

# PsTools

Copyright © 1999-2004 Mark Russinovich  
Sysinternals - [www.sysinternals.com](http://www.sysinternals.com)

The Windows NT and Windows 2000 Resource Kits come with a number of command line tools that help you administer your Windows NT/2K systems. Over time, I've grown a collection of similar tools, including some not included in the Resource Kits. What sets these tools apart is that they all allow you to manage remote systems as well as the local one. The first tool in the suite was *PsList*, a tool that lets you view detailed information about processes, and the suite is continually growing. The "Ps" prefix in *PsList* relates to the fact that the standard UNIX process listing command-line tool is named "ps", so I've adopted this prefix for all the tools in order to tie them together into a suite of tools named *PsTools*.

All of the utilities in the *PsTools* suite work on Windows NT, Windows 2000, Windows XP, and Server 2003 and none of the tools requires any special installation. You don't even need to install any client software on the remote computers at which you target them. Run them by typing their name and any command-line options you want. To show complete usage information, specify the "-?" command-line option.

The tools included in the *PsTools* suite are:

[PsExec](#) - execute processes remotely

[PsFile](#) - shows files opened remotely

[PsGetSid](#) - display the SID of a computer or a user

[PsInfo](#) - list information about a system

[PsKill](#) - kill processes by name or process ID

[PsList](#) - list detailed information about processes

[PsLoggedOn](#) - see who's logged on locally and via resource sharing

[PsLogList](#) - dump event log records [PsPasswd](#) - changes account passwords

[PsService](#) - view and control services

[PsShutdown](#) - shuts down and optionally reboots a computer

[PsSuspend](#) - suspend and resume processes

## **Requirements**

Some of the tools require that the default admin\$ share be available and/or that the Remote Registry service be active.

## **License**

No part of *PsTools* may be redistributed in any way, or by any means, without the prior written permission of Sysinternals LLC. If you wish to redistribute

*PsTools*, for example for use within an organization, please contact [licensing@sysinternals.com](mailto:licensing@sysinternals.com).

## PsExec

Utilities like Telnet and remote control programs like Symantec's PC Anywhere let you execute programs on remote systems, but they can be a pain to set up and require that you install client software on the remote systems that you wish to access. *PsExec* is a light-weight telnet-replacement that lets you execute processes on other systems, complete with full interactivity for console applications, without having to manually install client software. *PsExec*'s most powerful uses include launching interactive command-prompts on remote systems and remote-enabling tools like *IpConfig* that otherwise do not have the ability to show information about remote systems.

### Installation

Copy *PsExec* onto your executable path. Typing "psexec" displays its usage syntax.

### Usage

**usage:** [\\computer[,computer[,...]] | @file][**-u** user [**-p** psswd]][**-n** s][**-s**][**-e**][**-i**][**-c** [**-f**][**-v**]][**-d**][**-w** directory][**-<priority>**][**-a** n,n,...>] **cmd**  
**[arguments]**

<b>computer</b>	Direct PsExec to run the application on the computer or computers specified. If you omit the computer name PsExec runs the application on the local system and if you enter a computer name of \\* then PsExec executes the commands on all computers in the current domain.
<b>@file</b>	PsExec will execute the command on each of the computers listed in the file.
<b>-u</b>	Specifies optional user name for login to remote computer.
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>-s</b>	Run remote process in the System account.
<b>-e</b>	Loads the specified account's profile.
<b>-i</b>	Run the program so that it interacts with the desktop on the remote system.
<b>-c</b>	Copy the specified program to the remote system for execution. If you omit this option then the application must be in the system's path on the remote system.
<b>-n</b>	Specifies timeout in seconds connecting to remote computers.
<b>-f</b>	Force the copy of the specified program if it already exists on the remote system.
<b>-v</b>	Copy the specified file only if it has a higher version number or is newer on than the one on the remote system.

<b>-d</b>	Don't wait for application to terminate. Only use this option for non-interactive applications.
<b>-w</b>	Set the working directory of the process (relative to the remote computer).
<b>-priority</b>	Specifies -low, -belownormal, -abovenormal, -high or -realtime to run the process at a different priority.
<b>-a</b>	Separate processors on which the application can run with commas where 1 is the lowest numbered CPU. For example, to run the application on CPU 2 and CPU 4, enter: "-a 2,4"
<b>arguments</b>	Arguments to pass (note that file paths must be absolute paths on the target system)

You can enclose applications that have spaces in their name with quotation marks e.g. "psexec \\marklap "c:\long name\app.exe". Put arguments directed at the application outside of the parenthesis. Input is only passed to the remote system when you press the enter key, and typing Ctrl-C terminates the remote process.

If you omit a username the remote process runs in the same account from which you execute *PsExec*, but because the remote process is impersonating it will not have access to network resources on the remote system. When you specify a username the remote process executes in the account specified, and will have access to any network resources the account has access to. Note that the password is transmitted in clear text to the remote system.

## Examples

The following command launches an interactive command prompt on \\marklap:

```
psexec \\marklap cmd
```

This command executes IpConfig on the remote system with the /all switch, and displays the resulting output locally:

```
psexec \\marklap ipconfig /all
```

This command copies the program test.exe to the remote system and executes it interactively:

```
psexec \\marklap -c test.exe
```

Specify the full path to a program that is already installed on a remote system if its not on the system's path:

```
psexec \\marklap c:\bin\test.exe
```

## PsFile

The "net file" command shows you a list of the files that other computers have opened on the system upon which you execute the command, however it truncates long path names and doesn't let you see that information for remote systems. *PsFile* is a command-line utility that shows a list of files on a system that are opened remotely, and it also allows you to close opened files either by name or by a file identifier.

### Installation

Copy *PsFile* onto your executable path and type "psfile".

### Usage

The default behavior of *PsFile* is to list the files on the local system that are open by remote systems. Typing a command followed by "-?" displays information on the syntax for the command.

**usage: psfile [\\RemoteComputer [-u Username [-p Password]]] [[Id | path] [-c]]**

<b>-u</b>	Specifies optional user name for login to remote computer.
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>Id</b>	Identifier (as assigned by PsFile) of the file for which to display information or to close.
<b>Path</b>	Full or partial path of files to match for information display or close.
<b>-c</b>	Closes the files identified by ID or path.

## PsGetSid

Have you performed a rollout and only to discover that your network might suffer from the SID duplication problem? In order to know which systems have to be assigned a new SID (using a SID updater like Sysinternals' own NewSID) you have to know what a computer's machine SID is. Up until now there's been no way to tell the machine SID without knowing Regedit tricks and exactly where to look in the Registry. *PsGetSid* makes reading a computer's SID easy, and works across the network so that you can query SIDs remotely. *PsGetSid* also lets you see the SIDs of user accounts.

### Installation

Copy *PsPsGetSid* onto your executable path and type "psgetsid".

### Usage

**Usage: psgetsid [\\computer[,computer[,...]] | @file [-u username [-p password]]] [account]**

If you want to see a computer's SID just pass the computer's name as a command-line argument. If you want to see a user's SID, name the account (e.g. "administrator") on the command-line and an optional computer name.

Specify a user name if the account you are running from doesn't have administrative privileges on the computer you want to query. If you don't specify a password as an option *PsGetSid* will prompt you for one so that you can type it in without having it echoed to the display.

## PsInfo

*PsInfo* is a command-line tool that gathers key information about the local or remote system, including the type of installation, kernel build, registered organization and owner, number of processors and their type, memory size, the install date of the system, and if it's a trial version, the expiration date. *PsInfo* command-line switches also let you view installed hotfixes and software applications.

### Installation

Copy *PsInfo* onto your executable path and type psinfo.

### Usage

By default *PsInfo* shows information for the local system. Specify a remote computer name to obtain information from the remote system. Since *PsInfo* relies on remote Registry access to obtain its data, the remote system must be running the Remote Registry service and the account from which you run *PsInfo* must have access to the HKLM\System portion of the remote Registry.

In order to aid in automated Service Pack updates, *PsInfo* returns as a value the Service Pack number of system (e.g. 0 for no service pack, 1 for SP 1, etc).

**usage: psinfo [\\computer[,computer[,...]] | @file [-u username [-p password]]] [-h] [-s] [-d] [-c [-t delimiter]]**

<b>computer</b>	Run the command on the computer or computers specified. If you omit the computer name the command runs on the local system and if you enter a computer name of \\* then the command runs on all computers in the current domain.
<b>@file</b>	Execute the command on each of the computers listed in the file.
<b>-u</b>	Specifies optional user name for login to remote computer.
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>-h</b>	Shows installed hotfixes.
<b>-s</b>	Shows installed software.
<b>-d</b>	Show disk volume information.
<b>-c</b>	Dump in CSV format.



**-t**

The default delimiter for the `-s` option is a comma, but can be overridden with the specified character.

## PsKill

Windows NT/2000 does not come with a command-line 'kill' utility. You can get one in the Windows NT or Win2K Resource Kit, but the kit's utility can only terminate processes on the local computer. *PsKill* is a kill utility that not only does what the Resource Kit's version does, but can also kill processes on remote systems. You don't even have to install a client on the target computer to use *PsKill* to terminate a remote process.

### Installation

Copy *PsKill* onto your executable path and type pskill with command-line options defined below.

### Usage

Running *PsKill* with a process ID directs it to kill the process of that ID on the local computer. If you specify a process name *PsKill* will kill all processes that have that name.

**usage: pskill [-t] [\\computer [-u username] [-p password]] <process name | process id>**

<b>-t</b>	Kill the process and its descendants.
<b>-u</b>	Specifies optional user name for login to remote computer.
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>process id</b>	Specifies the process ID of the process you want to kill.
<b>process name</b>	Specifies the process name of the process or processes you want to kill.

## PsList

Most UNIX operating systems ship with a command-line tool called "ps" (or something equivalent) that administrators use to view detailed information about process CPU and memory usage. Windows NT/2K comes with no such tool natively, but you can obtain similar tools with the Windows NT Workstation or Server Resource Kits. The tools in the Resource Kits, pstat and pmon, show you different types of information, and will only display data regarding the processes on the system on which you run the tools.

*PsList* is utility that shows you a combination of the information obtainable individually with pmon and pstat. You can view process CPU and memory information, or thread statistics. What makes *PsList* more powerful than the Resource Kit tools is that you can view process and thread statistics on a remote computer.

### Installation

Copy *PsList* onto your executable path and type "pslist".

### Usage

The default behavior of *PsList* is to show CPU-oriented information for all the processes that are currently running on the local system. The information listed for each process includes the time the process has executed, the amount of time the process has executed in kernel and user modes, and the amount of physical memory that the OS has assigned the process. Command-line switches allow you to view memory-oriented process information, thread statistics, or all three types of data.

**usage: pslist [-?] [-d] [-m] [-x][-t][-s [n] [-r n] [\computer [-u username] [-p password]]] [[-e] name | pid]**

- d This switch has *PsList* show statistics for all active threads on the system, grouping threads with their owning process.
- m This switch has *PsList* show memory-oriented information for each process, rather than the default of CPU-oriented information.
- x With this switch *PsList* shows CPU, memory and thread information for each of

the processes specified.

**-t** Show process tree.

**-s [n]** Run in task-manager mode, for optional seconds specified.  
Press Escape to abort.

**-r n** Task-manager mode refresh rate in seconds (default is 1).

**-u** Specifies optional user name for login to remote computer.

**-p** Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.

Instead of listing all the running processes in the system, this parameter narrows *PsList's* scan to those processes that begin with the name process. Thus:

**name**                    **pslist exp**

would statistics for all the processes that start with "exp", which would include Explorer.

**-e** Use the -e switch if you want the process name to be treated as an exact match instead of a partial match.

Instead of listing all the running processes in the system, this parameter narrows *PsList's* scan to the process that has the specified PID. Thus:

**pid**                    **pslist 53**

would dump statistics for the process with the PID 53.

## PsLoggedOn

You can determine who is using resources on your local computer with the "net" command ("net session"), however, there is no built-in way to determine who is using the resources of a remote computer. In addition, NT comes with no tools to see who is logged onto a computer, either locally or remotely. *PsLoggedOn* is an applet that displays both the locally logged on users and users logged on via resources for either the local computer, or a remote one. If you specify a user name instead of a computer, *PsLoggedOn* searches the computers in the network neighborhood and tells you if the user is currently logged on.

*PsLoggedOn*'s definition of a locally logged on user is one that has their profile loaded into the Registry, so *PsLoggedOn* determines who is logged on by scanning the keys under the HKEY\_USERS key. For each key that has a name that is a user SID (security Identifier), *PsLoggedOn* looks up the corresponding user name and displays it. To determine who is logged onto a computer via resource shares, *PsLoggedOn* uses the NetSessionEnum API. Note that *PsLoggedOn* will show you as logged on via resource share to remote computers that you query because a logon is required for *PsLoggedOn* to access the Registry of a remote system.

### Installation

Copy *PsLoggedOn* onto your executable path and type "psloggedon".

### Usage

**usage: psloggedon [-?] [-l] [-x] [\\computername | username]**

<b>-l</b>	Shows only local logons instead of both local and network resource logons.
<b>-x</b>	Don't show logon times.
<b>\\computername</b>	Specifies the name of the computer for which to list logon information. If you specify a user name PsLoggedOn searches the network for computers to which that user is loggedon. This is useful if you want to ensure that a particular user is not logged on when you are about to change their user profile configuration.
<b>username</b>	

## PsLogList

The Resource Kit comes with a utility, *elogdump*, that lets you dump the contents of an Event Log on the local or a remote computer. *PsLogList* is a clone of *elogdump* except that *PsLogList* lets you login to remote systems in situations your current set of security credentials would not permit access to the Event Log, and *PsLogList* retrieves message strings from the computer on which the event log you view resides.

### Installation

Copy *PsLogList* onto your executable path and type "psloglist".

### Usage

The default behavior of *PsLogList* is to show the contents of the System Event Log on the local computer, with visually-friendly formatting of Event Log records. Command line options let you view logs on different computers, use a different account to view a log, or to have the output formatted in a string-search friendly way.

**usage: psloglist [-?] [\computer[,computer[,...]] | @file [-u username [-p password]]] [-s [-t delimiter]] [-m #|-n #|-h #|-d #|-w][[-c][[-x][[-r][[-a mm/dd/yy][[-b mm/dd/yy][[-f filter] [-i ID[,ID[,...]] | -e ID[,ID[,...]]] [-o event source[,event source][,..]]] [-q event source[,event source][,..]]] [-l event log file] <eventlog>**

- |              |  |
|--------------|--|
| <b>@file</b> | Execute the command on each of the computers listed in the file.   |
| <b>-a</b>    | Dump records timestamped after specified date.   |
| <b>-b</b>    | Dump records timestamped before specified date.  |
| <b>-c</b>    | Clear the event log after displaying.  |
| <b>-d</b>    | Only display records from previous n days.   |
| <b>-e</b>    | Exclude events with the specified ID or IDs (up to 10).  |
| <b>-f</b>    | Filter event types with filter string (e.g. "-f w" to filter warnings).                                      |
| <b>-h</b>    | Only display records from previous n hours.  |
| <b>-i</b>    | Show only events with the specified ID or IDs (up to 10).  |
| <b>-l</b>    | Dump records from the specified event log file.  |
| <b>-m</b>    | Only display records from previous n minutes.  |
| <b>-n</b>    | Only display the number of most recent entries specified.  |
| <b>-o</b>    | Show only records from the specified event source (e.g. "-o cdrom").   |
| <b>-p</b>    | Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password. |

- q** Omit records from the specified event source or sources (e.g. \'-o cdrom\').
  - r** Dump log from least recent to most recent.
  - s** This switch has *PsLogList* print Event Log records one-per-line, with comma delimited fields. This format is convenient for text searches, e.g. `psloglist | findstr /i text`, and for importing the output into a spreadsheet.
  - t** The default delimiter is a comma, but can be overridden with the specified character.
  - u** Specifies optional user name for login to remote computer.
  - w** Wait for new events, dumping them as they generate.
  - x** Dump extended data.
- eventlog** By default *PsLogList* shows the contents of the System Event Log. Specify a different event log by typing in the first few letters of the log name, application, system, or security.

## PsPasswd

Systems administrators that manage local administrative accounts on multiple computers regularly need to change the account password as part of standard security practices. PsPasswd is a tool that lets you change an account password on the local or remote systems, enabling administrators to create batch files that run PsPasswd against the computer's they manage in order to perform a mass change of the administrator password.

### Installation

Copy *PsPasswd* onto your executable path and type pspasswd with command-line options defined below.

### Usage

You can use *PsPasswd* to change the password of a local or domain account on the local or a remote computer.

**usage: pspasswd [\\computer[,computer[,...]] | @file [-u username [-p password]]] Username [NewPassword]]**

<b>computer</b>	Run the command on the computer or computers specified. If you omit the computer name the command runs on the local system and if you enter a computer name of \\* then the command runs on all computers in the current domain.
<b>@file</b>	Execute the command on each of the computers listed in the file.
<b>-u</b>	Specifies optional user name for login to remote computer.
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>Username</b>	Specifies name of account for password change.
<b>NewPassword</b>	New password. If omitted a NULL password is applied.



## PsService

*PsService* is a service viewer and controller for Windows NT/2K. Like the SC utility that's included in the Windows NT and Windows 2000 Resource Kits, *PsService* displays the status, configuration, and dependencies of a service, and allows you to start, stop, pause, resume and restart them. Unlike the SC utility, *PsService* enables you to logon to a remote system using a different account, for cases when the account from which you run it doesn't have required permissions on the remote system. *PsService* includes a unique service-search capability, which identifies active instances of a service on your network. You would use the search feature if you wanted to locate systems running DHCP servers, for instance.

Finally, *PsService* works on both NT 4 and Windows 2000, whereas the Windows 2000 Resource Kit version of SC requires Windows 2000, and *PsService* doesn't require you to manually enter a "resume index" in order to obtain a complete listing of service information.

### Installation

Copy *PsService* onto your executable path and type "pservice".

### Usage

The default behavior of *PsService* is to display the configured services (both running and stopped) on the local system. Entering a command on the command-line invokes a particular feature, and some commands accept options. Typing a command followed by "-?" displays information on the syntax for the command.

**usage: pservice [\computer [-u username] [-p password]]**

**<command> <options>**

- |                  |  |
|------------------|--|
| <b>-u</b>        | Specifies optional user name for login to remote computer.   |
| <b>-p</b>        | Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password. |
| <b>query</b>     | Displays the status of a service   |
| <b>config</b>    | Displays the configuration of a service  |
| <b>setconfig</b> | Specify the start type (auto, demand, disabled) of a service.  |

<b>start</b>	Starts a service
<b>stop</b>	Stops a service
<b>restart</b>	Stops and then restarts a service
<b>pause</b>	Pauses a service
<b>cont</b>	Resumes a paused service
<b>depend</b>	Lists the services dependent on the one specified
<b>find</b>	Searches the network for the specified service

## PsShutdown

*PsShutdown* is similar to the Resource Kit and Windows XP shutdown tools, providing you the same options and ability to shutdown, and optionally reboot, local and remote Windows NT/2K/XP/2003 systems. It also provided additional options that make it more powerful and flexible.

### Installation

Copy *PsShutdown* onto your executable path and type "psshutdown" with command-line options defined below.

### Usage

You can use *PsShutdown* to initiate a shutdown of the local or a remote computer, abort an imminent shutdown, logoff a console user, or lock the desktop.

**usage: psshutdown [\\computer[,computer[,...]] | @file [-u username [-p password]]] [-s|-r|-h|-d|-k|-a|-l|-o [-f] [-c] [-n s] [-t nn|h:m] [-e [u|p]:xx:yy] [-m "message"]**

<b>computer</b>	Run the command on the computer or computers specified. If you omit the computer name the command runs on the local system and if you enter a computer name of \\* then the command runs on all computers in the current domain.
<b>@file</b>	Execute the command on each of the computers listed in the file.
<b>-u</b>	Specifies optional user name for login to remote computer
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>-a</b>	Aborts a shutdown (only possible while a countdown is in progress)
<b>-c</b>	Allow the shutdown to be aborted by the interactive user
<b>-e</b>	Shutdown reason code. Specify 'u' for user reason codes and 'p' for planned shutdown reason codes. xx is the major reason code (must be less than 256) yy is the minor reason code (must be less than 65536)
<b>-f</b>	Forces all running applications to exit during the shutdown instead of giving them a chance to gracefully save their data
<b>-h</b>	Hibernate the computer
<b>-k</b>	Poweroff the computer (reboot if poweroff is not supported)
<b>-l</b>	Lock the computer
<b>-m</b>	This option lets you specify a message to display to logged-on users when a shutdown countdown commences
<b>-n</b>	Specifies timeout in seconds connecting to remote computers
<b>-o</b>	Logoff the console user

- r** Reboot after shutdown
- s** Shutdown without poweroff
- t** Specifies the countdown in seconds until the shutdown (default: 20 seconds) or the time of shutdown in 24 hour notation
- v** Display message for the specified number of seconds before the shutdown. If you omit this parameter the shutdown notification dialog displays and specifying a value of 0 omits the dialog.

## PsSuspend

*PsSuspend* lets you suspend processes on the local or a remote system, which is desirable in cases where a process is consuming a resource (e.g. network, CPU or disk) that you want to allow different processes to use. Rather than kill the process that's consuming the resource, suspending permits you to let it continue operation at some later point in time.

### Installation

Copy *PsSuspend* onto your executable path and type "pssuspend" with command-line options defined below.

### Usage

Running *PsSuspend* with a process ID directs it to suspend the process of that ID on the local computer. If you specify a process name *PsSuspend* will suspend all processes that have that name. Specify the `-r` switch to resume suspended processes.

**usage: pssuspend [-r] [\\computer [-u username] [-p password]]  
<process name | process id>**

<b>-r</b>	Resumes suspended processes.
<b>-u</b>	Specifies optional user name for login to remote computer.
<b>-p</b>	Specifies optional password for user name. If you omit this you will be prompted to enter a hidden password.
<b>process id</b>	Specifies the process ID of the process you want to suspend.
<b>process name</b>	Specifies the process name of the process or processes you want to suspend.