Ten More Useful NT Resource Kit Utilities

We've often mentioned what a crucial part of NT the Resource Kit is, even though it's supplied as a separate product. Previously, we described our 10 favourite RK utilities and their uses; here are 10 more you won't want to be without.

By Simon Pride

In the previous article [*Ten Useful NT Resource Kit Utilities, PCNA 116, File E1716*] I introduced around 10 of the Resource Kit utilities I reach for on an almost daily basis. The Windows NT Resource Kit comprises a set of documents describing the workings of NT in detail, and utilities that provide the missing commands and tools that system administrators coming from other OS backgrounds are used to having.

The Resource Kit is available in two versions, one for NT Workstation and one for NT Server; however, the Workstation Resource Kit is a subset of the Server Resource Kit, and the serious system administrator will want to use the latter.

Batch File Commands

First, let's look at some more commands which are most useful when used in scripts (batch files).

TIMEOUT And SLEEP

These are two time-related commands used in scripts. TIMEOUT is a grown-up version of MS-DOS's PAUSE command. In one mode it will do exactly what PAUSE does, waiting for the user to press a key. More usefully, it can take a parameter which tells it for how long to pause and invite user input before continuing with the script, unless a key on the keyboard is pressed in the interim.

TIMEOUT takes a single parameter, an integer which specifies the number of seconds the command will wait before resuming operations. The maximum time is 100,000 seconds, which is just under 28 hours. To use TIMEOUT like PAUSE - that is, to wait indefinitely until the user presses a key - specify an interval of -1 on the command line. Figure 1 shows a simple evocation of TIMEOUT from the NT command prompt and its user interface whilst running.

SLEEP is more or less an exact equivalent of the Unix sleep(1) command, suspending execution of the script from which it is called for the number of seconds passed to it as an integer on the command line.

MUNGE

MUNGE is a utility for altering the contents of text files. It is most often called from a script - if it was appropriate to edit the file manually then that is what you would be doing, using a good text editor to help you. MUNGE is for those situations when a file needs to be altered and a human can't be there to do the necessary work.

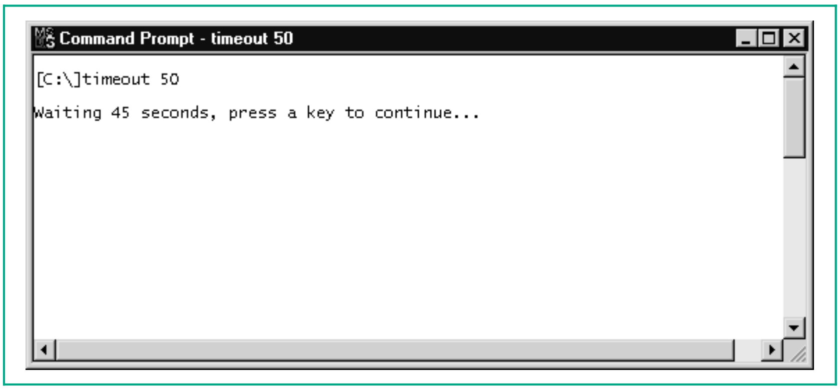
Unix users will be familiar with using sed, awk or the Perl language to do this in their environment, and although Perl is readily available on Windows NT the other two utilities are not. MUNGE more or less fills the gap left by their absence.

A simple use of MUNGE uses a single script file and a command-line filter. Imagine you have a large number of Web pages that reference each other using the extension ".HTM". You wish to move your files to a more compliant Web server and, in the process, convert the file names to end in the more canonical "HTML".

Renaming the files is trivial, but updating all the links in a set of tightly cross-referenced Web pages could take days. MUNGE can do this immediately. To do this simply create a small text file with a single line in it:

```
HTM HTML
```

Save the file as names-change.script (remember we're on NT now, we can



```
[C:\]timeout 50

Waiting 45 seconds, press a key to continue...
```

Figure 1 - Timeout counting down from 50 seconds.

forget about 8.3 limitations!) in the same directory as the files to change. Now issue the following command in that same directory:

```
MUNGE names-change.script *.HTM
```

MUNGE will run through every HTM file in the current directory (and those below, if you specify the -r command) and change each occurrence of HTM in the file to HTML.

RMTSHARE

How many times have you wanted to get to a particular directory on a remote server, but been frustrated because that directory is not shared or a subdirectory of an existing share? RMTSHARE lets you specify a directory on a remote computer as a share and even lets you assign the standard share controls to that share. Its simplest use is:

```
RMTSHARE \\SERVER\SHARENAME=<Drive>:-
\<Path>
```

So to share D:\My Documents\Public on the server \\TURING as Public use:

```
RMTSHARE \\TURING\Public="D:\My Docu-
ments\Public"
```

To set user permissions use the “grant” keyword. The syntax is of the form:

```
RMTSHARE \\SERVER\SHARENAME /grant
<user>:<permission>
```

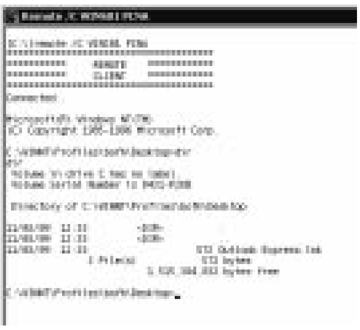
RMTSHARE uses its own set of one-character symbols to represent share permissions, being the initial letters of read, change, full control and none (No Access). To give read-only permission to the share Enigma on TURING to user account HilbertD, use:

```
RMTSHARE \\TURING\ENIGMA /grant Hilbe-
rtD:r
```

To share a printer, the syntax used is similar to that used to share a part of the file system, except that no drive letter or path is used, and the switch /printer indicates that the printer namespace should be used. To share the printer named Optra SC 1275 as FastColour on the server TURING use:

```
RMTSHARE “\\TURING\Optra SC 1275"=-
FastColour /printer
```

To review the shares on a remote server, RMTSHARE with just a server name as an argument lists all the shares on the server, in the same way as NET VIEW \\SERVERNAME will.



```
Remote /C WINS01 PCNA

C:\>remote /C WINS01 PCNA
**************************************
**********     REMOTE     ************
**********     CLIENT     ************
**************************************
Connected...

Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\WINNT\Profiles\[illegible]\Desktop>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is [illegible]

 Directory of C:\WINNT\Profiles\[illegible]\Desktop

[illegible]  [illegible]        <DIR>
[illegible]  [illegible]        <DIR>
[illegible]  [illegible]              [illegible] Outlook Express.lnk
               1 File(s)          [illegible] bytes
                          [illegible] bytes free

C:\WINNT\Profiles\[illegible]\Desktop>_
```

Figure 2 - REMOTE running as client.

```
[C:\NTReskit]rsetup \\WINS01
RSETUP 2.03 @1996-98. Written by Christophe Robert - Microsoft.
Connecting to registry of \\WINS01 ...
Checking existence of service RCONSVC ...
Copying file RCLIENT.EXE ...
Copying file RCONMODE.EXE ...
Copying file RCONMSG.DLL ...
Copying file RCONSTAT.EXE ...
Copying file RCONSVC.EXE ...
Copying file RCRUNCMD.EXE ...
Copying file RSETUP.EXE ...
Opening Service Control Manager ...
Installing Remote Console Service ...
Registering Remote Console service event sources ...
Getting domain information ...
Finding PDC for domain CSI-TUS ...
Found PDC \\WINS01 ...
Adding local group RConsole Users on \\WINS01 ...
Setting privilege “Log on as a batch job” ...
Remote Console has been successfully installed on \\WINS01.
Starting service RCONSVC on \\WINS01 .... started.
```

Figure 3 - RSETUP in action.

Remote Access Solutions

The Resource Kit has several solutions for remote administration of NT servers, some of which overlap in functionality.

REMOTE

REMOTE has been in the Resource Kit since the days of NT 3.1, and provides a bare bones command line on another computer. It works in a client-server fashion, the same binary doing duty as both server and client depending on which parameter it was started with.

REMOTE /S <Command> <id-tag> starts a server on one computer, and REMOTE /C <computername> <id-tag> connects to whatever <Command> spawned on the first computer. The value of <Command> is usually a shell, such as CMD.EXE. In this example I went to the computer WINS01 and typed:

```
REMOTE /S CMD.EXE PCNA
```

in order to start the remote server running an instance of CMD.EXE with the ID tag PCNA. I then went to a workstation and typed:

```
REMOTE /C WINS01 PCNA
```

which connected me to the session I had started on WINS01. This gave me a console window onto WINS01 (see Figure 2). To quit either a server or a client session, use Control-Break.

One nice thing about REMOTE is that the session on the server is preserved no matter how many times the client disconnects and reconnects. However, REMOTE is rather primitive, in that it simply redirects Stand-

NT Resource Kit

ard Input (STDIN) and Standard Output (STDOUT) from the command session spawned to the client running across the network connection. If the process being run (or started during the session) takes control of the console or the screen (such as the EDIT command), then REMOTE loses control of the session. To run such applications another kind of remote control service is needed, such as RCMD.

RCMDSVC

RCMD is REMOTE's big brother. It has few of REMOTE's drawbacks and can run full screen sessions. It comes in two parts, RCMDSCV, an NT service, and RCLIENT, which provides the local console. Install it by using RSETUP from the Resource Kit, specifying a computer to install on (see Figure 3).

To connect to a remote console use the client tool RCLIENT in the Resource Kit directory. Typing RCLIENT \\COMPUTERNAME gives you a command line on the remote computer (Figure 4) which you can use to run graphical console applications such as EDIT (Figure 5).

SU

SU is probably the most important utility in the whole of the Resource Kit. Its name is derived from the Unix utility of the same name. SU stands for Specified User, and allows one account to temporarily assume the security context of another. Why would you want to do this?

The answer is rooted in the traditions of best practice that have grown up around multi-user operating systems such as VMS and Unix, one of which is that you always use the account with the lowest privilege level to accomplish the task in hand. For ordinary users this means (in the NT context) an account that is a member of Users or Domain Users and no other.

Even systems administrators spend most of their time simply reading and responding to mail and news, reading Web pages and writing documents. None of these tasks requires any elevated levels of privilege, and therefore sysadmins should use a standard user account for their everyday work. The reason for this is that accidents will happen, and running with a highly privileged account can turn an accident with Explorer or File Manager into a disaster, followed by an eight-hour restore from backup tapes.

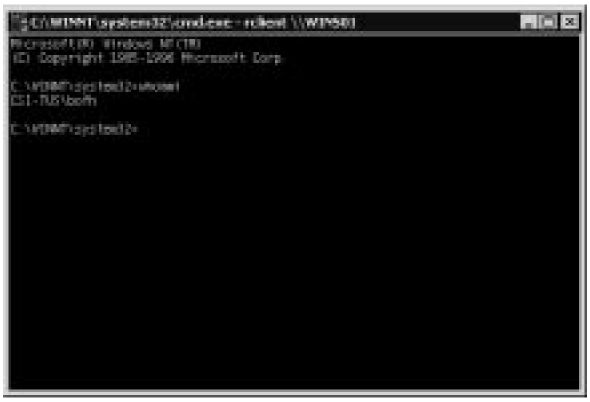
However, the sysadmin is also constantly interrupted by user requests to share a folder, add some new joiner to a workgroup, probe a printer to see why it's refusing jobs, or reset a forgotten password. The way to reconcile these conflicting pressures is to use the SU utility to temporarily assume Administrator powers.

Our colleagues in the Unix world have an easier time with this concept, since Unix generally has a command-line culture; any system administration task can be accomplished from a command shell. NT, however, is still closely wedded to the GUI, and only recently have the tools appeared (in the Resource Kit and natively) to provide a similar degree of access to the operating system and its administration. There are several remedies to hand for this situation, ranging from the spartan to the comfortable.

First, the spartan approach. This involves simply getting to work and learning NT's command-line tools, such as the many varied arguments to the NET command, and the command-line utilities in the Resource Kit. Instead of firing up User Manager for Domains to add a new user, use SU to run as Administrator and use NET GROUP to accomplish the same end. This requires the most effort, but in the end pays the greatest dividends, allowing you to accomplish in a few keywords tasks that otherwise take minutes of clicking in GUI utilities.

To use SU to do this, simply open a command prompt and type SU. This will bring up the SU parameter screen (see Figure 6) where you can fill in your credentials: the account you wish to impersonate, the domain to which the account belongs and its password. Completing these three elements and clicking OK (or more likely pressing Enter on the keyboard) will get you a new command prompt.

This prompt is usually NT's native command processor CMD.EXE, but you will get whatever has been specified by the %COMSPEC% environment variable - CMD.EXE, COMMAND.COM, 4NT (an improved shell based on the popular 4DOS) or even bash, the "Bourne-again" shell from

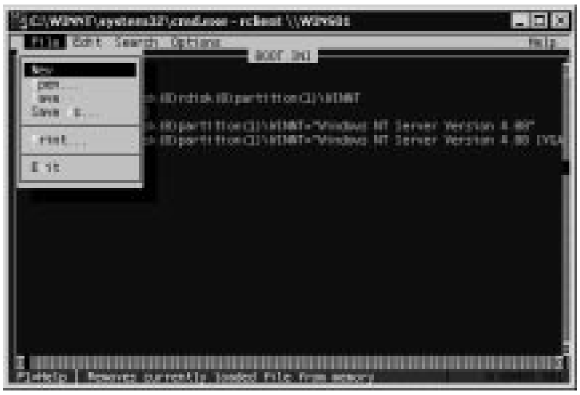


```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\WINNT\system32>whoami
CSI-TUS\boffn

C:\WINNT\system32>
```

Figure 4 - RCLIENT console onto remote server.



```
=\Drdisk(0)partition(1)\WINNT

=(0)partition(1)\WINNT="Windows NT Server Version 4.00"
=(0)partition(1)\WINNT="Windows NT Server Version 4.00 [VGA
```

```
F1=Help | Removes currently loaded file from memory
```

Figure 5 - RCLIENT running EDIT remotely.

the Cygwin distribution for diehard Unix users. Microsoft even includes a port of the Korn shell (ksh) licensed from Mortice Kern Systems in its Services For Unix product.

There are many good reasons for choosing an alternative shell for administrative work, including command and path completion and extra scripting abilities. Whichever shell you use, you can use a mixture of NT native commands, any added by your shell and the command-line Resource Kit tools to get the job done.

Secondly, we have the “start” approach: from a command shell SU’d to an appropriately privileged account, use START <GUI-program-name> to start an instance of a GUI tool running in the same security context as the shell which spawned it. A frequent example is installing software, which needs Administrator privileges but is a GUI process.

START D:\SETUP.EXE from a SU’d prompt will run the installation as Administrator. One odd problem that I have encountered is that the shell that is spawned has the path set to the value of OS2LIBPATH and nothing else. Not having discovered any way of correcting this I simply run a batch file which restores a sensible working environment.

Finally, SU can give you a complete desktop running as Administrator. In the SU dialog complete the Administrator credentials, then specify Winsta0\Desktop in the Desktop field and press OK (ensure Switch To Desktop is checked). You will see a new command prompt on a blank desktop. To get a GUI working environment type START EXPLORER and the familiar NT shell will appear. You can now run GUI tools as Administrator. To quit the second desktop type EXIT in the original command window from which you started EXPLORER.

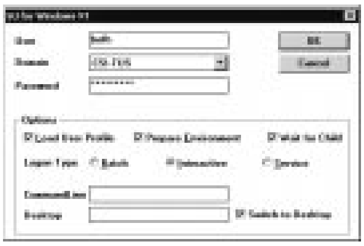


Figure 6 - SU’s GUI credentials screen.

SRVINFO

Keeping track of the patch level (or even OS level) of the computers on your network can be a time-consuming chore, especially if your corporate culture allows end-users to patch (or not) their own computers. In the current environment, with Internet-connected enterprises under attack from both determined hackers and amateurs using ready-made hacking scripts collected from Web sites, newsgroups, mailing lists, AOL forums or even the schoolyard, it’s vital to know which of your users’ machines have missed out on the most recent patches.

Microsoft provides a very poor mechanism for discovering this information, requiring the administrator to visit every machine and use the HOTFIX.EXE command to query the information the hotfix installations have left behind. SRVINFO lets the administrator obtain the same information over the network. It reports OS level, processor revision, service pack level and installed hotfixes, along with a listing of registered services and whether they are currently running.

The most important parameter to SRVINFO is \\ComputerName, which allows the administrator to direct their attentions to every computer on the network; this is most easily achieved by using the ever-versatile FOR command, with its /F argument, to read a list of values to be substituted from a text file.

SHUTDOWN

An appropriate command to end this series with, SHUTDOWN does what one would expect - shuts down a local or remote NT computer. Its syntax is:

```
SHUTDOWN <\\ComputerName> </L> </A>
</R> </T:nn> <“ Message ”> </Y> </C>
```

Interesting parameters are:

- **\\ComputerName** - if specified, shuts down the remote computer named. You must have administrative privileges over that computer or Domain Administrator privileges in the domain in which that computer participates in order to do this.
- **/A** - will abort a shutdown. You can only do this to cancel a shutdown which has had a timeout specified (see below).
- **/R** - reboots the computer targeted after it has shut down.
- **/T:nn** - sets the timeout in seconds. If no time is specified twenty seconds is used as the default.
- **“Message”** - lets the administrator send a warning message to any logged-in user of the affected computer.
- **/C** - forces all applications to close, without saving their data.

Conclusion

This article has looked at a few more of the most useful utilities in the Resource Kit, and has only looked at the most common options in some cases. Online documentation for all the utilities is available as a Windows Help file, supported by related Word documents and text files.

PCNA

Copyright ITP, 2000

The Author

Simon Pride can be contacted by email as simon.pride@itp-journals.com.