Cool storage

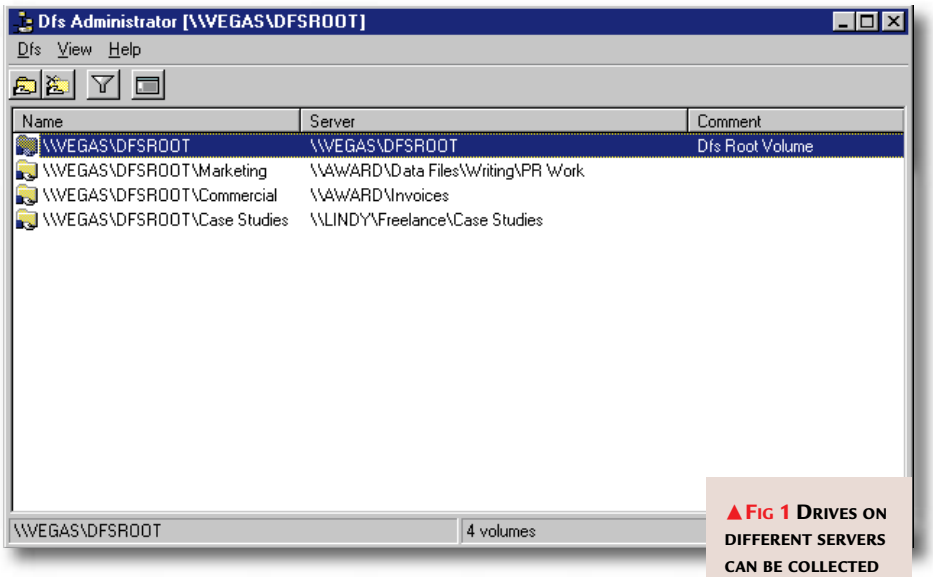
Andrew Ward assesses the advantages Microsoft's directory file system (DFS) for NT 4.0 and 5.0.

If you are a network administrator who has ever wanted to attach to a deep nested directory within a share, for example by typing something like this:

```
net use Q: \\SERVER\SHARENAME
\a\long\directory\path
```

you'll know that you can't do it if the server is Windows NT 4.0. Until now, that is, as you can install DFS, Microsoft's directory file system.

DFS will be part of Windows NT 5.0 and is available now for Windows NT 4.0. Its primary purpose is to allow network administrators to define a logical view of an organisation's file storage [Fig 1] which hides the physical structure from users. For example, you may have material stored in different folders on different drives on a number of servers around the organisation, which all relates to the marketing department. These folders can be collected together under a share named "Marketing", so that the user does not have to navigate the network to find all the marketing materials. Of course, this means extra work for administrators, since the DFS tree has to be built to combine the various shares from different servers under the DFS root directory. Two tools are provided for DFS administration to ease this burden: a graphical DFS manager and a scriptable command-line tool. There's no need to install any new software on the client systems, or even on the systems whose directories are being administered by DFS.



▲ FIG 1 DRIVES ON DIFFERENT SERVERS CAN BE COLLECTED TOGETHER TO PRESENT A LOGICAL VIEW OF STORAGE

The primary purpose of DFS is to allow network adminstrators to define a logical view of file storage

As far as I can tell, DFS needs to be installed only on the server which is subsequently going to own the share. Thereafter, from the point of view of the client systems, the DFS tree simply appears as a remote drive like any other. It appears that one server can own one DFS tree. With this extra level of indirection, server replacements can be hidden from users, who will now navigate via the DFS names rather than actual server names. If a server has to be changed for whatever reason, or a drive or its contents moved from one server to another, the user need not know. Another benefit of DFS is that you avoid the problem of running out of drive names: if you need to add more shares and directories, they can be added under the same DFS root. Installation is a little tedious. After executing the downloaded file, you have to go to the Network Control Panel, select the Services tab, click Add and then Have Disk (even if you see Distributed File System listed among the choices). There's no browse button in the Have Disk dialog box, so you have to type in the path manually, which will be something like:

```
C:\WINNT\system32\dfs
```

Then, after one of those infamous NT reboots, you're on your way.

Incidentally, you'll be pleased to hear that for Windows NT 5, the development team has been told to treat any requirement for a reboot as a bug, pure and simple. So most of these reboots will eventually become a thing of the past. Another feature of DFS is load balancing and resilience against failure. If you specify two directories on different servers for the same DFS path, then DFS will alternate requests between the two directories. If one of the servers disappears, then all requests will be directed to the other. However, DFS doesn't do any synchronisation or replication, so an administrator will have to ensure that the content of the two directories is identical, and this feature will clearly be of practical use only for read-only directories.

Being able to attach to a folder nested deeply within a DFS share is actually just a side-effect of DFS, but for some administrators, this will be a big enough

benefit to make DFS worth installing. To download DFS for Windows NT 4.0, visit www.microsoft.com\ntserver and look for Microsoft Distributed File System on the Downloads page.

Go ahead and su

No doubt you all remember the "su" (super user) command from Unix. Indeed, maybe some of you still use it. The Windows NT version will log you on as any user other than the one you are running, but the most common use is to grant yourself administrator rights so you can carry out a particular task without having to go through the pain of logging off and logging on again.

Many of us have probably (but unwisely) given in and resigned ourselves to permanently running with administrator rights, because we so often stumble across those little tasks that require administrator privileges. It would be much safer to use the "su" command, and only become an administrator when you need to, says Colin Simpson. Where do you find this magical su.exe? It's supplied with the Windows NT Resource Kit, but beware. The one that came with the original resource kit required a lot of user permissions to run in the first place. You needed rights to Act As Part Of The Operating System, Increase Quotas, Replace A Process Level Token, and Restore Files And Directories, which somewhat limited its purpose as a general-purpose user upgrade tool.

But no longer. Colin has spotted that with Resource Kit Supplement 2, a new service-based component avoids the need for these rights (although, predictably, you need to be an administrator to install this component in the first place). To run this component, from \I386\NETADMIN on the resource kit CD, type:

```
suss -install
```

The most obvious use of su to emulate the Unix functionality is to type:

```
su administrator
```

which will create a command prompt window logged in under the account administrator (assuming you have such an account on your system). Before that you will be prompted for a password, just as in Unix. You can circumvent the password request by redirecting stdin to a file containing the password, followed by a carriage return.Unfortunately, those of us who've been seduced by the GUI interface feel pretty helpless faced with a blank window and a command prompt [Fig 2], so what we really want is the Explorer. But, as Colin has found out the hard way, that doesn't work. You just end up with exactly the same rights as the already-running instance of Explorer which is managing your desktop.

As always, there is an answer, and that is to turn to the trusty old file manager (winfile.exe) and/or program manager instead.

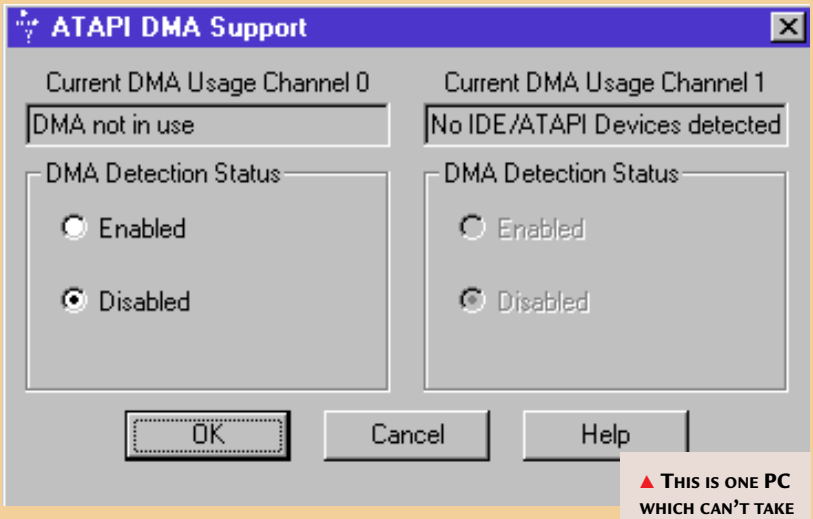
Colin has even found a way of making a control panel under the program manager, by creating a program group and putting in command lines like the example below. This one runs the desktop control panel applet, and obviously you can make others to run the various other applets.

```
rundll32.exe shell32.dll,
Control_RunDLL D:\WINNT\
system32\desk.cpl
```

The other interesting behaviour Colin has observed is that with Windows NT, drive mappings go with users and not the machine. Thus, if the desktop user has a particular drive mapping set, then the super user will not have access to that drive without setting up another mapping. And when you do, you can't use the same letter. For example, if the desktop user maps "F:" to \\VEGAS\DFSROOT, then the su session not only doesn't have access to that mapping, but can't use drive "F:".

With Windows NT, drive mappings go with users, not the machine

TAKE THE DMA TEST

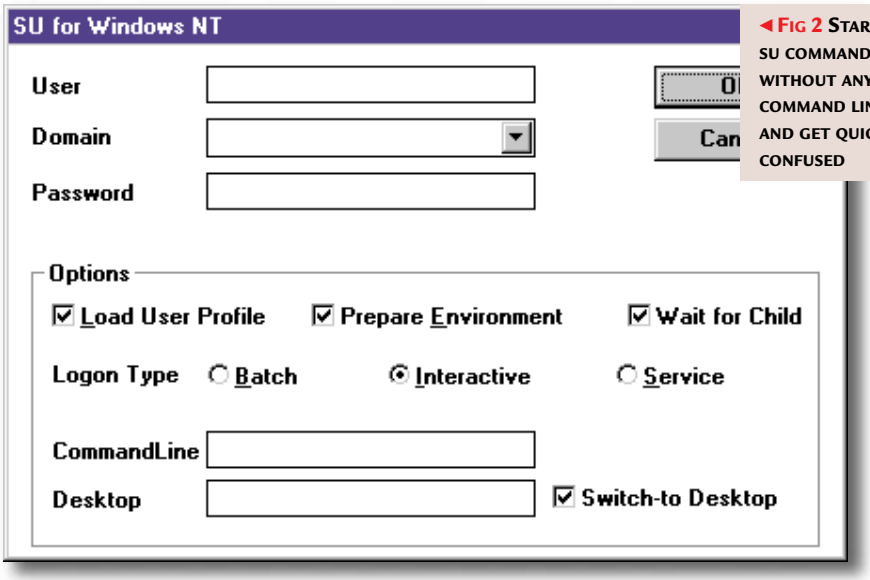


▲ **THIS IS ONE PC WHICH CAN'T TAKE ADVANTAGE OF HIGH-SPEED IDE DMA**

Many of Intel's more modern core logic chipsets include a high-performance, multiword DMA mode for transferring data to and from the IDE drive. A DMA bus-mastering capability that can make use of this mode was included in the atapi.sys driver that shipped with Windows NT Service Pack 3, but to turn it on, and potentially speed up hard-drive transfers, you first need to run a utility that also came with the service pack.

To find the utility, look for dmacheck.exe in the \support\utils\i386 directory. When you run it, observe the warnings given, and take a backup of all your data before you take the plunge.

Whether or not dmacheck.exe will succeed on a particular system is impossible to predict, since it depends not just on the chipset but on motherboard and BIOS factors as well. If it doesn't work, then there's some hardware limitation which means you can't use that facility. There's little point in looking to see what chipset version you have — that won't suddenly make it start working.



◀ FIG 2 START THE SU COMMAND WITHOUT ANY COMMAND LINE, AND GET QUICKLY CONFUSED

Conversely, if the super user sets up a mapping to a particular share, this won't be accessible to the desktop user. And Colin warns that you should remember to remove any drive mappings you make before terminating the super user session.

The one thing Colin hasn't found a way to do via a su session is to configure printers, since this is a function now carried out within the Explorer — there's no legacy printman.exe we can run. Can anyone help? Note also that the su command can usefully be combined with the scheduler service to run commands at specific times with required privileges.

At your service

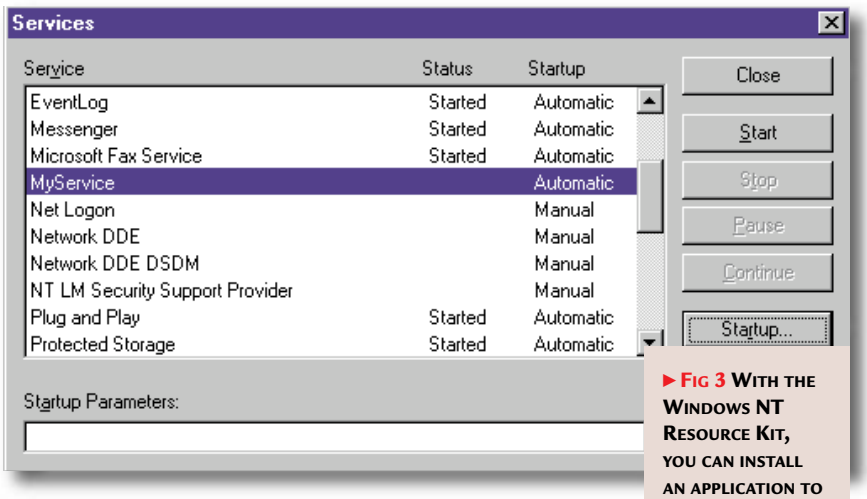
Jason Moxham writes to ask whether there's any mechanism to run a program under Windows NT as soon as a system boots up, so that there's no requirement to have a user log in first. As you probably know, there are many programs that run in this way on the average NT system, but they are all system services rather then normal executable application files. Fortunately, there is an answer. SRVANY.EXE allows you to run applications as services [Fig 3] and it comes with the Windows NT Resource Kit. To find it, look in the \I386\CONFIG directory of the CD-ROM.

Another benefit of using SRVANY, apart from removing the requirement to have a logged-on user, is that you can run an application with a different logon account than that of the currently logged-on user. This might be useful where you want something to run with administrator rights. To use SRVANY.EXE, it first needs to be installed as a service itself, using a command-line such as:

```
instsrv MyService "d:\program files\windows nt\accessories\srvany.exe"
```

INSTSRV.EXE can be found in the same location on the CD-ROM as SRVANY.EXE itself. You can install INSTSRV as many times as you like, but use a different service name each time ("MyService1", and so on). Now, go to the Services Control Panel, and specify the account that the service will run on (you can also do this via the INSTSRV command line). If you need to interact with the application via the screen and keyboard, specify the LocalSystem account and set the appropriate checkbox — but bear in mind that you then won't have network access. This isn't quite as bad as it sounds, because via a (documented) registry tweak you can name specific shares that will be made available to the LocalSystem account. Unfortunately, setting the application you want to run and its parameters requires fiddling with the registry, unless the service is configured to be started manually, which of course rather defeats the object.

On the subject of services and the registry, INSTSRV has another very convenient use, which is to remove unwanted services without needing to access the registry. Handy if you want to clear up leftover services (many applications don't seem to remove services when you uninstall them). For detailed instructions on the rest of the procedure, see the SRVANY.TXT documentation file with the Resource Kit. Remember, though, that you may have trouble with certain applications: they are running in a different context to the logged-in user and might not have access to the DLLs or other components they need. Also, some badly written applications running as a service will incorrectly terminate when the currently logged-in user logs off.



▶ FIG 3 WITH THE WINDOWS NT RESOURCE KIT, YOU CAN INSTALL AN APPLICATION TO RUN AS A SERVICE

PCW CONTACTS

Andrew Ward *can be contacted via the PCW editorial office (address, p10) or email* NT@pcw.co.uk