AFS on Windows

By Rodney M. Dyer

A presentation to the OpenAFS Best Practices Workshop

Stanford Linear Accelerator Center
Stanford, California
March 24-26, 2004

Abstract

Microsoft's SMB/CIFS networking solution dominates the world of Windows IT file serving. For many applications in small mono-cultural environments SMB/CIFS will suffice, however in heterogeneous large scale enterprises it shows inherent weaknesses. The Andrew File System (AFS) however, designed from the core to scale to world wide enterprise sizes is the network file system solution that can dramatically lower costs while increasing the efficiency and strength of a Windows IT infrastructure. This presentation describes the OpenAFS Windows client software, its features, and associated "best practices" used to access an existing AFS system.

Who I Am

I work for the University of North Carolina at Charlotte as a Windows system programmer. The title “system programmer” is a state position that's a hold over from the 70's mainframe era. Today the term system programmer can actually mean a number of things. I am one of the architects and programmers of our William States Lee College of Engineering Windows network. The Mosaic group, of which I am a member, manages the college computing platform. Please don't confuse our Mosaic group with NCSA Mosaic, we are unaffiliated and it's just a coincidence we used the same name.

Figure 1: The William States Lee College of Engineering at UNC Charlotte

Figure 2: Mosaic XP Lab Smith 249

The Mosaic group is a small team of 7 members plus T.A. helpers. I work closely with other members of our team to manage a diverse set of Windows XP and Sun machines used by the faculty, staff, and students of the college. I began as a student worker for the Mosaic group while earning my engineering degree. In all, we handle about 900 workstations, 25 servers, 150+ Sun Solaris apps, 80+ Windows XP apps, and 4,700 active engineering accounts.

We began working with NT 3.51 in mid 1996 just before the release of NT 4.0. Our goal was to setup our first fully secured and managed Windows platform. We had previously managed DOS, Windows 3.1, and Windows 95 workstations using NFS networking as well as a good bit of direct hands-on tech support. The goal for the new platform was to manage Windows much like we had managed our Sun Unix workstation environment. Using AFS in large part was the primary tool used to accomplish that task.

Client History on Windows

In late 1996 Transarc released the original AFS Client 3.4a for Windows NT 3.51 shown in Figure 3. Each major release was punctuated with later patch releases such as 3.4a patch x. As major client versions were released Transarc added features. With the release of client 3.5 in April of 1999 Transarc finally added a cache file, the server, the control center, and various other nice features. This was the first truly stable version. Version 3.6 was released in March of 2000 and was the last version to be released before Transarc was dismantled by IBM. Late in 1997 Transarc was renamed to IBM/Transarc Labs. Since that time IBM/Transarc has basically closed its doors on all further AFS information and has only released minor patch upgrades for clients that still use the last 3.6 version. IBM decided to open source AFS around September of 2000. The OpenAFS.org group was formed shortly thereafter and thus released the first version of OpenAFS 1.0.

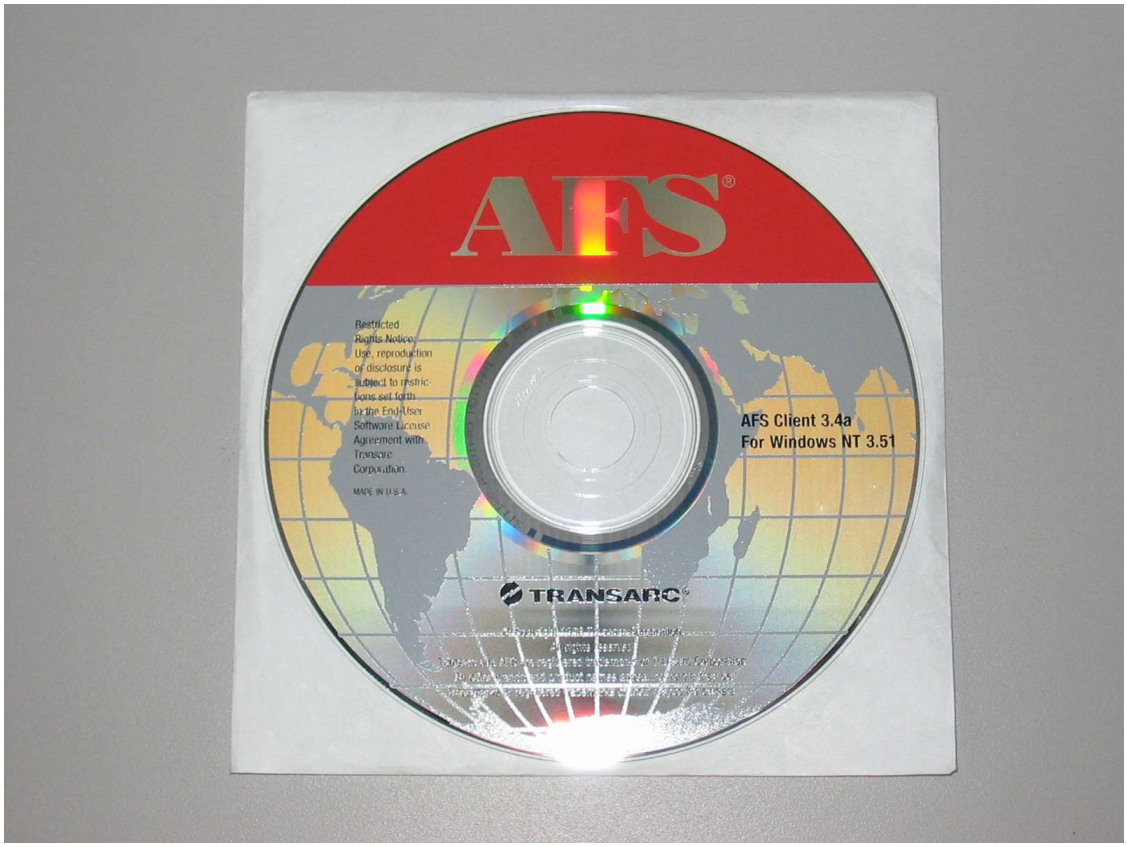


Figure 3: Transarc AFS Client for Windows NT 3.51

Key OpenAFS Windows Features

* Kerberos authenticated security.
* Continuously mountable global namespace.
* Token based credential security.
* Directory level user/group ACL security.
* User managed groups.
* Volume level quota.
* File and directory symbolic linking.
* Integration for heterogeneous environments of Windows, Unix, and Mac OS X.
* Encrypted network transfers.
* Good network utilization.
* Internet wide file sharing with other AFS institutions and businesses.

Setup of the OpenAFS Client

Setup of the AFS client only takes a couple of minutes and follows the general procedure of using a wizard like setup program.

1. Download the OpenAFS client. This is obtained by browsing to http://www.openafs.org then choosing the "Latest Release" under the "Downloads" section. The current production version available at the time of this document is 1.2.10. This version is for Windows 2000, Windows XP, or Windows Server 2003.

2. Run the executable. You will be presented with the “Choose Setup Language” dialog in Figure 4.

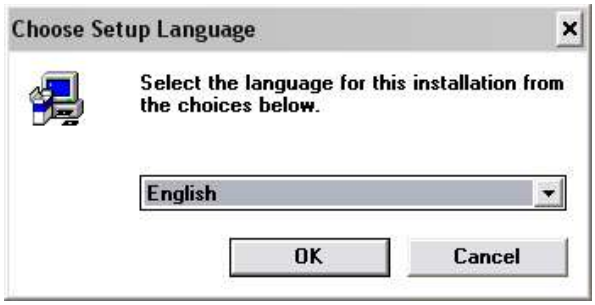


Figure 4: Select Language Dialog

3. After choosing your language you will be presented with the “Welcome” dialog shown in Figure 5.

Figure 5: OpenAFS Welcome Dialog

4. Choosing “Next” will bring up the “Select Components” dialog in Figure 6. In the "Select Components" dialog you can choose the various components you wish to install. Most people should choose to install only the client and documentation. You can also choose where you wish to install it and check available disk space required. Unless you have some need to install the AFS client elsewhere, you should allow it to install to the default location of "C:\Program Files\IBM\AFS".

Figure 6: Select Components Dialog

5. After selecting your components you will be presented with the “Select AFS Cell Data Base (afsdcell.ini)” dialog in Figure 7. The AFS cell data base selection dialog allows you to choose the initial source information to go in your new "afsdcell.ini" file. The "afsdcell.ini" file contains the list of AFS cells that you will be able to mount and their associated cell server IP addresses. This dialog can be somewhat confusing to beginning AFS users. Normally, you would put your own AFS cell into the "afsdcell.ini" and then add others as the need arises.

   The OpenAFS maintainers however keep contributed AFS cell information from around the world. In this case you can choose to use the default AFS cell data base file from "grandcentral.org" that comes in the OpenAFS setup executable, or you can choose to download a more current version directly from the website. You can also choose to use a previously installed "afsdcell.ini" file if it is still in an old location on the machine you are installing the AFS client on. Once you have a populated "afsdcell.ini" file, you can add or change your AFS cell information later with the AFS configuration utility (discussed later). In general the best option here is just to use the packaged installation file "CellServDB.GrandCentral".

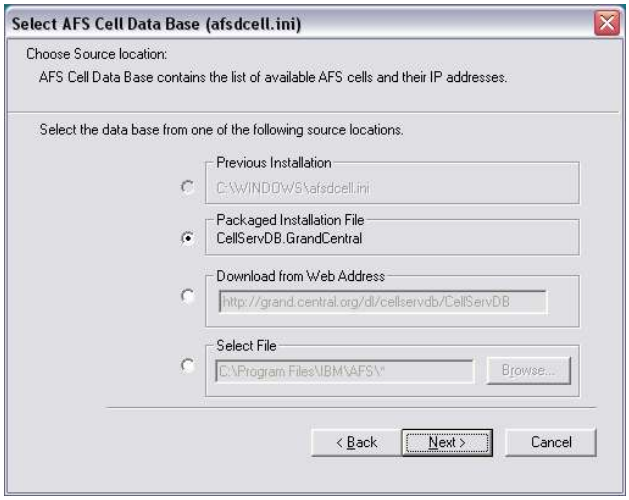


Figure 7: Select AFS Cell Data Base Dialog

6. After selecting the source for the cell data base you will be presented with the “Select AFS Cell Name” dialog in Figure 8. In the AFS cell name selection dialog you enter the name of your AFS cell, or more accurately the cell you will be authenticating to by default.

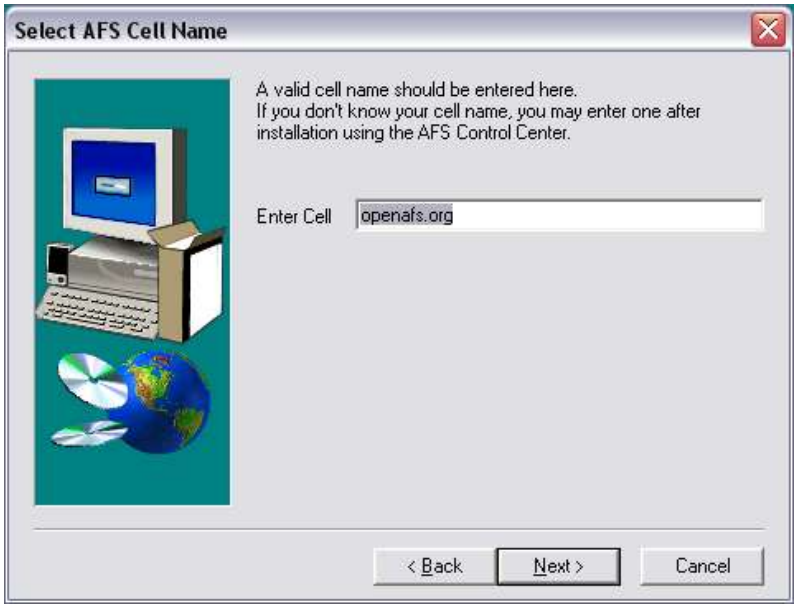


Figure 8: AFS Cell Name Selection Dialog

7. After entering your default cell you will be presented with the “Select Local Drive Mapping” dialog in Figure 9. In the local drive mappings selection dialog you can setup drives that will be mounted to AFS whenever a user logs on to the workstation. This dialog is simply an editor for the file "afsdsbmt.ini" shown in Figure 21. You will be able to edit or change these maps after AFS is installed by using the AFS configuration utility. At this point, it isn't absolutely necessary to create any drive maps, but the dialog allows this during setup for convenience. If you don't need any mapping done at this time, or you have an altogether different mapping scheme, then you may uncheck the "Enable Assignment" checkboxes.

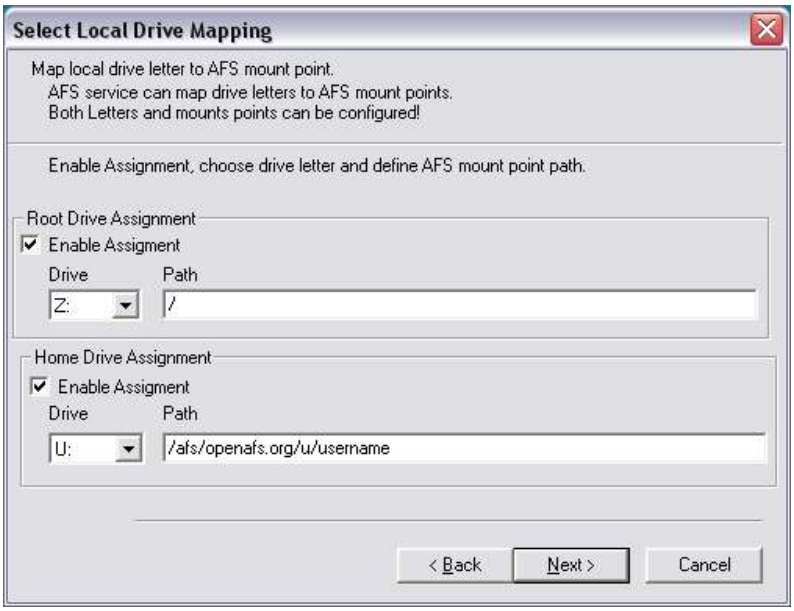


Figure 9: Drive Mappings Selection Dialog

8. After you have configured your drive maps you will be presented with the “Setup Complete” dialog in Figure 10. On the setup complete dialog you may either reboot the machine, or reboot the machine at a later time.

Figure 10: Setup Complete Dialog

9. After reboot the OpenAFS client service "afsd_service.exe" should start. Before you can use the AFS client with your organizations cell, you will need to fully configure it for your cell. You need to first logon as the local administrator. After logon, run the “AFS Client Configuration” utility shown in Figure 11. The AFS Client Configuration utility is found in the Control Panel list. Make sure you enable the Control Panel to view all the applet icons, also known as the "Classic" view.

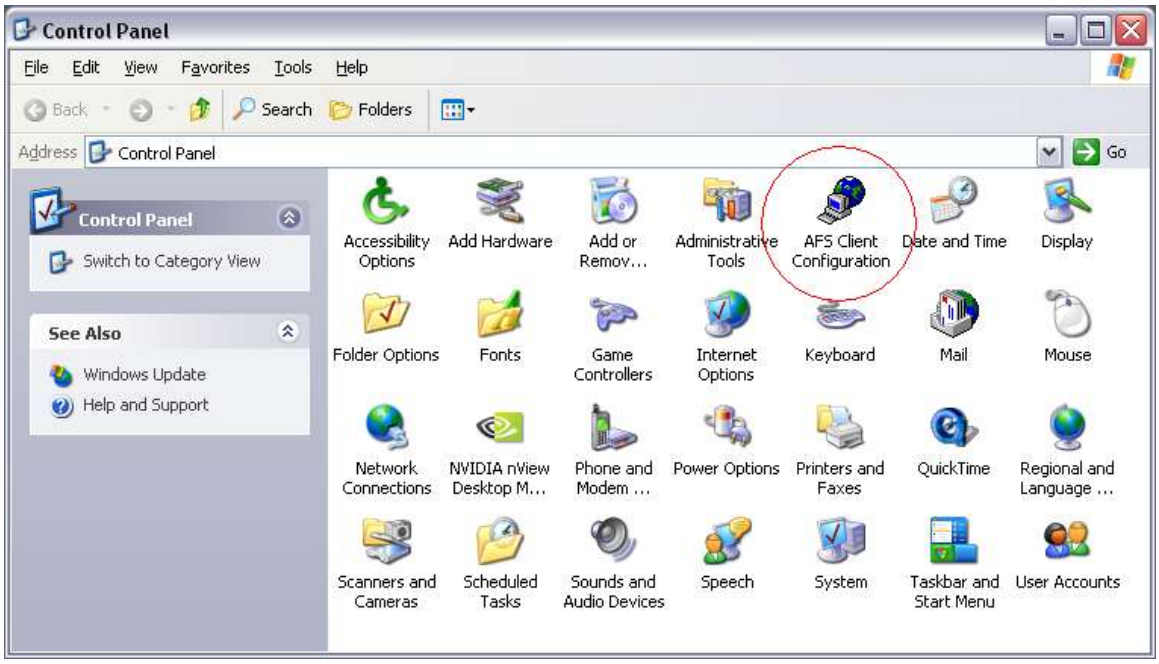


Figure 11: AFS Client Configuration Icon

Notes: You can also start the AFS Client Configuration utility by choosing the "Configure AFS Client" button on the AFS Client credentials dialog "Advanced" tab as shown in Figure 12. To start the AFS Client credentials dialog click on the icon shown in Figure 13.

Figure 12: AFS Credentials Advanced Options

Figure 13: AFS Client Credentials System Tray Icon

You can also simply choose to run the AFS Client Configuration utility directly at the command line, or on the Start Menu "Run" option, as in Figure 14 by typing:

afs_config.exe, or "c:\Program Files\IBM\AFS\Common\afs_config.exe"

(You could also create a desktop shortcut.)

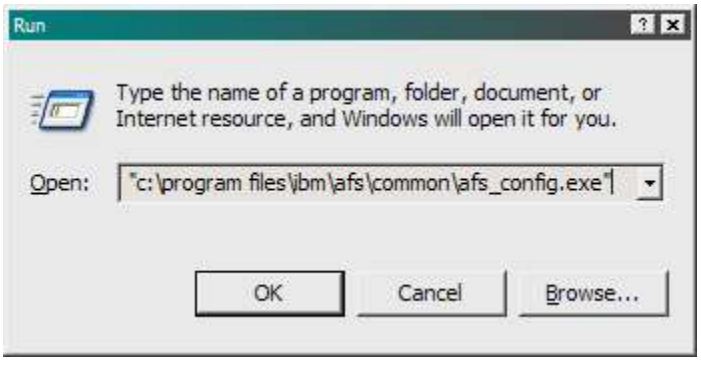


Figure 14: Starting the AFS Configuration Utility Directly

10. On the “AFS Client Configuration” utility shown in Figure 15, you will first need to make sure your cell is properly configured. Make sure your AFS cell name is listed under the "General Tab->Client Configuration->Cell Name".

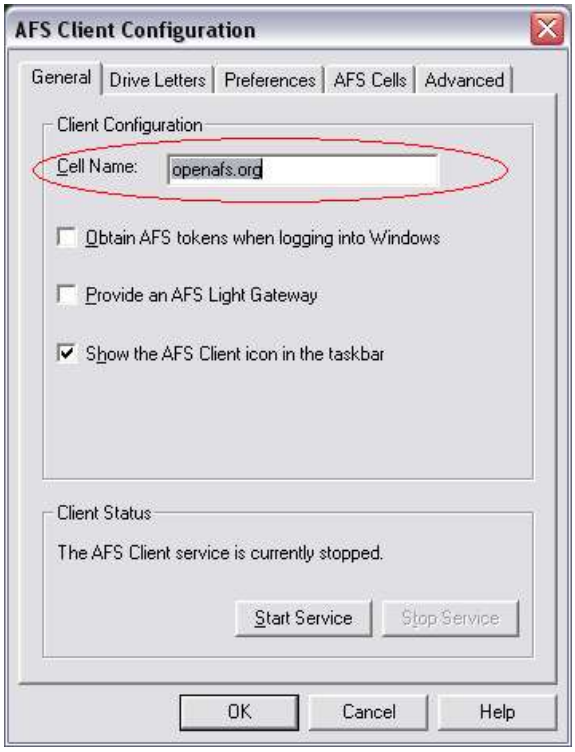


Figure 15: AFS Cell Name Configuration

11. Now choose the "AFS Cells" tab shown in Figure 16, then choose "Add..."

Figure 16: The AFS Cells Configuration

12. The “Add…” button brings up the “New Cell” dialog shown in Figure 17. In the “New Cell” dialog you enter your AFS cell name and a text description of it. Once you have done that then you can choose “Add…” to add your site cell servers.

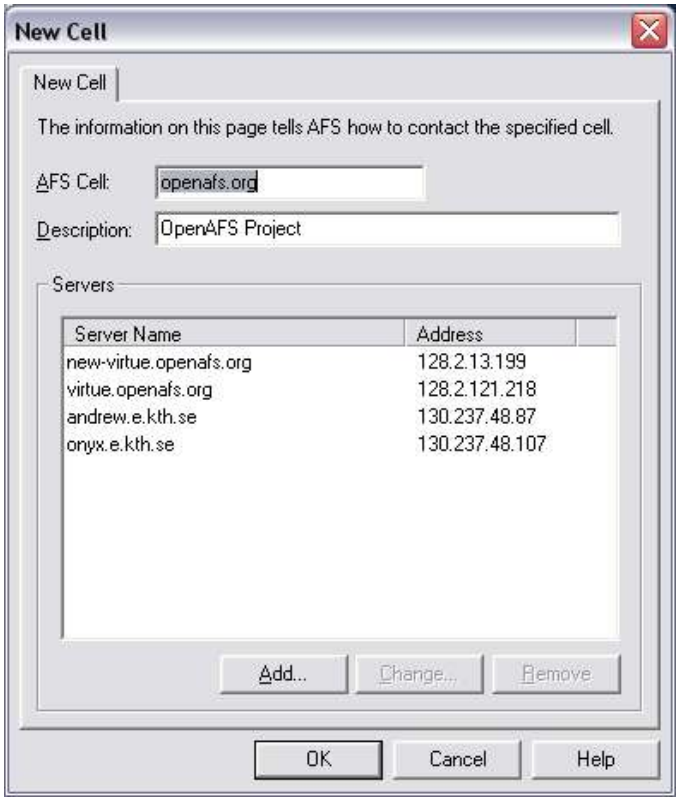


Figure 17: New Cell Dialog

13. The New Cell “Add…” button brings up the “Add Server” dialog shown in Figure 18. In the Add Server dialog you provide a single cell server name. You may either have the IP address looked up automatically, or you may provide it manually.

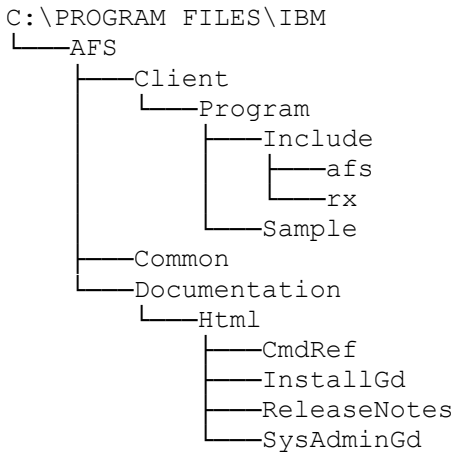
Figure 18: The New Cell Add Server Dialog

14. Once you have added all your cell servers you will need to restart your AFS client service to make your cell configuration active.

At this point you are finished with the setup of OpenAFS for your cell.

What Setup Does

The OpenAFS client setup installs software and registry settings. The software install involves placing binaries under the "C:\Progam Files\IBM" default location unless changed during setup. The following is the basic directory structure for the software:



Many of the files installed are libraries for AFS, others are command line executables and dialog DLLs. You will also see some included files for developers and necessary documentation.

A few other files are installed (or appear after proper configuration) elsewhere:

"c:\AFSCache".	This file is used to cache recently requested AFS information locally so that it is available quicker if requested again. See **Advanced Configuration Options** for more information.
"%SystemRoot%\afsdcell.ini" *.	This file is used to hold the AFS cell server information for the various AFS cells that you can mount. An example afsdcell.ini file is shown in Figure 19. This file is also known in UNIX as a CellServDB file. This file is edited using the AFS Client Configuration "AFS Cells" tab shown in Figure 16.
"%SystemRoot%\afsd_init.log".	This file is used to log debugging information by the AFS client service. An example afsd_init.log is shown in Figure 20. See **Advanced Configuration Options** for more information.
"%SystemRoot%\afsdsbmt.ini" *.	This file holds information about drive mappings and submounts. An example afsdsbmt.ini is shown in Figure 21. See **Mounting AFS** for more information.
"%SystemRoot%\system32\afs_cpa.cpl". . . .	This file is a control panel DLL used to provide access to the AFS Configuration dialog. The control panel icon created by this file is shown in Figure 11.

* These files are not true Windows INI file format. Do not treat them as such.

Figure 19: Example "afsdcell.ini" (CellServDB) File

Figure 20: The "afsd_init.log" Log File

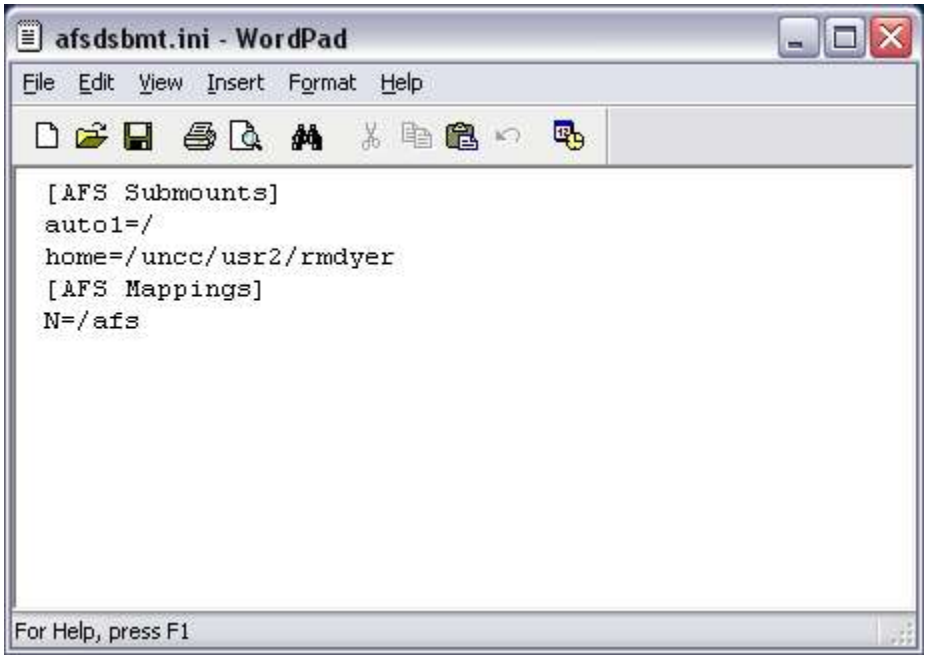


Figure 21: The AFS Client Drive Mappings and Submount Database File

Registry changes made during the install involve creating a software entry, a service entry, a registration for the Explorer shell extension, and some uninstall settings:

"HKEY_LOCAL_MACHINE\SOFTWARE\TransarcCorporation"

"HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TransarcAFSDaemon"

"HKEY_CLASSES_ROOT\CLSID\{DC515C27-6CAC-11D1-BAE7-00C04FD140D2}"

"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\AFS Client"

Note: These registry entries are only the top level keys. There are sub-keys and values below these root keys for configuration data.

Component Details

afs_config.exe - This is your primary GUI interface for configuring the AFS client. This was used during setup in Figure 15.

afscreds.exe - This is a GUI system tray utility. It will allow you to authenticate and discard tokens, create drive mappings, and configure other settings where necessary. This was seen during the install as Figure 12 and Figure 13.

afsshare.exe - This utility can be used for creating submount share entries. It is not typically used from the command line since you can do the same thing with the afs_config.exe GUI tool. You can create a submount entry simply by specifying the submount name and the AFS path as arguments. You remove an entry simply by specifying the existing name of the submount. This command has no help whatsoever at the command line, nor does it output any response.

afsd_service.exe - This is the AFS service binary (read daemon in Unix-land). This is the process that houses a credential manager, the cache manager, and the virtual SMB server among others. When this service isn't running, then AFS isn't running. This service can be stopped and started at the command line by performing:

c:\>sc start|stop TransarcAFSDaemon

or,

c:\>net start|stop TransarcAFSDaemon

or,

c:\>net start|stop "IBM AFS Client"

symlink.exe – A command line utility for creating symbolic links in AFS from Windows. Since Windows has no support for symbolic links under the Win32 subsystem one is provided in the client.

The following AFS standard command line system commands are provided:

backup.exe
bos.exe
fs.exe
kas.exe
klog.exe
kpasswd.exe
pts.exe
rxdebug.exe
tokens.exe
translate_et.exe
udebug.exe
unlog.exe
vos.exe

Also included is an Explorer shell extension DLL called "afs_shl_ext.dll". This shell extension allows a GUI interface to some of the AFS file system utilities. The shell extension is shown in Figure 22.

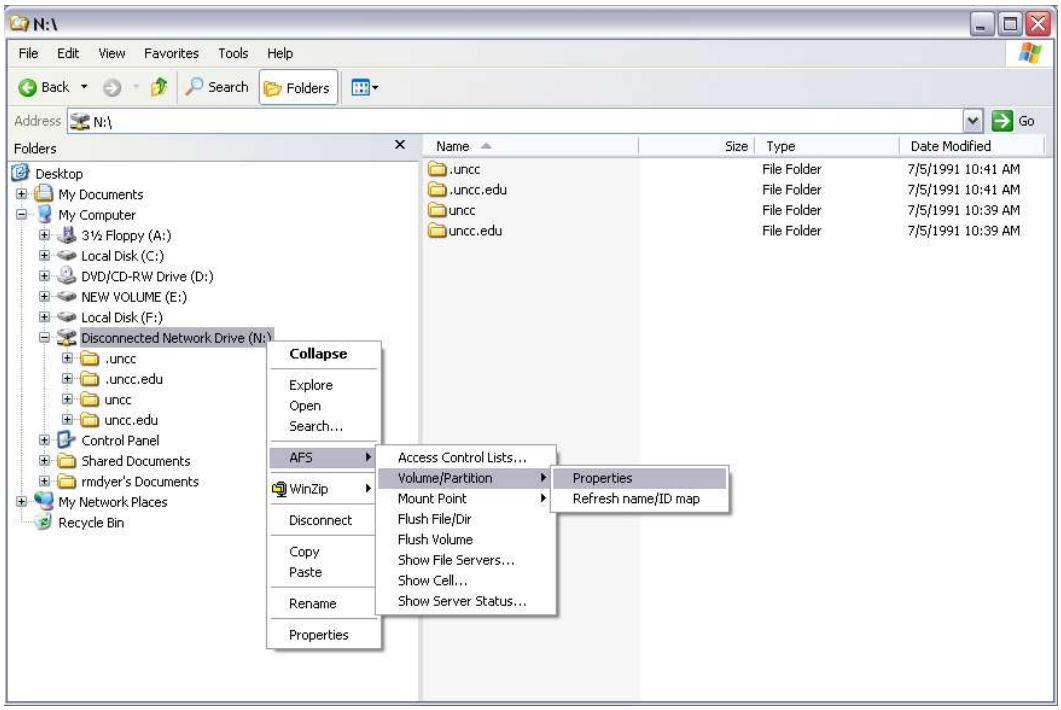


Figure 22: The AFS Client Explorer Shell Extension

Advanced Configuration Options

There are many configuration options under the “Advanced” tab of the “AFS Client Configuration” dialog shown in Figure 23. Few of the options need to be changed in normal use. We will discuss a few that might be changed here.

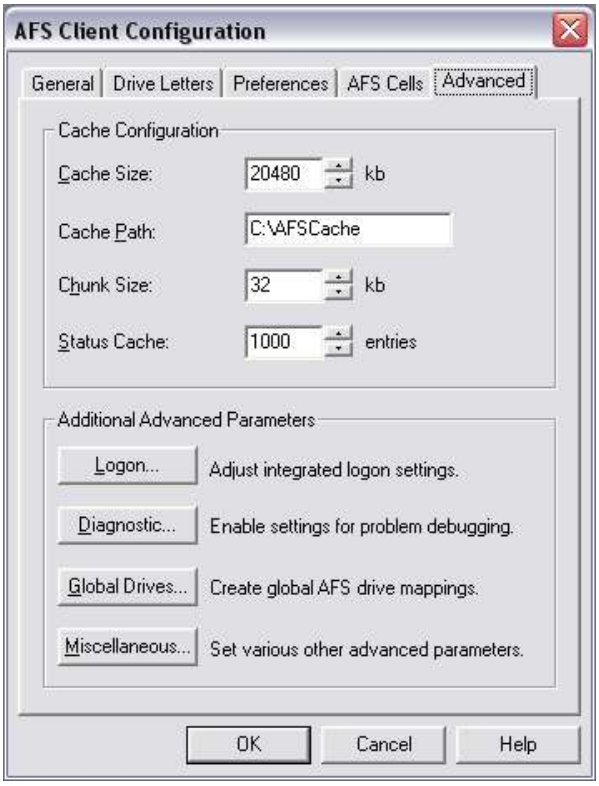


Figure 23: The AFS Client Advanced Configuration Options

1. **Cache Size** – The cache size refers to the amount of disk space set aside for local caching of AFS file system information. The cache increases the speed for requested information while at the same time reducing the overall network traffic. Due to recent bug report information on OpenAFS 1.3.52 and previous versions, it was discovered that because of the AFS Windows cache manager code design, the cache size should not be made very large. If made too large, the cache tends to reduce the available PC application memory and can cause performance issues.

   Recent work has been done to correct flaws in the AFS Windows cache code that will increase performance and prevent memory leaks. This work should appear when OpenAFS version 1.4 appears.

2. **Chunk Size** – The AFS client requests information from the file servers in chunks (also known as blocks). In general a chunk of file system information is requested, stored in the cache, and a file handle is allocated for reference to the chunk. To prevent excessive allocation of file handles which may cause Windows performance issues the cache size shouldn’t be made too large in relation to the chunk size. In general the maximum number of handles allocated would be determined by cache size divided by chunk size.

3. “**Diagnostic…**” – The AFS client service keeps a log of important debugging information in a file called “afsd_init.log”. Example contents of the “afsd_init.log” are shown in Figure 20. You can change certain log parameters using the diagnostic button.

4. "**Miscellaneous…**" - AFS options that rarely need to be changed can be found here. You can change such parameters as network adapter association, background threads, and service threads as well as others. About the only thing most people would need to change would be the background and service threads, especially if you run many applications that access AFS at the same time.

Mounting AFS

Mounting AFS, also known as “Drive Mapping” under the AFS client configuration utility, will allow you to associate a network drive letter with a specified AFS tree directory. In the following discussion, the terms “mounting” and “drive mapping” are used synonymously.

Under Windows accessing or mounting AFS is as simple as mounting any other Microsoft file system share. The root of the AFS tree is considered to be mountable by everyone and so requires no authentication at mount time. Subdirectories of AFS may or may not require pre-authentication via obtaining a token depending on the method used to mount the directory.

There are at least 6 methods that can be used to access or mount (map) network drives to AFS on Windows:

1. If you are already logged on, you can mount AFS by using the various Microsoft standard Windows Explorer GUI shell interfaces. This is generally done by right clicking on the “My Computer” icon, then choosing “Map Network Drive…” option.

   By using this method you are allowing the Windows Explorer shell to create the mapping. If you choose the option to “Reconnect at logon” when you create your mapping then whenever you logon Windows will try to re-establish your AFS connection. The information to do this is saved in the registry under "HKEY_CURRENT_USER\Network". If the AFS service is not running at the time the drive is being mapped then the drive will show up in Explorer as unavailable and will not be usable until the service is running again.

2. You can mount AFS by using the AFSCreds system tray icon tool. This is done by choosing the “Drive Letters” tab then choosing “Add...”

   With this method you are using one of the programs that came with the AFS client suite of tools to create a drive mapping. When creating the mapping if you choose the option “Restore this mapping whenever I logon” then Windows will try to re-establish your AFS connection at logon. The information about current and previous drive mappings as well as information about what drive mappings to restore at logon are saved in a file called “afsdsbmt.ini” shown in Figure 21. The actual restore drive mapping information for Windows is saved in the registry as in method 1.

3. You can mount AFS by using the AFS Configuration utility “Drive Letters” tab, then choosing “Add…”

   This case, except for the different interface dialog, does exactly the same job as described in mounting option 2 above. An extra bonus feature here is the ability to create submounts (discussed below).

4. You can mount AFS at the command line using the “NET USE” command.

   This method is commonly used by people proficient with the command shell interface. It is the historical way to mount drives on Windows using the old Microsoft LAN Manager syntax. This method is useful in batch scripting such as logon scripts.

5. You can allow the AFS client service to map a “global drive” for you whenever the service is started.

   This method creates a continuously mounted AFS network drive that exists even if no one is logged on. This is especially useful for management of the PC via scheduled tasks. Batch scripts can be written that can update the local machine with information from various AFS subdirectories. This method is also good to be used with “Roaming User Profiles”, or RUPS. A plus to using this method is that the users cannot disconnect the drive.

6. You can access AFS without needing to map a network drive by using the UNC path syntax.

   This method is useful for quick data lookups without the need to permanently or temporarily mount a drive.

Basic Mounting Syntax

This section uses the following typographical conventions:

Convention	Description
hostname	The client PC hostname. In most cases this can be substituted with the environment variable %COMPUTERNAME%.
username	An AFS user's AFS account name.
drive	An available Windows network drive letter usually D to Z.
cell	Your AFS cell name.
dir1, dir2, dir3	Subdirectories in your AFS tree.
-afs	The appended designation that refers to the AFS client SMB server.
all	The word "all" is a permanently built-in mount name for the AFS client that refers to the root of the AFS tree.

1. Current Syntax

   A. Using AFS directly via a UNC path:

      C:\>dir "*hostname*-**afs****all***cell**dir1**dir2*"

      or,

      C:\>dir "\\%COMPUTERNAME%-**afs****all***cell**dir1**dir2*"

   B. Mounting the AFS root as a drive:

      C:\>net use *drive*: "*hostname*-**afs****all**"

   C. Create an abstraction by associating a drive with an AFS subdirectory.

      Method 1 (direct* mount):

      C:\>net use *drive*: "*hostname*-**afs****all***cell**dir1**dir2*"

      * Depending on the AFS ACL security at the directory level you are mounting, this method may require you to be pre-authenticated with a token.

      Method 2 (indirect mount):

      Create a submount (See **Creating Submounts**) then mount the submount name like a share.

      Example, if submount name "*username*" = /afs/*cell*/*dir1*/*dir2*/*dir3*/*username*

      C:\>net use *drive*: "*hostname*-**afs***username*"

2. Future Syntax

Anywhere the "*hostname*-**afs**" is used you can simply replace the whole string with the word "AFS".

C:\>net use *drive*: "**AFS****all***cell**dir1**dir2*"

You may also remove the "**all**" designation and instead begin directly with the "*cell*". Example,

C:\>dir "**AFS***cell**dir1**dir2*"

Creating Submounts

In Windows you can share out a subdirectory on your hard drive using a share name. This is done by using the NET SHARE command, or by using the Explorer interface. With the AFS client you can similarly create reference names for subdirectories in AFS and map a drive to those names. These names are known as submount names. The root of a drive mapped using a submount name will be the AFS subdirectory you specified.

Submounts can be used as an abstraction in case you don't want to use a full path specification to a directory from the root of the AFS tree. They can also be used if you need to relocate a subdirectory to some other location in the AFS tree. By using the submount name, the application doesn't need to know the data moved.

Submounts are created with the AFS configuration utility shown in Figure 24.

Figure 24: AFS Drive Configuration Dialog

Clicking on the "Advanced" button will give you the dialog shown in Figure 25.

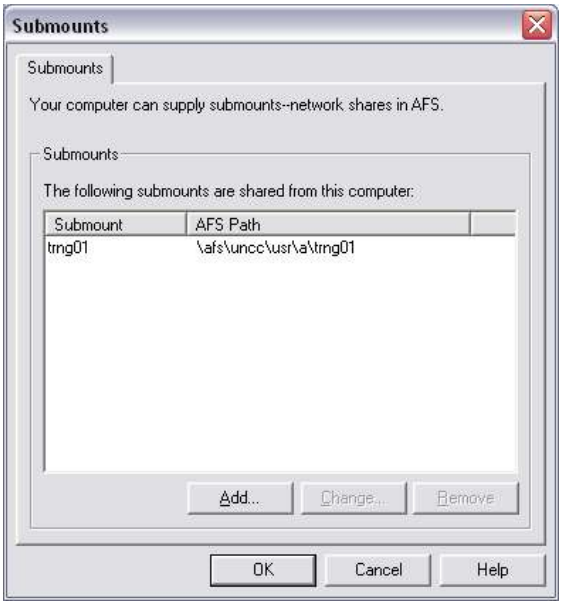


Figure 25: AFS Configuration for Submounts

Clicking on the "Add…" button gives you the dialog shown in Figure 26.

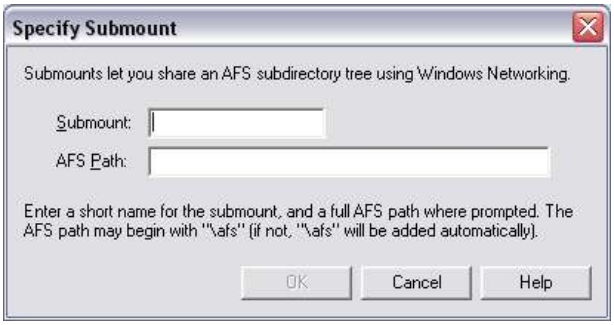


Figure 26: Adding an AFS Submount Name

You can also create a submount at the command line by using the "afsshare.exe" utility.

Example:

C:\>afsshare *username* /afs/*cell*/*dir1*/*dir2*/*dir3*/*username*

This will create the submount named "*username*" pointing to the AFS directory "/afs/*cell*/*dir1*/*dir2*/*dir3*/*username*".

To remove a submount name at the command line just do the following…

C:\>afsshare *username*

You should note that the "afsshare.exe" command does not have any help associated with it at the command line. The "afsshare.exe" command would most likely be used for any real-time logon scripting you may need.

Drive mappings and submount names are actually databased in the file "afsdsbmt.ini" as shown in Figure 21. This file is usually stored under "%systemroot%\afsdsbmt.ini".

Mounting Techniques

There are several techniques a user or administrator may use to mount drives to AFS. Listed here are a few of them.

1. You can mount drives to AFS manually after you logon.

   This is the simplest most basic case. This technique is useful for people who are knowledgeable about how to use the drive mapping tools and know where their data is located in the AFS tree.

2. You can have Windows re-map your drives for you after you logon.

   This is the reasonable extension of technique 1 where you have a recurring need for a common drive mount. The user would still require knowledge about how to setup the mapping.

3. You can have the AFS service mount a global drive at reboot.

   When the AFS service starts up such as at reboot, you can configure the service to map a global network drive. This technique is useful for administrators who are providing a common and continuously mounted drive across a network of machines. This method prevents normal unprivileged users from removing or changing the mount. With a drive mounted as SYSTEM, administrators can use task scheduled batch processes to access data in AFS to update the local PC. Global drives are also useful for accessing roaming profiles at user logon. The only downside to this technique is that you have little control over error checking of the drive mapping process.

   To setup global AFS drives, see the "Create global AFS drive mappings" option on the AFS Client Configuration dialog in Figure 27.

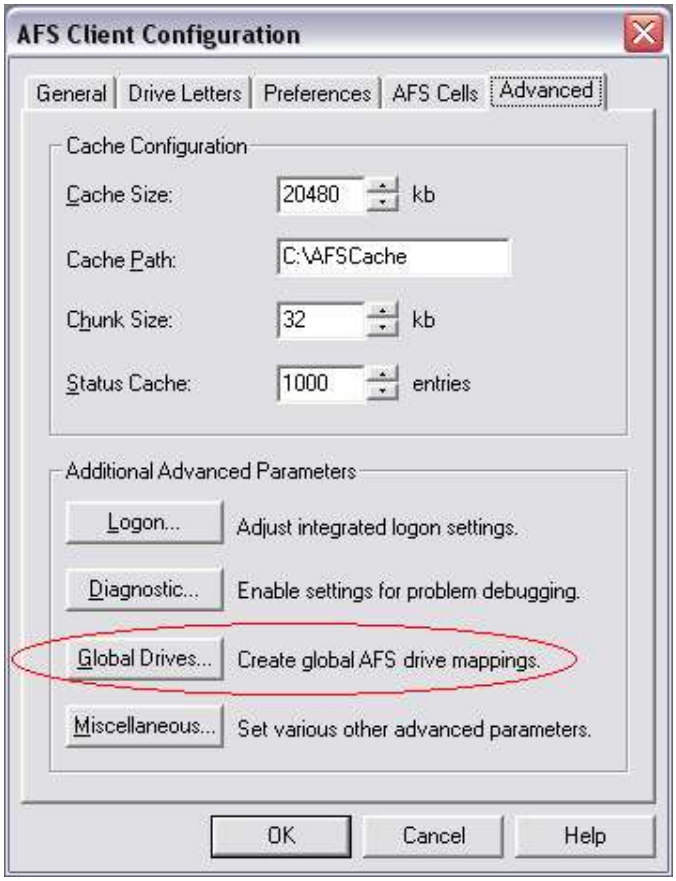


Figure 27: AFS Global Drive Setup

Clicking on the "Global Drives…" button brings up the global drive dialog as shown in Figure 28.

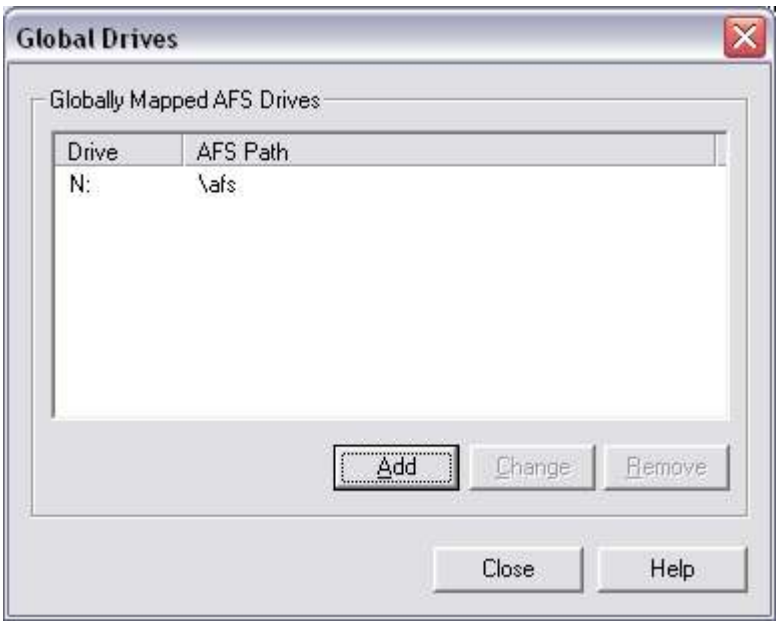


Figure 28: The AFS Global Drive Dialog

Clicking on the “Add” button brings up the configuration dialog for a global drive as shown in Figure 29.

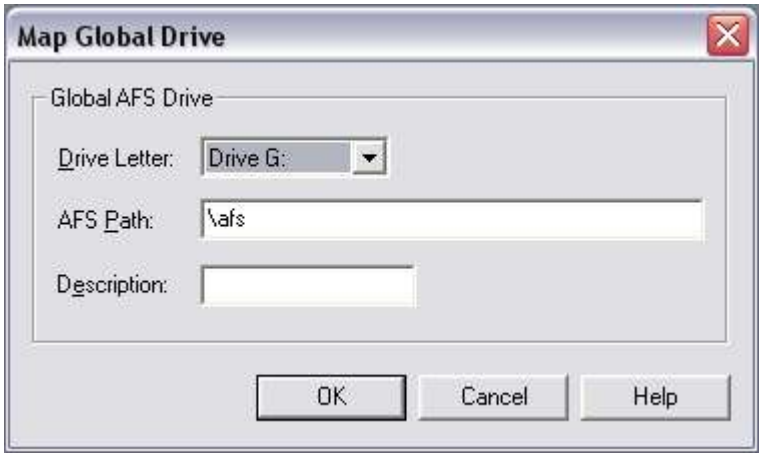


Figure 29: Global AFS Drive Configuration Dialog

4. You can create a group policy system startup script that mounts a system drive at reboot.

   This technique results in all the advantages of technique 3 except it gives you a bit more administrative control over how your global drive is mounted. You can also perform better error checking if there is a mounting problem.

   The location of the AD group policy startup script configuration is shown in Figure 30.

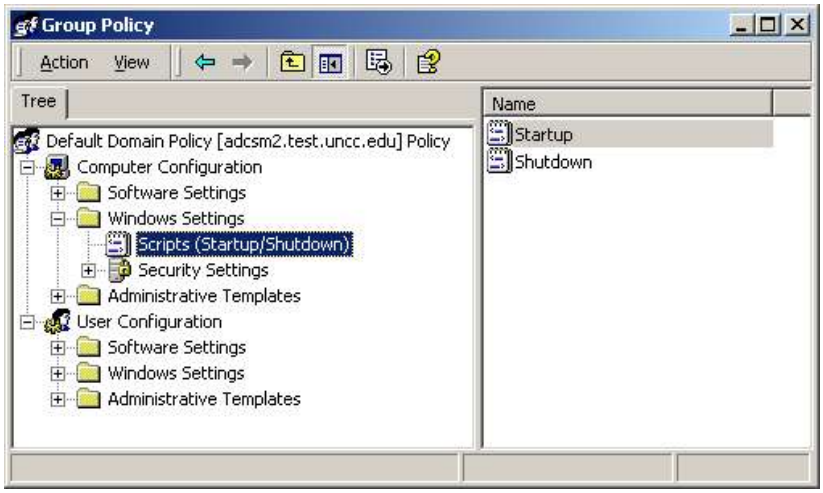


Figure 30: Group Policy Startup Script Location

Clicking on the startup script brings up the properties as shown in Figure 31.

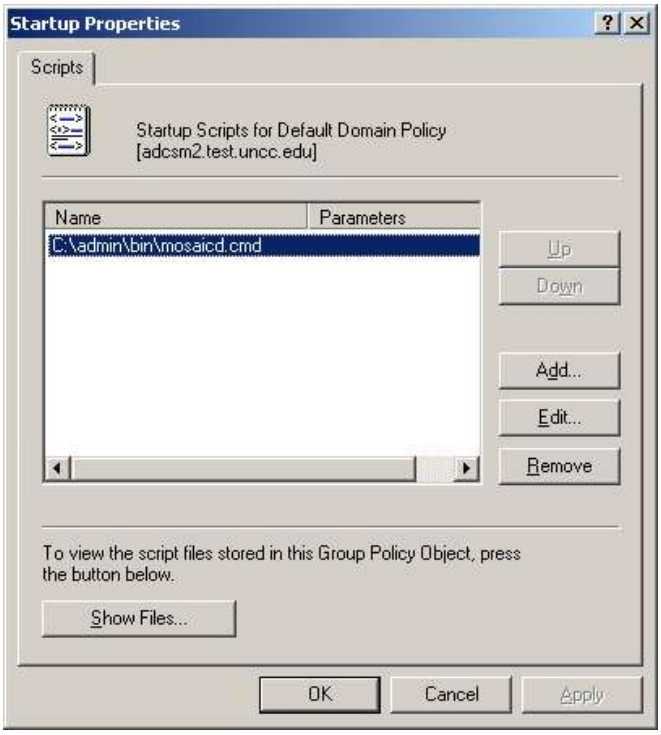


Figure 31: Group Policy Startup Script Properties

5. You could create a one-shot service to map a global drive at reboot using the SRVANY.EXE program from the Windows resource kit.

   This would be done by configuring the SRVANY service to execute a command shell script that will map the global drive as user SYSTEM. When the PC is rebooted, the SRVANY service would auto-startup, the script would be run, and your global drive would be mapped. Before the command shell script is complete, it would stop the SRVANY service, hence the nature of the “one-shot” at startup. The drive mounting procedure would be the same as mounting technique 4. The only advantage here is that you are not depending on Microsoft group policy services for starting your script. This is a “roll your own” technique for those who are masochists.

Authentication

Once you have a drive mounted to the AFS tree you are running unauthenticated. Running unauthenticated, you can access any directories that have ACLs set to "system:anyuser" with some access rights like 'read' or 'look'. For more substantial rights you will need to authenticate. In AFS terms this means obtaining a "token". There are many methods to obtain a token for AFS. Many of these methods depend on your needs as a user or systems administrator. The following is a general outline.

Basic Authentication

To authenticate to AFS you can use the GUI or the command line. This is done after you have logged on the Windows system and are ready to use AFS.

To use the GUI method you simply bring up the "afscreds.exe" system tray utility. See Figure 32.

Figure 32: AFSCreds System Tray Icon

Whether or not this utility shows up in your system tray is configured with the option "Show the AFS Client icon in the taskbar" on the "afs_config.exe" dialog shown in Figure 33.

Figure 33: Setup of AFSCreds System Tray Tool

On the "Tokens" tab, choose the button "Obtain New Tokens..." This brings up the authentication dialog in Figure 34.

Figure 34: AFS Authentication Dialog

To authenticate via the command line just use "klog.exe" as shown in Figure 35.

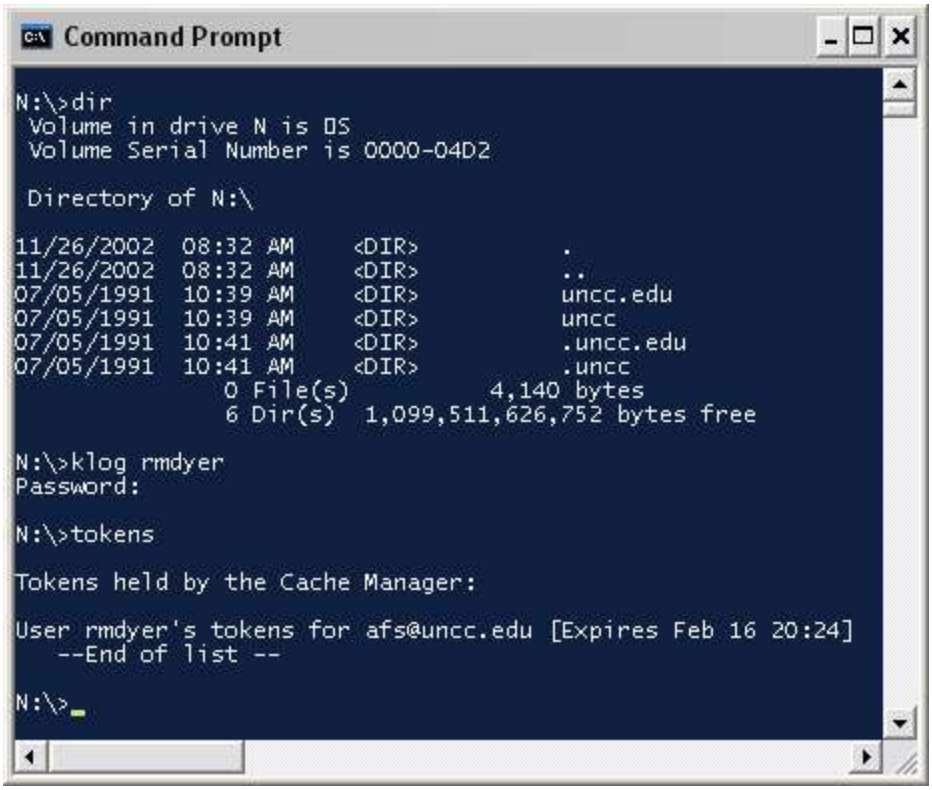


Figure 35: Using Klog to Authenticate at the Command Line

As you can see in Figure 35, the "tokens" command is used to determine if you are authenticated at the command line.

Logon Authentication

It is a bit annoying to need to "klog", or run AFSCreds every time you want to authenticate to AFS independent of your logon to Windows, so the AFS client can be setup to automatically authenticate you to AFS just immediately after Windows authentication. To do this choose the option "Obtain AFS tokens when logging into Windows" on the "afs_config.exe" program shown in Figure 36.

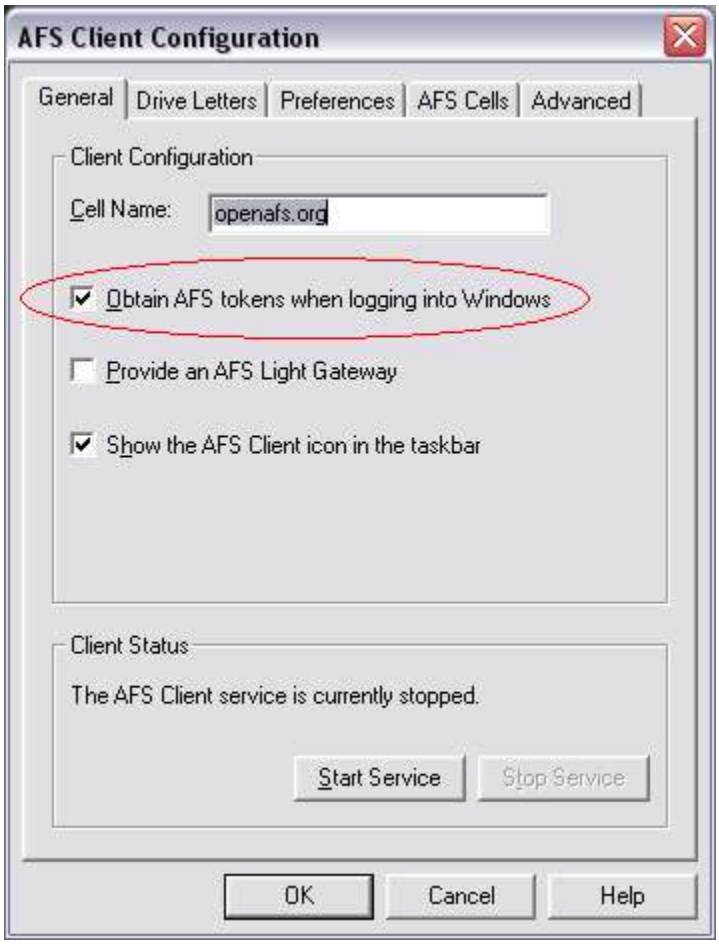


Figure 36: Setup for Logon Authentication

When logon authentication is enabled, you are authenticated to AFS using the username and password obtained from your Windows logon. Of course in order for this to work you must use account names on your Windows system that match those for AFS. Most AFS sites do keep their Unix, AFS, and Windows accounts in sync, but some do not.

How Logon Authentication Works

When you press Ctrl+Alt+Del on the Windows logon dialog then enter your username and password, the Winlogon process calls upon the "MSGina.DLL" (assuming it hasn't been replaced) to authenticate you to the system. Once the "MSGina.DLL" has verified your access to the system, you will then need to be authenticated to the various network providers that provide network services to the machine. To do this the Winlogon process calls all the logon providers listed in the "ProviderOrder" value under the registry key shown in Figure 37.

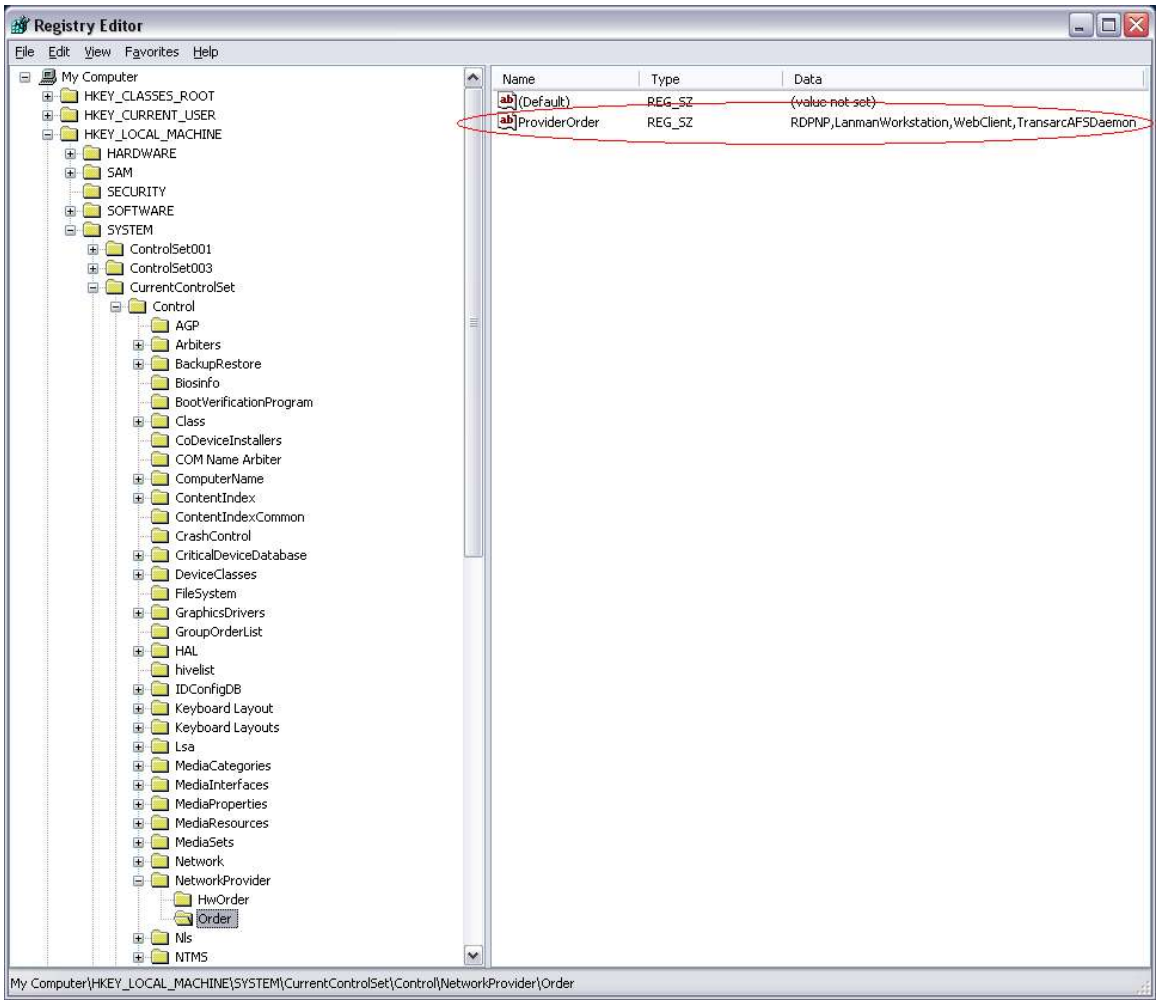


Figure 37: Network Provider Registry Key and Value

The "TransarcAFSDaemon" entry in the "ProviderOrder" registry value tells the Winlogon process to find the "logon provider" DLL used for authentication to AFS and execute the "NPLogonNotify" function within it. It is the responsibility of the "NPLogonNotify" function to authenticate the user to AFS using the username and password passed to it by Winlogon. When the "NPLogonNotify" function completes then control returns back to Winlogon where it then initializes the logon user session. The user logon initialization usually starts with downloading the user profile (if it needs to be downloaded), then calling upon "userinit.exe" to execute any administrative logon scripts and the user's shell (usually Explorer).

The Winlogon process finds the "logon provider" DLL by looking for the "NetworkProvider" registry entry under the "TransarcAFSDaemon" service registry key as shown in Figure 38.

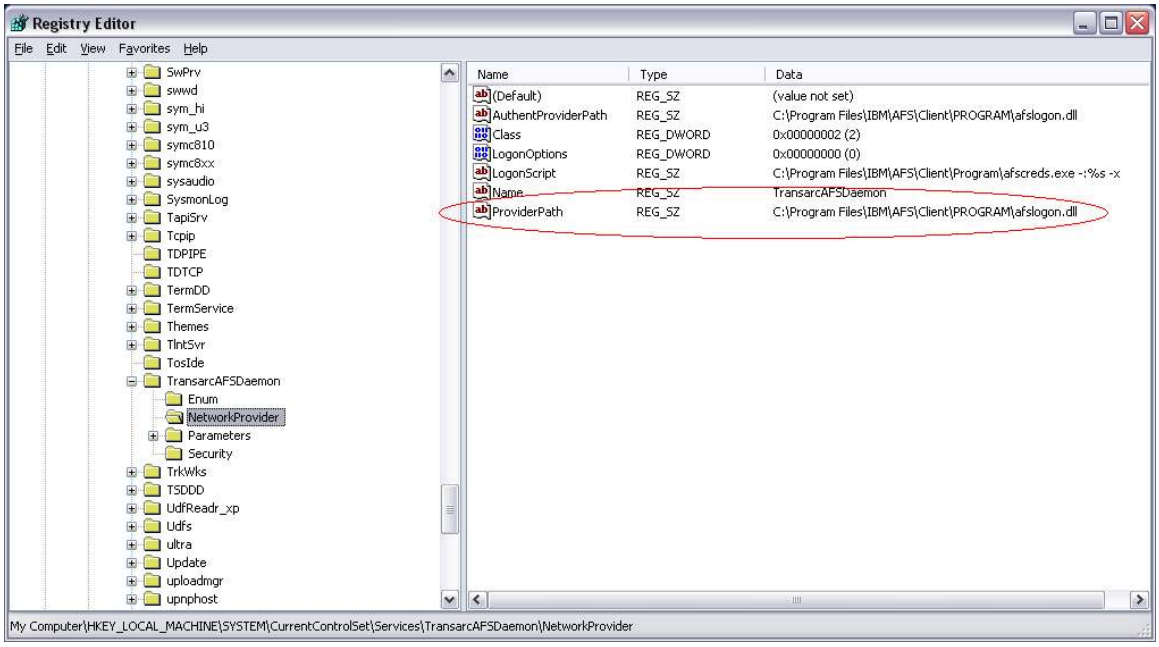


Figure 38: Logon Provider Location Specification

AFS Logon Authentication Process

The entire logon process is detailed in Figure 39.

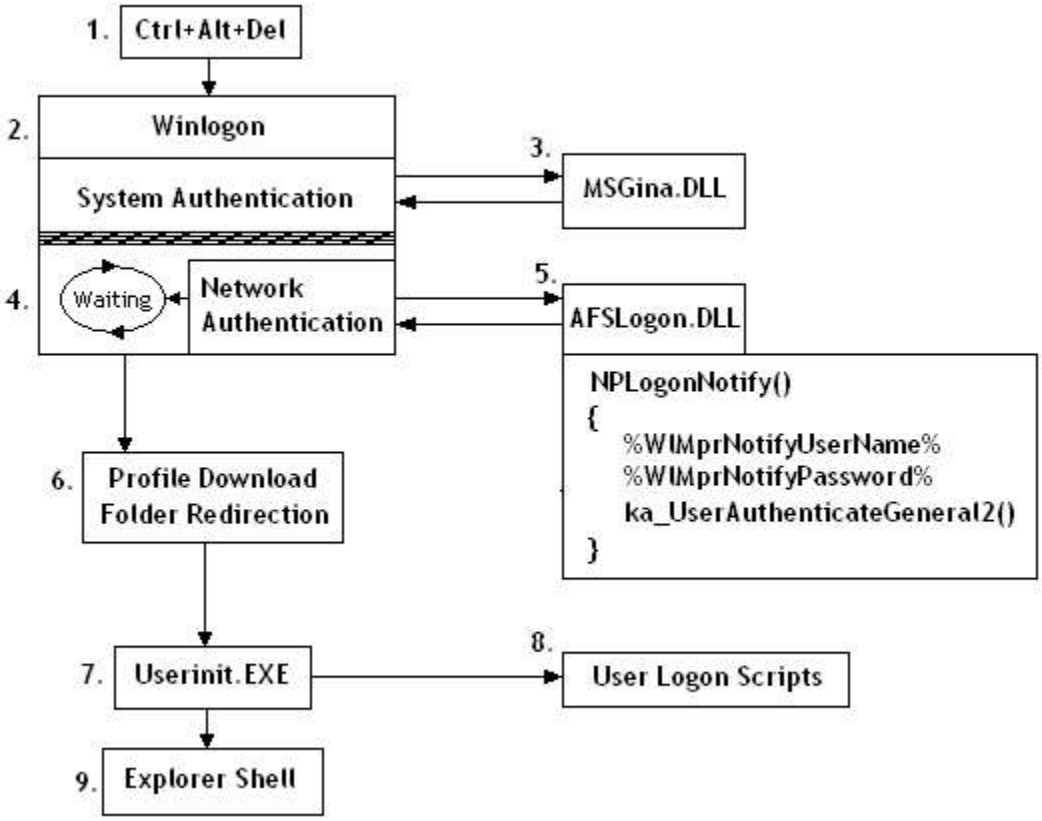


Figure 39: The AFS Client Logon Authentication Process

The Loopback Adapter Is Your Friend

The AFS client service presents AFS to Windows networking as a mountable server share. This is accomplished by the AFS service because it creates a virtual SMB server. The virtual server name will be the client's hostname with "-afs" appended to the end. The virtual SMB server must be associated internally with a network adapter. Any virtual SMB service that is associated with a real network adapter will receive requests from the real network as well as internal requests. This can sometimes cause problems for example if the network cable is unplugged, or the SMB ports are probed from the outside world, the AFS service can be DOS'ed, and the service may crash.

To add a higher level of reliability to the AFS client service you can install another network adapter, one that doesn't interact with the outside world, the Microsoft Loopback Adapter. Starting with OpenAFS version 1.2.8 and above, a change was made to allow the AFS service to detect whether the loopback adapter was installed. If the loopback adapter is found, then it will be used instead of the real network adapter. Note that this will prevent the "AFS Light Gateway" from functioning because your AFS client will no longer be able to receive requests from machines on the real network.

Loopback Adapter
http://grand.central.org/twiki/bin/view/AFSLore/WindowsLoopBackAdapter

The Loopback Adapter installer was made possible by the following contributors…

Scott Williams
Ben Creech
Kim Kimball

Alternative Authentication Mechanisms

Over the years alternative mechanisms have been developed to create AFS tokens on Windows. The following is a short list of some of the more popular methods.

Wake - http://www.rose-hulman.edu/TSC/software/wake

Wake is an excellent utility written by Bill Richardson at Rose-Hulman Institute of Technology in Terre Haute, Indiana. The Wake utility can provide solutions to just about any authentication need. Wake allows obtaining Kerberos credentials, viewing tickets, creating AFS tokens, mapping drives, and viewing Microsoft credentials. Wake also comes with a nice Kerberos authentication provider similar to the OpenAFS logon authenticator. The Wake user interface is shown in Figure 40.

Figure 40: The Wake User Interface

MSKlog - https://lists.openafs.org/pipermail/openafs-info/2004-January/011753.html

MSKlog is a program written by Douglas E. Engert (deengert@anl.gov) of Argonne National Laboratory. MSKlog is a solution to create an AFS token directly from the users Microsoft SSPI Kerberos credentials. As described by the MSKlog README…

> "AFS has evolved over the years, and recent capabilities have been added to OpenAFS to allow it to use Kerberos V5 tickets in the AFS token. The MSKLOG program takes advantage of this, and is designed to use as much of the Microsoft built in Kerberos code as possible. When run on a machine as part of a domain the User's login credentials can be used to obtain the AFS token.
>
> Unlike aklog or gssklog, no additional Kerberos libraries are needed on the client, and no additional daemons like krb524d or gssklogd are needed.
>
> The program relies only on the Microsoft DLLs and OpenAFS libraries. No changes are needed to the AFS servers. (The servers need to be at 1.2.8 at least.) So the program can easily be used in a Microsoft only environment, as well as a mixed environment."

GSSKlog - https://lists.openafs.org/pipermail/openafs-info/2002-April/004268.html

GSSKlog is a program written by Douglas E. Engert (deengert@anl.gov) of Argonne National Laboratory. GSSKlog is a solution to create and AFS token directly by performing a Generic Security Services authentication (GSS). As described by the GSSKlog README…

> "Obtain an AFS token, using a Generic Security Services (GSS) implementation for authentication thus giving the user and administrators greater flexibility in authenticating to an AFS cell. This can still be used in conjunction with current AFS authentication methods. This separates the dependency of AFS on using Kerberos V4 tickets to obtain a token.
>
> When used with Kerberos V5, it replaces the the aklog command, and does not require the use of the krb524d.
>
> No modifications to any AFS code or servers are required, but the AFS administrator will need to run an additional daemon on one or more of the AFS servers.
>
> A modified klog program uses the GSS to authenticate to a daemon process running on one or more of the AFS database servers. If authentication is accepted, and the user is found in a map file on the server, and AFS token is returned."

Kerberos for Windows - http://web.mit.edu/kerberos/www/dist/index.html

The Kerberos for Windows (KfW) toolset from M.I.T is an excellent package used for management of Kerberos tickets and AFS tokens. The KfW package comes with a number of useful command line tools. The KfW Leash GUI user interface is shown in Figure 41.

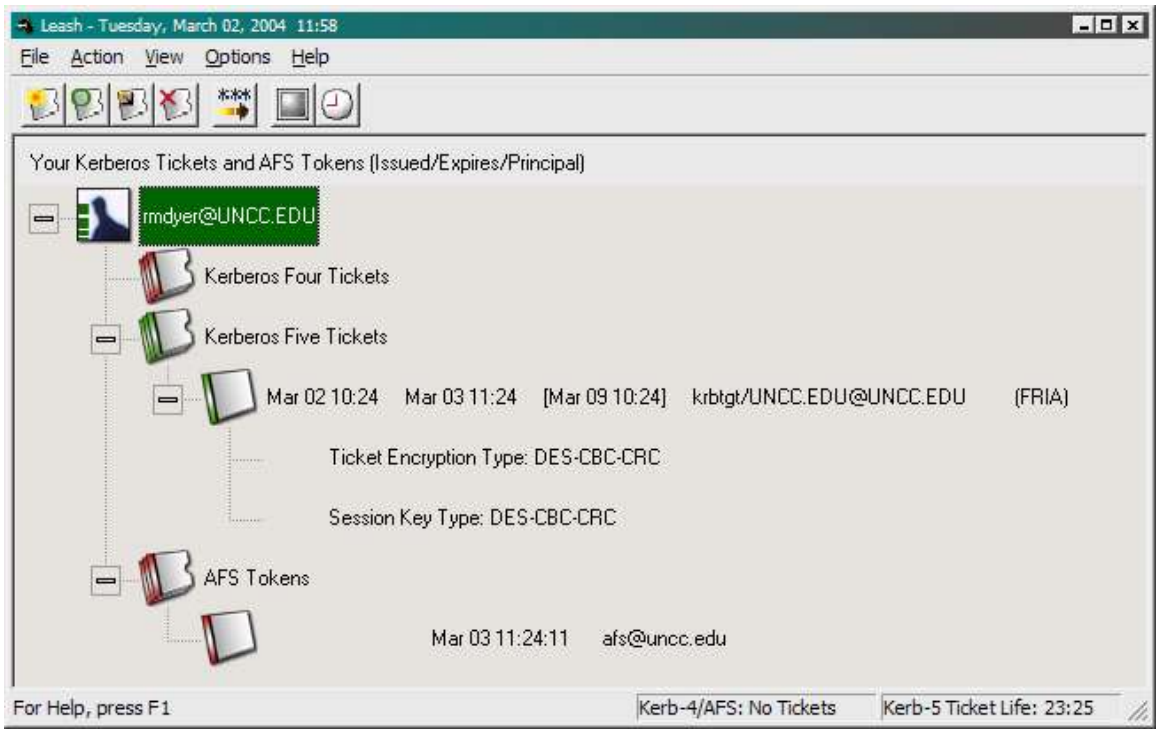


Figure 41: Leash User Interface

AFSLogonShell - http://www.coe.uncc.edu/~rmdyer/

The AFSLogonShell is a solution to provide scripting to the AFS logon process. Scripting can serve to enhance administrative capabilities before the users process space has been initialized, especially before profile download and folder redirection, as well as provide for alternate methods of authentication. Note: This should not to be considered a "best practice" it is more of a quick and dirty interim solution. As described in the AFSLogonShell document…

> "AFSLogonShell provides an augmentation of the traditional AFS client authentication process under the Windows operating system. AFSLogonShell allows an administrator to write their own child process in almost any compiled or scripted language that will be executed during the AFS authentication process at logon. The child shell process will run as the SYSTEM user and will have available to it the username and password of the user logging on to the system. The shell can also be made hidden so that the user never sees it. Many things can be accomplished in the child shell that may help administrators in their administration of Windows machines that are using the AFS file system.
>
> AFSLogonShell is specifically a single C function that can be applied to the C source file called "afslogon.c". The "afslogon.c" file is provided in the OpenAFS Windows client source code. The AFSLogonShell function was made to be added into the file so that it is statically linked in. No libraries or DLLs are required other than those already used by the AFS client."

Important Note: Because the SYSTEM account is shared during a logon of multiple users in this design, this solution doesn't currently work with Microsoft Terminal Services, or Citrix based systems. This solution was created with single user workstations in mind.

Figure 42 shows the modified AFS logon authentication execution sequence using the AFSLogonShell.

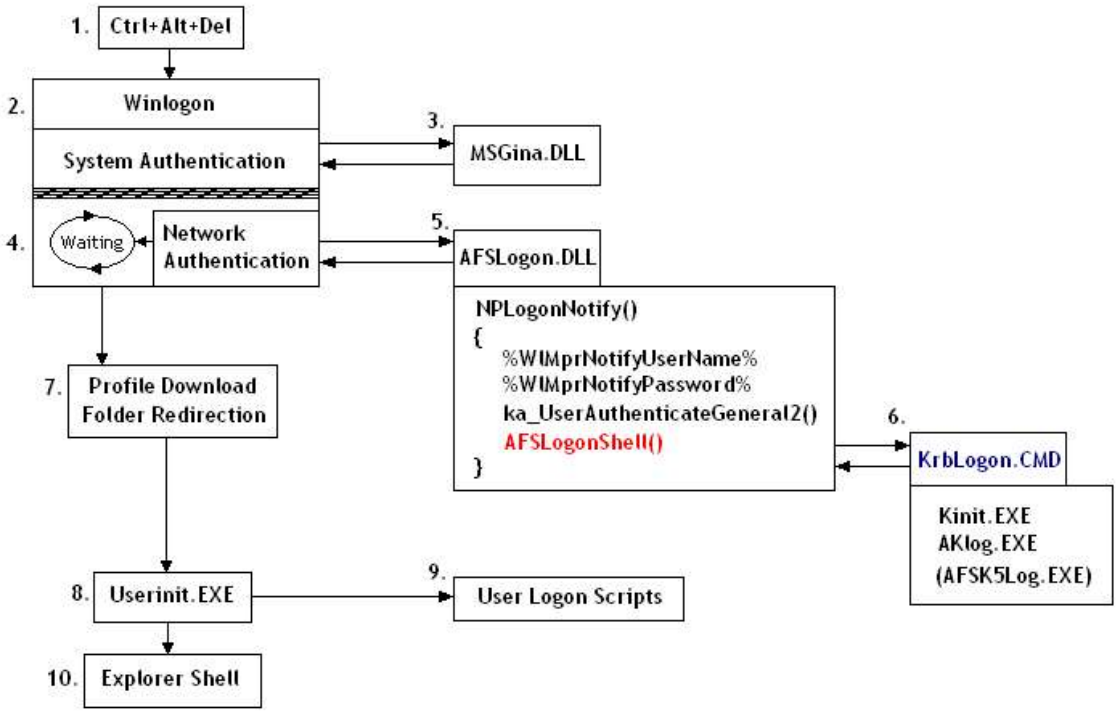


Figure 42: AFSLogonShell Scripting Process

AFSK5Log - http://www.coe.uncc.edu/~rmdyer/

The AFSK5Log program is an updated (modified) version of Ken Hornstein's AKLOG. This is an AKLOG written specifically for Microsoft Windows NT/2K/XP. The AFSK5Log program was written to be used with the AFSLogonShell scripting method so that an AFS user token can be generated from a Kerberos V5 credential during the logon process.

Positives of this version are...

* Written for Windows 2k/XP only.
* The original code has been cleaned and optimized extensively.
* Uses Microsoft's standard C libs.
* Maintained for the latest versions of OpenAFS and MIT Kerberos.
* Written as a pure Win32 console app.
* Added option flag -stlgtk to obtain a user logon token.
* Better logging.

Setup of Roaming Profiles and Folder Redirection

The question of whether AFS supports Microsoft Windows Roaming User Profiles, or RUPs, is asked quite often. The answer is definitely yes! However the method by which you implement roaming profiles on AFS has been a little vague. This section provides the missing details.

First what is a roaming user profile? A profile is simply a store of the user's Windows environment preferences. Some of these preferences include the desktop background colors and images, the user's application settings, window metrics, recent file lists, favorites, etc. The information in a profile is stored in a file system data archive under a single directory (folder) usually on the client PC in "c:\documents and settings". The profile folder name is the same as the user's logon account name. See Figure 43 for an example profile.

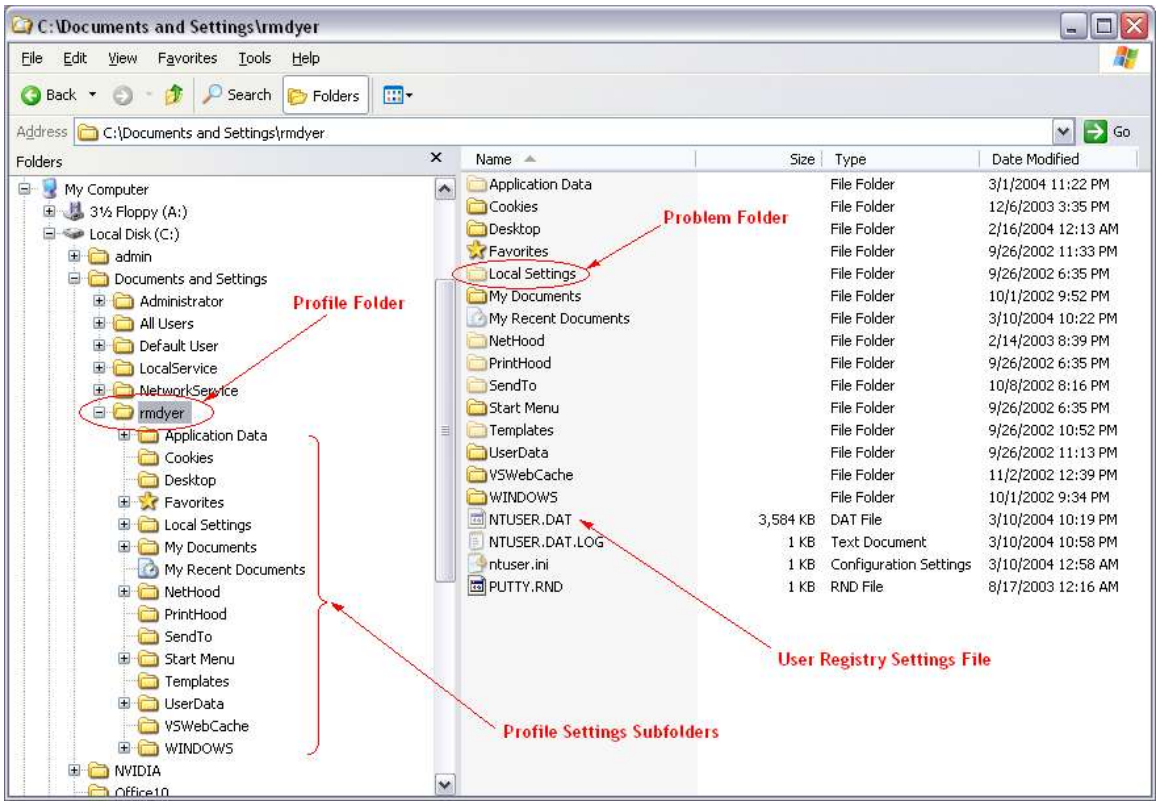


Figure 43: A Local User Profile

For most people who use and manage their own PCs the profile, and the information within it, is managed by

Windows and they are usually oblivious to it. If the user doesn't roam from machine to machine their local profile folder never moves.

For a user who does roam from machine to machine on the network it would be nice if their settings roamed with them. The method to accomplish this is simply to store the user's profile on a network file server. When the user logs onto a machine their profile is retrieved from the server and stored locally. When the user logs out, the profile information that changed during their session is synced back to the server profile folder.

For more information on profiles go to http://technet.microsoft.com and search for "roaming user profiles".

See also: http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/xpusrdat.mspx

Microsoft has made it easy to store roaming profiles on Microsoft server systems which serve a network file space via the SMB or CIFS protocol. With AFS however there are some obvious problems that creep in. Let's look at those problem areas one at a time.

1. First we need to decide where we are going to store the user profile in AFS. Most sites end up storing the profile in the user's AFS home directory under some name. At our site for example, when we were running NT 4.0 we stored the profile by the name "*~username*/.nt_profile". Placing a "." (dot) in front of the profile folder name will hide the folder from most Unix tools. An problem existed with the AFS Windows 2000 client because of placing a dot in front of the profile folder name, but that problem has been resolved. Because we were in the middle of a transition from Windows NT 4.0 to Windows XP at the time, we couldn't wait for the dot problem to be resolved so when we moved to XP we just changed our profile name to "*~username*/xp_profile". You can essentially name the profile folder by any valid Microsoft folder name. When the profile is copied locally when the user logs on, its local name will always be set to the user's account name.

2. Second we need to find some way of making the user's AFS home directory available to Windows when the user logs on. Here is the sticky part…the profile is downloaded by Windows "before" any network drives are mounted for the user. So we need to somehow mount AFS first so Windows can download the profile. The historical method to do this is to mount a global system drive. A global system drive is a network drive that is mounted by the SYSTEM account when the PC is booted. The global drive typically stays mounted continually until the PC is rebooted. The SYSTEM account is the built-in local Windows user which most services run as, including AFS. Mounting a global drive at boot allows access to AFS when the user logs on. The global drive is typically mounted to the root of AFS. The global drive was discussed previously. See the section "**Mounting Techniques**".

3. Third we need to find some way of getting the user authenticated to AFS at logon so that the profile can be downloaded from the user's AFS home directory. Remember that Windows can't just pull the user's profile without being authenticated. The user can be directly authenticated to the Windows OS itself, but not as easily to a third party network file system like AFS. Thankfully the creators of AFS saw fit to create an AFS logon authentication mechanism. This is made possible by the use of a Microsoft network provider authentication DLL. This was discussed previously. See the section "**How Logon Authentication Works".**

Ok, so now we've got a place to store the user's profile information in AFS. We've got a method of AFS network access at logon. And, we've got a method of authenticating the user to AFS at logon. All we need to do now is to specify to Windows that a user has a roaming profile and where that profile is at logon.

At our site, our global network drive was specified to be "N". The "N" stands for "network drive". Our account management process on our Microsoft Active Directory™ simply changes a user's "Profile Path" property to indicate where down the "N" drive tree the profile exists. This can be seen in Figure 44. This is basically done once when the account is created. Our account maintenance process basically reads our UNIX password file and creates an account when one by the same name is not found in the Active Directory.

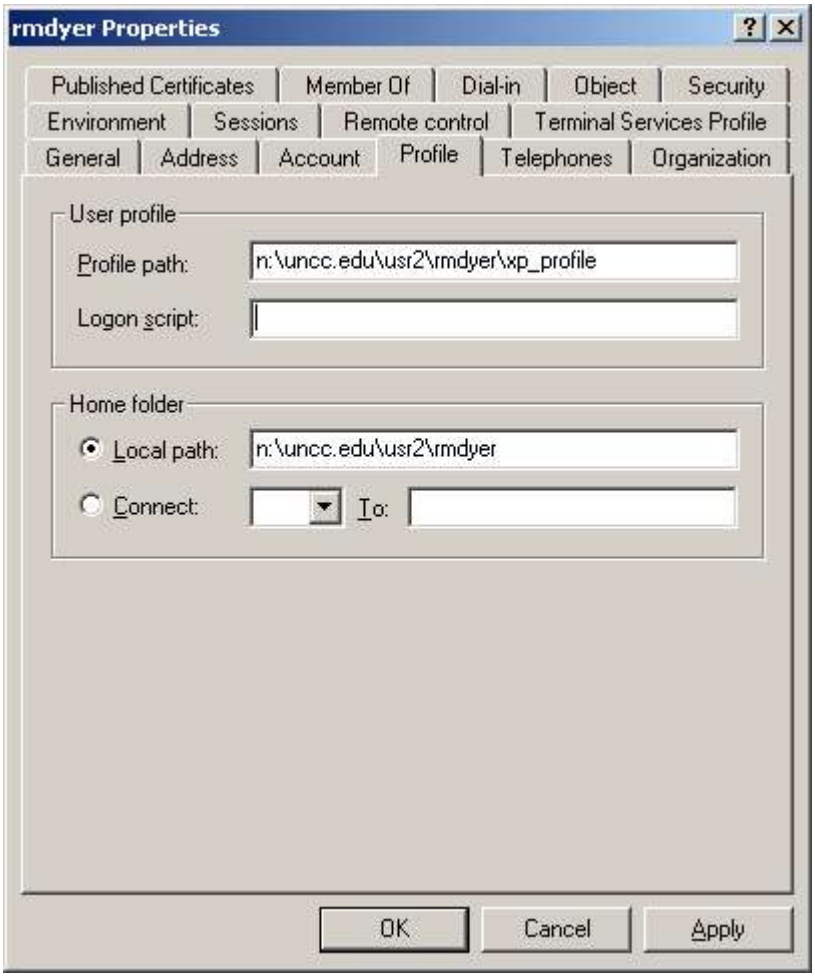


Figure 44: Active Directory User Account Profile Properties

Roaming Profile Problems

No good deed goes unpunished. You may think roaming profiles are nice but things can get ugly. Here is a short list of problems.

1. One problem you will have with roaming profiles is that users may stay logged on beyond their AFS token time (the token expires). Without a token, the profile can't be saved to AFS when the user logs out, this can cause settings to be lost if the user moves to another machine later. When using roaming profiles it is always important to create a policy/mechanism so that the user has a valid token at logout.

2. Windows default settings are to keep the local cached copy of the profile even when the user is logged out. We didn't like this, so at our site we set our Active Directory Group Policies so that the "local profile cache" is erased at logout.

"Delete cached copies of roaming profiles" **Enabled**

If you don't delete the cached copy of the user's profile at logout then you will end up with many profiles on many machines. The cache deletion doesn't always work anyway, sometimes local profile caches can become "stuck" on the PC. In our case we had to create a method to remove those profiles during our PC maintenance updates. Related to issue 1, if you lose your AFS token during your session, logging out of our system will cause your settings to be lost because of the cache deletion process.

3. The AFS client service has been known to crash, or tokens have been lost during the profile save at logout. When this happens, the state of the user's profile in AFS is suspect. Many times when this happens the user logon may stop working. The only thing that can be done is to pull a backup of the older profile, or delete the

bad one and let Windows regenerate a new one at the next logon.

4. The user's AFS account quota is too low or becomes too low at logout while the profile is being saved. This issue is the same as issue number 3, the profile may become corrupted.

5. Profiles don't work on XP Service Pack 1? Sure they do. Microsoft just changed the method they used to validate profile security with SP1. See the following for fixing this problem…

> "Windows XP SP1 checks permissions on an existing profile folder when a roaming profile is created?"
>
> http://www.jsifaq.com/SUBL/tip5700/rh5745.htm
>
> also,
>
> "Windows XP SP1 Checks for Existing Roaming User Profile Folders When a Roaming User Profile Is Created"
>
> http://support.microsoft.com/default.aspx?scid=KB;EN-US;q327462&

6. Roaming profiles are too big. The profile data storage scheme that Microsoft uses has changed between NT 4.0 and Windows 2000/XP. Under NT 4.0 everything, including temporary data, was stored in the profile. Many enterprise NT administrators cried foul because roaming profiles took way too long to download because of their size. In order to alleviate this problem Microsoft made changes that keep many folders from saving with the profile. In many cases this is exactly what was needed in others it just created trouble.

As you can see in Figure 43, a folder called "Local Settings" is circled as a "Problem Folder". The local settings folder is included in a list of folders to be excluded from saving when the user logs out. While this may seem like a good thing, a second look will reveal problems. The following is the default XP exclude list for roaming profiles (with Microsoft Office installed).

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Winlogon]
"ExcludeProfileDirs"="Local Settings;Temporary Internet Files;History;Temp;Local Settings\\Application Data\\Microsoft\\Outlook"
```

You can see that the "Local Settings" folder is excluded. But this very folder contains some valid settings that you may want to save. For example:

> A. If you want to use a JPEG as background wallpaper, XP converts the wallpaper to a BMP file and stores it in "Local Settings\Application Data\Microsoft\Wallpaper1.bmp" so when you logout and move to another machine, the bitmap won't be there. Microsoft has confessed to us through bug support that this is a problem. Watch for a fix in SP2.
>
> B. Microsoft Visual Studio 7.0 (.NET) stores some of the user's preferences under "Local Settings\Application Data\Microsoft\VisualStudio\7.0\VCComponents.dat". So, when the users logout and move to another machine, they won't be there. Don't know when this will be fixed.
>
> C. Microsoft Outlook Email Wizard creates the user's standard mailbox in "Local Settings\Application Data\Microsoft\Outlook\Outlook.pst". So, when the users logout and move to another machine, they won't be there. I can't see a fix for this one on the horizon.

The solutions around these roaming profile problems are varied and depend largely on your own IT groups policies.

Setup of Folder Redirection

Folder redirection allows some of the user's local profile folders to remain in AFS. One of the most important folders used with this feature is the desktop folder. Many users enjoy storing icons (shortcuts), folders, and files right on their desktops. Without folder redirection this data is stored put into the user's locally cached roaming profile. When the user logs out, the local cache is pushed back to AFS. If the user has stored many megabytes of data on their desktop, then the logon/logout processes may take a very long time as well as use up valuable network bandwidth. Folder redirection allows the desktop and some other folders to remain in AFS, never to be copied back and forth at logout and logon. The folders still appear to reside locally. Any files placed on the desktop for example get magically synced to AFS by Microsoft's IntelliMirror™ feature in real-time as they are placed into the folders.

For further folder redirection information see:
http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/xpusrdat.mspx

The location of the folder redirection setup within group policy is shown in Figure 45.

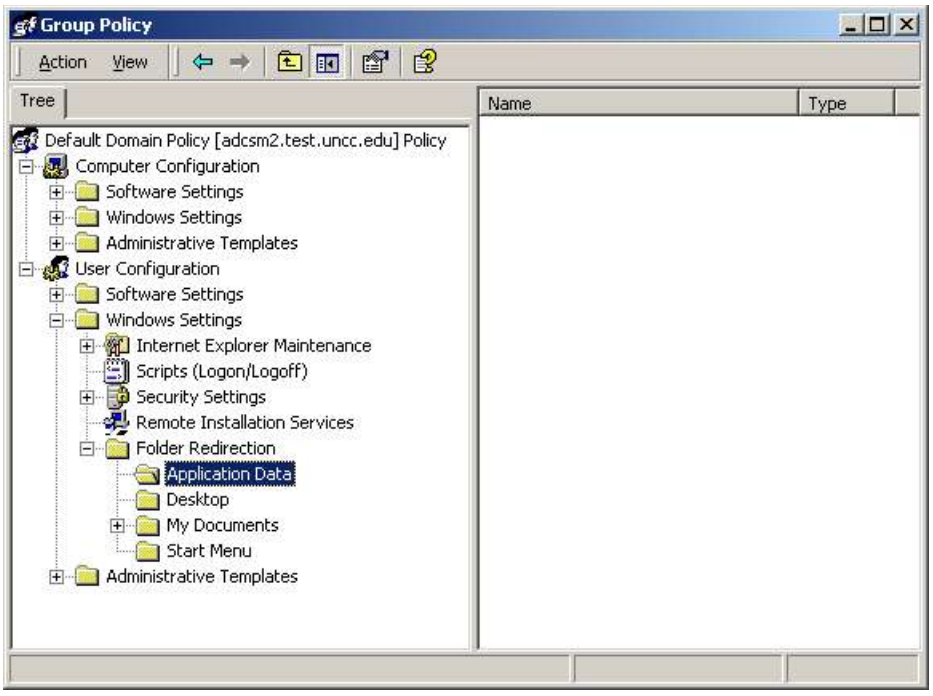


Figure 45: Group Policy Folder Redirection Configuration

The setup of folder redirection with AFS mirrors the setup of roaming profiles with AFS. You need a place to store the redirected folders, you must have access to AFS at logon, and you must be authenticated to AFS at logon. But here ends the easy part.

Folder redirection can be a bit more complicated than roaming profiles, not because of AFS, but because of the way AFS administrators setup their user home directory locations within AFS. Microsoft's group policy doesn't allow the use of certain environment variables in the path specification for the relocated folders.

You will note from the Microsoft Folder Redirection documentation:

> "**Folder Redirection and environment variables**
>
> The folder redirection client side extension is only able to process two environment variables: %username% and %userprofile%. Other environment variables such as % logonserver%, %homedrive% and %homepath% will not work with folder redirection."

Here we have the problem. The group policy folder redirection setup isn't per-user it's global. So to specify the path for the redirected folders you have to use something like…

"n:*cell*\users\%*username*%\redirected_folders"

But this method won't work at all sites because some AFS administrators have their AFS cell directory structures such that user home directories are spread over several main directories as in…

"n:*cell*\usr\a*username*"
"n:*cell*\usr\b*username*"
"n:*cell*\usr\c*username*"

This is typically done to eliminate having too many users under one volume. The problem cannot easily be solved. You have a situation where you need to use the %homepath% environment variable but can't. The % homepath% environment variable only points to the local user profile under "c:\documents and settings". This doesn't help at all.

Even if we could somehow specify the AFS subdirectory path to the redirected folder for the users, we are still left with how to specify the AFS mount syntax*. Using a UNC path doesn't currently work because the users will be on different systems when they logon. You can't currently specify "\\%computername%**-afs\all**" for example because the %computername% environment variable won't be expanded by group policy at logon. Figure 46 shows the configuration dialog for the Desktop folder redirection.

* Update: This will soon be remedied in a future 1.3 or 1.4 version of the AFS Windows client. See the subsection "Future Mounting Syntax", under section **Basic Mounting Syntax** for more information.

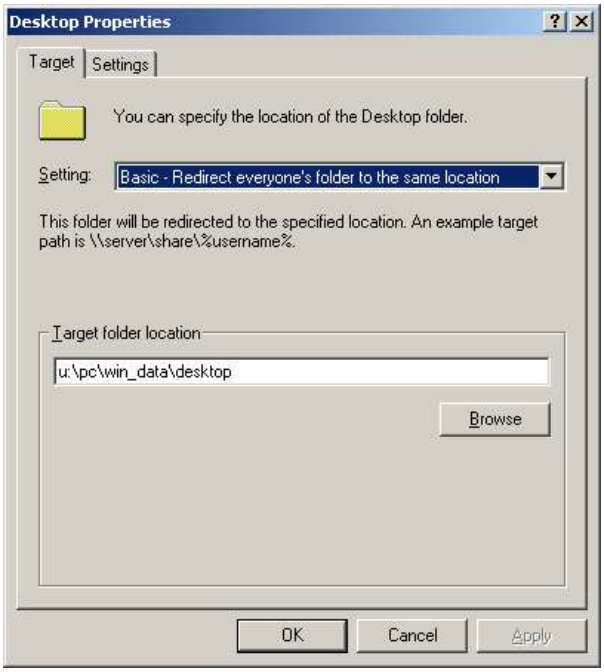


Figure 46: Group Policy Desktop Folder Redirection Dialog

At our site we were able to develop a short-term solution for the folder redirection problems by making use of the **AFSLogonShell** described in section **Alternative Authentication Mechanisms**. Using the AFSLogonShell we simply created an abstraction "U" drive that pointed to the user's AFS home directory during logon. As you can see in Figure 46, we then simply specify "u:\pc\win_data\desktop" as the user's redirected folder. Whether or not your site may take advantage of this method depends on many factors.

Setting AFS Server Preferences

If your AFS cell uses many replicated volumes spread across multiple file servers and you use AFS extensively from the Windows client it may be important for you to specify server preferences. Specifying server preferences will distribute the AFS network traffic between multiple systems. In effect you are increasing your network bandwidth by preventing too many clients from using the same file server for a replicated AFS volume.

When the AFS Windows client service starts it will set the file server preferences based on an internal algorithm.

See: http://www.openafs.org/pages/doc/AdminReference/auarf162.htm#HDRFS_SETSERVERPREFS

A paragraph from the above **fs setserverprefs** documentation states…

> "After assigning a base rank to a file server machine interface, the Cache Manager adds to it a number randomly chosen from the range 0 (zero) to 14. As an example, a file server machine interface in the same subnetwork as the local machine receives a base rank of 20,000, but the Cache Manager records the actual rank as an integer between 20,000 and 20,014. This process reduces the number of interfaces that have exactly the same rank."

The random value added to the base rank is all well intentioned, but the random number algorithm the AFS client uses doesn't quite generate a great server preferences rank list. Every time the server rank is calculated on service startup the rank list comes out the same on every system that has the same subnet. This is probably because it is using the same random number seed. Also, many of our machines tended to have multiple servers with that exact same rank.

At our site we decided to write a script to set our own server preferences based on our own algorithm. We decided to set base ranks depending on what building the client workstation was in. The clients in a building with file servers, or closest to a building with file servers (in router hops), would get the best starting ranks. Then, we created a list of 16 non-repeating random values, each value being a number in the range 0-15. We then multiplied each random value by 16 and added the result to the starting base rank for the workstation. This is done to override any value that the AFS random number generator will provide. We guarantee each starting base rank to be at least a multiple of 256 so that overlap doesn't occur.

So the server preferences algorithm becomes:

```
foreach intra_building_server ( x = 0 to 15 )
        server_pref(x) = starting_base_rank + afsrand* + ( random_list(x) * 16 )
next server
```

* You will note that because the afsrand variable (generated by the AFS client) takes on some random value of 0-15, this algorithm makes the afsrand value moot.

Since our random number algorithm for each workstation guarantees uniqueness, no two machines within a building will have the same server rank list. This distributes the client network load within the buildings.

Futures and Keeping the Faith

One can only hope that with more input from open source Windows devotees the OpenAFS Windows client has a strong future.

Some features for upcoming releases are:

> **Freelance Client** – This allows the AFS client to operate without needing a home cell. This would primarily be used by mobile notebook users. This feature already exists in the current production version of the AFS client 1.2.10, but it can only be enabled by manually editing the registry. For more information see http://grand.central.org/twiki/bin/view/AFSLore/FreelanceClient.
>
> **New Installer** – Rob Murawski from the University of Pittsburgh is working on a new installer for the OpenAFS 1.3.5x series. The current production version of OpenAFS uses the InstallShield installer software. The new installer is made possible by using a software package called the Nullsoft Scriptable Install System. For more information see http://nsis.sourceforge.net/site/index.php.
>
> **New Mounting Syntax** – The current mounting syntax requires the PC hostname together with "-afs" to reference the AFS client virtual SMB server. The syntax also needs the word 'all' to be added to reference the AFS root. The new syntax can only be used with a Loopback Adapter installed, but allows the AFS tree to be referenced by "**AFS***cell*\dir1\dir2" etc. This was discussed previously in **Basic Mounting Syntax**. This feature complements of M.I.T and Jeffrey Altman. Thanks Jeff!

Further Information

> **OpenAFS.ORG**
> http://www.openafs.org/
>
> **OpenAFS TWiki.AFSLore.WebIndex**
> http://grand.central.org/twiki/bin/view/AFSLore/WebIndex
>
> **The Integration of Kerberos V5, AFS, and Windows XP using the AFSLogonShell**
> Rodney Dyer, Mosaic Windows Systems Programmer
> http://www.coe.uncc.edu/~rmdyer/krblogon.htm
>
> **User Data and Settings Management for Windows XP in a Windows 2000 Environment**
> Microsoft Windows XP Web Site
> http://www.microsoft.com/technet/prodtechnol/winxppro/maintain/xpusrdat.mspx
>
> **AFS Consulting**
> Sine Nomine Associates, Inc.
> http://www.sinenomine.net/afs.php

Random Thoughts

* Global drives do not appear in, and cannot be changed by the afscreds system tray tool.
* Submounts must be mounted using the "\\machine-afs\share" format.
* Not a great idea to have all the AFS cells listed in the afsdcell.ini.
* Regular users can set server preferences.
* Silent install via /r /s doesn't fully work with InstallShield?
* File ownership cannot be determined under Windows???