realtimepublishers.com™

*The How-To Guide™ To*

# Windows Server 2003 Terminal Services

triCerat®

*Greyson Mitchem*

## Copyright Statement

# 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.

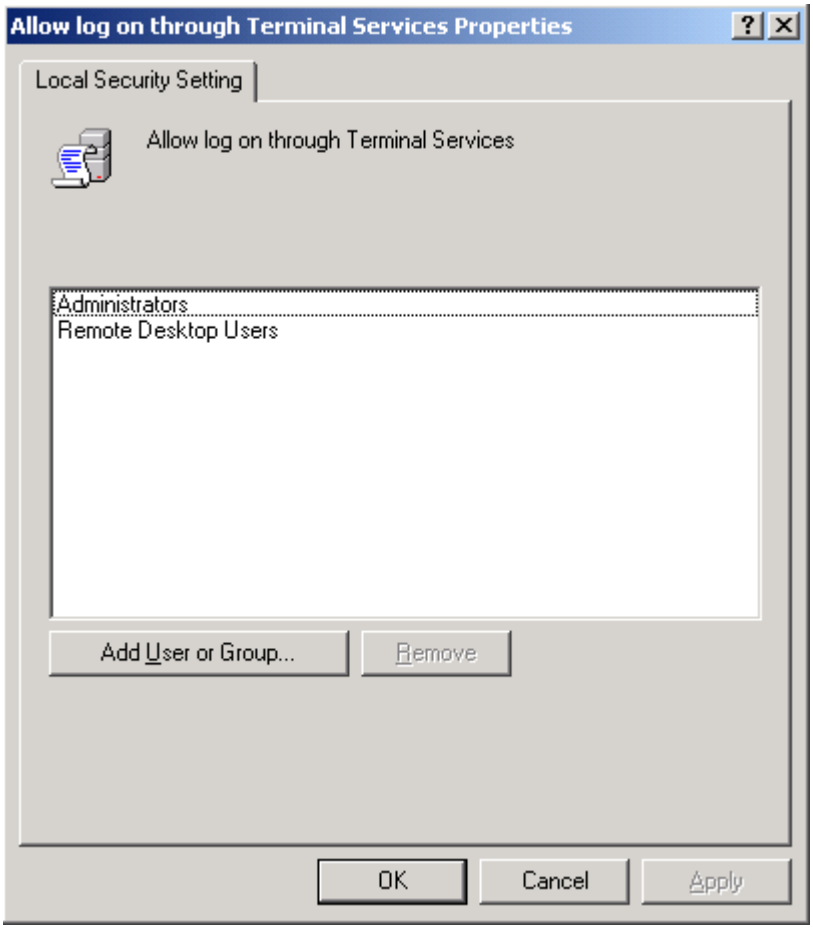


***Figure 4.1: The* Allow log on through Terminal Services *user rights assignment.***

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.

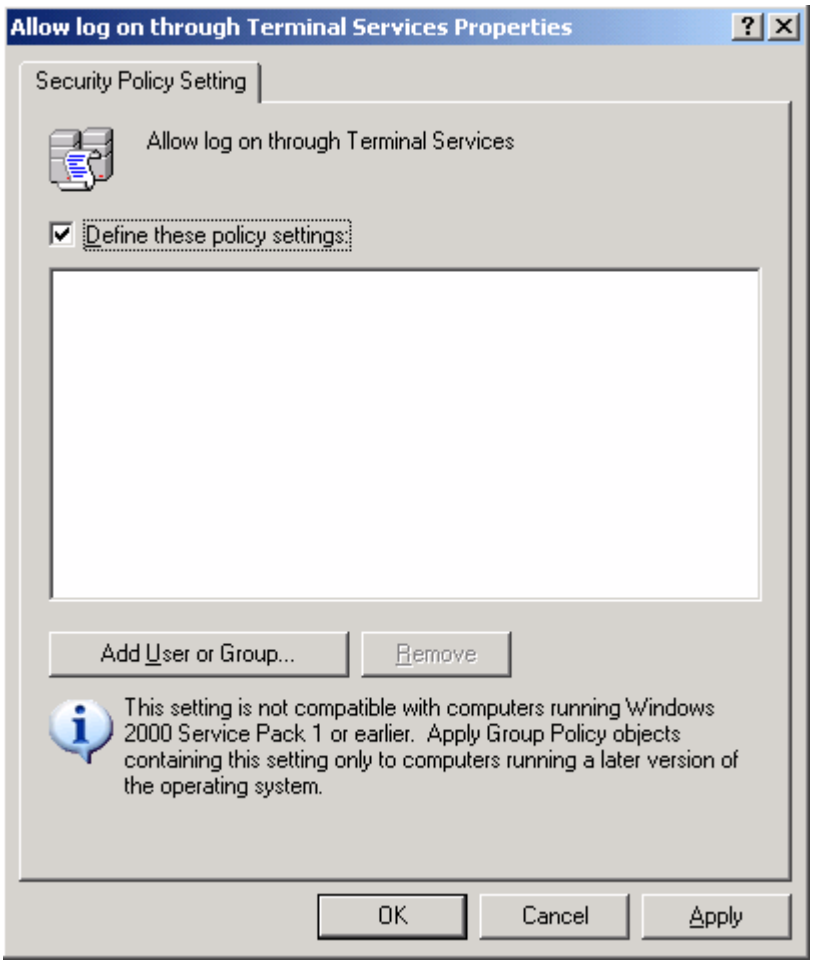


***Figure 4.2: Managing user rights assignment via Group Policy.***

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.

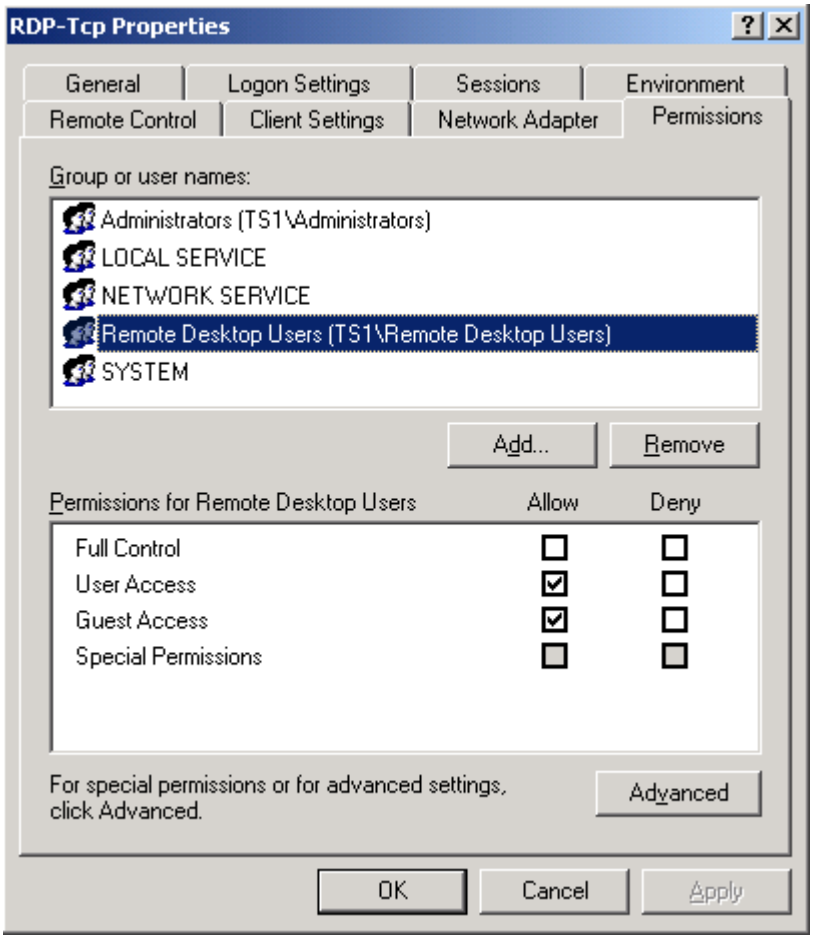


***Figure 4.3: Default permissions on the RDP protocol.***

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).

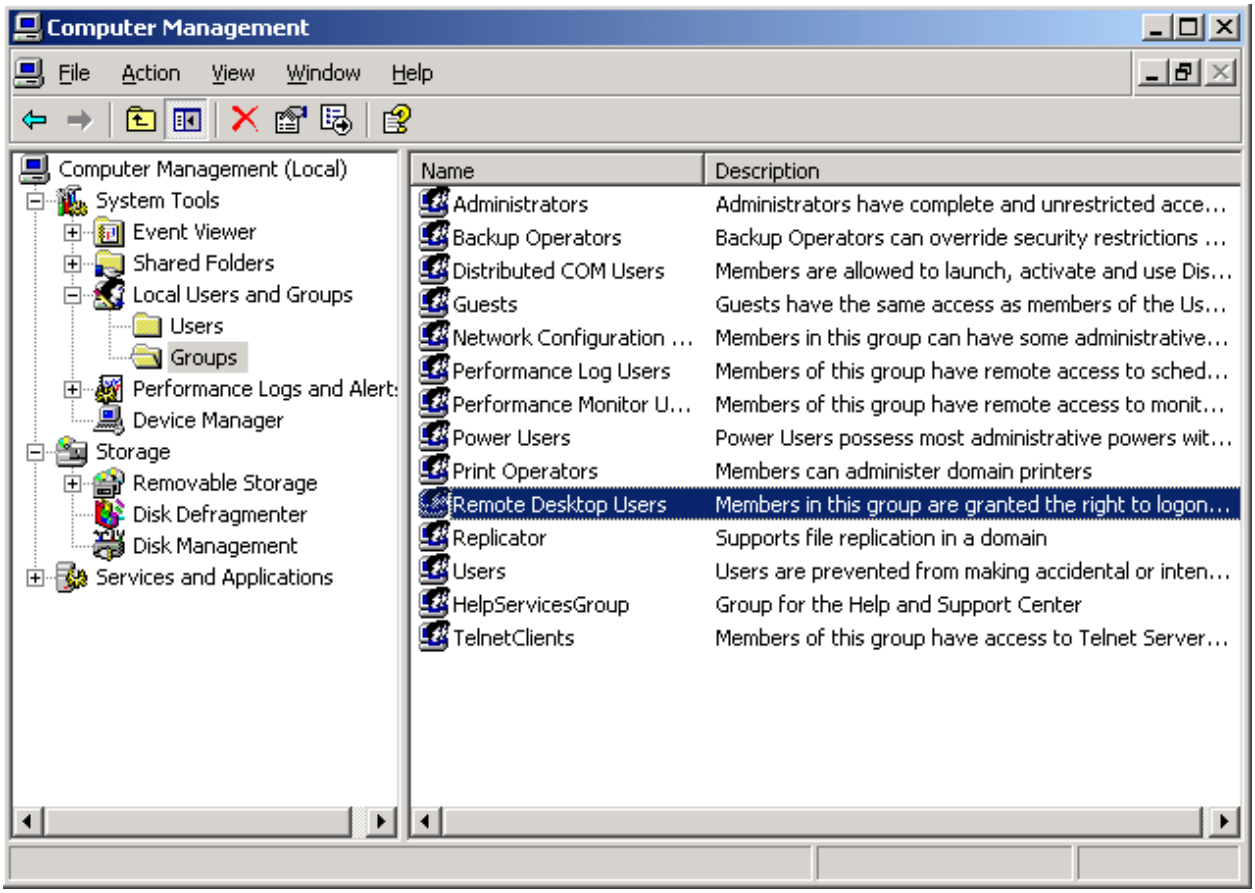


***Figure 4.4: Managing the Remote Desktop Users group in the Computer Management console.***

### *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 orJScript 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.

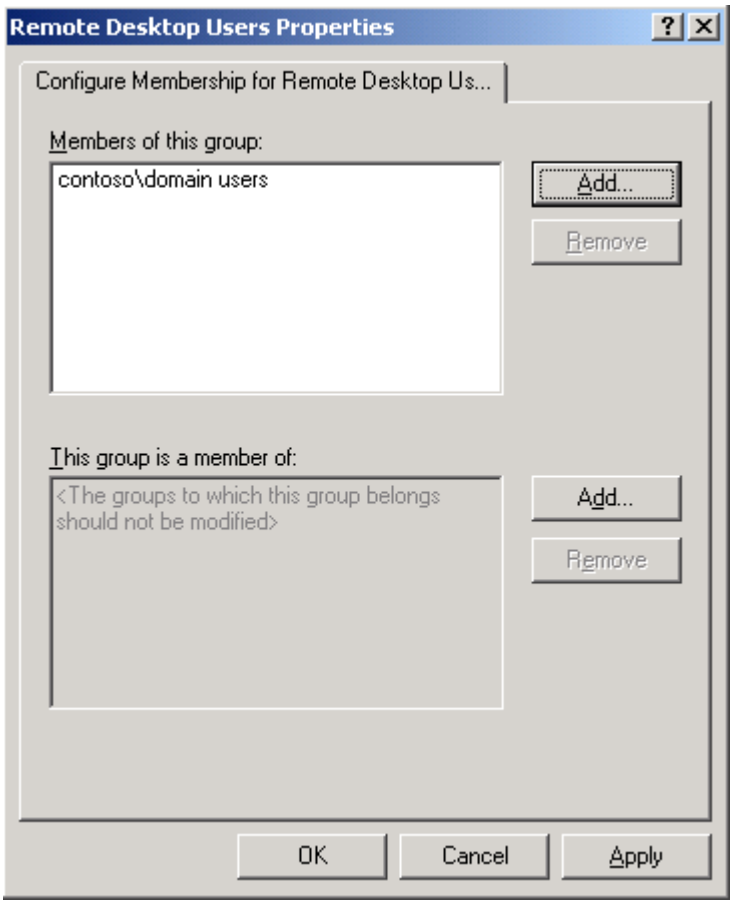


***Figure 4.5: Configuring a restricted group.***

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.

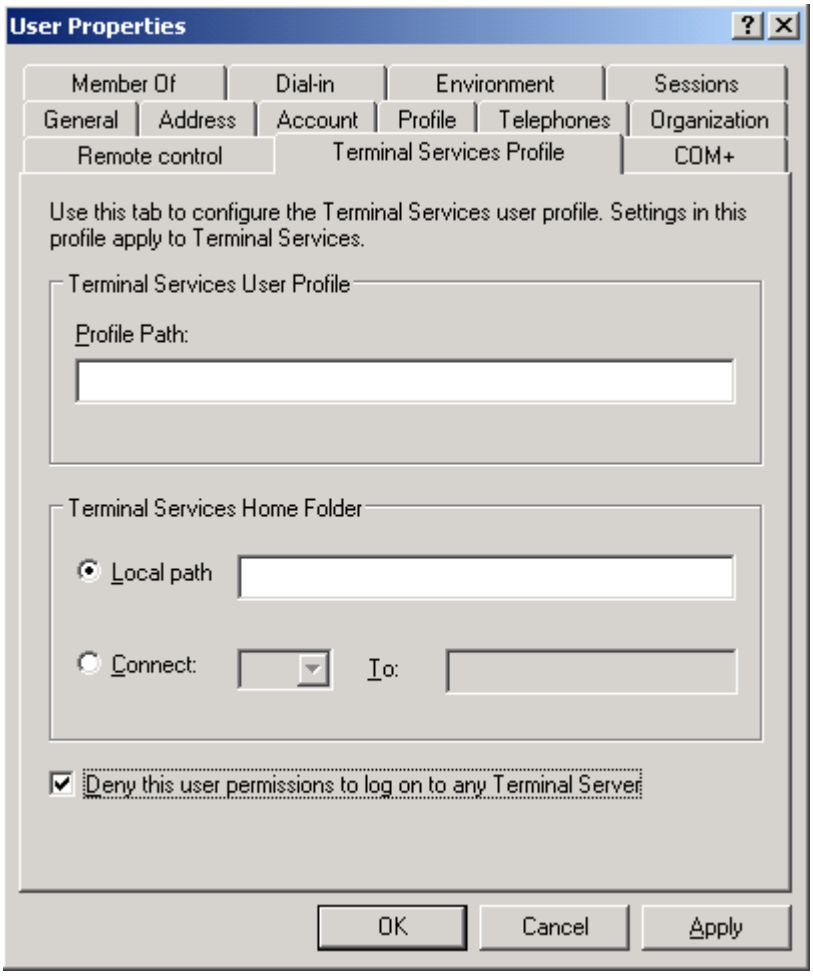


***Figure 4.6: Preventing a specific user account from logging on to any terminal server.***

### *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.

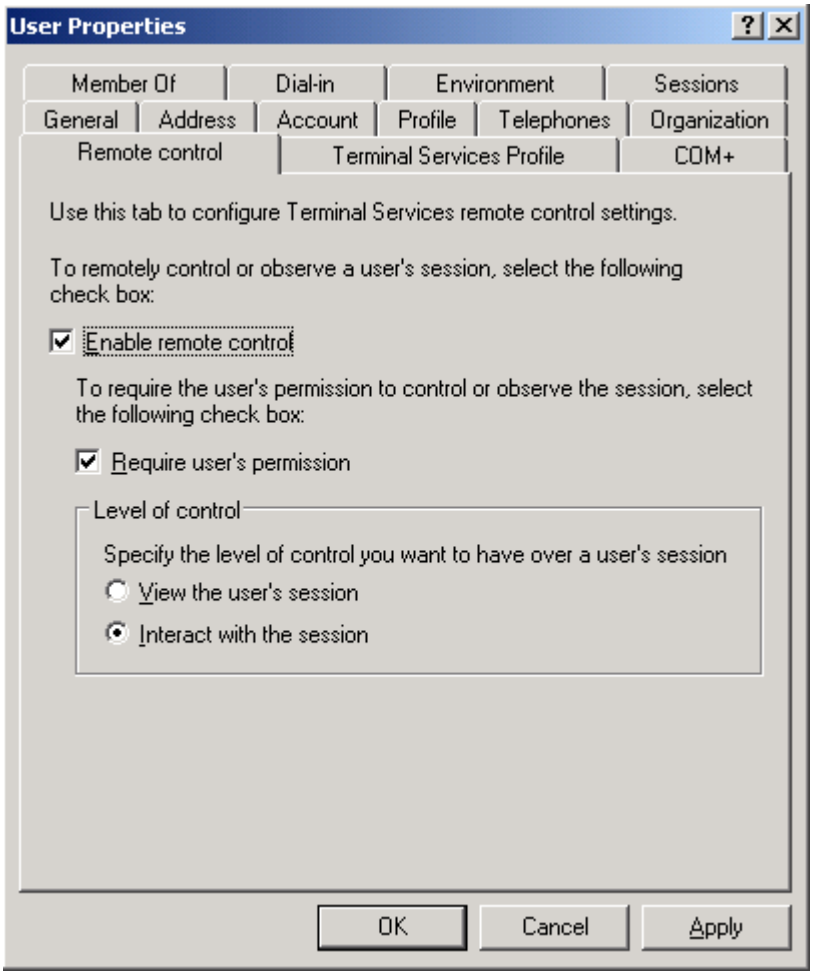


***Figure 4.7: Shadow settings on a user object.***

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.

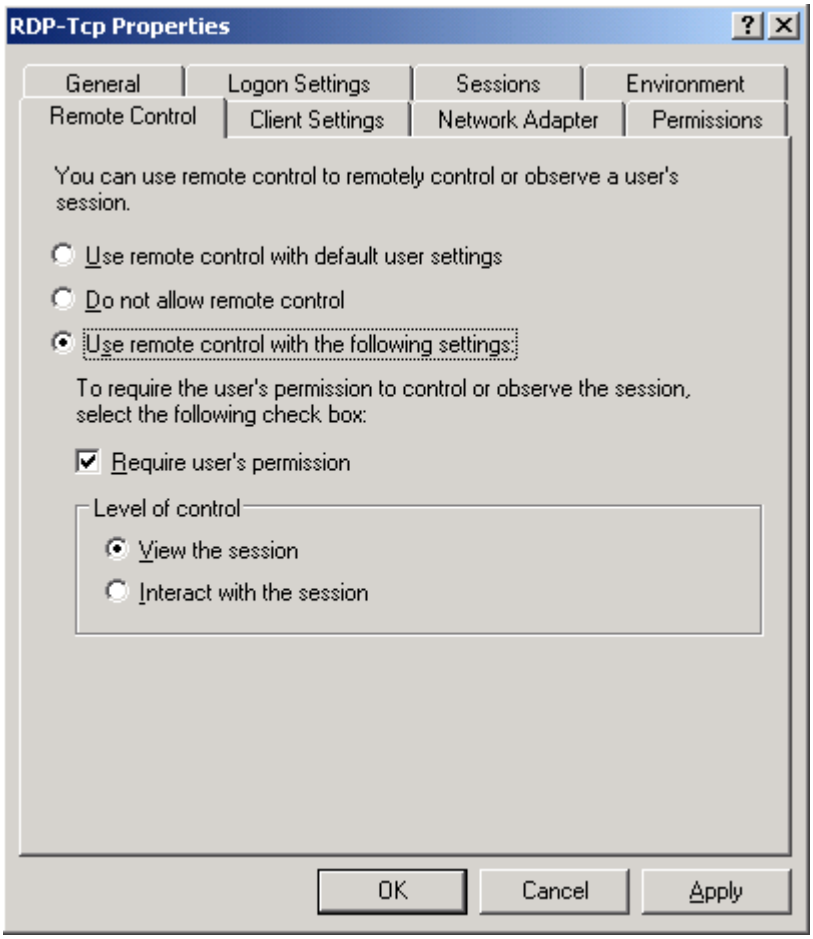


***Figure 4.8: Configuring remote control on a per-server basis.***

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.

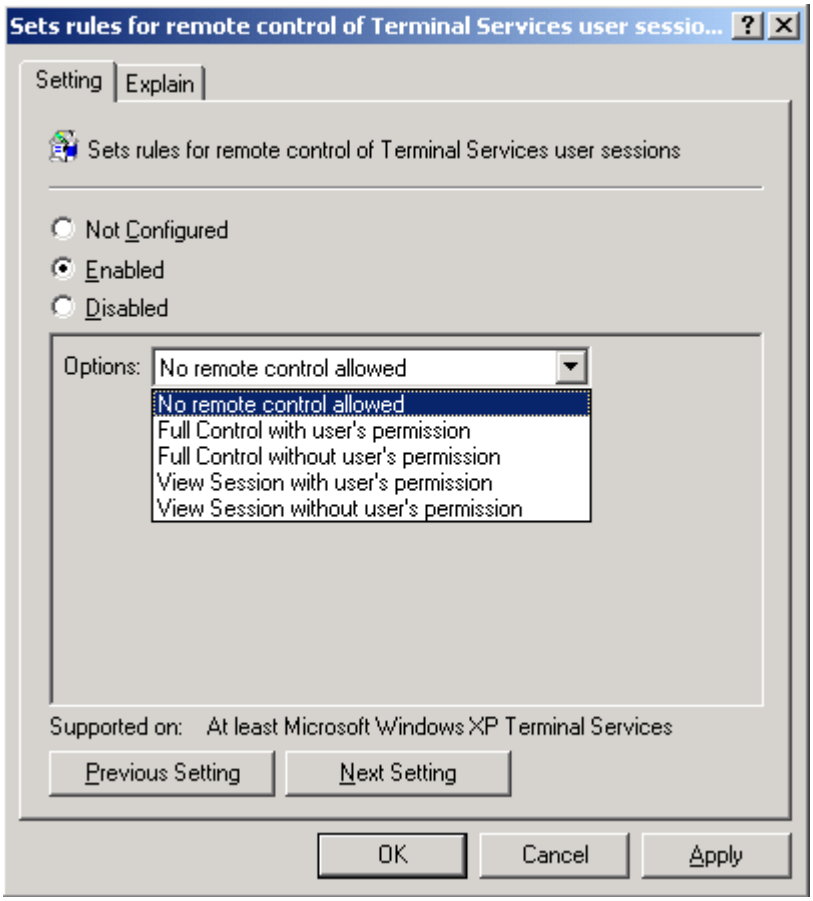


***Figure 4.9: Configuring remote control via Group Policy.***

### *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 singe 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.

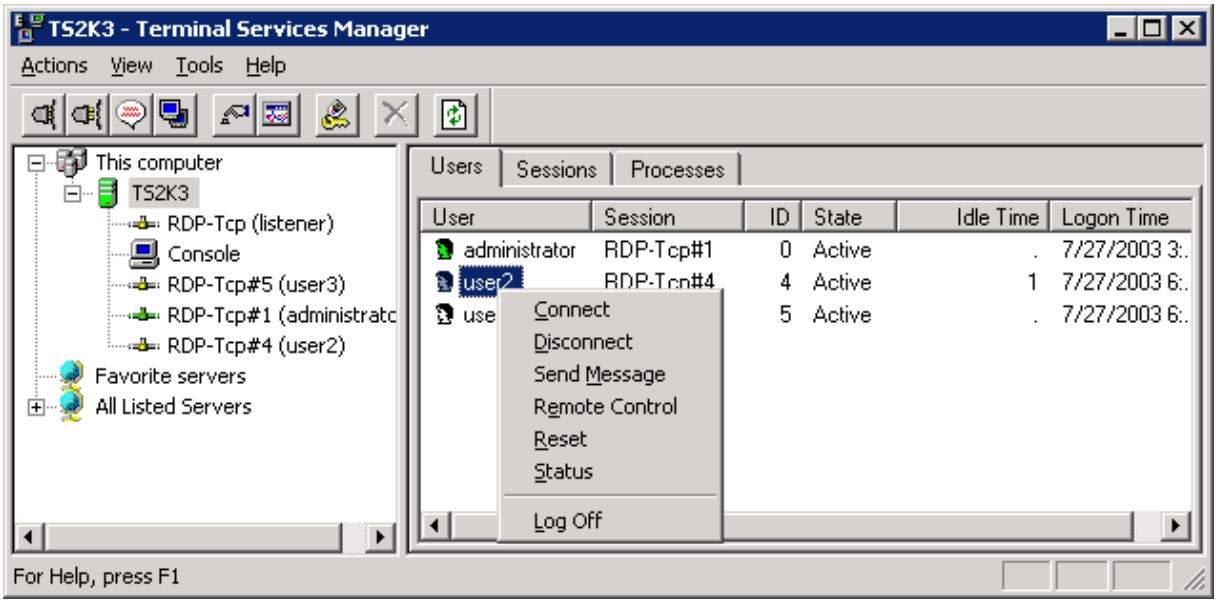


***Figure 4.10: The Terminal Services Manager administrative tool.***

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).

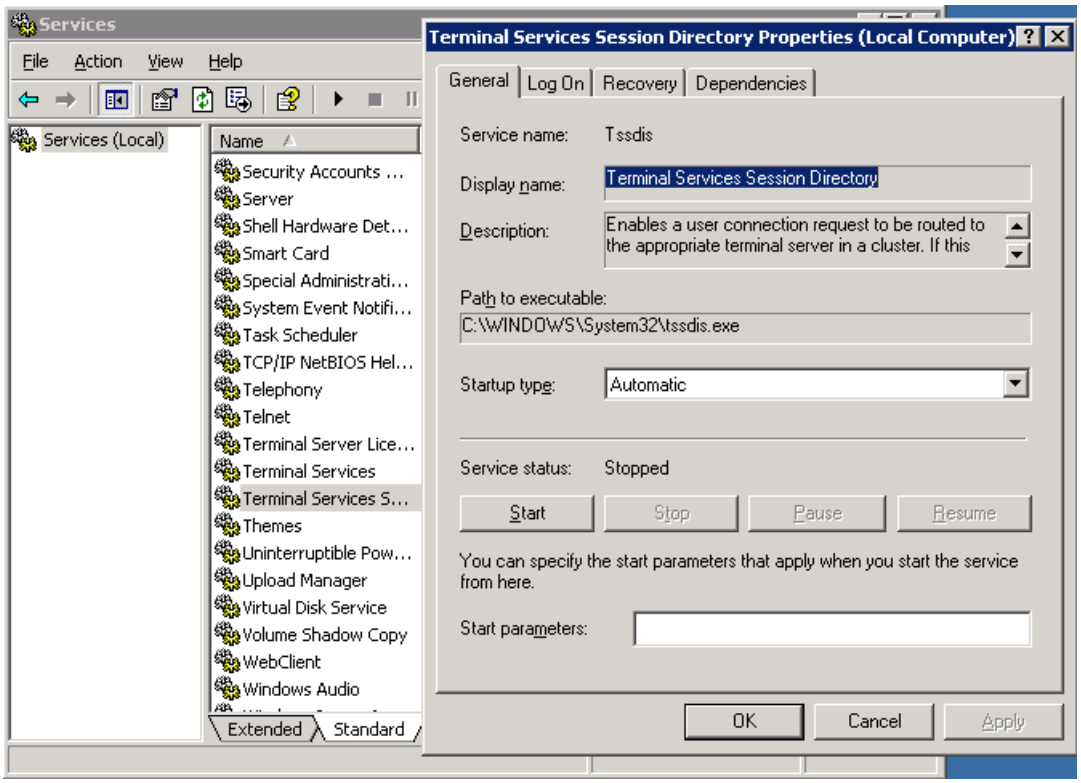


***Figure 4.11: Enabling the Terminal Services Session Directory service.***

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.

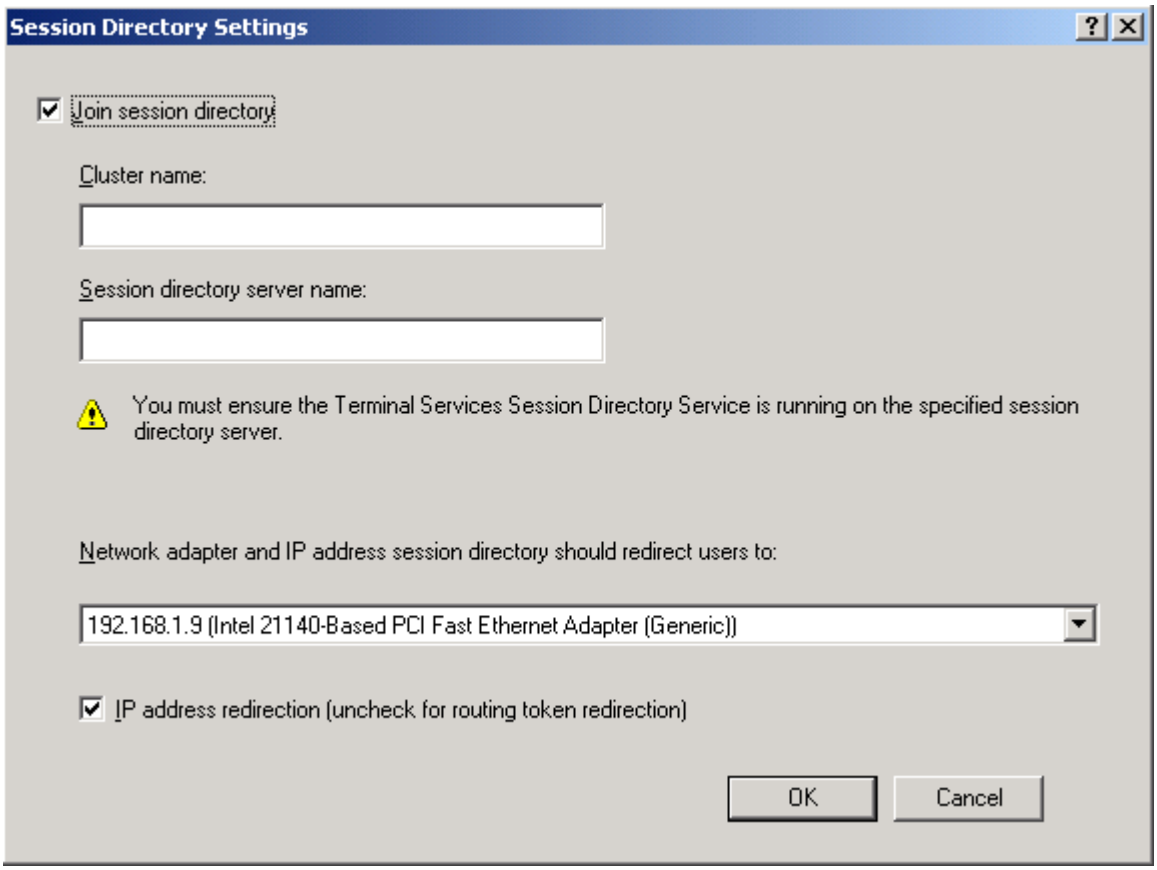


***Figure 4.12: Adding a terminal server to a Session Directory.***

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 or 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/wsrm.mspx. 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.

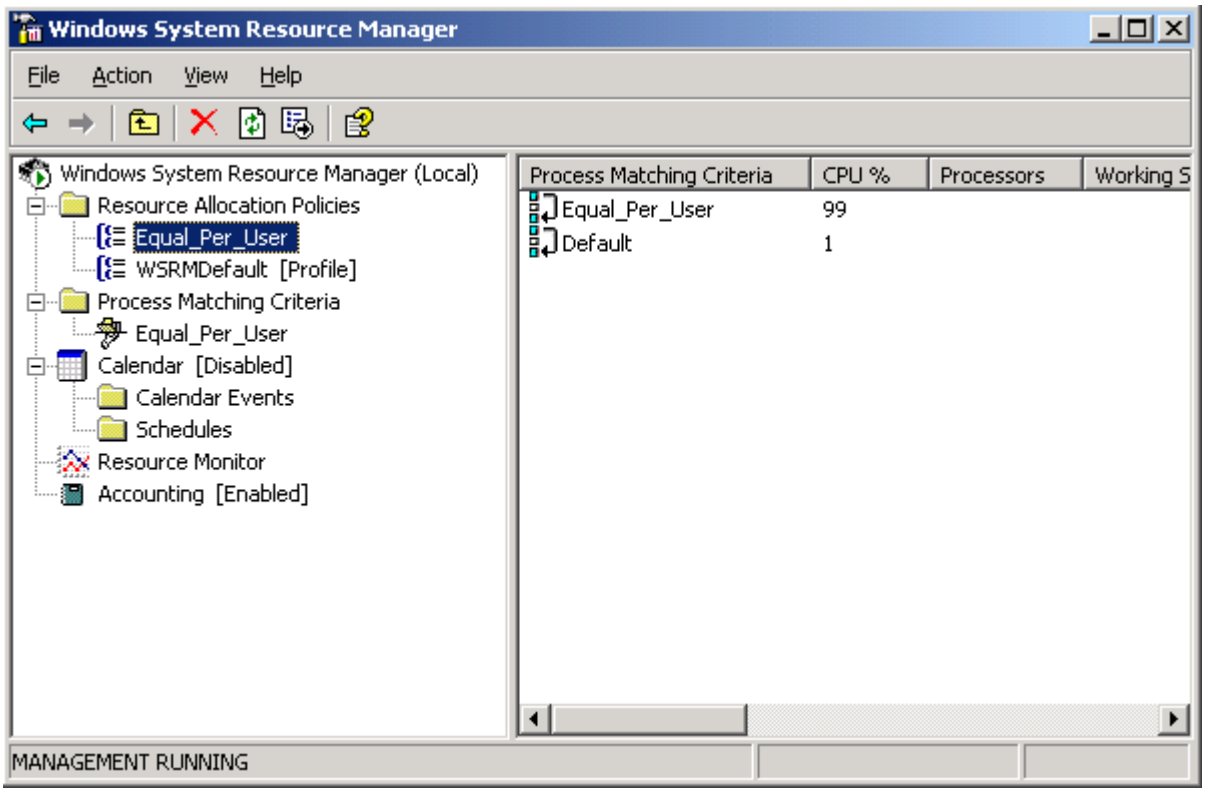


***Figure 4.13: The WSRM tool.***

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 into 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.