policy Editor, drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services, and double-click the *Use the specified Terminal Server license servers* setting (see Figure 1.10). In the dialog box, set this setting to enabled, and enter the NetBIOS names, FQDN, or IP addresses of the license servers in the text box, separating them with commas.

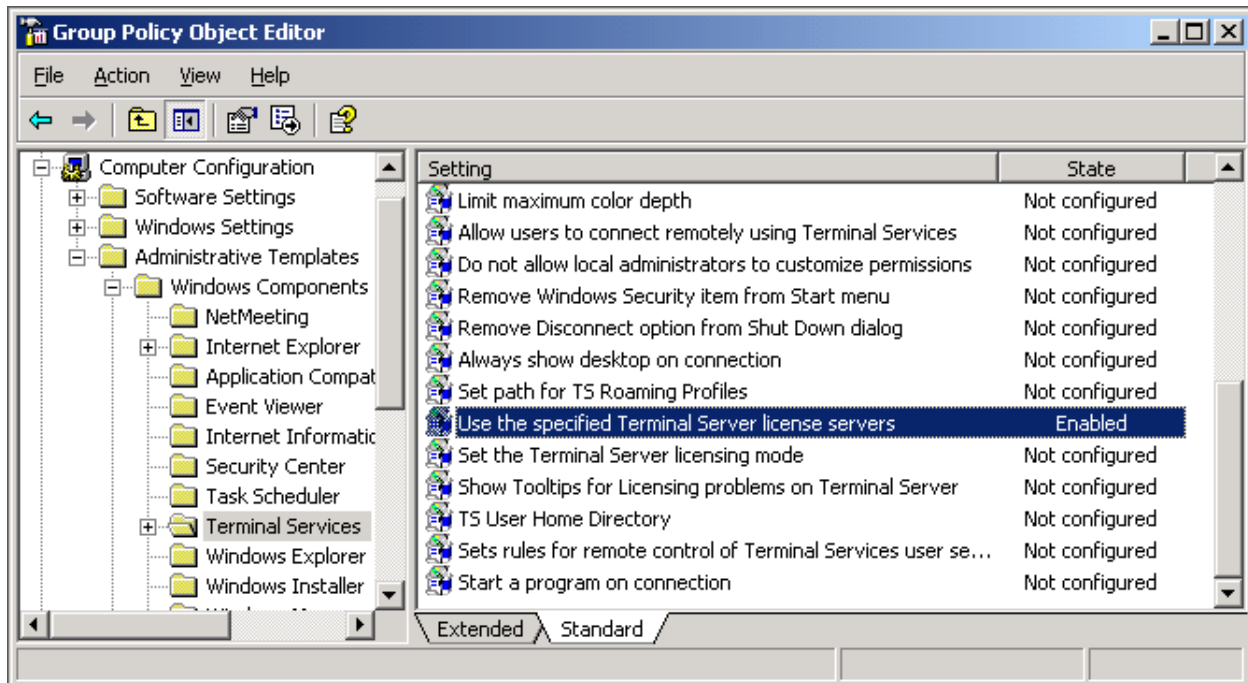


Figure 1.10: Configuring licensing mode via Group Policy.


How to Restrict Access to Terminal Server License Servers

By default, a TSLS will issue tokens to any terminal server that requests them. In a large, environment, this default action can cause problems if unmanaged terminal servers are added to the domain or rogue servers are manually pointed at your license servers. You can, however, restrict access to your license servers by using the License Server Security Group feature. When enabled, a TSLS will only issue tokens to terminal servers that are members of the Terminal Services Computers group.

In a domain environment, this group is a domain local group and is shared among all domain controllers. In a workgroup, this group is a local SAM group on the server running TSLS. If you enable the License Server Security Group in a workgroup environment, TSLS must be installed on the terminal server, and it will then only issue licenses to itself.

To enable the License Server Security Group feature, take the following steps:

1. Use GPMC, Active Directory Users and Computers (in a domain environment), or the Group Policy Editor (gpedit.msc) in a workgroup environment to edit a GPO that applies to your license servers.

 In a domain, this GPO will most likely be your domain controller security policy.

2. Drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Licensing, and set the License Server Security Group setting to enabled.
3. Add your terminal servers to the Terminal Services Computers group.

After policy refresh occurs, the TSLs will only issue tokens to terminal servers that are in the Terminal Services Computers local group. By default, this group is empty, so you must immediately add your terminal servers to it in order for them to continue to receive tokens from the license servers.

Group Policy Reference

To configure terminal server licensing mode:

Computer Configuration | Administrative Templates | Windows Components | Terminal Services, and double-click the *Set the Terminal Server licensing mode*

To override automatic license server discovery:

Computer Configuration | Administrative Templates | Windows Components | Terminal Services, and click *Use the specified Terminal Server license servers*

To enable the License Server Security Group feature:

Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Licensing, and select *License Server Security Group*

Summary

As terminal servers are increasingly utilized in today's workplace, systems administrators must become familiar with terminal server technologies and installation and administrative steps. This chapter provided an overview of the native options available from Microsoft as well as the basic requirements of a Terminal Services implementation. The next chapter will move forward with an exploration of application installation and configuration.

Chapter 2: Application Installation and Configuration

Starting with the release of Windows 2000 (Win2K), Microsoft has included terminal server compatibility in its Certified for Windows logo program, so most current applications can be installed and run in the multi-user environment of a terminal server without modification. However, you should still be familiar with the process of installing applications on a terminal server as well as the application-compatibility subsystems in case you encounter a legacy program that you must integrate into your terminal server environment.


The Windows logo certification specification instructs programmers to take advantage of several Windows component services that would make the application natively compatible with WS2K3 and Terminal Services. The following list quotes and describes the elements of the specification that are of particular interest to the Terminal Services administrator:

- *Do not read from or write to Win.ini, System.ini, Autoexec.bat, or Config.sys on any Windows operating system based on NT technology*—Programs that don't obey this rule might store per-user settings in these per-machine configuration files.
- *Install using a Windows Installer-based package that passes validation testing and Ensure that your application supports advertising*—The Windows Installer service uses a process called *advertising* to ensure that per-user registry keys and files are installed for each user of a computer and not just the user who installed it.
- *Default to My Documents for storage of user-created data*—Compliance with this item ensures that per-user files (documents, macros, templates, and so on) are stored in a per-user location, not in the program's directory.

In the real world, however, systems administrators have to deal with many applications—both current and legacy—that don't adhere to these guidelines or were written before the specification was established. To assist in integrating these types of applications, Terminal Services utilizes several application compatibility subsystems.

Application Compatibility Subsystems


There are three main application compatibility subsystems: registry mapping, INI file mapping, and root drive. The subsystems have two modes—install and execute. Install mode is used during the installation of user applications, and execute mode is the normal operating mode for the terminal server.

 It is often tempting to install Terminal Services purely to overcome the two session limitation imposed by Remote Desktop. You need to be aware that in addition to eliminating the limit on the number of sessions available, Terminal Services enables these application compatibility subsystems, which may effect the way that server-based applications behave.

Registry Mapping

During installation, many applications add registry information to the Current User registry hive (HKCU). If the application does not take advantage of the Windows Installer Service, or have its own way of populating HKCU keys upon launch, only the person installing the application will have the correct values. In a workstation environment, this consideration is not usually an issue as the application is installed under the context of the user that will run it. On a terminal server, however, such is not the case, as many users will run the application on the same server.

Registry mapping takes care of this problem. While in install mode, the terminal server monitors any write actions to HKCU during the install process and replicates the keys to a special subsection of HKLM. Then, while in execute mode, the terminal server monitors read requests to HKCU by the application. If the application attempts to read from a key that does not exist, the terminal server will look in HKLM for the values and copy them up to HKCU before the application is aware that they are missing. The repository for registry mapping data is `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Terminal Server\Install\Software`.

 If you want to modify the default values for an application that uses registry mapping, simply add or edit values in the just-mentioned HKLM key.

INI File Mapping


Some applications continue to store settings (either per-machine or per-user) in INI files instead of in the Windows registry. This action poses a challenge on a terminal server, as no single user can have exclusive access to the INI file, nor can per-user settings be stored there, as changes made by one user will affect everyone on the server.

INI file mapping allows each user to have his or her copy of the INI file without having to recode the application to look in a user-specific location for the file. While in install mode, the terminal server monitors modifications to the WIN.INI file as well as the creation of new INI files in the Windows or Program Files directory. Then, in execute mode, the terminal server will create copies of the INI files in either the user's profile or the user's home directory. When the application attempts to read from or write to the original file, the subsystem redirects the action to the user's copy. Changes to INI files are handled by comparing the date stamp on the user's copy with that of the original file, and if the original is found to be newer, the two files are merged to create a new file for the user.

Root Drive

Older versions of Windows were not able to map drives to a subdirectory of a share. A user's home directory, for example, would be represented by H:\%username% and not simply H:\ as the server could only map to the home share itself. This shortcoming posed a challenge, as many applications do not allow for system variables when referencing files. The root drive concept was developed to create a uniform path that referenced a per-user location. This way, you could simply reference the root drive letter (R:\ perhaps) and have the destination be either the user's specific home directory or user profile.

During logon, a script called `usrlogon.cmd` is run on all terminal servers. If root drive has been enabled on the server, the script performs a `SUBST` command that aliases the user's home directory to the defined root drive letter.

 The `SUBST` command works much like the `NET USE` command, but instead of aliasing a network path to a drive letter, `SUBST` aliases a local path. The `SUBST` command can also be used to alias a subfolder of an existing *mapped network drive* to another drive letter.

Starting with Win2K, Windows is able to map drives to subfolders of network shares, so a user's home directory can be represented by H:\ even if it is a subfolder of the HOME share. This feature has made the use of a root drive almost obsolete. Many existing application compatibility scripts, however, are still written to reference `%rootdrive%`. Thus, you may still need to define and use a root drive. You can, however, modify the `usrlogon.cmd` file on your server to take advantage of the existing drive letter instead of using the `SUBST` command to alias another letter to the same location. Listing 2.1 provides an example of how you can do so (changes are in red).

```
Cd /d %SystemRoot%\ "Application Compatibility Scripts"
Call RootDrv.Cmd

If "A%RootDrive%A" == "AA" goto done

REM If the user has a network Home Directory already mapped
REM on the ROOTDRIVE, we do not need to do anything.

if /I "%rootdrive%" == "%homedrive%" goto NoSubst

:DoSubst
Net Use %RootDrive% /D >NUL: 2>&1
Subst %RootDrive% "%HomeDrive%%HomePath%"
if ERRORLEVEL 1 goto SubstErr
goto AfterSubst
:SubstErr
Subst %RootDrive% /d >NUL: 2>&1
Subst %RootDrive% "%HomeDrive%%HomePath%"
:AfterSubst

:NoSubst
```

Listing 2.1: Modified USRLOGON.CMD.

How to Toggle Between Install and Execute Mode

Both registry mapping and INI file mapping have different behaviors based on whether the terminal server is in install or execute mode. You should be careful to make sure that the server is in the proper mode at all times. To switch the server to install mode, open a command shell (cmd.exe) and type:

```
change user /install
```

To switch the server to execute mode, open a command shell and type:

```
change user /execute
```

To determine which mode the server is currently in, open a command shell and type:

```
change user /query
```

You can use any of these commands in shell scripts (batch files) as well.

Alternatively, to have the server automatically switch between install and execute modes, use the Add/Remove Programs Control Panel applet. To do so, click Add New Programs, then click CD or Floppy (see Figure 2.1). Doing so will start the Install Program wizard.

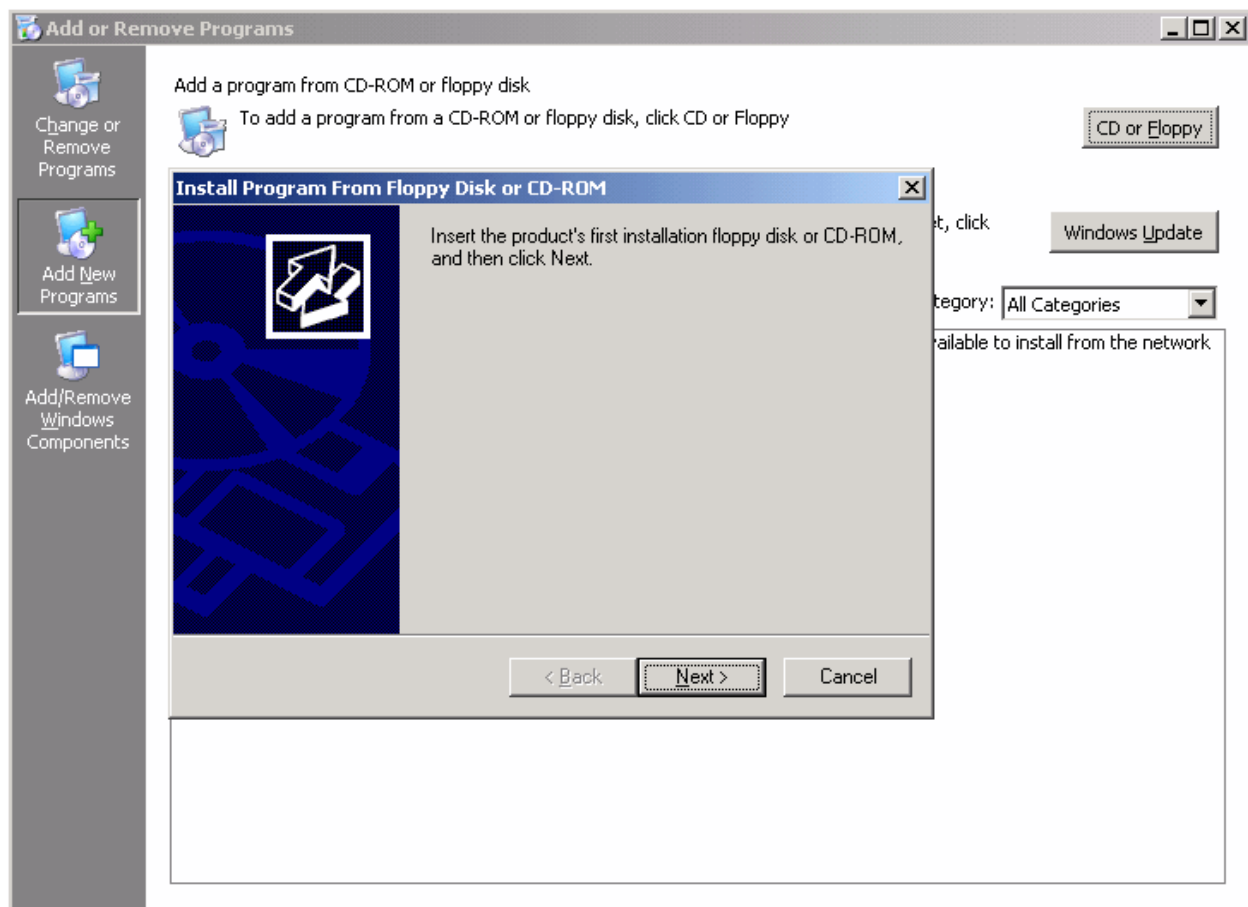



Figure 2.1: Activating install mode via the Add/Remove Programs Control Panel applet.


After you select the setup program, the wizard will automatically switch the server to install mode and run the installation program. When the installation is complete, click Finish in the wizard, and the server will switch back to execute mode.

 After a reboot, the server will always come up in execute mode, regardless of which mode it was in when it was shut down.

WS2K3 will detect most installation packages and automatically invoke the Install Program wizard for you. However, it is still a good idea to use the Control Panel or command line when installing new applications just to be safe.

Windows Installer Service


The Windows Installer Service (msiexec.exe) and its ability to “advertise” settings has virtually eliminated the need for registry mapping. When installing applications that come packaged in MSI format, you usually do not need to place the server into install mode.

 Some applications are not packaged to take full advantage of MSI advertising, so you might want to continue to use install mode just to be safe.

When a user launches an “advertised” application, the Windows Installer service is invoked and any per-user settings are added to HKCU. The package can even be set up to add per-user files to the user’s profile under application data or local settings. This functionality eliminates the need for many application compatibility scripts.

How to Install MSI Packages on Terminal Servers

To install an application that has been packaged in MSI format, simply double-click the MSI file. Use the installation wizard to select the components you want to install.


 If the installation wizard asks you whether you want to install the application for yourself or for all users, be sure to select “All users” as doing so will register the components to be advertised to other users.

You can also install MSI packages via the command line by using the msiexec command. Later, this chapter provides a complete list of arguments used to install, uninstall, and repair packages.

Via IntelliMirror/Group Policy

If you are managing a large terminal server farm or need the ability to expand your farm quickly, you will want to use an automated software installation technology to install MSI packages on terminal servers. In an AD environment you can use Group Policy to assign applications to your servers. When the server boots, the machine policies are processed and any applications that are assigned will be installed.

To use Group Policy to install applications, your applications must be in MSI format. Group Policy also supports a script format called ZAP, which can trigger a non-MSI installer. However, the use of this format is not recommended with terminal servers because such applications do not support advertising, so your users may not receive necessary HKCU registry keys.


 You can repackage software into the MSI format by using a third-party utility such as Wise for Windows Installer. Be sure to thoroughly test the application on Terminal Services before and after repackaging to determine whether any modifications need to be made for terminal server compatibility.

The basic steps required for Group Policy-based software installation are:

1. Create a network share to store your MSI packages.
2. Create Administrative Installations of your MSI packages and place them on the share.
3. Add the packages to a GPO that applies to the terminal server computers in AD.
4. Modify the permissions on the packages in the GPO if you want to filter which terminal servers receive each package.
5. Reboot the terminal servers.

Create a Share

To assign or publish an application via Group Policy, you need a central location to reference for the application source files. The path to this location must be resolvable and accessible by all computers to which the policy applies. The easiest way to accomplish this task is to create a share on a file server and copy the source files to it.

 Make sure that the machine accounts of your terminal servers have read and execute rights on the share. The Authenticated Users and Everyone groups both include machine accounts.

Create Administrative Installations

MSI packages typically come with all the files needed for the application compressed into CAB files. To optimize the installation of the software, uncompress the files in advance by creating an Administrative Installation.

To create an Administrative Installation, execute the Windows Installer Service with a “/a” switch, and specify the path and name of the MSI package you want to uncompress:

```
msiexec /a d:\proplus.msi
```

A wizard, similar to the one that Figure 2.2 shows, should appear asking you for the destination directory for the Administrative Installation. If the application requires a license key for installation, the wizard will prompt you for this as well. The key is then encrypted into the Administrative Installation so that installs performed from that source will not be prompted for the key.

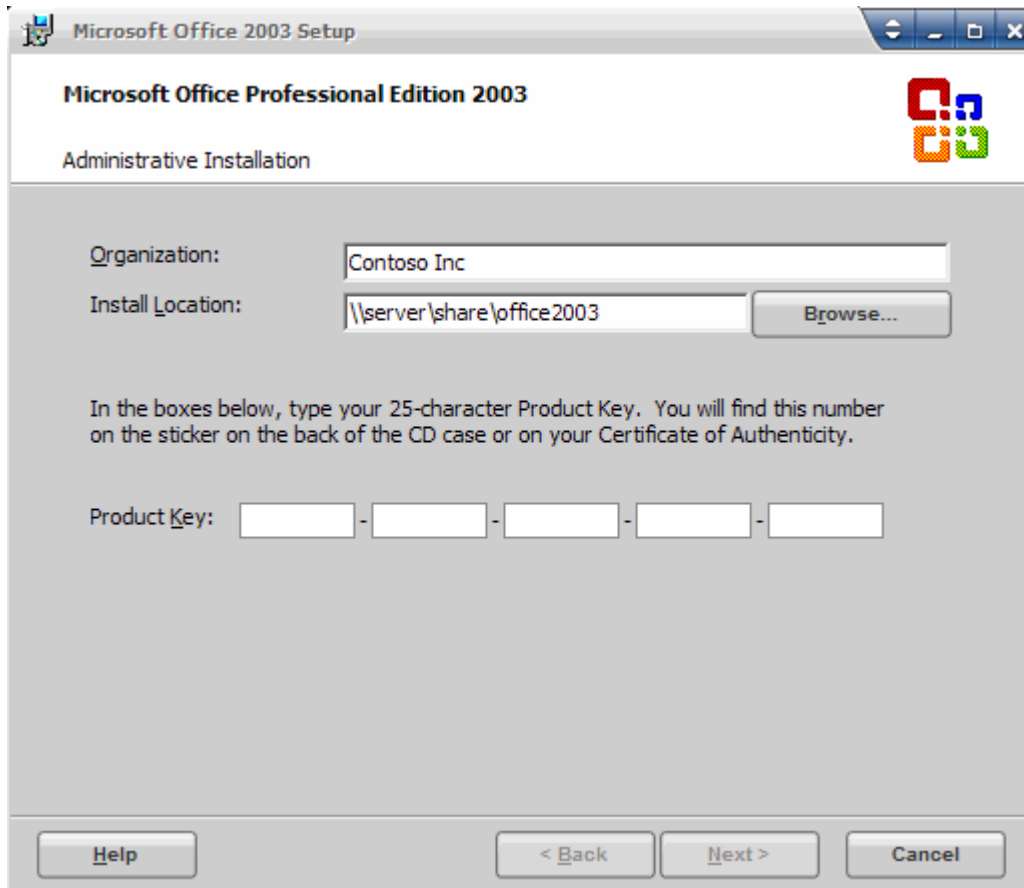


Figure 2.2: An Administrative Installation wizard.

Once the Administrative Installation is complete, copy the new source files to your share.

Add the Packages to a GPO

To add a package to a GPO, edit the policy, and expand Computer Configuration, Software Settings. Right-click *Software installation*, and select New, Package (see Figure 2.3).

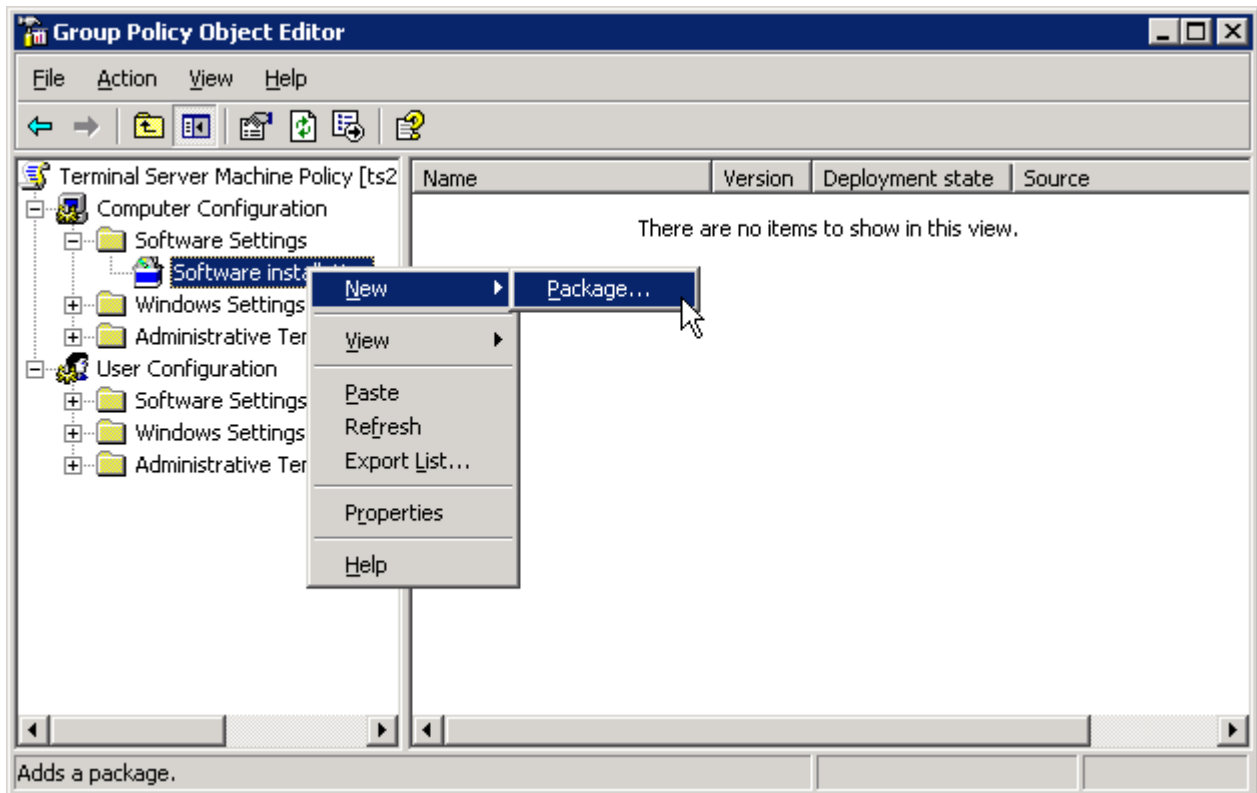



Figure 2.3: Adding a software package to a GPO.

You will be prompted for the MSI package that you want to add. Navigate to your share and select the MSI file that you want to install. You will then be asked whether you want to assign the package or open the Advanced Settings interface. In most cases, you can select Assign and accept all the default options. If, however, you need to specify a transform to be applied during the installation, select Advanced.

 A transform (MST file) specifies options or makes changes to the default settings of a Windows Installer package (MSI file). You can create transforms by using the Microsoft Office Custom Installation Wizard or with a third-party utility.

Filtering Applications

You can use a single GPO that applies to all your terminal servers and still filter which applications are installed on each server by using security filters on the packages in the GPO. Figure 2.4 shows a GPO that contains three software packages—Adobe Acrobat Reader 6.0, Office XP, and the Microsoft Group Policy Management Console.

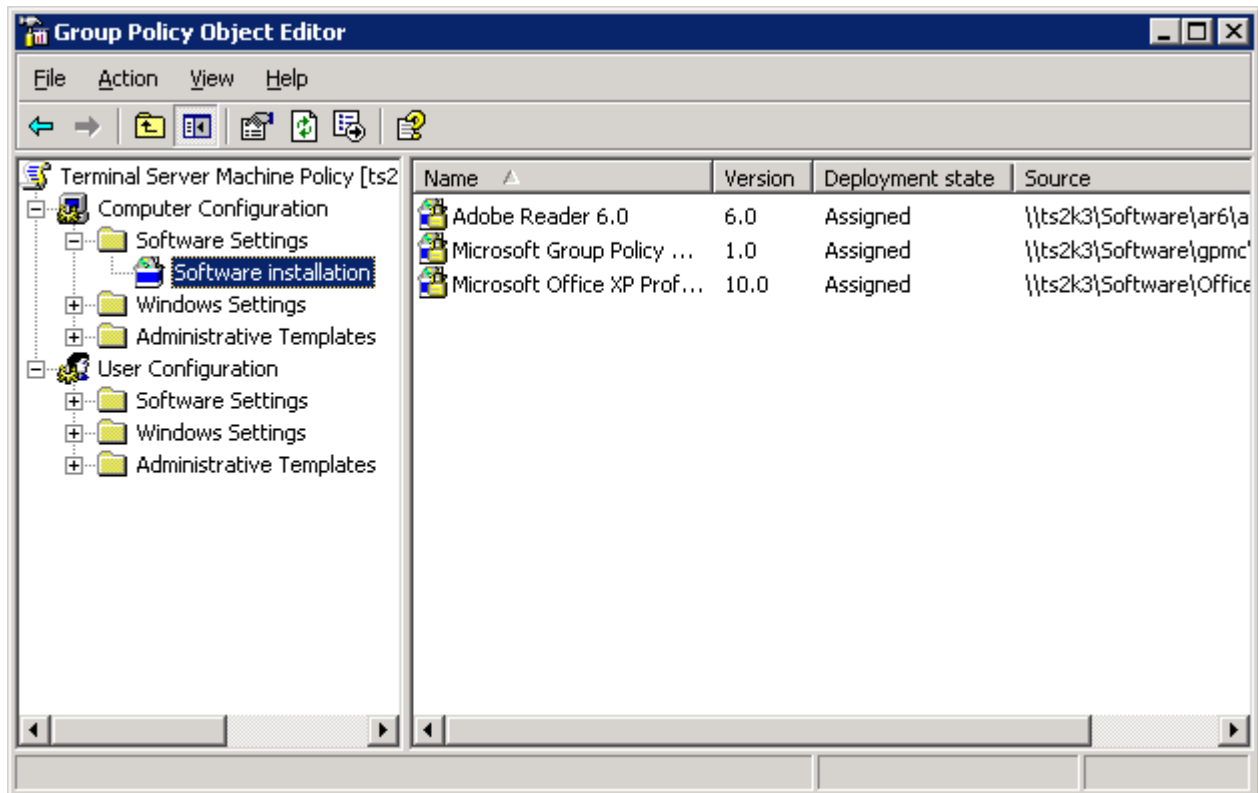


Figure 2.4: A Group Policy that has three software packages applied.

Let's assume that you want to install Acrobat Reader and Office XP on all terminal servers to which the GPO applies, but you only want to install the Group Policy Management Console on the terminal servers that administrators use. By default, the packages will inherit security settings from the GPO, so any computers that have read permission on the GPO will also have read permission on the package, and will therefore install the software during boot up.

You can change the permissions on the package and limit the ability to read the package to only a specific group of computers. By doing so, all computers in the organizational unit (OU) will process the policy but only those with read permission on the package will install it. Figure 2.5 shows the default permissions on a package and the permissions after they have been modified so that only the Admin Terminal Servers group can read it.

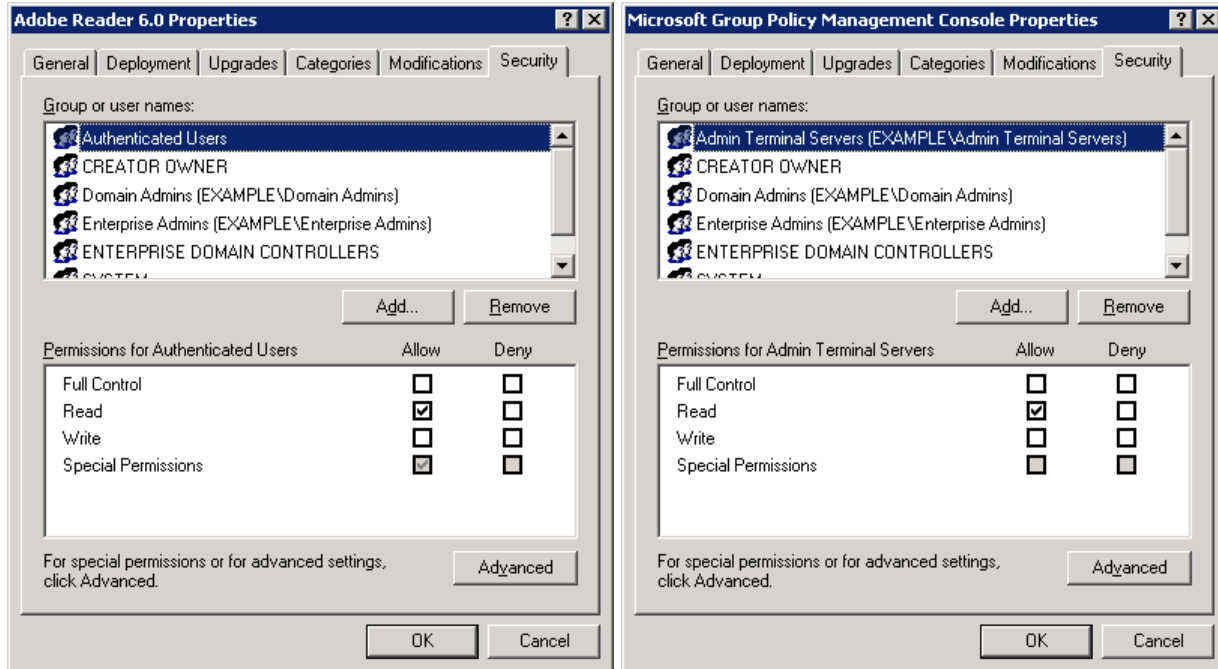



Figure 2.5: Filtering security on a software package.

For this security filter to work, you will need to create a Domain Security group—Admin Terminal Servers, for example—and place the servers you want to receive the console into that group.

 To modify the permissions of a package, you need to break inheritance on the GPO's permissions. To do so, click **Advanced**, and clear the *Allow inheritable permissions* check box. You can then choose to copy the existing permissions and use them as a starting point for your modifications.

MSIEXEC Command-Line Reference

The following list highlights some common command-line arguments for Windows Installer (msiexec.exe):

- /I—Installs an msi package
- /f—Repairs an installed package
- /a—Creates an Administrative Installation package
- /x—Uninstalls a package
- /l logfile—Creates a log at the specified path/filename
- /lv logfile—Creates a verbose log
- /p—Applies a patch in MSP format
- /q with n|b|r|f—Sets the user interface (UI) level
- /qn—Silent installation
- /qn+—Silent installation with completion dialog box


For example, the command:

```
Msiexec /i pro11.msi /lv c:\temp\office.log /qn+
```

Installs Office 2003 silently with a completion dialog box and logs the installation to c:\temp\office.log.

You can modify the default installation behaviors of an MSI file by using a *transform*—an MST file. You can specify a transform from the command line by using the syntax:

```
Msiexec /i pro11.msi /lv /qn+ transforms=terminalserver.mst
```

 If the MSI and MST files are not both in the current working directory, you should specify their exact path. UNC paths are recommended so that the Windows Installer service can locate the source files in the event that a repair is required.

Some MSI files also take custom arguments. For example, if you want to install Office 2003 on a terminal server via the MSI package (skipping the setup.exe wrapper), you must add

```
Terminalserver=1
```

to the msiexec command line or to the package properties in the Group Policy.

Application Compatibility Scripts

During the logon process, all terminal servers call a built-in script called `usrlogon.cmd`; this script is used to map the root drive (if one is defined) as well as call any application compatibility scripts that have been installed. Between the application compatibility subsystems and the Windows Installer Service, very few applications still require application compatibility scripts. In fact, WS2K3 only comes pre-loaded with one—Eudora 4.

Application capability scripts are used to copy files to a user's home directory or user profile, or to modify registry keys that are not handled via registry mapping. They can also be used to map drives that are required for specific applications.

How to Add an Application Compatibility Script that Does Not Require a Root Drive

If your application compatibility script does not reference the root drive, copy the script to C:\Windows\Application Compatibility Scripts\logon, then add a call statement to the `usrlogn1.cmd` file located in the Sytem32 directory. If this script is the first application compatibility script you are installing on the server, you will need to create this file. The call statement should look like this:


```
call scriptname.cmd
```

How to Add an Application Compatibility Script that Requires a Root Drive

To call an application compatibility script that requires a root drive, you must first define the root drive letter. To do so, run the CHKROOT.CMD script found at C:\Windows\Application Compatibility Scripts. Once you do so, the root drive variable will be defined during the logon process, and you can reference it in your application compatibility script logon scripts.

Next, copy your application compatibility script logon script to C:\Windows\Application Compatibility Scripts\logon. Finally, go to the system32 directory and add a call statement to the usrlogn2.cmd file. If this script is the first application compatibility script that you are installing on the system, you will need to create this file. The call statement should look like this:

```
call scriptname.cmd
```

 Usrlogn2.cmd only gets run if the root drive letter has been defined.

User Logon Process and Scripts

The logon process to a terminal server can involve several scripts depending on the environment. Scripts can be written in whatever scripting language you prefer; however, natively, WS2K3 only supports Shell Script (BAT and CMD files), Visual Basic Script (VBS Files), and Java Script (JS files). If you want to use another language (KIX or Perl, for example), be sure to deploy the appropriate runtime to your servers. Figure 2.6 offers a flowchart showing the scripts and the order that they are processed.

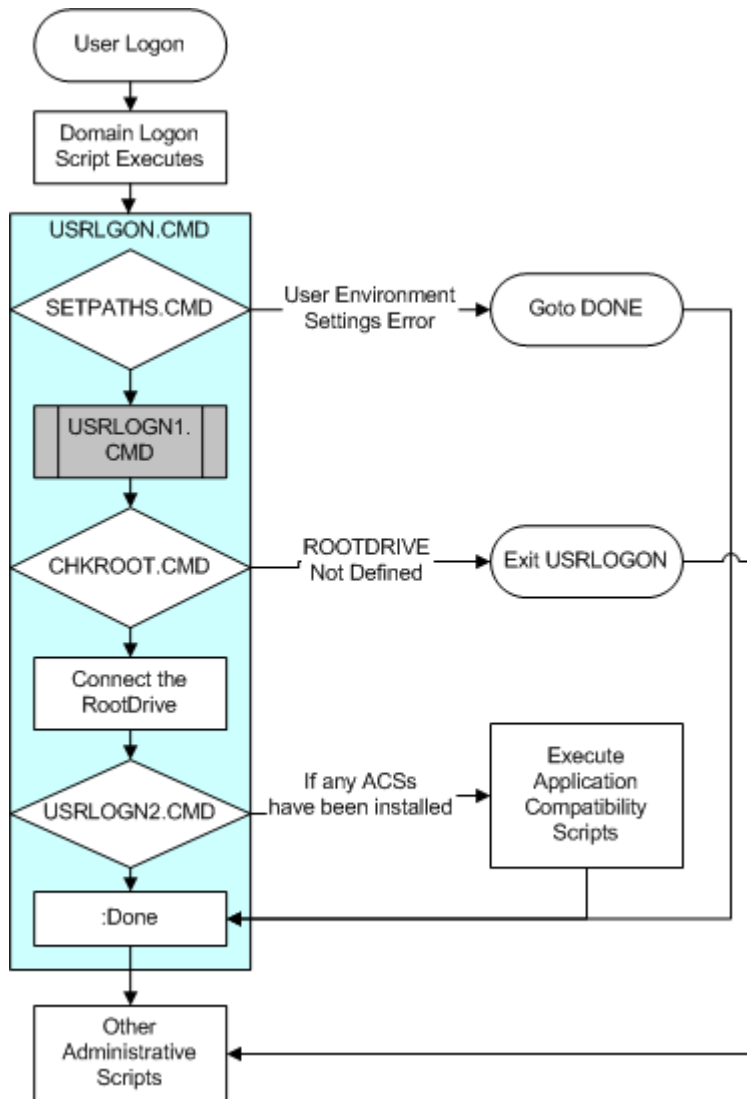


Figure 2.6: The user logon process, and its associated scripts

As you can see, there are several options for invoking user logon scripts. You should select the one that is right for your environment. Some things to consider:

- Does the script perform operations specific to a given application? If so, you might want to use an application compatibility script so that the logon script is only run on servers that have the application installed.
- Is the script specific to your terminal servers (and should not be run on workstations)? If so, consider a Group Policy-based logon script.
- Is the script universal? A drive mapping to a public drive for example? Then either a Group Policy or per-user script might be the best option.
- Is the script unique to a given user? If so, then a per-user script is the way to go.

How to Invoke a Per-User Logon Script

Per-user logon scripts are usually domain based and are configured as an attribute of a user object in AD. To configure a user account to run a logon script:

- Copy the script and any required files (resource kit tools, command-line utilities, and so on that are called by the script) to the domain netlogon share ([\\domain\netlogon](#)).
- Launch Active Directory Users and Computers, and open the properties dialog box of a user object (see Figure 2.7).
- On the Profile tab, enter the name of the script in the Logon script field.

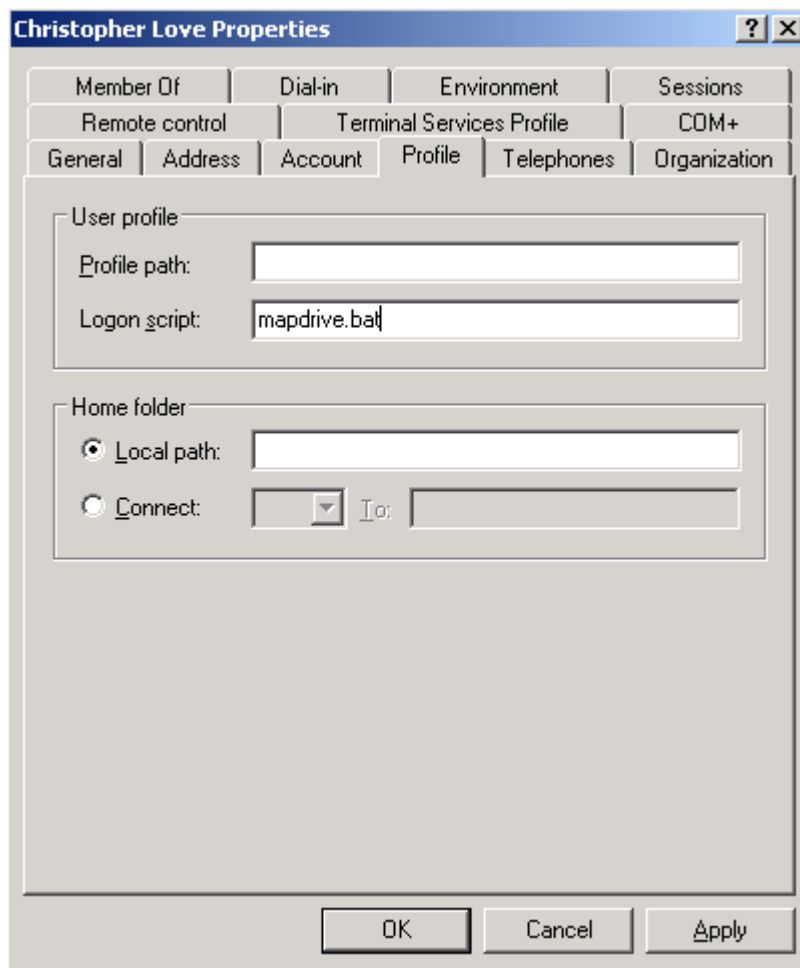


Figure 2.7: Adding a logon script to a user object.

How to Invoke a Logon Script via Group Policy

Group Policy-based logon scripts are executed for any user account that is within the scope of the GPO. The following list provides the steps to add a logon script to a GPO (see Figure 2.8):

- Edit a GPO that applies to the desired user objects.
- Drill to User Configuration | Windows Settings | Scripts.
- Double-click Logon in the right pane of the editor.
- In the dialog box that appears, click Show Files.
- Copy the script and any dependant files to the directory that appears.
- Close the Explorer Window, then click Add in the dialog box.
- Click Browse, and select the script.

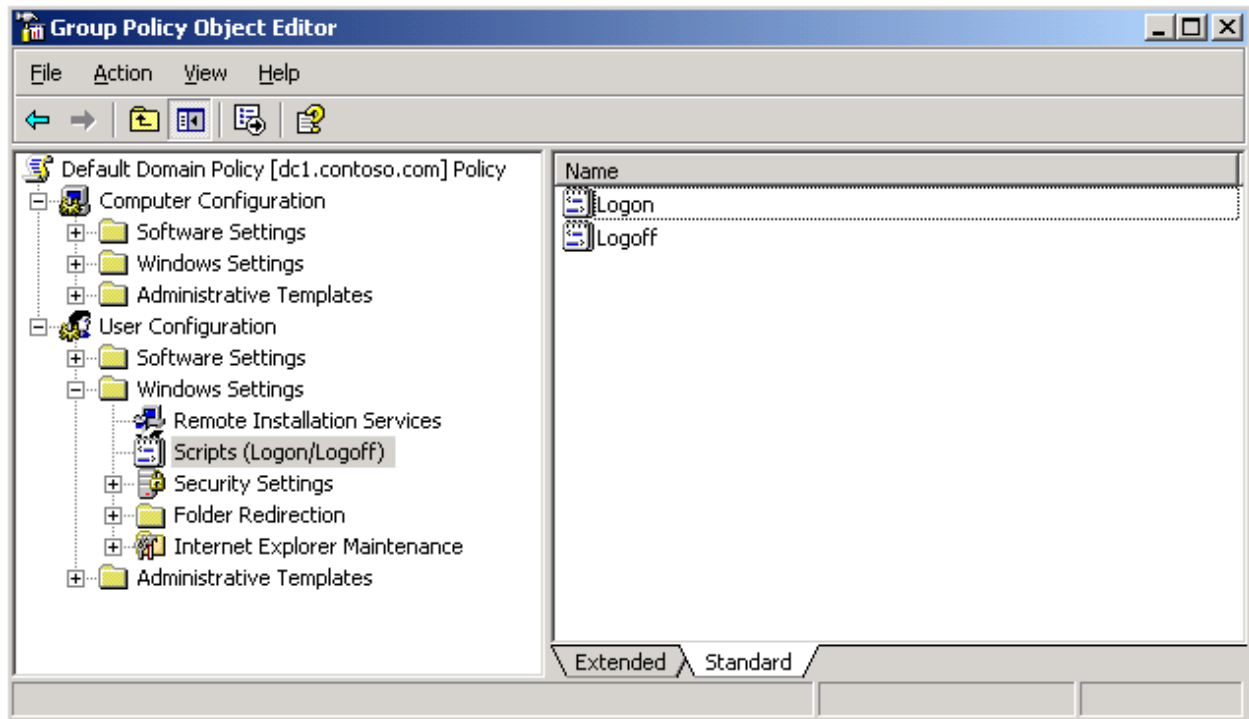


Figure 2.8: Adding a logon script to a GPO.

Normally, the GPO you use will be linked to the OU that the user objects are in, but it is common to configure terminal servers to run in Loopback Policy Processing mode, in which case the GPO will be linked to the OU that contains the terminal server computer objects.

By default, you use Active Directory Users and Computers to access and edit GPOs. Strongly consider installing and using the Group Policy Management Console to manage your GPOs instead. The console is a free download from Microsoft and can be found at <http://go.microsoft.com/fwlink/?linkid=21813>. The Group Policy Management Console provides a single interface to display, manage, and edit all GPOs in your forest. It also has built-in tools to check the resultant set of policy (RSOP) as you perform GPO modeling "What if?" scenarios.

Summary

Although most applications can be installed and run in a terminal server environment without requiring any changes by the administrator, it is useful to have the knowledge for how to deal with applications that necessitate modification. In such cases, you can rely on tools such as application registry mapping, INI file mapping, and root drive specification as well as modified application compatibility scripts and logon scripts. We'll build on this knowledge in the next chapter, which explores user sessions and profiles in a terminal server environment.

Chapter 3: User Session and Environment Configuration

Once you have the Terminal Server role enabled and applications installed, you are ready to allow users to start logging onto your terminal server. Before you do so, however, you should consider configuring their user profiles, sessions, and resource redirection settings. These settings are central to the overall experience your users will have when working on your terminal server. Although your users might like to have the freedom to customize every setting in their terminal server environment—such as unlimited session lengths, and the ability to access every drive, printer, and peripheral on their local workstations—this freedom would not be a very efficient use of your terminal servers resources. Your goal in configuring them should be to find a balance between end-user functionality and server performance.

Session Length Limits

There are three distinct user session timeouts you can configure on a terminal server:

- **Active session**—Limits the amount of time that a user can actively work on the terminal server. At the end of the timeout duration, the user will be logged off of the server. This setting is commonly used in public kiosk environments.
- **Idle session**—This setting limits the amount of time that a user can leave a session idle. If the user minimizes the terminal server window or locks his or her workstation, the terminal server session becomes idle but continues to take up some resources on the server to maintain the user session.
- **Disconnected session**—When a user closes a terminal server window without logging off or experiences a network interruption, the session can become disconnected. This setting determines how long the session will remain active on the server waiting for the user to reconnect. Like an idle session, a disconnected session continues to use some server resources.


If your servers have the capacity to handle all your users simultaneously, you might choose to leave these timeouts set to unlimited so that your users can leave sessions running for days at a time and disconnect and reconnect at will. More often, however, you will want to configure at least the disconnected session timeout to reclaim the resources on the server. The next section will show you how to configure these settings.

Types of User Profiles

A user profile consists of the user's HKCU registry hive as well as all per-user files and data (Internet Explorer—IE—Favorites, Outlook Signature files, and so on). Since the release of Windows NT, Windows has supported three distinct types of user profiles:

- Local profiles—These profiles are specific to one computer and are stored on the local hard disk only (on WS2K3 systems, they are stored in C:\Documents and Settings).
- Roaming profiles—These profiles are stored centrally on a file server and are copied down to every machine that a user logs onto. At logoff, any changes made during the user session are copied back up to the central file server. Roaming profiles allow users to have consistent settings across multiple computers.
- Mandatory profiles—A mandatory profile is a preconfigured profile that is stored on a central file server. At logon, a copy of the profile is made for the user with all the administrator-defined settings in tact. At logoff, any changes that the user might have made during the session are discarded. Mandatory profiles provide a consistent user experience and minimize the chances of profile corruption by creating a fresh profile for the user at every logon.

In a terminal server environment, there are some additional options available, but they are based on these three types of profiles. You can configure a separate roaming or mandatory profile to be used exclusively on terminal servers so that your users can have separate settings for servers and workstations. You can also configure a single server or group of servers to use a distinct roaming profile from the one defined in the user account.

 There are also third-party tools that offer alternatives to the native profile types, which will be introduced later in this chapter.

Client Device Redirection and Resource Mapping

In order for a user to have a rich, full-featured terminal server experience, you may want to take advantage of client device redirection and resource mapping. These features allow the user to seamlessly work across both local and terminal server-based applications by automatically connecting them to resources available on the user's local workstation. RDP 5.2 supports the following features:

- Client drive mapping—Automatically connects to the client workstation's local drives (hard disk, diskette, and CDROM) so that users can access files on these drives from within applications on the terminal server.
- Client printer mapping—Automatically connects to printers that users have mapped on their workstations (both locally attached and network printers) so that users can print without needing to re-map the printers on the terminal server.

- Client time zone mapping—Detects the time zone to which the client workstation is set and adjusts the time within the user session to match. This mapping is vital when your users are distributed across multiple time zones.
- Client COM port mapping—Allows users to access peripherals that are connected to the client workstation’s serial ports. This mapping can be used for POS devices (bar-code readers, cash drawers, and so on) as well as serial-port–connected PDAs.
- Client audio redirection—You can configure sounds that are generated by applications on the terminal server to be played through the client workstation’s speakers, played through the terminal server’s speakers (if it has a sound card), or disabled altogether.



Each of these client redirection features consume additional bandwidth. Take this consideration into account before enabling these features if your users are working over a low-bandwidth connection.

How to Configure User Session Timeouts

If you want each user to have unique session timeout settings, you can configure them on a per-user basis via Active Directory Users and Computers, ADSI, and Group Policy, and on a per-server basis via GUI and Group Policy. The following sections explore how to do so.

On a Per-User Basis via Active Directory Users and Computers

The Active Directory Users and Computers tool enables you to set session timeouts on each user account independently. To do so, launch Active Directory Users and Computers, then open the properties interface for the user account you want to configure. Figure 3.1 shows the Sessions tab of a user account.

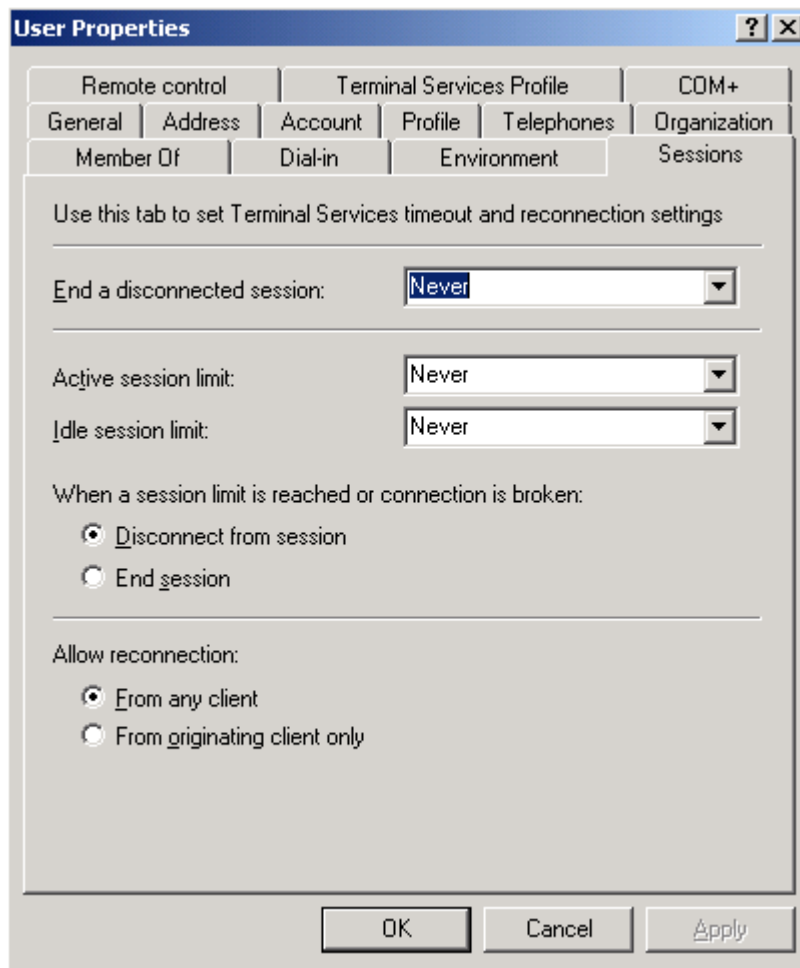



Figure 3.1: Configuring session timeouts on a user account.

In addition to setting active, idle, and disconnected session timeouts with this tool, you can configure whether to disconnect from the client or end the session when the idle or active session limit is reached. You also have the option of limiting the user to only reconnecting to a disconnected session from the client device that originally established the session. In other words, you can prevent your users from moving a terminal server session from one client device to another.


 Per-server and Group Policy-based timeouts override per-user settings, so if you want the settings on the user account to apply, you must leave the timeouts on the server unconfigured.

On a Per-User Basis via Active Directory Scripting Interface

If you want to use per-user session timeout settings and you have a large number of users in your environment, configuring each account manually can be very time consuming. WS2K3 allows you to configure these settings through the Active Directory Scripting Interface (ADSI). This way, you can batch-edit a large number of user accounts with very little effort.

You access ADSI by using the Windows Script Host (WSH); thus, you can choose whether to write your scripts in Visual Basic Script (VBScript) or Java Script. The following examples are in VBScript.

Configuring user properties through ADSI is a three-step process. First, you must open a connection to the user account, then set the properties, and finally write the changes back to the user account. To open a connection to the user account, use either the WinNT provider or the Lightweight Directory Access Protocol (LDAP).

 Although you can use the scripts in this section to configure both NT 4.0 domain and Win2K AD accounts, you can only run the scripts on a WS2K3 server; they will not work if run on Win2K or even Windows XP.

The syntax for the connection is either

```
Set objUser = GetObject("WinNT://<domain name>/<username>,user"
```

or

```
Set objUser = Get Object("LDAP://<distinguished name of user>")
```

As you can see, to use LDAP, you must know the distinguished name of the user object (for example, cn=joe.user,ou=users,dc=example,dc=domain,dc=com), which is difficult if your users are spread across multiple organizational units (OUs). To make it easier, Microsoft enables the ability to use the WinNT provider for AD accounts as well. The domain controller will automatically translate the WinNT call into an LDAP call for you.

Once you have the user account open, set the parameters that you want to change. The names of the Terminal Services session timeouts and the syntax for setting them are provided in Listing 3.1.


```
objUser.MaxDisconnectionTime = [minutes, 0 for never]
objUser.MaxConnectionTime = [minutes, 0 for never]
objUser.MaxIdleTime = [minutes, 0 for never]
objUser.BrokenConnectionAction = [1,0]
    1 = end session, 0 = disconnect the session
objUser.ReconnectionAction = [1.,0]
    1 = original client only, 0 = any client
```

Listing 3.1: Names and syntax for setting Terminal Services session timeouts.

Finally, you must write the changed attributes back to the user account:

```
objUser.SetInfo
```

Obviously, if you want to configure a single user account, it would be faster to just use the GUI tool. However, ADSI is a useful option for configuring properties for multiple users at the same time.

 The Command Line Reference section at the end of this chapter provides an example script that configures all the terminal server attributes of a user account.

 The Microsoft TechNet Script Center (<http://www.microsoft.com/technet/scriptcenter>) is a useful resource for administrative scripting. With a little scripting know-how, you can modify the example scripts to meet your unique needs.

On a Per-User Basis via Group Policy

If the desired session timeouts align with either AD group membership or OU structure, you can configure per-user session timeouts via Group Policy. To do so, use the Group Policy Management Console to create or edit a GPO that applies to the user accounts you want to configure. Then drill to User Configuration | Administrative Templates | Windows Components | Terminal Services | Sessions and configure the timeouts (see Figure 3.2).

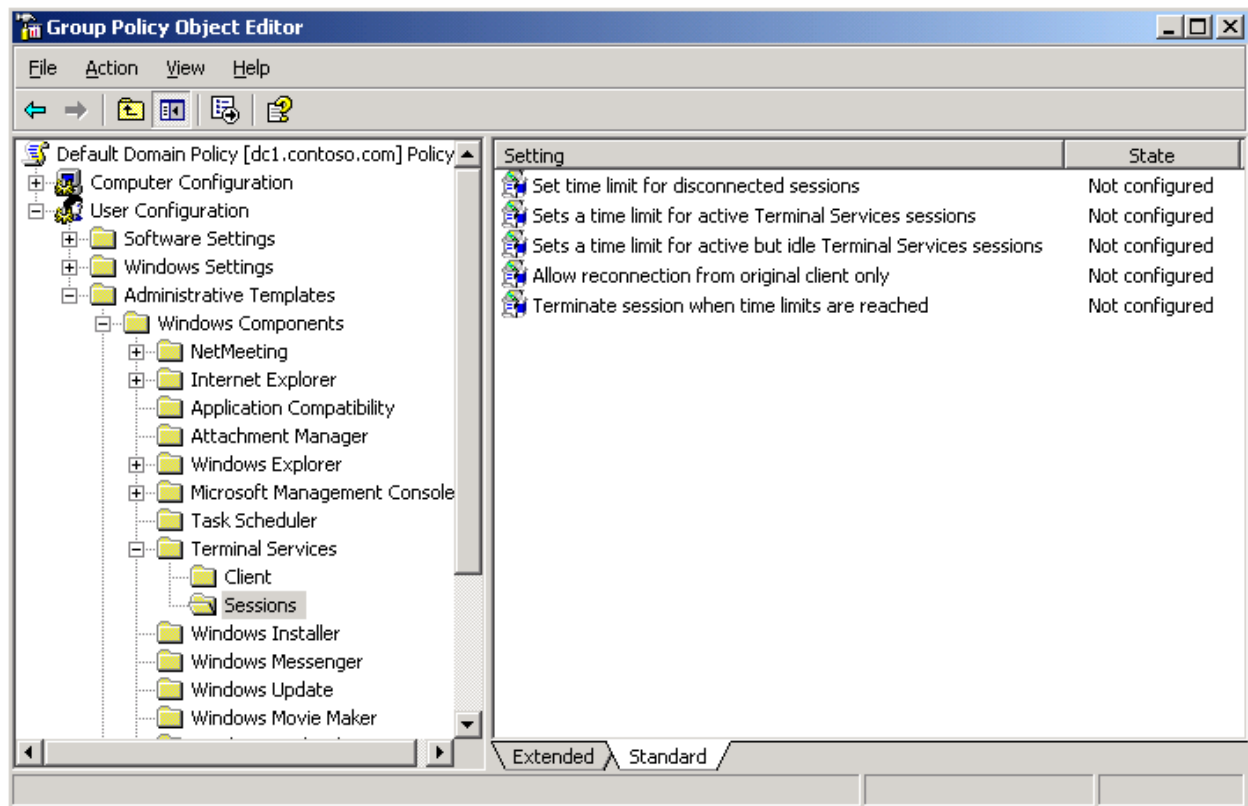



Figure 3.2: Configuring per-user session timeouts via Group Policy.

 It is common to use Loopback Policy Processing mode in a terminal server environment. Loopback mode applies user configuration settings from the computer object's GPOs instead of those that apply to the user object. If you use Loopback mode, session timeouts you configure in a GPO that is scoped to the user object may not apply when logging into a terminal server.

On a Per-Server Basis via GUI

Although configuring session timeouts on a per-user basis provides the greatest amount of flexibility, it can also be very time consuming and difficult to manage. Most terminal server administrators choose to find settings that are appropriate for all users and configure timeouts on a per-server basis.

To configure per-server session timeouts, launch the Terminal Services Configuration administrative tool. Open the properties dialog box of the RDP-Tcp protocol, and go to the Sessions tab, as Figure 3.3 shows.

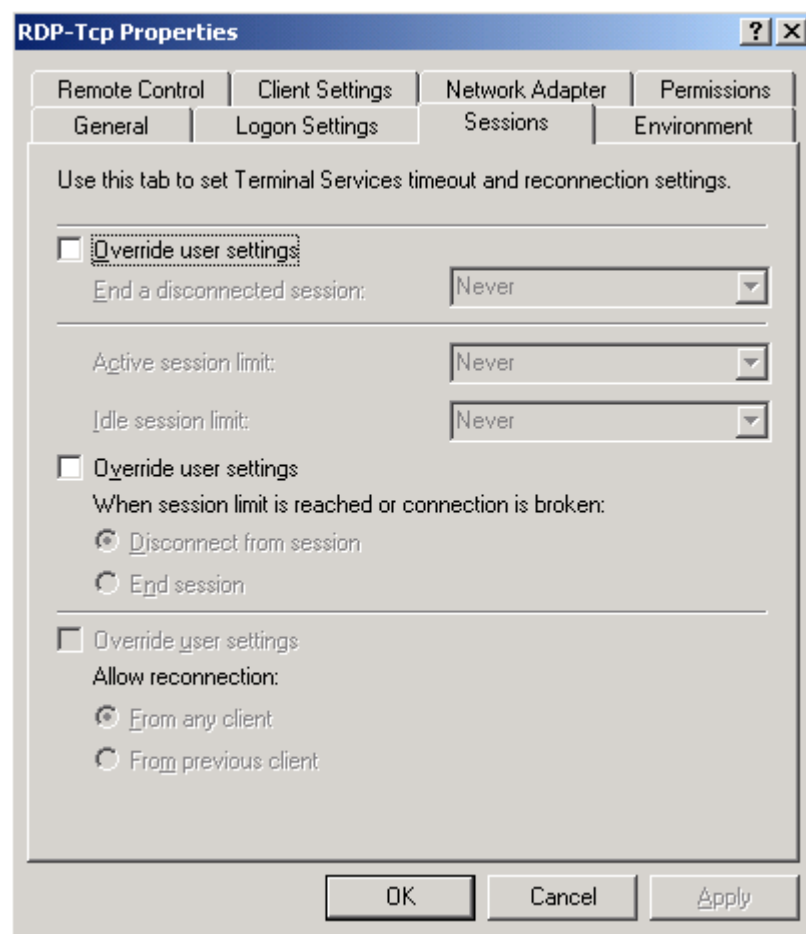


Figure 3.3: Configuring session timeouts on a per-server basis.

On this tab, you can enable the server to override the settings configured on the user account, then set server-specific timeouts. You have the ability to override the client reconnection setting as well.

On a Per-Server Basis via Group Policy

If you want to use per-server session timeout but have multiple servers to configure, it may be easier to centrally configure the session timeouts via Group Policy. This way, any new terminal servers that are added to your domain automatically acquire the correct timeout settings (provided they are placed in AD within the scope of the GPO).

To configure session timeouts via Group Policy, use the Group Policy Management Console to create or edit a GPO that applies to your terminal servers. Then drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Sessions and configure the timeouts (see Figure 3.4).

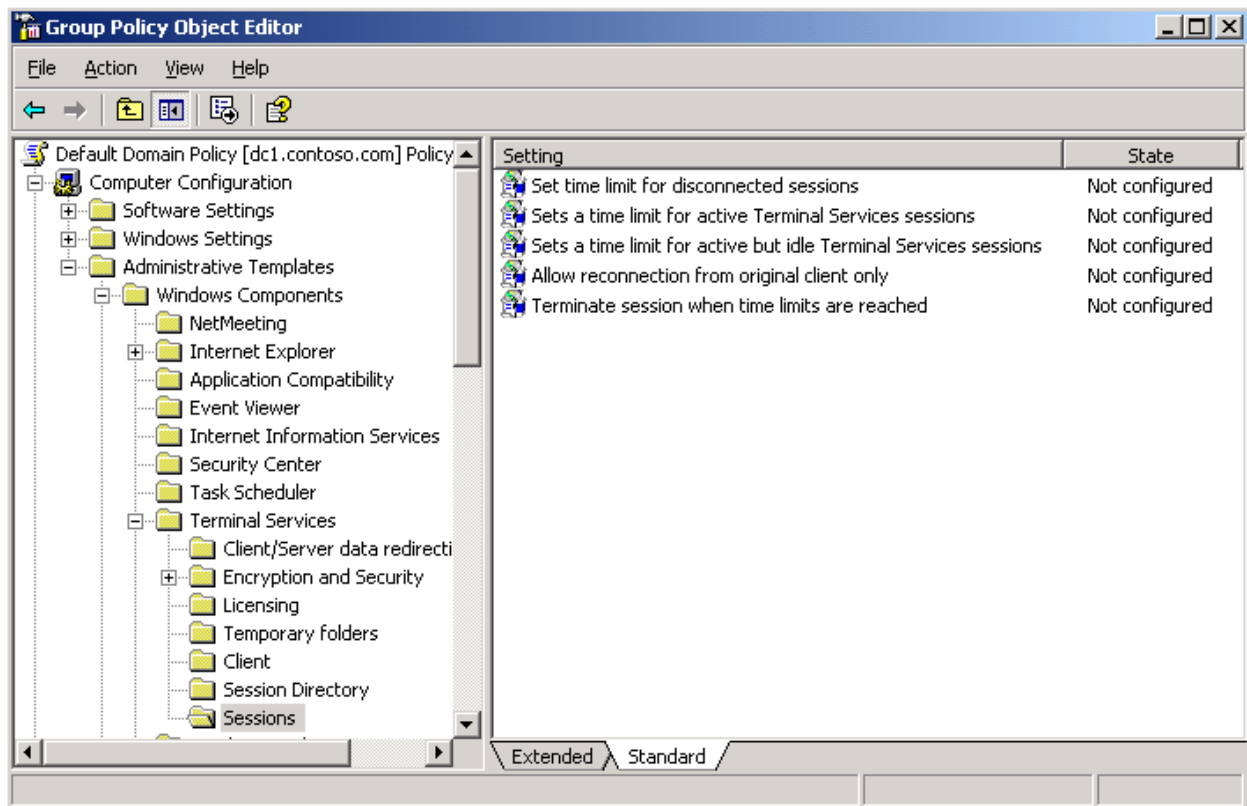


Figure 3.4: Configuring per-server session timeouts via Group Policy.

It is also possible to configure per-server session timeouts via Windows Management Instrumentation (WMI). To learn more about this option, see Microsoft's Web site at <http://www.microsoft.com/technet/prodtechnol/windowsserver2003/library/ServerHelp/f826112b-c88d-42cc-a52c-c99ea467ab87.msp>.

How to Configure Client Device Redirection and Resource Mapping

Like session length limits, you can configure client device redirection and resource mapping on a per-user or per-server basis. To do so, you can use Active Directory Users and Computers and ADSI for per-user settings or on a per-server basis via a GUI or Group Policy.

On a Per-User Basis via Active Directory Users and Computers

To configure client device redirection and resource mapping via Active Directory Users and Computers, launch the tool, and open the properties window of the user object you want to configure. Figure 3.5 shows the Environment tab of a user object.

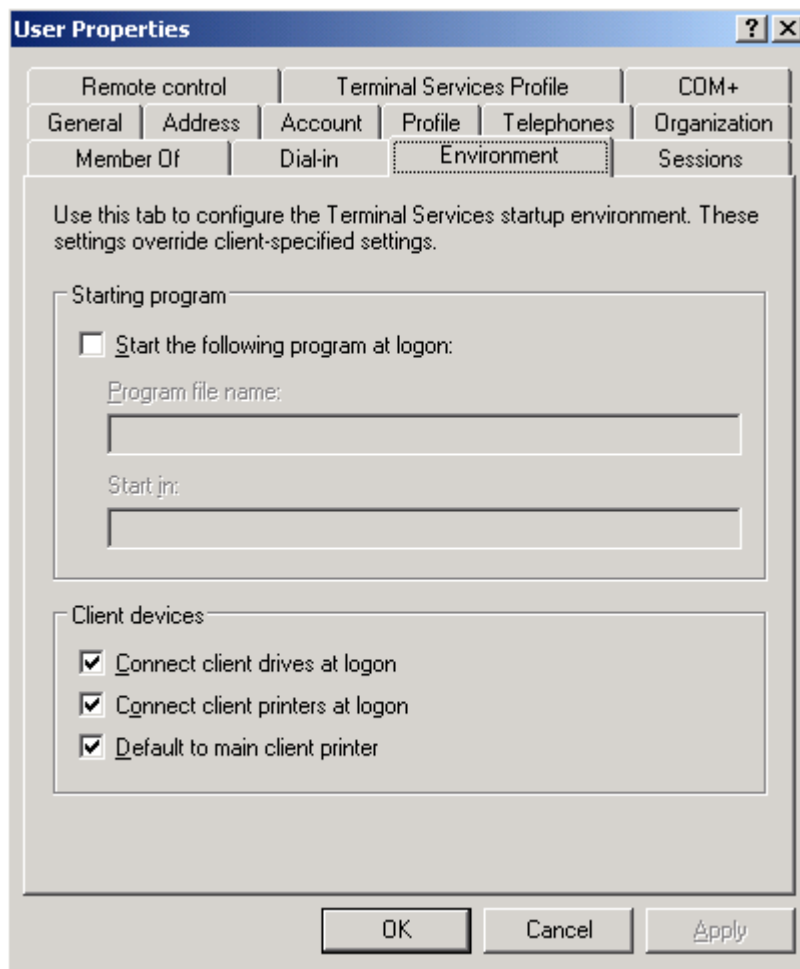



Figure 3.5: Configuring client redirection via Active Directory Users and Computers.

On this tab, you can enable or disable client drive mapping and client printer mapping and set whether to default to the main client printer if it is different than the default printer set in the user's terminal server profile. This tab also allows you to set a specific program to be used instead of the Windows Explorer Shell whenever this user account logs into a terminal server.

 The starting program setting does not launch the specified program within a desktop shell on the terminal server. If you configure this setting, the user will only see the specified application and will not have access to a desktop shell.

On a Per-User Basis via ADSI

If you want to script the configuration of these user settings, use the following ADSI interfaces:

```
objUser.ConnectClientDrivesAtLogon = [1,0]
objUser.ConnectClientPrintersAtLogon = [1,0]
objUser.DefaultToMainPrinter = [1,0]
objUser.TerminalServicesInitialProgram = ["path to program"]
objUser.TerminalServicesWorkDirectory = ["path to directory"]
```

Remember, you must first open a connection to the user object using either the WinNT or LDAP provider:


```
Set objUser = GetObject("WinNT://<domain name>/<username>,user")
```

or

```
Set obUser = Get Object("LDAP://<distinguished name of user>")
```

And then save your changes after configuring the attributes:

```
objUser.SetInfo
```

 Once again, if you want the per-user settings to be obeyed, you must leave both the per-user and Group Policy settings unconfigured.

On a Per-Server Basis via GUI

To configure client device redirection at the server, launch the Terminal Services Configuration administrative tool, and open the properties dialog box of the RDP-Tcp protocol. Figure 3.6 shows the Client Settings tab.

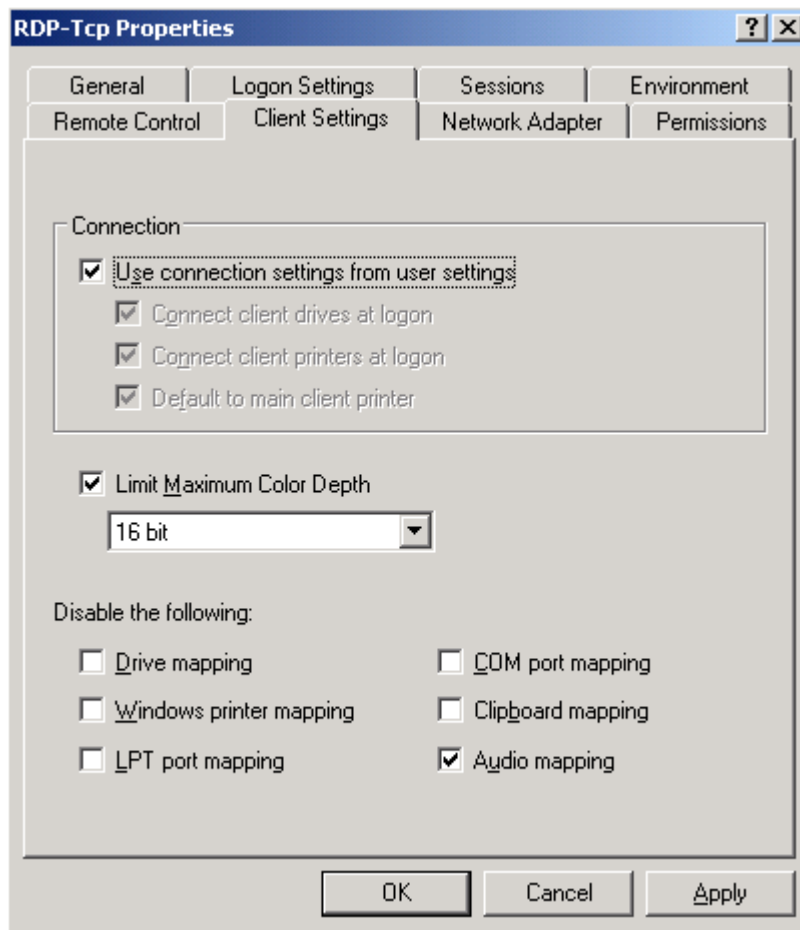



Figure 3.6: Configuring client device redirection at the server.

On this tab, you can override the same three settings that are available on the user account. In addition, you can limit the maximum color depth that is sent to the client and disable specific device mapping features.

 Audio mapping is disabled by default on WS2K3 SP1.

To configure the starting program feature that you saw on the user account, go to the Environment tab, which Figure 3.7 shows.

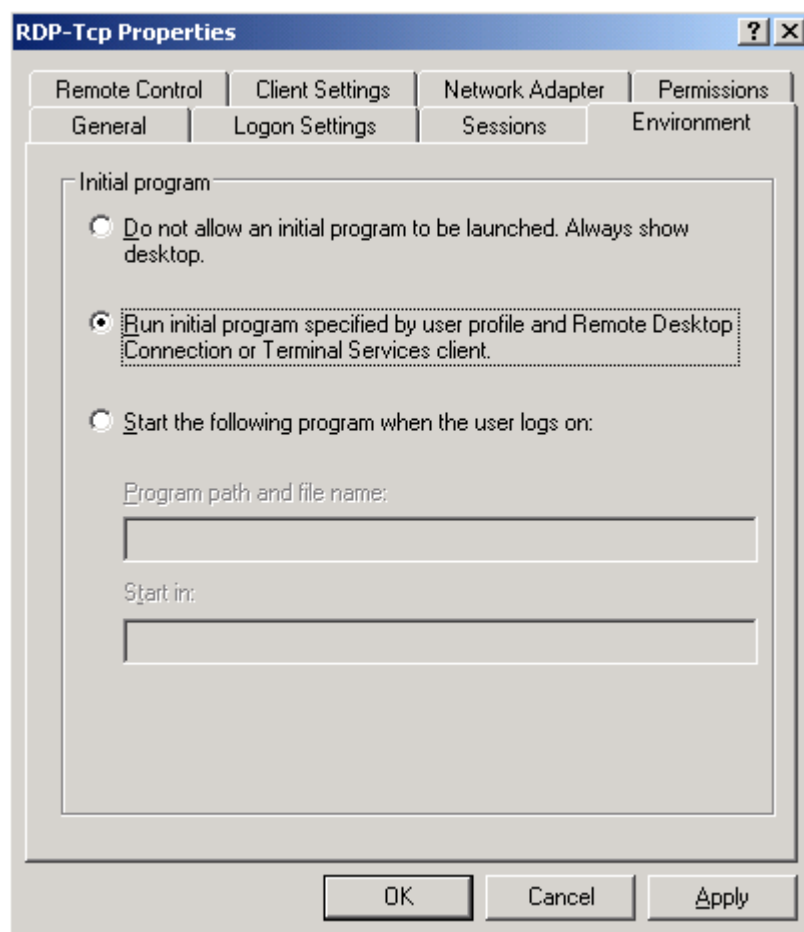



Figure 3.7: Configuring initial program settings at the server.

On this tab, you have three options: override both terminal server client and per-user settings and always display a desktop shell, obey the client and per-user settings, or always start a specific program.

 Be very careful when configuring an initial program at the server level as doing so will apply to administrative accounts as well as users, and you might lose your own ability to see a desktop shell.

On a Per-Server Basis via Group Policy

You can centrally configure client device redirection via Group Policy. To do so, launch the Group Policy Management Console, and create or edit a GPO that applies to your terminal servers. Drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Client/Server Data Redirection, as Figure 3.8 illustrates.

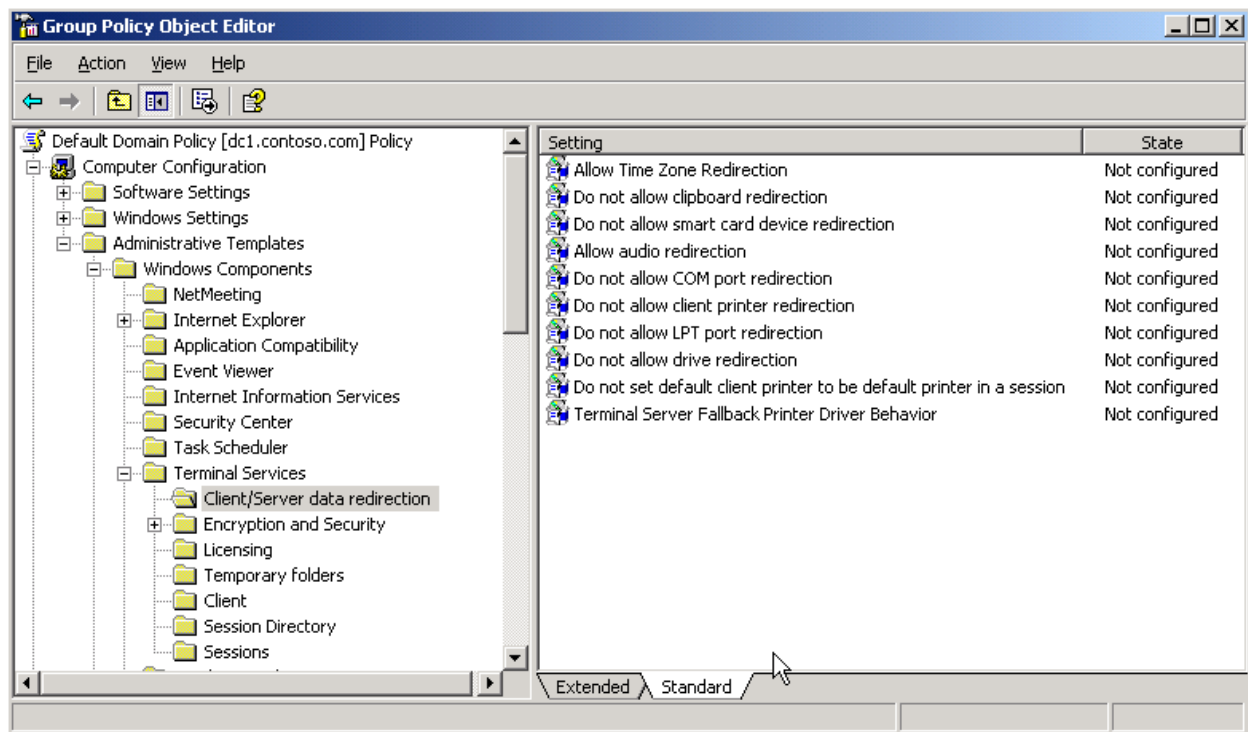




Figure 3.8: Configuring client device redirection via Group Policy.

In addition to the settings you have in the Terminal Services Configuration tool, you can also control client time zone mapping and smart card redirection from the Group Policy Management Console.

 Client time zone mapping is disabled by default on WS2K3.


 To configure an initial program via GPO, go up a level to Computer Configuration | Administrative Templates | Windows Components | Terminal Services, and configure the application in the *Start a program on Connection* setting. Once again, be careful as doing so will also prevent an administrator from receiving a desktop shell over Terminal Services.

How to Configure User Profiles

By default, all Windows systems will use local profiles. If you have only one terminal server, this default can be fine; however, most environments have multiple servers that are either load balanced or provide different applications. If you want users to have a consistent experience across the servers, you must configure either roaming or mandatory profiles.

On a Per-User Basis via Active Directory Users and Computers

To configure a profile on a per-user basis, use Active Directory Users and Computers, and open the properties dialog box for the user account. You will see that there is both a Profile and a Terminal Services Profile tab. The Profile tab is primarily used for workstation logons, and the settings on the Terminal Services Profile tab will be loaded only by terminal server sessions.

 If you use roaming or mandatory profiles for your workstations and you do not configure a separate profile on the Terminal Services Profile tab, the terminal server will load the same profile that is defined on the Profile tab.

To configure a specific roaming or mandatory profile to be used on terminal servers, go to the Terminal Services Profile tab, which Figure 3.9 shows.

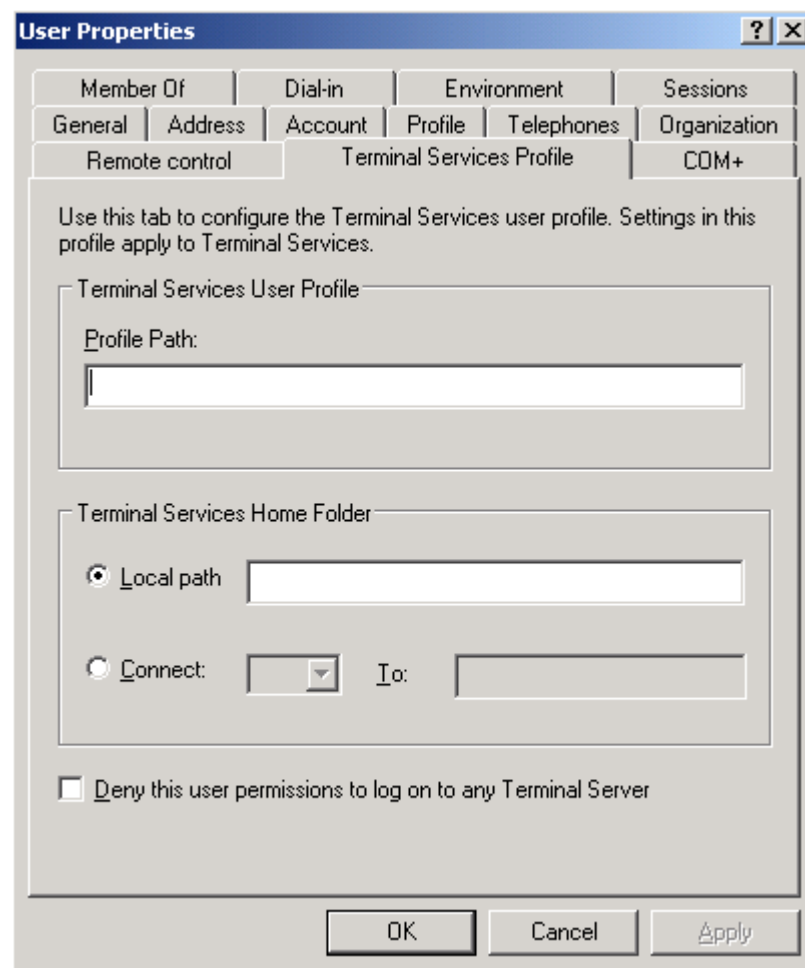



Figure 3.9: Configuring a Terminal Services profile on a user account.

On this tab, specify the full UNC path to the profile you want this user to load at logon. If you want to use mandatory profiles, set all users to the same path, configure the central profile, and change the NTUSER.DAT filename to NTUSER.MAN. On this tab, you can also configure a separate user home directory to be used on terminal servers, but doing so is fairly rare, as most likely your users will need the same home directory for both workstations and terminal servers.

 You can also flag a specific user account to deny access to any terminal server on this tab.

On a Per-User Basis via ADSI

Like the other Terminal Services user attributes, you can configure Terminal Services profile settings via ADSI. After opening a connection to the user object:

```
Set objUser = GetObject("WinNT://<domain name>/<username>,user"
```

or

```
Set objUser = Get Object("LDAP://<distinguished name of user>")
```

Set any of the following attributes:

```
objUser.TerminalServicesProfilePath = ["path to directory"]
objUser.TerminalServicesHomeDirectory = ["path to directory"]
objUser.TerminalServicesHomeDrive = ["drive letter:"]
objUser.AllowLogon = [1,0]
```

and then save your changes:

```
objUser.SetInfo
```

Via Group Policy

A feature that is new in WS2K3 is the ability to override the user account settings and configure a terminal server to use only local profiles or to use a separate roaming profile share from the one defined in the user account. To take advantage of either of these options, use the Group Policy Management Console to create or edit a GPO that applies to your terminal servers.

To set an alternative roaming profile path, drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services. Configure the *Set Path for TS Roaming Profiles* setting with the UNC path of the file share you want to use. Do not specify the %username% variable, as it is automatically added to the path for each user. Figure 3.10 shows the UI for this setting.

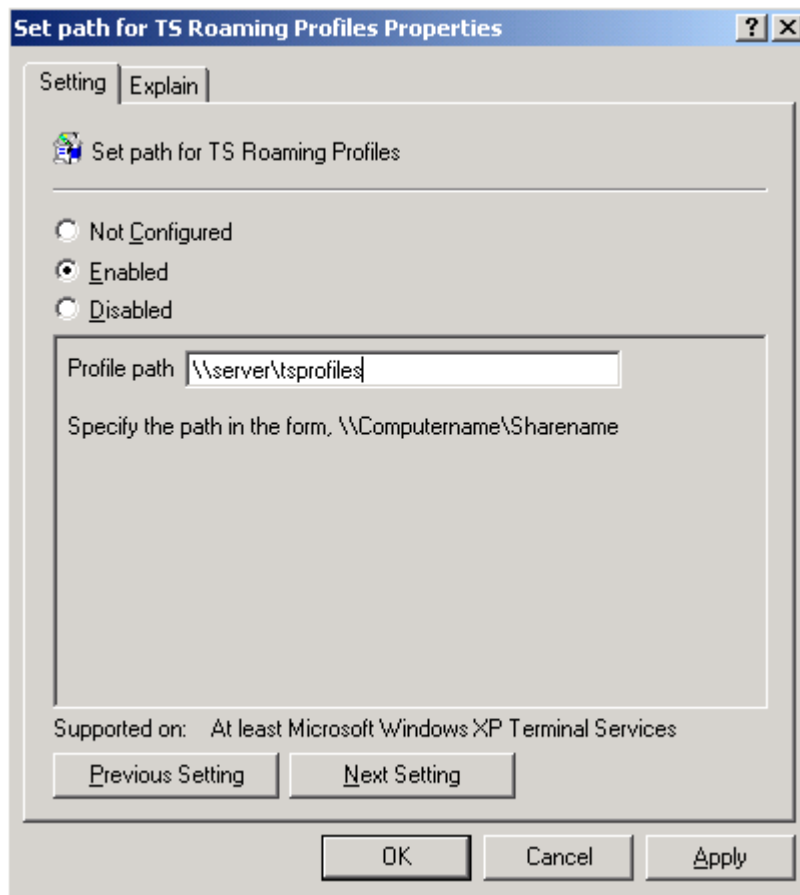


Figure 3.10: Configuring an alternative Terminal Services profile share via GPO.

To configure the terminal server to ignore the profile path in the user account and exclusively use local profiles, drill to Computer Configuration | Administrative Templates | System | User Profiles, and set *Only allow local user profiles* to enabled (see Figure 3.11).

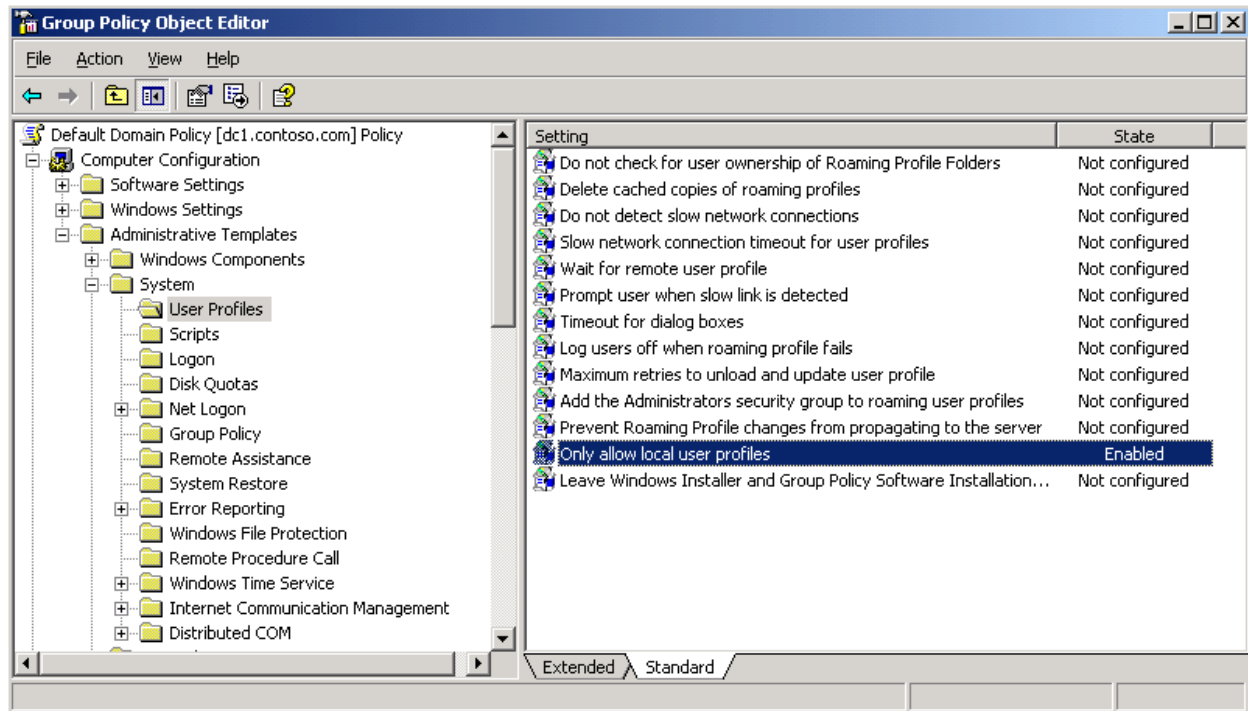


Figure 3.11: Configuring a server to only use local profiles.

Per Server (Local Machine Policy)

You can configure either of the previously mentioned settings on a per-server basis without creating a separate GPO in your domain. To do so, edit the local machine policy by launching the Group Policy Editor (GPEDIT.MSC) from a Run dialog box.

Third-Party Products for Profile Management

User profiles can be the biggest challenge for any Windows systems administrator. Users want the convenience of roaming profiles, but with frequent logons to multiple machines, they can become corrupt. Systems administrators want the central control of mandatory profiles but have to face the challenge of editing them when changes are needed. There are third-party products—such as triCerat’s Simplify Suite, AppSense’s AppSense Management Suite, and BrsSuite from BrainSyS—that help you bridge the gap between the two.

One of the components of Simplify Suite is Simplify Profiles, which gives you the ability to leverage the stability of mandatory profiles while still enabling users to save designated settings across sessions and servers. Simplify Profiles also makes changes to settings in the profiles easy by providing a central UI for edits. Alternatively, BrsSuite is a free option that stores user profile information in a SQL database, allowing you to make single-point changes to all profiles while enabling your users to store designated per-user settings.

User Profile Hive Cleanup Service

Regardless of whether you use local, roaming, or mandatory profiles, user sessions can get locked in memory at logoff. This can occur because of software defects, quick logon/logoffs, or hung applications. When a user session is locked, the server cannot properly unload the profile and cannot reclaim all the resources used by the session. Microsoft's User Profile Hive Cleanup service can assist in this process.

The User Profile Hive Cleanup service monitors the user logoff process, and if any user process handles are keeping the user hive from being unloaded, the service will proactively kill the process so that the profile can be unloaded. The User Profile Hive Cleanup service is fully supported by Microsoft and should be installed on all your terminal servers.

How to Install UPH Clean

To install the User Profile Hive Cleanup service, download the MSI package from Microsoft's Web site at <http://www.microsoft.com/downloads/details.aspx?familyid=1b286e6d-8912-4e18-b570-42470e2f3582&displaylang=en>. Either install the package manually or assign it to your terminal servers via Group Policy. The User Profile Hive Cleanup service is a simple "set it and forget it" tool, although there are some advanced logging options that you can control via the registry. See the readme.txt file for the current information about these options.

How to Manage Printing in a Terminal Server Environment

When a user connects to a terminal server, if client printer mapping is enabled, the terminal server enumerates all printers defined on the workstation and attempts to connect to them. RDP 5.2 supports USB, Serial (COM), Parallel (LPT), and network printers. Before the release of WS2K3 SP1, the terminal server had to have a native driver for the printer pre-installed by an administrator for the mapping to be successful. SP1 introduced a fallback printer driver that can be used whenever a native driver is not available.

Microsoft Fallback Printer Driver

The Microsoft fallback printer driver can be used whenever the native driver is not installed on the terminal server. The fallback driver feature is disabled by default, so if the native driver is not available, the client printer will not be mapped. The driver supports both Printer Control Language (PCL) and PostScript printer languages, but only basic functionality, so advanced features—such as duplex, multiple paper trays, high resolution, and so on—will not be available.

To enable the fallback printer driver, use the Group Policy Management Console to create or edit a GPO that applies to your terminal servers, then drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Client/Server Data Redirection. Open the *Terminal Server Fallback Printer Driver Behavior* setting, which Figure 3.12 shows.

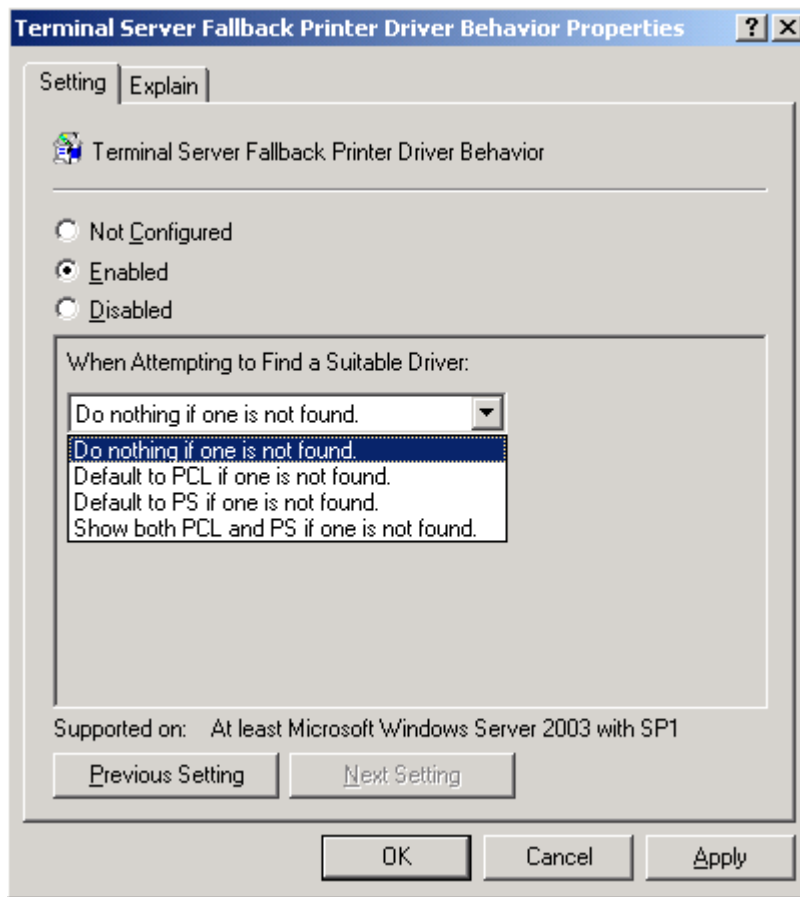



Figure 3.12: Configuring the fallback printer driver behavior.

After enabling this setting, you must select the appropriate behavior for the server to take:

- Do nothing if one is not found—Disables the fallback driver. This option is the same as not configuring or disabling the setting.
- Default to PCL if one is not found—If no suitable printer driver can be found, the terminal server uses the Hewlett-Packard compatible PCL fallback printer driver.
- Default to PS if one is not found—If no suitable printer driver can be found, terminal server uses the Adobe PostScript fallback printer driver.
- Show both PCL and PS if one is not found—In the event that no suitable driver can be found, show both PostScript-based and PCL-based fallback printer drivers.

 You can also enable the fallback printer driver locally on a specific server by editing the local machine policy (GPEDIT.MSC).

Third-Party Products for Printer Driver Management

The fallback printer driver is a useful advancement for Microsoft Terminal Services, however, it is still a very basic driver and does not support advanced features of many printers. If your users need the full functionality of their printers when working on the terminal server, you should look to third-party tools to enhance the native printing functionality.

One such tool is triCerat's Simplify Printing, part of the Simplify Suite. Simplify Printing enables users to access the local printer properties and features even when working in a terminal server-based application. Simplify Printing does require both a server and client install, but the advantages of not having to install drivers on your terminal servers while still providing users with the full features of their printers is worth the effort.

Another third-party option for printer management is ThinPrint, which uses Driver Free Printing to spool documents to local printers while managing the bandwidth and performance latency that can arise during printing. ThinPrint also requires both client and server components.

Group Policy Reference

The following lists highlight the Group Policy settings user environment considerations discussed in this chapter:

- To configure per-user session timeouts—User Configuration | Administrative Templates | Windows Components | Terminal Services | Sessions
- To configure per-server session timeouts—Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Sessions
- To configure client device redirection—Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Client/Server Data Redirection
- To configure an alternative path for Terminal Services profiles—Computer Configuration | Administrative Templates | Windows Components | Terminal Services; configure the *Set Path for TS Roaming Profiles* setting
- To configure a server to use only local profiles—Computer Configuration | Administrative Templates | System | User Profiles; set *Allow only local user profiles* to enabled
- To enable the Microsoft fallback printer driver—Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Client/Server Data Redirection; set the *Terminal Server Fallback Printer Driver Behavior* setting

Command-Line Reference

Listing 3.2 provides an example script to configure all the terminal services-related attributes of a user object via ADSI.

```
Set objUser = GetObject _
    ("LDAP://cn=joe.user,ou=users,dc=example,dc=domain,dc=com")
` or: Set objUser = GetObject("WinNT://example/joe.user,user")
objUser.ConnectClientDrivesAtLogon = 1
objUser.ConnectClientPrintersAtLogon = 1
objUser.DefaultToMainPrinter = 1
objUser.TerminalServicesInitialProgram = "C:\windows\notepad.exe"
objUser.TerminalServicesWorkDirectory = "c:\windows"
objUser.TerminalServicesProfilePath = _
    "\\server\tsprofiles\joe.user"
objUser.TerminalServicesHomeDirectory = _
    "\\server\home\joe.user"
objUser.TerminalServicesHomeDrive = "H:"
objUser.AllowLogon = 1
objUser.MaxDisconnectionTime = 15
objUser.MaxConnectionTime = 0
objUser.MaxIdleTime = 180
objUser.BrokenConnectionAction = 0
objUser.ReconnectionAction = 0
objUser.EnableRemoteControl = 1
objUser.SetInfo
```

Listing 3.2: An example script to label all the terminal services-related attributes of a user object via ADSI.

Summary

This chapter covered session timeouts, client device redirection, and user profile management. It walked you through the steps needed to configure these features manually, via Group Policy, and by script. The next chapter will cover the configuration of load balancing, session directory, and user session management.

Chapter 4: Management, Load Balancing, and Optimization

An important skill in terminal server administration is configuring the settings required for a user to establish a terminal server session, and the tools used to manage the session once it is established. This chapter will show you how to configure and join a Session Directory to help users find orphaned sessions across multiple terminal server farms. Finally, it will introduce you to Windows Systems Resource Manager as well as third-party products that can help manage the allocation of resources to keep your terminal servers running at maximum capacity.

Requirements for Logging on to Terminal Server

WS2K3 has three distinct layers of protection that enable you to control who can log on to a terminal server. For a user to log on to a terminal server, these settings must be in place:

- The Allow log on through Terminal Services right—Under Win2K, you are required to grant the Log on locally right to all users who need access to a terminal server. This requirement poses a potential security hole, as it allows users to log on at the console of the server, thus bypassing any restrictions you configured for RDP. WS2K3 separates the right to log on to the console from the right to log on through Terminal Services. By default, on WS2K3, the Allow log on through Terminal Services right is granted to Administrators and to the Remote Desktop Users group.
- Permission to use RDP—An administrator can set permissions on RDP through the Terminal Services Configuration tool. As Chapter 2 mentioned, Microsoft's new focus on security has changed the default permissions for the protocol in WS2K3. Under Win2K, the local Users group is granted access to RDP; WS2K3 restricts this right to the local Remote Desktop Users group. Thus, you must add your users to this group in order for them to log on to the terminal server.
- The Deny this user permissions to log on to any Terminal Server check box—In the properties of each user object in AD, there is a Deny this user permissions to log on to any Terminal Server check box that controls whether the user is enabled to log on to a terminal server. This check box is unchecked by default.

If a user receives a *You do not have permission to access this session* error message, one of these three settings is the culprit. The following sections will explain how and where to adjust these settings.

How to Manage User Rights Assignments

User rights assignments are established during the installation of Windows from the local security templates. To modify them individually after the installation, use either the local Group Policy Editor (GPEDIT.MSC) or the Local Security Policy Administrative tool.

In either tool, drill to Security Settings | Local Policies | User Rights Assignment. (In the Group Policy Editor, the Security Settings node is located under Computer Configuration | Windows Settings.) You will then see a list of the available User Rights. By default, the Administrators and the Remote Desktop Users groups are granted the *Allow log on through Terminal Services* right, as Figure 4.1 shows.

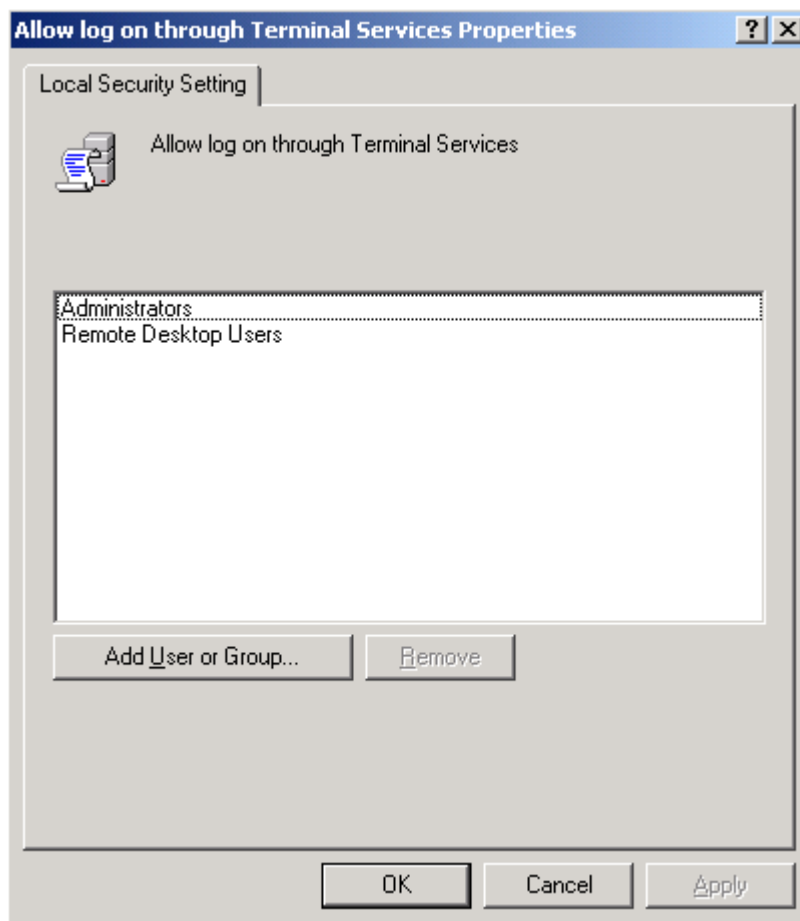


Figure 4.1: The Allow log on through Terminal Services user rights assignment.

In this dialog box, you can add or remove users or groups from having this right. It is recommended, however, that you manage this particular right by adding and removing users from one of the default groups.

Via Group Policy

To centrally manage user rights assignments, use a Group Policy Object (GPO) that applies to the server objects you want to configure. Use the Group Policy Management Console (GPMC) to edit the GPO, then drill to Computer Configuration | Windows Settings | Security Settings | Local Policies | User Rights Assignment. Double-click the *Allow log on through Terminal Services* right, and select the *Define these policy settings* check box, as Figure 4.2 shows.

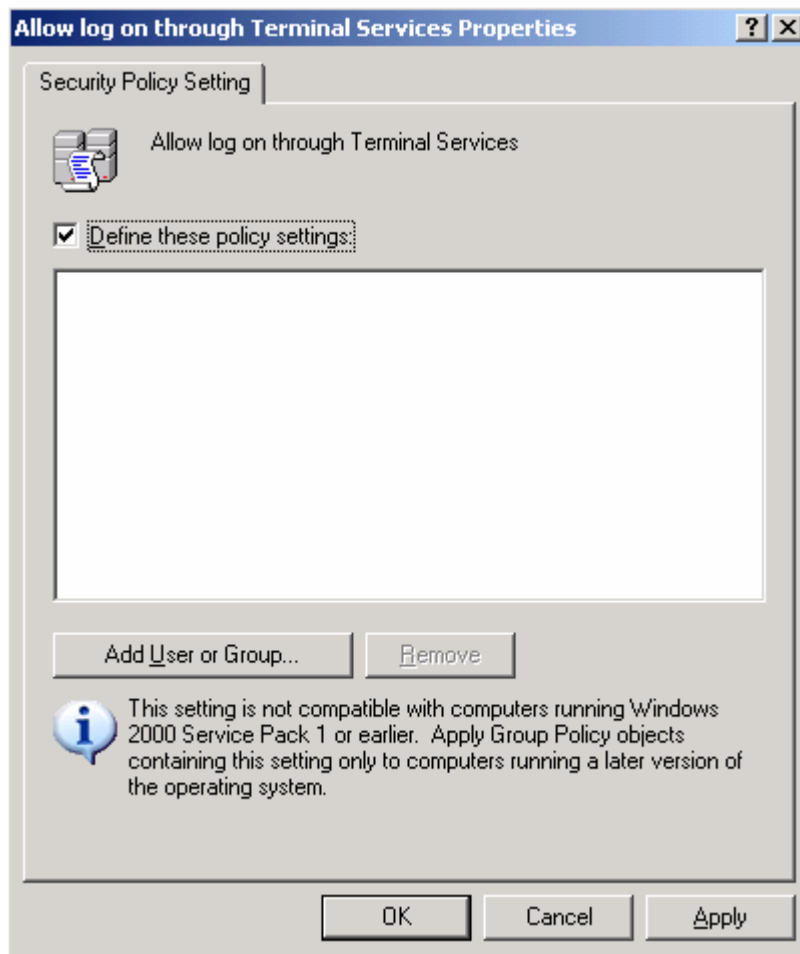



Figure 4.2: Managing user rights assignment via Group Policy.

Next, add the users or groups you want to have this ability. When the policy is refreshed on the target servers, the Group Policy setting will override the local setting.

 Keep in mind that defining the user rights assignment via Group Policy overrides the local setting, so doing so will prevent you from making exceptions on individual servers.

How to Manage RDP Protocol Permissions

The Terminal Services Configuration Administrative Tool (TSCC.EXE) is used to access the permissions on the RDP protocol. Within the tool, select the Connections node on the left, then open the properties window for the RDP-Tcp protocol. Figure 4.3 shows the default settings on the Permissions tab.

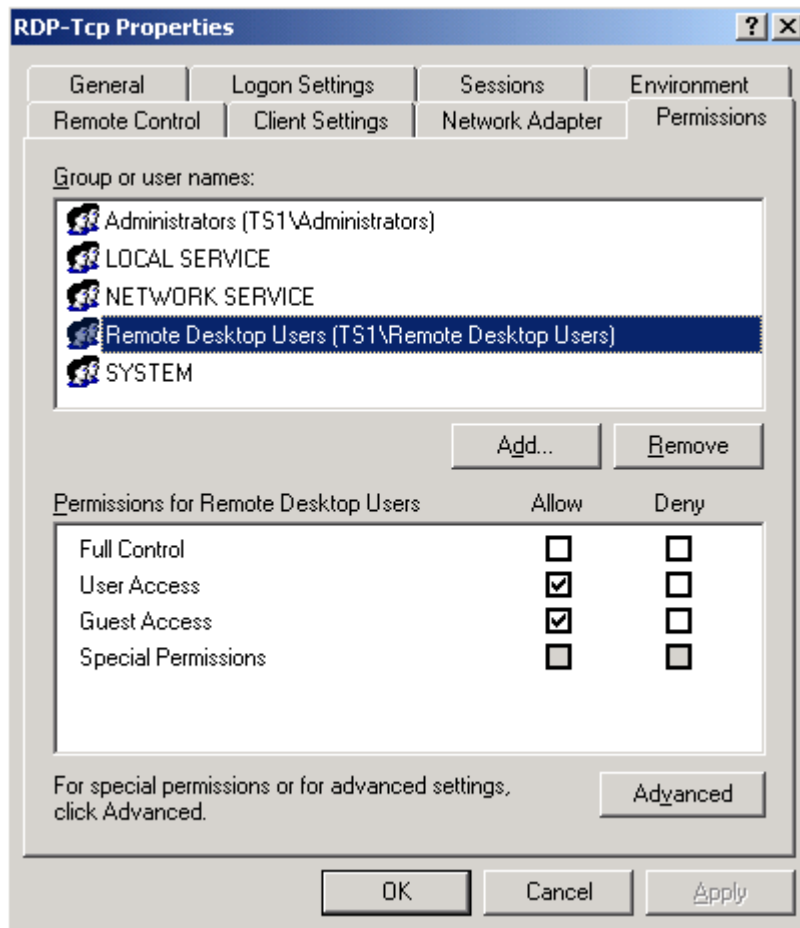


Figure 4.3: Default permissions on the RDP protocol.

By default, the following groups have access to the protocol:

- Administrators—Full Control
- Local Service—Full Control
- Network Service—Full Control
- Remote Desktop Users—User Access
- System—Full Control

In this dialog box, you can add users or groups to the permissions set as well as modify the access level of existing groups. It is common, for example, to grant your Help desk staff full control of the protocol so that they can shadow and reset user sessions without having local administrative rights on the server.

Via Group Policy

Microsoft does not provide a way to centrally manage the permissions on the RDP protocol. However, there is a Group Policy setting to disable the Permissions tab altogether so that local administrators cannot modify the default permissions via the GUI. To enable this setting, use the GPMC tool to edit a policy that applies to your terminal servers.

Drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services and set *Do not allow local administrators to customize permissions* to enabled. Once this is done, the Permissions tab in the TSCC tool becomes read-only.

How to Manage the Remote Desktop Users Group

Membership in the Remote Desktop Users group grants you both the *Allow log on through Terminal Services* right and user level access to the RDP protocol. Thus, it is best to manage access to your terminal servers via this group.

Manually

Use the Computer Management administrative tool to modify the membership of the Remote Desktop Users group. To do so, drill to System Tools | Local Users and Groups | Groups (see Figure 4.4).

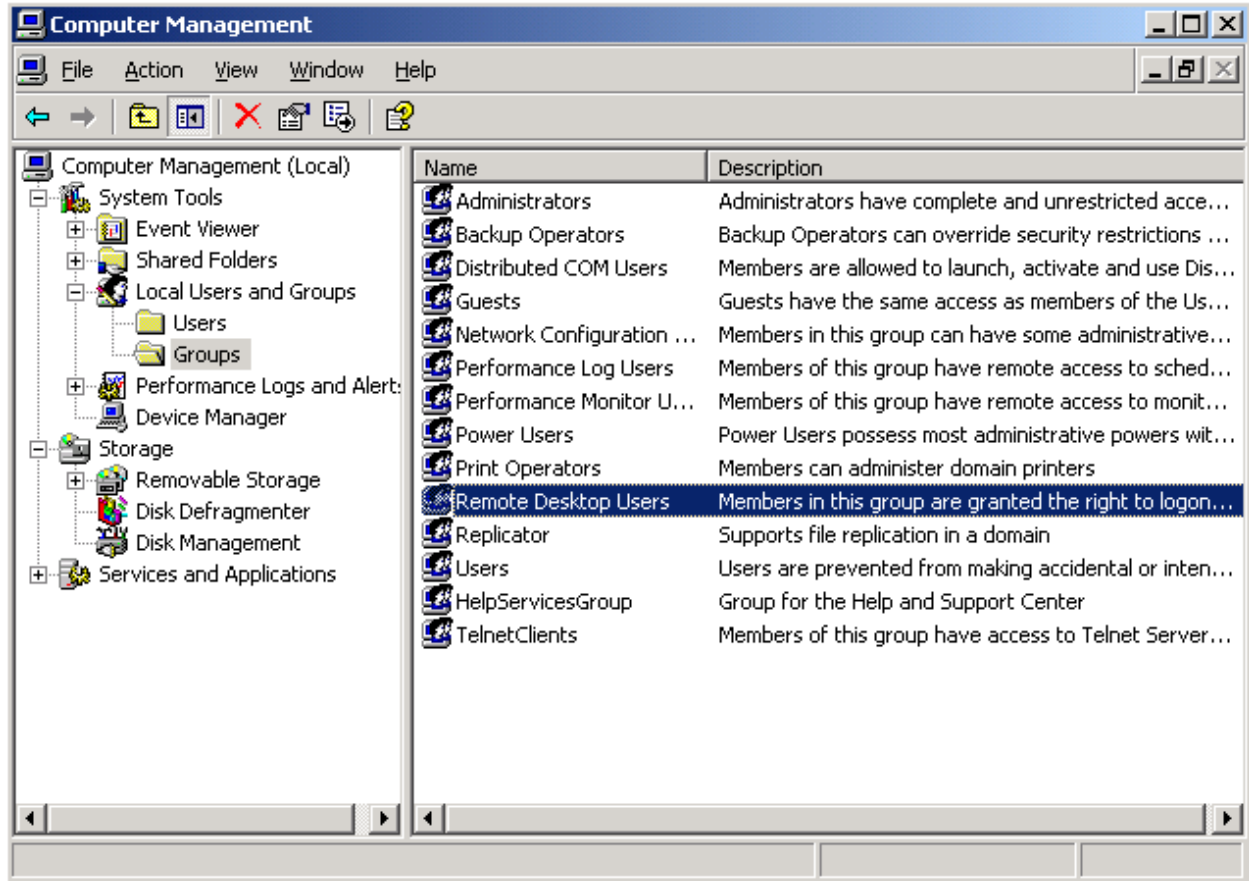


Figure 4.4: Managing the Remote Desktop Users group in the Computer Management console.

Via Script


You can programmatically add or remove members in the Remote Desktop Users group via script. In Shell script, use the NET LOCALGROUP command:

```
net localgroup <group> <member> {/add | /delete}
```

Example:

```
net localgroup "Remote Desktop Users" "contoso\Domain Users" /add
```

In the example, you would be adding the Domain Users group from the Contoso domain to the local Remote Desktop Users group.

 You can also modify group membership in VBScript or JScript by using the WinNT:// object.

Via Group Policy

To centrally manage membership in the Remote Desktop Users group, use a Restricted Group setting in a GPO. Launch the GPMC tool, and edit a GPO that applies to the server objects you want to manage. Next, drill to Computer Configuration | Windows Settings | Security Settings | Restricted Groups.

Next, right-click and select *Add Group...* and enter *Remote Desktop Users*. You will then see the Restricted Groups properties window in which you can explicitly control both the users and groups that are a member of this group as well as which groups this group is a member of. Figure 4.5 shows the interface with the Domain Users group added.

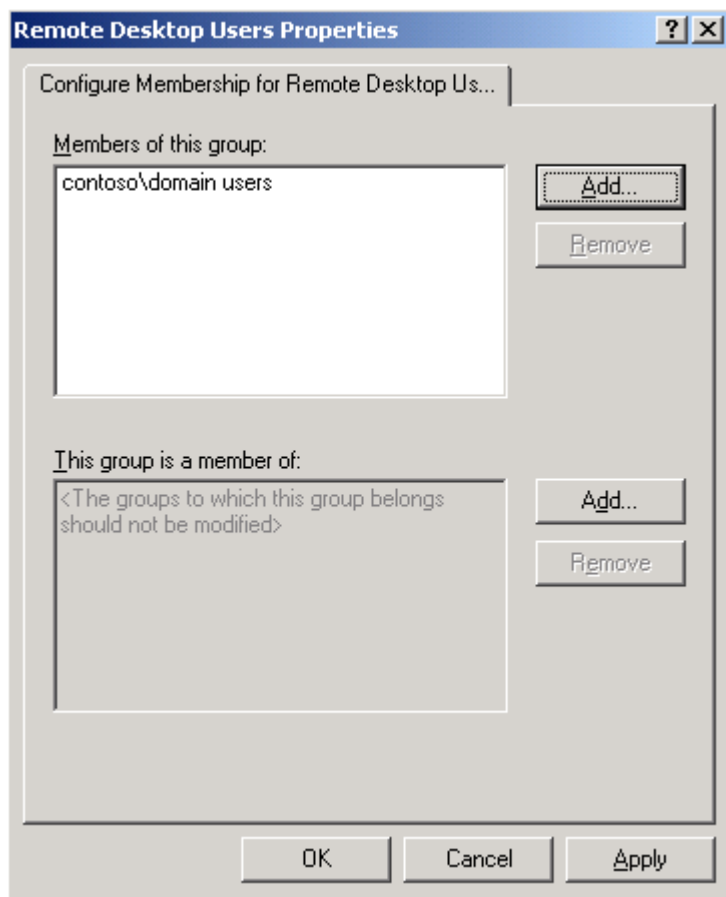



Figure 4.5: Configuring a restricted group.

 As with most Group Policy settings, the configuration in the GPO overrides the local setting (the member lists do not merge), so once you define a restricted group, you will lose the ability to add or remove members on individual servers without first removing them from the scope of the entire policy.

How to Manage the *Deny this user permissions to log on to any Terminal Server* Setting

Regardless of how the user rights assignments, protocol permissions, and group memberships are configured, you can still prevent specific users from being able to log on to terminal servers at the user account level. Every user object has an attribute called *Deny this user permissions to log on to any Terminal Server*, which can be enabled to achieve exactly that.

Manually

To enable this setting manually, use either the Computer Management administrative tool (for local accounts) or Active Directory Users and Computers (for domain accounts), and open the properties dialog box for the user you want to restrict. Select the *Deny this user permissions to log on to any Terminal Server* check box on the Terminal Services Profile tab, as Figure 4.6 shows.

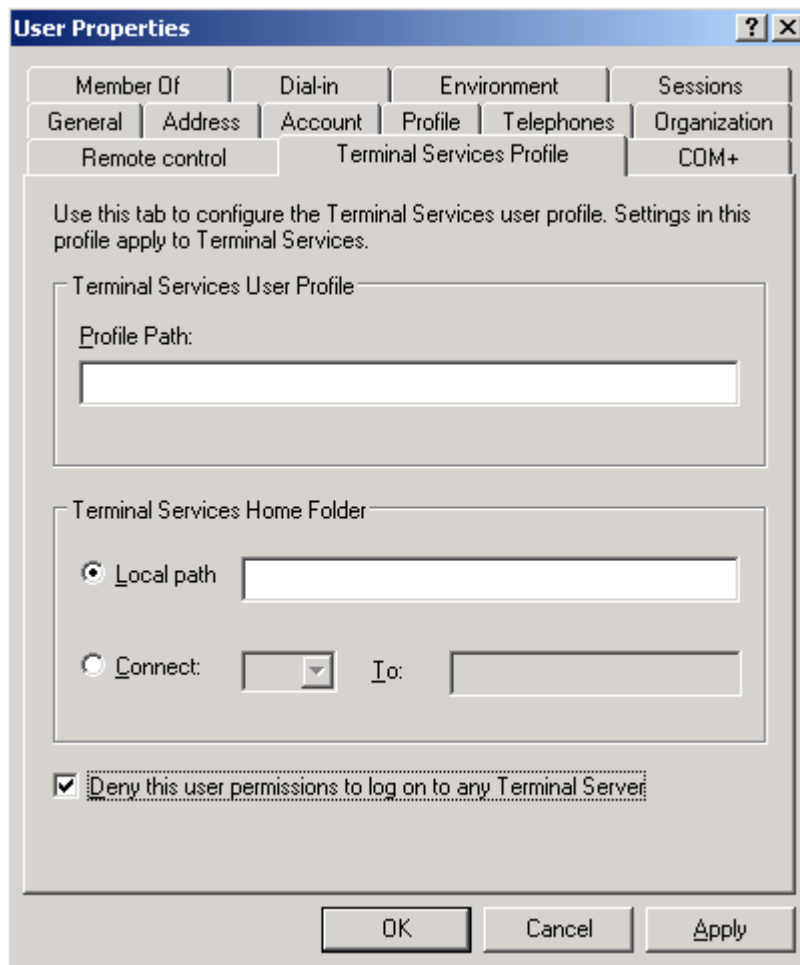


Figure 4.6: Preventing a specific user account from logging on to any terminal server.

Via ADSI

Like the other Terminal Services user attributes, you can configure Terminal Services profile settings via ADSI. After opening a connection to the user object:

```
Set objUser = GetObject("WinNT://<domain name>/<username>,user")
```

or


```
Set objUser = Get Object("LDAP://<distinguished name of user>")
```

Set the following attribute:

```
objUser.AllowLogon = [1,0]
```

and then save your changes:

```
objUser.SetInfo
```

 In WS2K3 Service Pack 1 (SP1), the setting name in the GUI was changed from *Allow logon to Terminal Server* to *Deny this user permissions to log on to any Terminal Server*, but the ADSI attribute name was not changed. Thus, if you want the deny box to be enabled, you have to set the AllowLogon attribute to 0 (zero).

Managing User Sessions

Terminal Services gives you the ability to remotely manage user sessions. You can remote control them to view what your users are doing or assist them with problems or application configuration. You can also forcibly log off users or hard-reset a session if it has become hung or the user has disconnected from it without logging off.

How to Configure Remote Control Options

As you saw in Chapter 3, you can configure most Terminal Services setting on a per-user basis. Remote control settings are no exception. To do so, launch the Active Directory Users and Computers tool, and open the properties window of the user object you want to configure. Figure 4.7 shows the Remote Control tab of a user object.

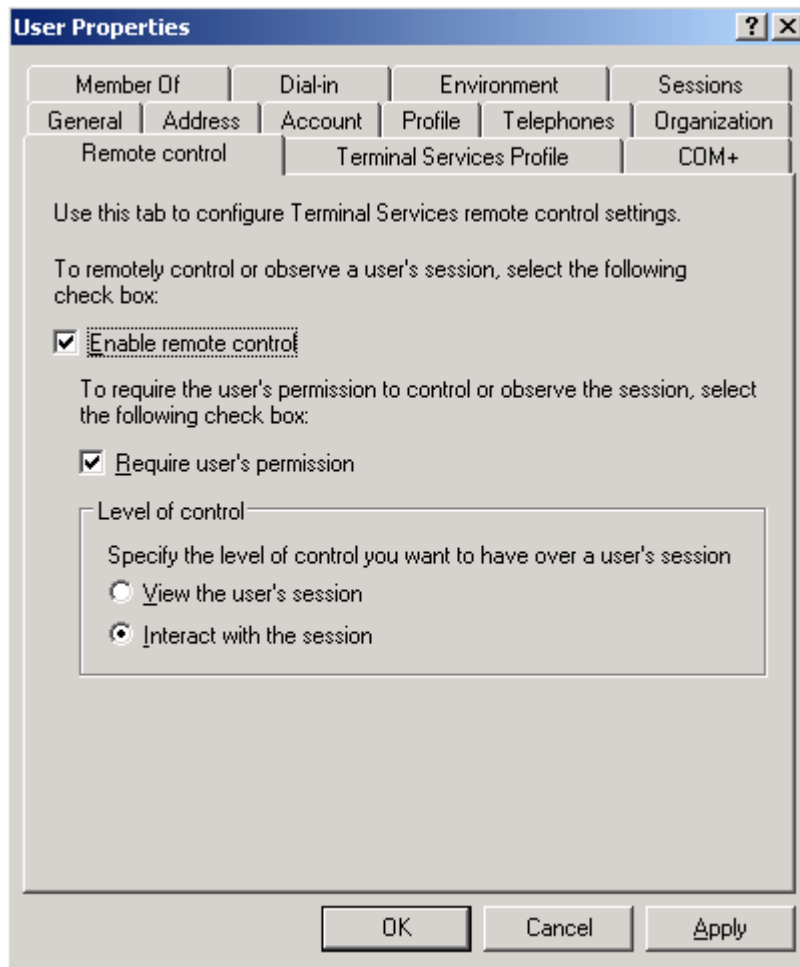


Figure 4.7: Shadow settings on a user object.

In this window, you can enable or disable the ability to shadow this user account, and if it is enabled, you can set whether to prompt the user for permission before allowing the administrator to view the session as well as set the level of control the administrator has over the user's session once shadowing begins. You can select a view-only mode or allow the administrator to interact with the session by controlling the user's mouse and keyboard.

On a Per-User Basis via ADSI

Like the other Terminal Services user attributes, you can configure remote control settings via ADSI. After opening a connection to the user object:

```
Set objUser = GetObject("WinNT://<domain name>/<username>,user"
```

or

```
Set objUser = Get Object("LDAP://<distinguished name of user>")
```

Set following attribute:

```
objUser.EnableRemoteControl = [0,1,2,3,4]
```

```
0 = Disable Remote Control
```

```
1 = Enable Notify & Enable Interact
```

```
2 = Disable Notify & Enable Interact
```

```
3 = Enable Notify & Disable Interact
```

```
4 = Disable Notify & Disable Interact
```

And then save your changes:

```
objUser.SetInfo
```

On a Per-Server Basis via GUI

Although configuring remote control settings on a per-user basis provides the greatest amount of flexibility, it can also be very time consuming and difficult to manage. Most terminal server administrators choose to find settings that are appropriate for all users and configure timeouts on a per-server basis.

To configure per-server remote control settings, launch TSCC.EXE. Open the properties dialog of the RDP-Tcp protocol, and go to the Remote Control tab, as Figure 4.8 shows.

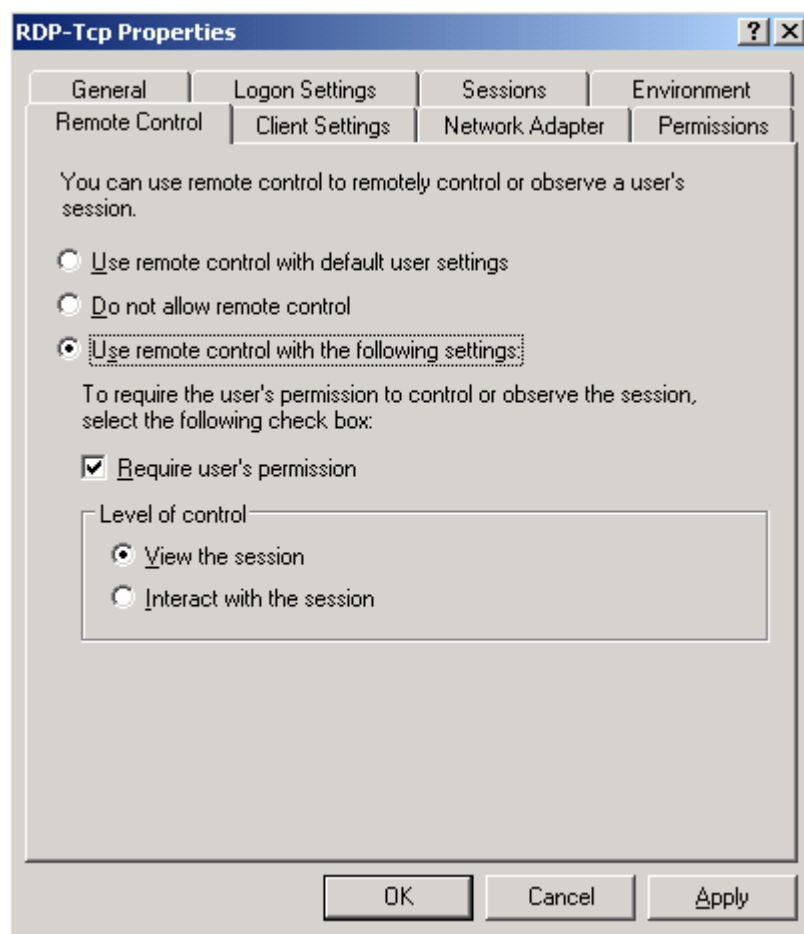


Figure 4.8: Configuring remote control on a per-server basis.

In this window, you can place the server into one of three modes—use the per-user settings, disable remote control on this server altogether, or override the user settings and use the per-server settings. If you select the third option, you then specify whether to require the user's permission and what level of control the administrator will have over the user's session once shadowing begins.

Via Group Policy

You can also centrally configure remote control via Group Policy. You can do so on a per-user or a per-server basis. To configure the settings, use the GPMC tool to edit a policy object that applies to either the user objects or the computer objects you want to receive the settings, then drill to Computer (or User) Configuration | Administrative Templates | Windows Components | Terminal Services. Set *Sets rules for remote control of Terminal Services user sessions* to Enabled, and select from one of the four options that Figure 4.9 shows.

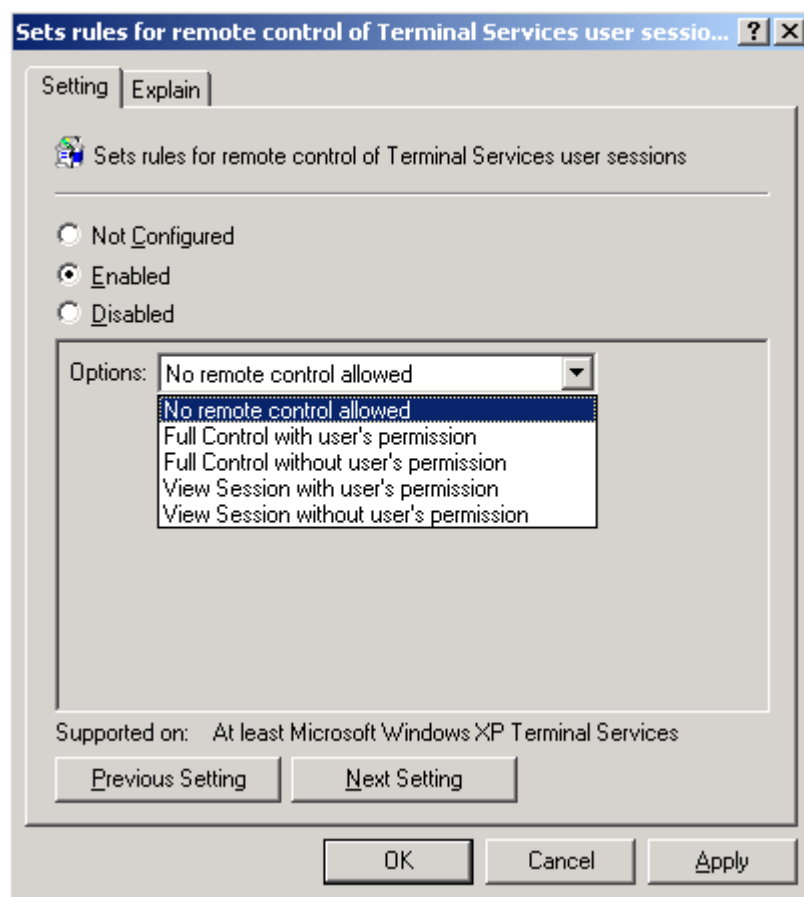


Figure 4.9: Configuring remote control via Group Policy.

Order of Precedence

With all the options available to configure remote control settings, it can be difficult to troubleshoot where the setting is coming from when shadowing does not work. If possible, you should select a single method to configure all remote control settings. If, however, you have a complex environment that requires multiple configurations, you should be aware of the order of precedence that the options take.

The settings are applied in the following order, and the last set applied is the final result. The settings do not merge:

1. Settings on the user object
2. Per-user Group Policy settings
3. Per-server settings in the Terminal Services Configuration tool
4. Per-server Group Policy settings

How to Control a User Session

The primary tool for interacting with user sessions is the Terminal Services Manager administrative tool. With this tool, you can connect to any server and enumerate all user sessions on it. You then right-click the target session and select one of the available options, as Figure 4.10 shows.

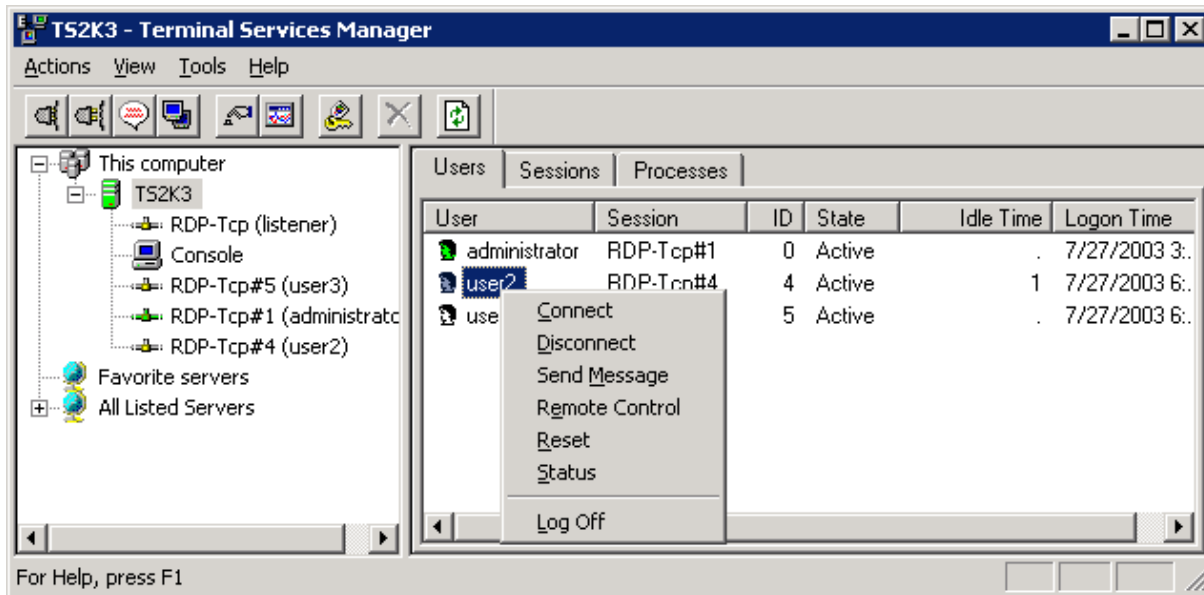


Figure 4.10: The Terminal Services Manager administrative tool.

The default setting for this tool is to the local computer, but you can either browse the domain for other terminal servers through the All Listed Servers node or connect to a specific server by name by right-clicking the All Listed Servers node and selecting *Connect to computer...*

☞ Connecting manually can be very helpful as only terminal servers are listed automatically. If you want to enumerate remote desktop sessions on a non-terminal server, you must connect manually.

Via Command Line

The command line equivalents to the Terminal Services Manager tool are:

```

QUERY USER or QUSER - enumerates user sessions on a server
SHADOW - initiates a remote control session
LOGOFF - logs off a specific user session
RESET - resets a specific user session
MSG - sends a message to a user session

```

All of these commands except for RESET support both local and remote sessions. Listing 4.1 shows the syntax for each command.

```

QUERY USER [username | sessionname | sessionid] [/SERVER:servername]
username          Identifies the username.
sessionname       Identifies the session named sessionname.
sessionid         Identifies the session with ID sessionid.
/SERVER:servername The server to be enumerated
Example:
QUSER /server:TermServ01
would enumerate all user sessions on the Terminal Server named
TermServ01

SHADOW {sessionname | sessionid} [/SERVER:servername]
sessionname       Identifies the session by name.
sessionid         Identifies the session by ID.
/SERVER:servername The server containing the session
Example:
SHADOW 3 /server:TermServ01
would remote control session 3 on server TermServ01


LOGOFF [sessionname | sessionid] [/SERVER:servername]
sessionname       The name of the session.
sessionid         The ID of the session.
/SERVER:servername The server containing the session
Example:
LOGOFF 3 /server:TermServ01
would logoff session 3 on server TermServ01

RESET [sessionname | sessionid]
sessionname       The name of the session.
sessionid         The ID of the session.
Example:
RESET 3
would logoff session 3

MSG {username | sessionname | sessionid | @filename | *}
  [/SERVER:servername] [/TIME:seconds] [/V] [/W] [message]
username          Identifies the specified username.
sessionname       The name of the session.
sessionid         The ID of the session.
@filename         Identifies a file containing a list of
usernames, sessionnames, and sessionids to send the message to.
*                Send message to all sessions on specified
server.
/SERVER:servername server to contact (default is current).
/TIME:seconds     Time delay to wait for receiver to acknowledge
msg.
/V               Display information about actions being
performed.
/W               Wait for response from user, useful with /V.
message          Message to send. If none specified, prompts for
it or reads from stdin.
Example:
MSG 3 /server:TermServ01 Hello there!
Sends the message "Hello there!" to session 3 on server TermServ01

```

Listing 4.1: Syntax for the Terminal Services Manager tool command-line equivalent commands.

 To manage sessions on the server you are logged into, you can also access them via the Users tab in Task Manager.

Session Directory

If your terminal server environment needs to service more concurrent users than a single server can support, you will need to distribute users across multiple servers. The easiest way to do so in a native Microsoft environment is to use Network Load Balancing (NLB). Doing so groups servers together into a logical farm and creates an alias for the entire farm of servers. When users want to connect to a terminal server, they connect to the cluster name instead of a specific server and NLB redirects them to an available server.

One of the challenges of creating a load-balanced cluster of native Windows terminal servers is how to deal with disconnected sessions. As Chapter 3 explored, administrators have the ability to configure timeouts for idle and disconnected sessions on a terminal server. You can either set these timeouts very low, giving only enough time for a user to reconnect from the same client device and IP address in the event of a network failure, or you can set it fairly high, giving your users the ability to disconnect from a session, leave applications running, then reconnect at a later time to pick up where they left off.

These settings work perfectly well in a single server environment. When the user reconnects to the server, Session Manager reconnects them to his or her existing session. If, however, you have a load-balanced cluster of terminal servers, Session Manager is unaware of sessions on the other servers in the cluster. Microsoft addresses this challenge in WS2K3 with Session Directory.

The Session Directory server maintains a dynamic database that maps user names to open sessions on all terminal servers in the cluster. This feature enables users to be reconnected to existing sessions on any server in the cluster, regardless of which server the NLB service initially directs/connects them to.

Required Components

A Session Directory farm requires a group of terminal servers that are all members of the same NLB cluster as well as a server to host the Session Directory itself. To take advantage of the Session Directory feature, all terminal servers in the cluster must be running WS2K3 Enterprise or Datacenter edition. The Session Directory server can use any edition of WS2K3. You can even create the Session Directory on one of the terminal servers in the cluster, although it is not recommended because that would prevent you from taking that terminal server offline for software installations/upgrades without impacting the entire cluster.

To configure Session Directory, begin with the server you want to host the session database. On this server, open either the Computer Management or Services administrative tool to access the Terminal Services Session Directory (among the available services) Go into this service's properties, set the startup type to automatic, and start the service (see Figure 4.11).

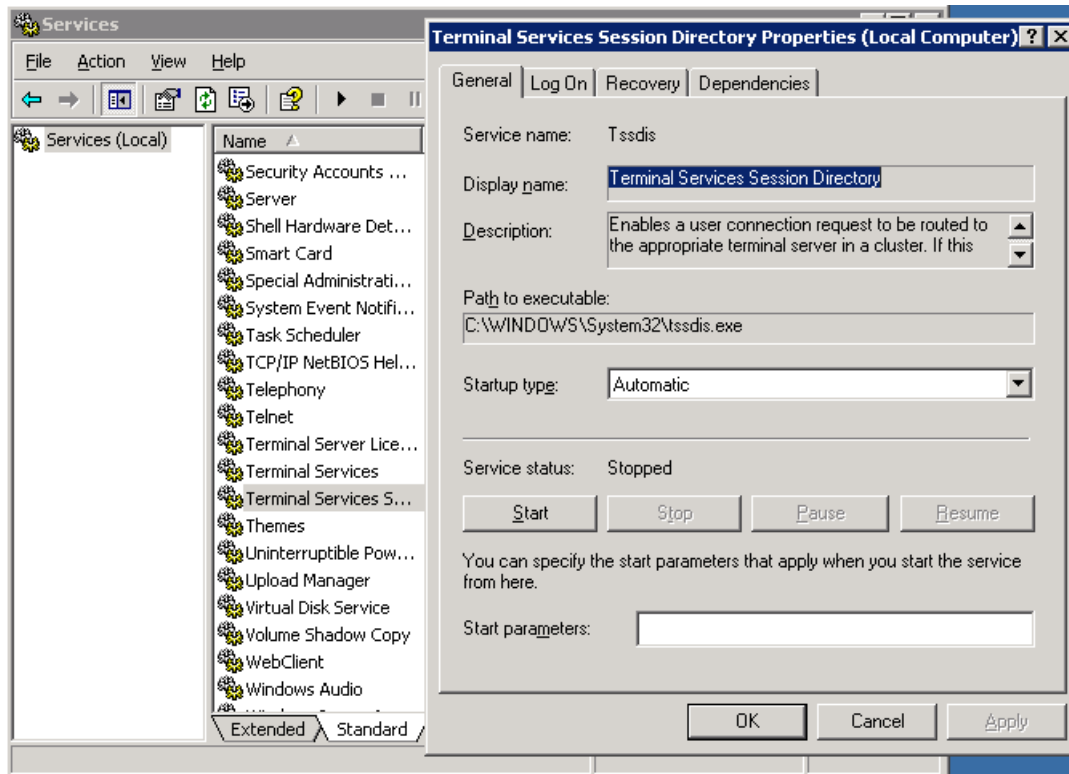


Figure 4.11: Enabling the Terminal Services Session Directory service.

The first time the Session Directory service is started, it will create a new local group on the server called Session Directory Computers. For a terminal server to inform the Session Directory server of the terminal server's sessions or query the Session Directory for sessions on other servers, the terminal server must be a member of this group. You can either add the individual computers in the cluster to the group or create a domain group containing the terminal servers and add that group to Session Directory Computers.

How to Join a Terminal Server to a Session Directory

You can join a terminal server to a Session Directory manually by using TSCC.EXE. In the Server Settings node, open the properties dialog box for Session Directory, as Figure 4.12 shows.

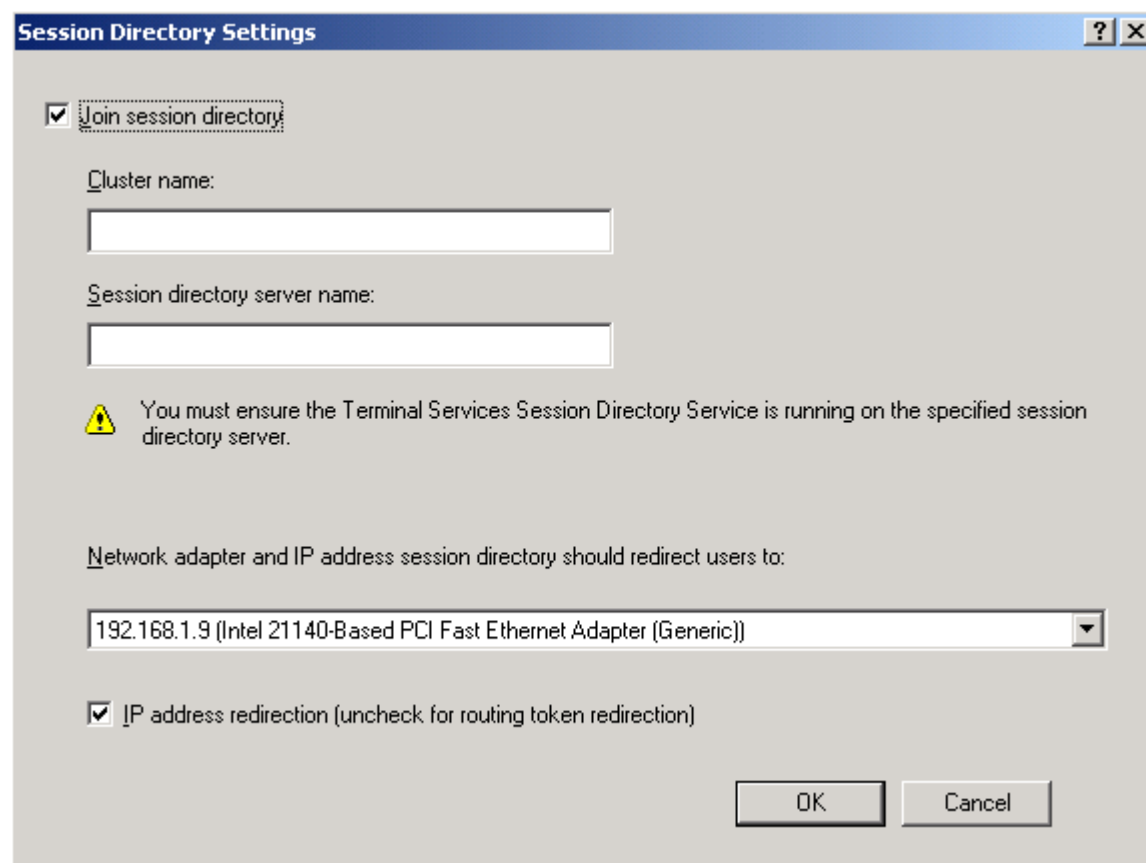



Figure 4.12: Adding a terminal server to a Session Directory.


In this window, select the *Join session directory* check box, and enter the Session Directory cluster name of which this server will be a member as well as the name of the server hosting the Session Directory. If your terminal server is multi-homed, you will need to specify which IP the RDP traffic should be directed to. You also need to specify which type of redirection your NLB cluster uses. Leave this box selected if you are using native Microsoft NLB services.

 Don't forget, you need to add the server to the Session Directory Computers group on the Session Directory server before you can join it to the session directory.

Via Group Policy

Instead of manually entering the cluster name and Session Directory server name on each server, you can centrally configure the Session Directory settings via Group Policy. Use GPMC to edit a policy that applies to all the server objects you want to join to the Session Directory. Then drill to Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Session Directory. Next, you must configure all four settings:

- Join Session Directory—Enabled joins the server to a session directory
- Session Directory Server—The name of the server hosting the Session Directory
- Session Directory Cluster Name—The name of the cluster
- Terminal Server IP Address Redirection—Enabled for IP address redirection and disabled for token-based redirection

 If you are configuring your Session Directory settings via Group Policy, create a domain group for all of the servers in the cluster and add it to the Session Directory Computers group on the Session Directory server. You can even use the group as the scope of your GPO so that by simply adding a terminal server to the group, the server is added to the Session Directory Computers group and receives the GPO settings.

Windows System Resource Manager

Windows System Resource Manager (WSRM) is a free add-on to WS2K3 Enterprise and Datacenter editions. WSRM manages the allocation of processor and memory resources based on a rule set that you can configure. You can define resource priorities based on process name or user/group name, thus ensuring that no single process or user can consume all available resources on your terminal server, impacting other users.

How to Install WSRM

WSRM comes on a CD included with WS2K3 Enterprise and Datacenter editions. If you do not have the CD, it can be downloaded in ISO format from Microsoft at <http://www.microsoft.com/technet/downloads/winsrvr/wsrw.msp>. Once you have the CD, run the setup program for your server architecture (x86, AMD64, or IA64) to install WSRM. No reboot is required.

How to Configure WSRM

WSRM has both a GUI and a command-line interface. Figure 4.13 shows the WSRM administrative tool.

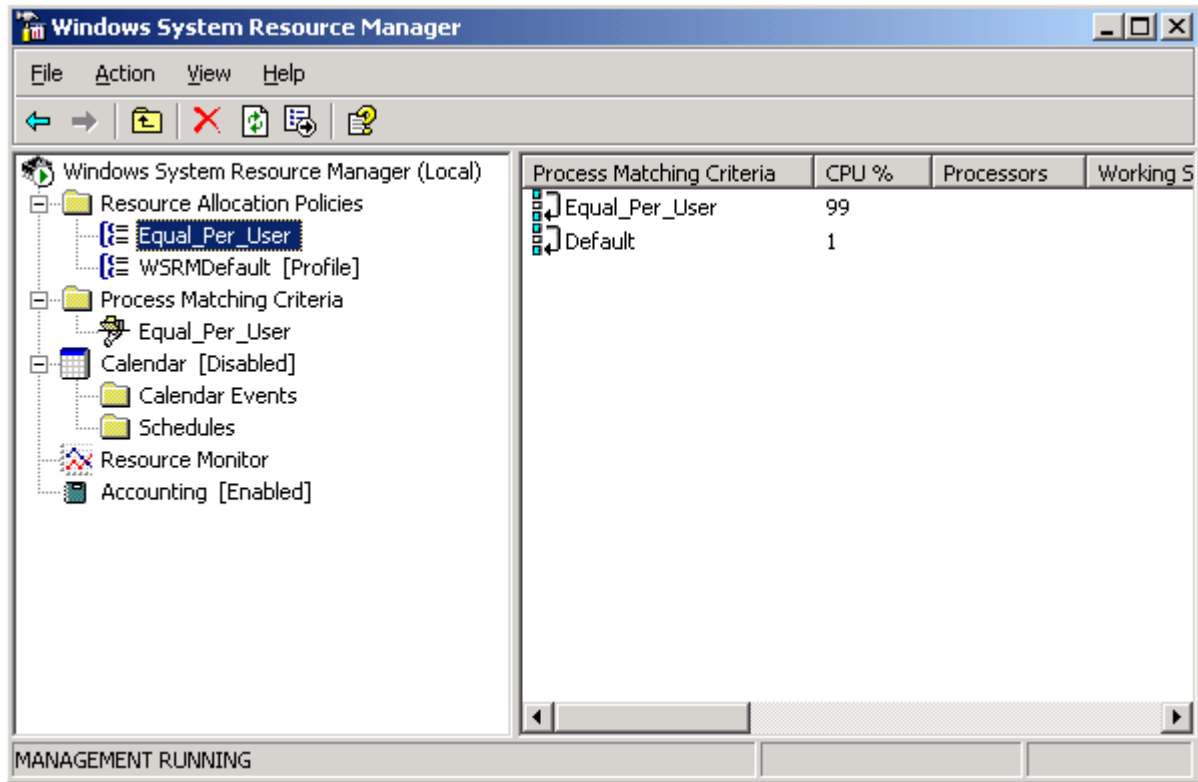


Figure 4.13: The WSRM tool.

WSRM allows you to create multiple resource allocation policies. Each policy defines a set of rules used to allocate server resources. For example, you could specify that Outlook.EXE is entitled to 50 percent of available resources while Winmine.exe (the Mine Sweeper game) is only entitled to 1 percent. This way, one user's game cannot slow down another user's Outlook session.

You can also set up rules that will terminate applications if they try to consume more than their allocated share of resources—a useful feature for runaway applications on a terminal server. WSRM also lets you set up a calendar to change which resource allocation policy is in effect at different times of day. This way, you can provide maintenance or batch processes more resources at night and user applications more resources during the day.

Third-Party Products for Server Resource Management

In addition to WSRM, there are third-party products available to help you manage resources on your terminal servers. These tools are tuned to the specific needs and configurations of a terminal server environment, whereas WSRM is designed to fit a broad range of server types. Two such third-party tools are triCerat Simplify Resources and Citrix Presentation Server.

triCerat Simplify Resources

Simplify Resources is part of triCerat's Simplify Suite, which is considered the standard for managing user profiles, printing, user environment, and resource management in a Terminal Services environment. The company describes this tool in the following manner:

Simplify Resources overcomes application performance problems by taking control of the way Windows allocates and manages CPU and memory resources automatically. It dramatically improves the user experience and increases the number of sessions that can be effectively supported by your servers. Simplify Resources also provides full dll rebasing technology and real-time system monitoring and reporting capabilities tracking performance across an entire farm, per server, session, and application.

Citrix Presentation Server

Citrix Presentation Server is far more than just a resource management tool, as it also replaces Session Directory features and enhances Terminal Services with application publishing abilities. From a resource management perspective, Presentation Server 4.0 includes a number of features to keep your terminal servers running at their maximum capacity.

Group Policy Reference

The following list highlights Group Policy settings for terminal server management, load balancing, and optimization.

To manage user rights assignment: Computer Configuration | Windows Settings | Security Settings | Local Policies | User Rights Assignment—double-click the *Allow log on through Terminal Services* right

To restrict administrators from modifying RDP protocol permissions: Computer Configuration | Administrative Templates | Windows Components | Terminal Services—set *Do not allow local administrators to customize permissions* to Enabled

To manage restricted groups: Computer Configuration | Windows Settings | Security Settings | Restricted Groups

To configure Session Directory: Computer Configuration | Administrative Templates | Windows Components | Terminal Services | Session Directory

Join Session Directory—Enabled joins the server to a session directory

Session Directory Server—The name of the server hosting the Session Directory

Session Directory Cluster Name—The name of the cluster

Terminal server IP address redirection—Enabled for IP address redirection; disabled for token-based redirection

Command Line Reference

The following list highlights command-line settings for terminal server management, load balancing, and optimization.

QUERY USER [username | sessionname | sessionid] [/SERVER:servername]

username	Identifies the username
sessionname	Identifies the session named sessionname
sessionid	Identifies the session with ID sessionid
/SERVER:servername	The server to be enumerated

Example:

```
QUSER /server:TermServ01
```

would enumerate all user sessions on the terminal server named TermServ01

SHADOW {sessionname | sessionid} [/SERVER:servername]

sessionname	Identifies the session by name
sessionid	Identifies the session by ID
/SERVER:servername	The server containing the session

Example:

```
SHADOW 3 /server:TermServ01
```

would remote control session 3 on server TermServ01

LOGOFF [sessionname | sessionid] [/SERVER:servername]

sessionname	The name of the session
sessionid	The ID of the session
/SERVER:servername	The server containing the session

Example:

```
LOGOFF 3 /server:TermServ01
```

would logoff session 3 on server TermServ01

RESET [sessionname | sessionid]

sessionname	The name of the session
sessionid	The ID of the session

Example:

```
RESET 3
```

would logoff session 3

MSG {username sessionname sessionid @filename *}	
[/SERVER:servername] [/TIME:seconds] [/V] [/W] [message]	
username	Identifies the specified username
sessionname	The name of the session
sessionid	The ID of the session
@filename	Identifies a file containing a list of usernames, sessionnames, and sessionids to send the message to
*	Send message to all sessions on specified server
/SERVER:servername	Server to contact (default is current)
/TIME:seconds	Time delay to wait for receiver to acknowledge msg
/V	Display information about actions being performed
/W	Wait for response from user, useful with /V
message	Message to send. If none specified, prompts for it or reads from stdin
Example:	
MSG 3 /server:TermServ01 Hello there!	
Sends the message "Hello there!" to session 3 on server TermServ01	
WSRMC.EXE	Command Line Interface to Windows System Resource Manager

Summary

This chapter covered how to configure the settings required for a user to establish a terminal server session, and the tools used to manage the session once it is established. It also showed you how to configure and join a Session Directory to help users find orphaned sessions across multiple terminal server farms. Finally, I introduced you to WSRM as well as third-party products to help manage the allocation of resources to keep your terminal servers running at maximum capacity.

Overall, this guide provides the basic concepts and steps needed to set up and maintain a terminal server environment, making it a useful overview to systems administrators new to Terminal Services or as a handy reference guide for those who only occasionally need to interact with terminal servers.

Download Additional eBooks from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this eBook to be informative, we encourage you to download more of our industry-leading technology eBooks and video guides at Realtime Nexus. Please visit <http://nexus.realtimepublishers.com>.