OpenAFS for Windows Status Report: December 2006

1. Introduction

This document summarizes the improvements in OpenAFS for Windows (OAFW) since Secure Endpoints Inc. accepted responsibility for maintaining it in October 2003. It also outlines the remaining known issues and provides a roadmap for future development plans.

As of 1 December 2006, the most recent OpenAFS for Windows releases are maintenance build 1.4.2a and features build 1.5.12. These OpenAFS releases implement all of the standard AFS client functionality permitting access to data stored in AFS cells. OpenAFS provides Microsoft Windows users the benefits of a globally distributed location independent file system. Tight integration with the Microsoft Windows environment is obtained for client authentication during the login process as well as via extensions to the Windows Explorer Shell allowing graphical manipulation of AFS access control lists, volume mount points, symlinks, and object properties. The AFS Client Service and its associated tools are regularly stress tested and hundreds of thousands of copies are in use.

Both OpenAFS releases are supported on 32-bit versions of Microsoft Windows 2000, XP, 2003. In addition, OpenAFS 1.5.12 is supported on 64-bit Microsoft Windows XP, 2003, and all versions of Windows Vista.

OpenAFS is deployed on more than one hundred thousand desktops the world over. End users come from diverse communities including financial, manufacturing, medical, academic and research institutions.

Although much has been accomplished, there are still implementation weaknesses and architectural design choices in AFS that prevent a completely natural integration with Microsoft Windows. Native Microsoft Windows file systems implement a mandatory locking model based upon byte ranges, Unicode object names, multiple data streams, extended attributes and referrals to Microsoft's Distributed file system; features that are not supported in the current AFS implementation. In addition, the use of the CIFS-to-AFS gateway architecture instead of a native Windows File System Redirector imposes limitations on the performance and reliability of the AFS access when using the Windows AFS client.

The other components of OpenAFS for Windows product include administration tools and the AFS servers (file, volume, pts, bos). These components have received very little

developer attention. The AFS Server Management tool has been updated to work with Kerberos 5 but continues to be unstable due to thread safety issues. The AFS User Manager tool is not being worked on as it is a dedicated kaserver tool. In order for the User Manager to become a generic tool it would have to support a variety of Kerberos administration protocols including MIT Kerberos kadmin, Heimdal kadmin, and Microsoft Active Directory. As for the AFS Server processes, they run but are not being tested from release to release and should be considered experimental. The AFS Server installation wizards are known to be highly unstable. They assume the use of kaserver and have some serious thread safety issues that frequently result in unresponsive behavior. At the present time it is advised that users deploy AFS servers on MacOS X, Solaris, or Linux.

One of the primary concerns to Information Technology managers when selecting a file system is product longevity and timely support for future operating system releases. Secure Endpoints Inc. and the OpenAFS community have twice been challenged by major revisions in the Microsoft Windows operating system: Windows XP Service Pack 2[1] and Windows Vista[2]. New releases of OpenAFS for Windows supporting the new architectures were available within a day of the official Microsoft date.

Secure Endpoints Inc. has published a road map for the continued evolution of the OpenAFS for Windows client including a new user interface based upon the Network Identity Manager[3]. As OpenAFS is an open source effort, contributions from organizations that use OpenAFS are crucial to its continued improvement. Completion of the roadmap is dependent upon resource availability. The eventual goal is for AFS to be a first class file system for Microsoft Windows operating systems.

Beyond the OAFW focused development that will improve the AFS user experience, there are many changes to the AFS servers that must also be performed. These include the deployment of a new security class based on GSSAPI that will bring military grade authentication and data secrecy; an extension to the Protection Server database to allow the multiple authentication names to be associated with AFS Identifiers; server side support of byte range locking and the mandatory locking model; directory format changes to support Unicode object names and multiple data streams per file; and performance enhancements to the RX remote procedure call library. These changes once implemented will require client side support before they become useful.

With the continued support of the OpenAFS community, all of these projects will be successfully accomplished.

1.1 History

The history of the AFS client for Microsoft Windows operating systems leaves much to be desired. Even the most recent client from IBM Pittsburgh Labs, 3.6 2.55, is prone to

[1] https://lists.openafs.org/pipermail/openafs-announce/2004/000081.html
[2] http://www.openafs.org/openafs-vista-announce.html
[3] Network Identity Manager is an extensible multiple identity credential manager distributed as part of MIT Kerberos for Windows.

crashing; leaks memory and kernel objects; and is difficult to integrate into both the Windows operating system as well as authentication environments based on Kerberos 5.

On 31 Oct 2000, IBM released an open source version of their AFS for Windows product. While the OpenAFS community made substantial improvements to AFS on the UNIX platforms, the Windows product languished until November 2003. Organizations with substantial AFS deployments struggled with the question of how to support AFS on Microsoft Windows. Those that attempted to support Windows clients as first class AFS citizens were repeatedly burned. In response, many organizations have looked for an alternative distributed file system to migrate to although there are few choices available which provide the scalability and volume management capabilities of AFS.[4]

In November 2003, Secure Endpoints Inc. began a concerted effort to support OpenAFS for Windows and add new functionality. The goal has been to improve stability; performance; interoperability; end user transparency; ease of deployment; and integration with Kerberos 5 environments via use of MIT's Kerberos for Windows product. With each subsequent release since March 2004 the OpenAFS for Windows product has improved. As problems were experienced by end users, they were debugged and corrected in the subsequent release.

Testing of AFS clients and servers over the years has been ad-hoc in nature. Transarc Labs is rumored to have tested their new releases by running them in the production andrew.cmu.edu cell. If there were no problems reported by end users, everything must have been ok. The lack of a robust environment for stress testing and a reliance on end users to deploy new releases in order to test them caught up with the AFS community in 2004. Race conditions between clients and servers were discovered in production code which resulted in repeated downtime throughout the community as the performance of the hardware improved and the number of clients increased. Mobile clients and those behind network address translators[5] also resulted in serious performance degradation.

In December 2004, MIT's Information Services and Technology group began developing a test suite for use in stress testing the OpenAFS for Windows client. The 1.3.81 release was the first version of OpenAFS for Windows capable of passing the stress test. As the quality of the OpenAFS client has been improved the stress test has been improved. The stress test is now a standard part of the OpenAFS for Windows development process. The OpenAFS client is not only reliable and easy to deploy but its performance is comparable to its UNIX counterparts.

---

[4] One of the primary benefits of AFS is the minimal impact that occurs for end users during periods of server maintenance, load rebalancing, and system failure. Organizations that have attempted to migrate to other networked storage solutions have discovered that the outages for end users tend to be more frequent and of longer duration. AFS volumes can be moved from server to server while in use and can be restored to servers other than the original server in case of catastrophic outage. In addition, outage of a single file server cannot disable access to other resources in the AFS name space.

[5] The 1.4.1 OpenAFS file server was the first to provide support for multiple AFS clients using the same IP address. The 1.4.2 release is the first release to support transparent migration of mobile clients from IP address to IP address without delay.

On 1 November 2005, OpenAFS released version 1.4.0. OpenAFS.org celebrates its fifth anniversary.

On 16 Feb 2006, OpenAFS released version 1.5.0, the first development release to support 64-bit Microsoft Windows operating systems and CIFS support of byte range locking.

On 10 March 2006, OpenAFS released version 1.4.1

On 11 June 2006, OpenAFS released version 1.4.2-beta-1 and development version 1.5.2., the first releases to support Microsoft Windows Vista and Longhorn Server from the command prompt.

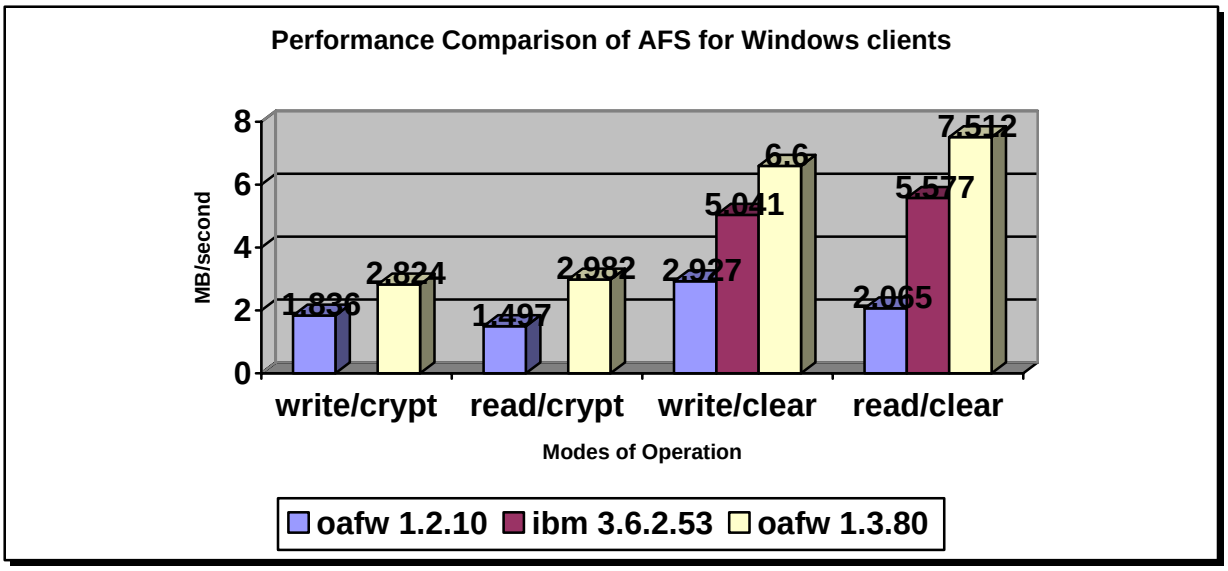
On 6 September 2006, OpenAFS released version 1.4.2-rc3 and development version 1.5.8. The 1.5.8 build implements the ability to fetch status data in bulk, can read and write files greater than 2GB, and updates the CIFS server interface to support the Microsoft Windows Vista Explorer Shell.

On 1 December 2006, OpenAFS released version 1.5.12, the first release to support all of the Microsoft Windows Vista versions and receive the "Works with Windows Vista".

2. Improvements in OpenAFS for Windows since 1.2.10

Version 1.4.0 was a milestone release for OpenAFS for Windows and things have only gotten better since. There have been more than 500 improvements since the August 2003 1.2.10 release. The majority are changes to the client affecting stability, performance and Windows integration. The details are available in the afs-changes-since-1.2.txt file available at http://www.openafs.org/dl/openafs/1.5.12/winnt/afs-changes-since-1.2.txt. This section will focus on some of the highlights.

The resulting performance improvements can be summarized in the following chart.



The percentage improvement of OpenAFS for Windows 1.3.80 over 1.2.10 are write/crypt mode: 54%; read/crypt mode: 125%; write/clear mode: 143%; read/clear mode: 263%. The percentage improvement of OpenAFS for Windows 1.3.80 over IBM AFS 3.6.2.53 are: write/clear mode: 31%; read/clear mode: 35%.

2.1. Resource Management

The OAFW 1.2 and earlier releases suffer from a large number of programming errors which adversely affect the stability and performance of the OpenAFS Client:

- use of un-initialized variables
- memory leaks due to reference count errors
- premature object destruction due to reference count errors
- kernel object leaks due to reference count and usage errors
- thread deadlocks due to improper lock management
- thread deadlocks due to a failure to wake up sleeping threads
- memory de-allocation errors
- queue management errors
- the number of allocated NetBIOS Control Blocks exceeded the number of objects Windows can monitor simultaneously resulting in NetBIOS operations which never complete
- race conditions between threads left data structures in invalid states
- the use of longjump() to restore the state of program registers is not safe to use in multi-threaded programs
- non-atomic combined pioctl/rpc operations produced race conditions between processes simultaneously setting or retrieving tokens from the AFS Client Service
- string table resources were improperly assigned index values resulting in memory corruption
- a failure to manage references to AFS RPC connections produced a “use once and discard” error which can overwhelm the file servers

2.2. SMB/CIFS improvements

- Extended SMB/CIFS messages were not supported resulting in file operations being rejected for a subset of files
- Authenticated connections between the Windows CIFS client and the OpenAFS SMB/CIFS Server are now negotiated using GSS SPNEGO. The actual authentication is performed using NTLM and the contents of the Windows logon cache.
- Unique virtual connection IDs are used for all connections
- Garbage collection of invalid or expired objects implemented
- Virtual Connection keep alive messages are periodically sent providing automatic detection of premature termination
- Virtual Connection termination forces cleanup of all open file handles and locks.
- Proper reporting of unsupported CIFS functions have been implemented
- Partial support for SMB/CIFS browsing has been added:
    - NETSHAREENUM
    - NETSHAREGETINFO
    - NETSERVERENUM2
    - NETSERVERGETINFO

- Force SMB/CIFS reconnects in the pioctl() library when Windows reports a downgrade attack error
- Properly follow soft symlinks
- Added support for hard symlinks
- Symlinks to \\AFS\all\path and \\AFS\path are now equivalent to /afs/path
- Directory Searches no longer produce invalid handle errors after 64K FindFirst operations
- The use of foo.exe.local files or directories as a means of redirecting the location from which DLLs are loaded is now supported
- File times are now reported entirely in UTC. This prevents problems with backup software when switching back and forth from daylight savings time.
- Short file names (8.3 notation) were being generated using an algorithm that would produce invalid file names.
- Dynamic priority adjustments based upon the age of the outstanding CIFS request being processed.
- Support for CIFS byte range lock requests

2.3. Cache Manager Improvements

- Directory Name Lookup Cache is now case-sensitive
- Cell name comparisons are now case-insensitive
- New algorithm for computing filename pattern matches
- Memory utilization is now fixed
- Callback management improved
- Call timeout management improved
- Fixed Root Stat Cache entry initialization
- AFS RPC (RX) connections are no longer used once and discarded
- Cached data both stat and buffers are stored across AFS Client Service sessions
- UUIDs are now used to identify the AFS Client to the AFS servers. The UUID is kept across AFS Client Service sessions
- The lists of ACL entries no longer become corrupted
- IP addresses are no longer obtained at service startup and used for the life of the AFS Client Service. IP addresses are now obtained as needed allows the AFS client to report the correct set of IP addresses to the AFS servers upon request
- All AFS RPC callback interfaces are implemented including CM Debugging
- The default cache size has been increased to 96MB. The maximum cache size is 1.2 GB.
- The default number of cache entries has been increased to 10,000.
- Volume and Bulk Callback revocations no longer deadlock (eventually causing the AFS Client Service to panic.)
- The logic used to determine if volumes are available, busy or offline has been fixed. Failover now works.
- The default @sys name list for 32-bit x86 systems is now “x86_win32 i386_w2k i386_nt40". The default for Itanium is "ia64_win64" and for AMD X86-64 “amd64_win64”.

- Multi-homed servers are now supported.
- Threading optimizations reduce the number of locks that must be held for many operations. This increases the ability to pipeline operations and in turn improve performance especially of write operations.
- A number of race conditions and object reference errors in the RX library have been fixed. These improve the stability of the program.
- The RX library has been optimized to reduce the number of global locks. This improves the ability of AFS to take full advantage of multiprocessor and hyper-threaded systems.
- Byte range locks are managed by the client. In 1.5.2, allocated locks are backed by full file locks obtained from the AFS file server. AFS file locks which cannot be renewed block access to the file until the file is closed.
- AFS File Server capabilities are now queried.
- Universal AFS Error codes are now supported.

2.4. Installation Packaging

Two open source installation options are supported: NSIS and WiX.

2.4.1. NSIS 2.0

- Rob Murawski implemented a new executable installer using the open source Nullsoft Scriptable Installer Framework 2.0
- Supports new installs, uninstalls and upgrades from previous releases
- Designed for interactive installations by individual users

2.4.2. WiX 2.0

- Asanka Herath implemented an MSI installer for OpenAFS utilizing the open source WiX installation builder
- Designed for automated installation via Windows Group Policy but may be used interactively as well
- Supports new installs and uninstalls
- The OpenAFS.org distributed MSI can be customized by the use of MSI Transforms for use by all organizations without requiring the ability to build OpenAFS for Windows from source

2.4.3. Digital Signatures

- All binary files and installers distributed by OpenAFS.org are digitally signed by “Secure Endpoints Inc.” using a code signing certificate issued by Thawte.
- All digitally signed files are timestamped by the Verisign Timestamping Server
- Digital signatures may be used to ensure that installed files have not been replaced or modified

2.5. Improved Build System and Development Tool Compatibility

- The latest Microsoft Development Tools are now supported
  - Visual Studio .NET 2003

  - Visual Studio .NET 2005 (aka Visual Studio 8)
- Windows XP SP2 SDK or 2003 SP SDK required
- Only Windows 2000 and above. Windows 9x no longer supported

2.6. Improved Windows Integration

2.6.1. UNC handling

- UNC paths of the form \\afs\cellname are now supported when the Microsoft Loopback adapter is installed
- The "NetbiosName" registry value can be used to specify alternatives to "afs"
- No longer need to use \\afs\all\cellname

2.6.2 Byte Range Locking

- All versions of AFS for Windows prior to 1.4.1 and 1.5.0 would grant a lock to the requestor whether or not such a lock could be obtained.
- OAFW 1.4.1 and later locally manage lock allocations but do not back the allocations with AFS file server locks.
- OAFW 1.5.0 and later locally manage lock allocations but only grant locks that are backed by AFS file server locks unless the user access is no better than "rl" or the volume is read-only.

2.6.3. Integrated Login

- Kerberos 5 is used to obtain tokens
- All uppercase user names authentication attempts are retried using all lowercase upon failure
- Closed security hole which leaked plaintext passwords on the wire
- WinLogon Event Notification handler added to destroy AFS tokens at logout
- Domain specific configuration is now supported
- The "TheseCells" registry key enables the retrieval of AFS tokens for multiple cells.
- Timeout processing has been fixed. A request to 'retry' by the end user will now wait for a full timeout period.
- If the service is in the START_PENDING state, login will not timeout until the state changes.
- Kerberos 5 tickets obtained during the login process are now preserved and passed into the user's logon session for storage in the user CCAPI credential cache.

2.6.4. Windows Explorer Shell

- AFS UNC paths now supported in the Explorer
- Browsing of the "AFS" Server is now supported (limited to 13 character names)
- The AFS Context Sensitive Popup Menu works on all files and directories located within the AFS name space

- When the AFS Client Service is disabled, the AFS Shell Extension is dynamically disabled to prevent performance delays

2.6.5. Microsoft Loopback Adapter

- Installed by both OpenAFS.org installers
- Provides a locally visible adapter to bind the SMB/CIFS Service Name
- Provides an adapter for the AFS Client Service to bind to when network connectivity is not available
- Prevents the AFS Client Service from halting due to dynamic reconfiguration of plug and play network devices

2.6.6. Select LAN Adapter By Name

- The display name of the LAN Adapters can be used as a means of specifying which LAN adapter should be used by the AFS Client Service.
- Simply name the desired LAN Adapter "AFS"
- This functionality may be disabled using the "NoFindLanaByName" registry value

2.6.7. Power Management

- Receipt of Standby, Suspend or Shutdown notifications by the AFS Client Service force all dirty buffers in the cache to be written back to the server.

2.6.8. VPN Compatibility

- OpenAFS for Windows was found to be incompatible with the Cisco IPSec VPN client.
- In order for AFS RPC requests to pass through the Cisco IPSec VPN, the maximum size of Rx packets must be kept no larger than 1292 bytes.
- The OpenAFS.org installers sets the new "RxMaxMTU" registry value to 1260 to provide compatibility

2.6.9. Profile Data

- HKLM\Software\OpenAFS\Client key used to set system default values
- HKCU\Software\OpenAFS\Client key used to store user configuration data
- Used for:
    - Token Expiration Reminders
    - Use of MIT Kerberos for Windows for Kerberos 5
    - Use of the Kerberos 5 to Kerberos 4 translation service
    - Show Tray Icon (afscreds.exe auto start)
    - afscreds.exe shortcut parameters
    - Freelance data (mount points and symlinks)
    - Submount entries
    - Drive mappings
    - Default Authentication Cell
    - Windows' SMB Client Side Caching configuration

2.6.10. Terminal Server and Citrix Compatibility

- All configuration files have been removed from the %WINDIR% directory
- All user configuration data is now stored in the per-user registry allowing for multiple users and user instances
- SMB/CIFS sessions are authenticated to the logon session thereby removing the need for the random SMB Names utilized by "High Security" mode to enforce token ownership separation
- Corrected detection of the current logon user name
- Power Management support is multi-user safe

2.6.11. Windows XP SP2, 2003 SP1, 2003 R2, and Vista Support

- Sets a magic registry value to permit the use of SMB/CIFS Service Names which do not match the local hostname
- Sets a magic registry value to allow GSS SPNEGO authentication over a loopback connection
- Communicates with the Windows Integrated Firewall to dynamically open the ports used for callback messages.
- DLLs no longer initialize the AFS RPC library in DllMain entry-point which caused the Integrated Logon support code to block the successful startup of the Microsoft Windows operating system.
- OpenAFS can now be used in multi-domain Windows forests when users log in with a Kerberos 5 principal from a non-Windows realm. (Roaming profiles cannot be used in such a configuration due to bugs in Windows XP.)
- Experimental Support for Microsoft Vista and Longhorn Server as of OAFW 1.5.8 and Vista RC1.

2.7. Kerberos 5 Support

- Integrates with MIT Kerberos for Windows 2.6.5 and greater. KFW 3.0 ships with the Network Identity Manager. An AFS plug-in for NetIDMgr is available from Secure Endpoints Inc.
- Obtains tokens using Kerberos 5 (Kerberos 5 to Kerberos 4 conversion is not used by default)
- Imports credentials from both the MSLSA and CCAPI credential caches
- The AFS System Tray tool (afscreds.exe) automatically renews tokens and tickets as they approach expiration
- Tokens can be obtained for multiple cells from a single Kerberos 5 TGT (limited User Interface functionality)
- Can be disabled either per machine or per user (no UI)
- Maximum token size increased to 12,000 bytes to allow for large Kerberos 5 tickets issued by Windows 2003 Active Directory

2.8. DNS AFSDB Support

- Configuration information for cells not specified in the CellServDB file may be discovered via DNS

- Use of DNS AFSDB records is enabled by default
- DNS support extended to all operations which referenced the CellServDB file
- Windows DNS Query API now used instead of home grown implementation
- Controlled by “UseDNS” registry value
- AFS Server records obtained via DNS AFSDB are valid for the DNS AFSDB time-to-live period.

2.9. Dynamic Root Volume (Freelance Mode)

- On by default
- Fixed initialization code
- Added support for read-write mount points
- Added support for symlinks
- Stores locally defined mount points and symlinks in the Registry
- Configurable via “FreelanceClient” registry value
- Timestamp from most recent update to mount point data used for all mount point stat entries
- Algorithm used for detecting the fake root.afs volume replaced to avoid conflicts with existing deployed cells

2.10. Hidden Dot Files

- Following Unix tradition, files/directories whose names begin with a period are given the Hidden attribute when the “HideDotFiles” registry value is set
- This functionality is enabled by default

2.11. Logging Changes

- afsd_init.log and afsd.log moved to the %TEMP% directory (usually %WINDIR%\TEMP for the SYSTEM account)
- Stack Trace data logged to afsd_init.log during assertion failure or unhandled exceptions
- The maximum size of the afsd_init.log file is now restricted to either 100Kb or the value specified by the “MaxLogSize” registry value
- “fs trace” logging defaults to off for release builds and on for debug builds.
- “fs trace” logging can be configured to write to the Debug Output stream via a registry setting.
- “fs trace” logging also controls logging of AFS RPC debug output to the Debug Output stream.

2.12. Graphical User Interface Tools

2.12.1. AFS System Tray Tool (afscreds.exe)

- No longer requires administrator account to operate
- Uses Kerberos 5 instead of kauth to obtain tokens when MIT Kerberos for Windows is available
- Automatically renews Kerberos 5 tickets and derived tokens prior to expiration

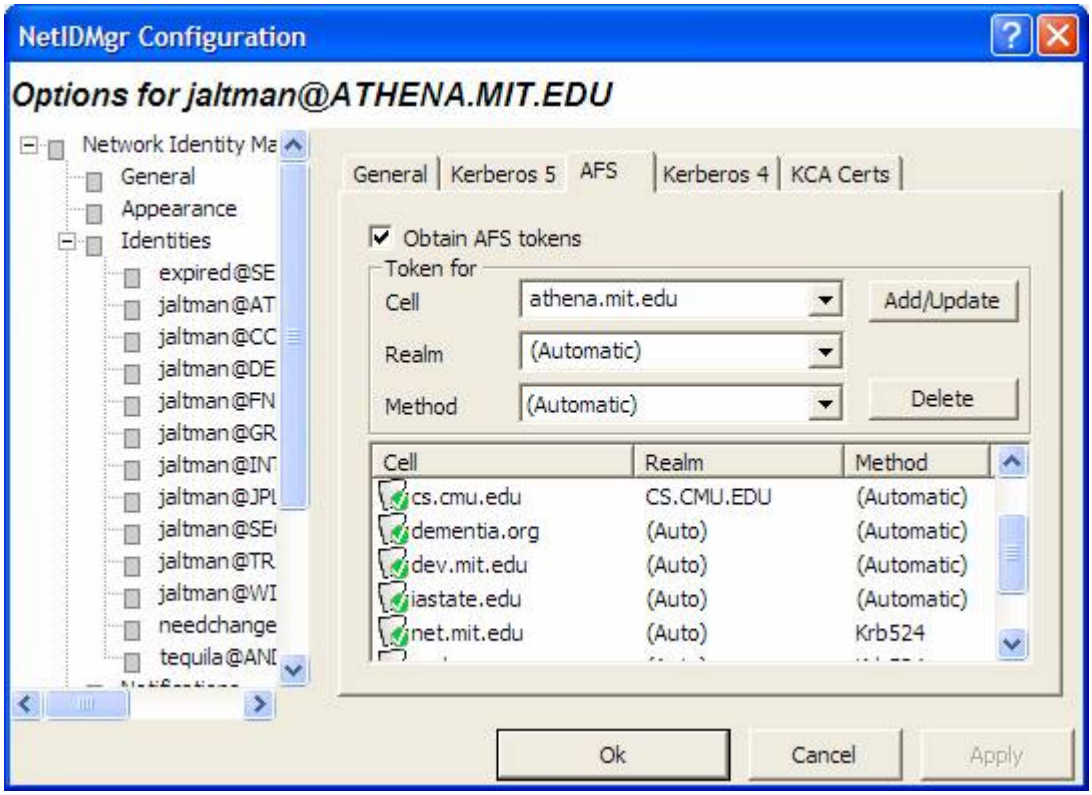
- Supports multiple Kerberos 5 principals
- A single Kerberos 5 principal may be used to obtain tokens for multiple cells without repeated password entry (including environments which require Kerberos 5 cross realm authentication)
- Monitors network connectivity by watching for IP address changes and probing the KDC. If network access is restored and the AFS Client Service has not been started, the service will be started. If network access is restored and the user does not possess any valid tokens, the user will be prompted to obtain tokens.
- Drive mappings are restored upon startup
- PTS registration of new users to foreign cells has been added

2.12.2. AFS Control Panel Tool (afs_config.exe)

- Fields which modify the AFS Client Configuration now require logon account membership in the Windows "AFS Client Admin" group.

2.12.3. Network Identity Manager Plug-in

The Network Identity Manager replaces the former KFW ticket manager, Leash", and when combined with the OpenAFS plug-in is intended to be used as a replacement for the AFS System Tray Tool (afscreds.exe). Unlike both Leash and the AFS System Tray Tool, Network Identity Manager with the OpenAFS plug-in can easily manage AFS tokens for multiple cells from one or more Kerberos 5 identities.



2.13. Command Line Tools

- aklog.exe has been added
- afsshare.exe modified to support the new registry entries
- several coding errors which would result in crashes fixed in fs.exe and vos.exe
- **fs cscpolicy** command added
- UNC paths supported in fs commands
- cmdebug.exe has been added
- afsdacl.exe has been added
- "vos.exe listvol –format" is now supported
- **fs uuid** command (1.5.8) allows the AFS client UUID to be changed on the fly

2.14. Debugging Tools

- **cmdebug**
- **fs minidump** command added to generate Debugging Mini Dump files without requiring a debugger.[6]
- **fs trace**
- OutputDebugString
- Debug Symbols

[6] On Windows 2000, the Microsoft Debugging Tools for Windows must be installed on the machine in order for this command to work. On Windows XP and above, the required debugging APIs ship as part of the operating system.

3. Mobile Client and Network Address Translation Support

AFS was originally designed for a world in which clients were assigned unique IP addresses which did not change over time. The addition of the *WhoAreYou* and *TellMeAboutYourself* Cache Manager RPCs combined with the assignment of Universally Unique Identifiers (UUIDs) to each AFS client in AFS3 were meant to separate the identity of the client from its address. Unfortunately, the AFS file server implementation was very much incomplete and AFS file servers and clients continued to experience performance problems associated with cache manager callbacks that other file systems such as CIFS and NFS did not.[7]

In order to improve the response time of repetitive read operations, AFS unlike other network file systems employs a client side cache. When a client reads directory information (performs a *FetchStatus* RPC), a callback is registered with the file server. When directory contents change, the file server contacts the registered clients to notify them that the affected contents of the cache must be invalidated. Performance problems occur when the *Callback* RPCs cannot be successfully completed.

The performance problems can be categorized as follows:

- AFS volumes could not be released while outstanding callbacks could not be delivered to registered clients. This was fixed in OpenAFS 1.4.0 by a change to the file server. Instead of blocking the release of a volume until all callbacks could be delivered, the clients with undelivered callbacks are flagged. The next time the client is heard from, the callback is delivered before any other file server requests can successfully complete.
- NATs permit multiple clients to contact the file server from the same IP address. The file server only tracked clients by IP address and ignored the port number. Although the file server was able to distinguish two clients by their UUID, the file server was unable to maintain state information for more than one client. This prevented the AFS clients from being able to properly maintain the contents of their caches. This was fixed in OpenAFS 1.4.1 by tracking clients by both their IP address and port number.
- AFS clients which migrate to a new IP address or port[8] do not receive callbacks from the file server and do not notice directory changes until either the directory status expires or the client attempts to modify the contents of the directory. This prevents the file server from being able to contact the client. As of OpenAFS 1.4.1, the Windows client pings the file servers once every ten minutes in order ensure that directory changes will be detected.

[7] CIFS and NFS do not use a callback mechanism to enable the file server to notify the client of directory and file modifications.

[8] NATs allow multiple machines to share one public IP address. They do so by mapping external port numbers to internal IP addresses and ports. These mappings are transient and frequently have a lifetime of less than five minutes. If no outbound communications take place using the port mapping, the mapping is removed. Subsequent outbound communications result in a new port mapping being assigned.

- AFS client which have migrated to a new IP address or port and contact the file server after a callback failure has occurred have experienced 56 second delays while the AFS file server attempts to contact the client on its old address. This was fixed in OpenAFS 1.4.2 by replacing the callback connection as a side effect of migration detection.

As of OpenAFS 1.4.2, client mobility and network address translation is no longer a concern for AFS users.

4. Quality Assurance

The quality of OpenAFS for Windows releases in ensured through stress testing, examination of crash reports collected by Microsoft's *Windows Quality Online Service* and End User testing and bug reporting.

4.1 The Stress Test

The OpenAFS for Windows stress test developed by MIT's Information Services and Technology (IS&T) group is modeled on the Samba Team's SMB torture test. The test engine allows for scripted operations to be performed against an SMB server by a client. A set of test files is provided along with scripted operations which are designed to simulate the behavior of a large number of popular Windows applications including Microsoft Office, the Paradox database, Corel Draw, and many others.

The test engine allows for multiple client processes to be started. Each process in turn can be configured to execute separate instances of the script in independent threads. The processes can be started with a specified delay between each one to ensure that the widest range of simultaneous operations are being performed against the SMB server at a time

This test engine is useful for testing the OpenAFS for Windows client because the AFS Client Service is essentially an SMB to AFS gateway. Sitting between an SMB server and the AFS servers is an AFS cache manager. SMB operations are mapped to cache manager operations which in turn result in AFS remote procedure calls being issued against the AFS servers.

IS&T designed the test engine to integrate with MIT's AFS locker and Moira administration tools. In addition, they implemented reporting functionality that allows AFS trace log output to be triggered when errors are detected during test runs.

IS&T executes the stress test on a variety of platforms including Windows 2000 workstation, Windows XP workstation, Windows 2003 server, Windows 2000 Citrix Terminal Server. The Windows 2003 Server and 2000 Citrix Terminal Server machines are dual-processor systems. One of the Windows XP workstations is a dual-processor hyper-threaded system providing the equivalent of a 4-way system.

A typical run on the 2003 Server would utilize ten processes each with ten threads of operation against a volume which is released (taken offline) every 15 minutes and cloned once an hour. The ability to execute 100 simultaneous threads each performing simultaneous operations is an important milestone because the Windows architecture limits the number of simultaneous SMB operations to 63. The fact that 50% more operations could be successfully queued and processed without data loss is a major achievement.

On the Windows 2000 Citrix Terminal Server the typical test scenario would include starting three test processes per user session with each process reading/writing to a

different volume. Like the previous tests, the servers were configured to release the volumes every fifteen minutes.

All clients previous to 1.3.81 would at some point during the testing reach a deadlock condition or trigger a reference count assertion. It took four months of testing to shakeout the entire set of known deadlock and assert conditions. The end result is a fast and robust client. As time goes on additional testing will be performed to ensure that new errors are not introduced. This statement is not meant to indicate that OpenAFS for Windows is bug free. No software is. The claim is simply that OpenAFS for Windows is significantly more robust than any prior AFS client and it can be trusted to work without issues under all known circumstances.

4.2 Windows Quality Online Service (Crash Reporting)

As of Microsoft Windows XP, Microsoft has begun to automatically collect mini-dumps of processes and device drivers that crash. These dumps are then provided to the application author to assist in improving the quality of the application. Secure Endpoints Inc. is registered with Microsoft and receives all of the crash reports. Crash reports for resolved problems or from non-current AFS releases result in the user being directed to a page at the Secure Endpoints Inc. web site specifying where the user can obtain the necessary corrective action.

4.3 End User Testing and Bug Reports

As with any software product, OAFW does have its fill of errors. End user testing and reporting of discovered bugs is a critical requirement for bug fixing.

5. Known Issues

The 1.5.12 release of OpenAFS for Windows is a stable and functional AFS client which provides 32-bit and 64-bit Windows 2000, XP, 2003, and Vista systems access to the AFS global file system space. However, there remain a number of deficiencies:

- The User Interface is sorely lacking
  - Poor separation of functionality between tools. On Windows Vista, the appropriate separation of administrative and non-administrative functions is required in order to support the User Account Control[9] model.
  - No configuration for integrated logon options
  - No configuration for all of the new functionality since 1.2
- Changes to the local machine's network configuration cause the AFS Client Service to panic if the Microsoft Loopback Adapter is not installed and enabled.[10]
- SMB/CIFS implementation deficiencies:
  - Restricted to 8-bit OEM Code Pages
  - Path names restricted to 256 characters
  - Share names restricted to 13 characters
  - Authentication restricted to 8-bit OEM Code Pages
  - No support for Microsoft Dfs Referrals
  - Remote Administration Protocol is incomplete
  - Per message integrity protection and authentation (digital signatures) is not supported
- No file permission integration with Windows Security model
- Performance can be improved by replacing the SMB/CIFS gateway server with a native File System driver running in kernel mode.
- No disconnected mode functionality similar to the Windows SMB/CIFS Client Side Caching (aka Offline Folders)
- The strength of data confidentiality and integrity protection provided for use by AFS RPC calls leaves much to be desired. (fcrypt is a weakened version of DES.)
- DOS Attributes such as Hidden and System cannot be associated with files stored in AFS
- Extended Attributes associated with files stored in AFS are lost
- Multiple data streams are not supported. Streams are increasingly used by Microsoft and third parties to store meta-data associated with the file. This meta-data is used to enhance search capabilities and enforce security boundaries.
- Drive mapping within the AFS System Tray tool requires "AFS Client Admin" group membership[11]

[9] http://www.microsoft.com/technet/windowsvista/library/0d75f774-8514-4c9e-ac08-4c21f5c6c2d9.mspx

[10] The MLA is installed by default and is necessary for portable \\AFS UNC names, this is rarely an issue. Some organizations deploy remote administration tools that are dependent on workstation reporting of the IP address and which are incapable of filtering out loopback devices. At these organizations, use of the MLA is often discouraged even though the proper fix would be to correct the behavior of the administration tools.

[11] Drive mapping is best performed via the Explorer Shell.

- An audit of the UNIX and Windows code needs to be performed in order to determine the full list of features supported by the UNIX AFS Client that are not implemented in Windows.

6. Future Implementation Roadmap

At the 2004 AFS Best Practices Workshop held at Stanford Linear Accelerator Center in March 2004 it was the consensus of the attendees that the future growth of AFS was dependent upon the availability of a Windows client which is secure, robust, fast, and transparent to the end users. With 95% of the world's desktops running Microsoft Windows, if the Windows clients are not robust then supporting end users will be a nightmare; if the Windows clients are slow and inefficient then the load on the AFS servers will be too high; if the Windows clients do not integrate transparently with the operating system then the users will become frustrated and will use something else to do their job; if the client is not secure then organizations with data confidentiality and integrity requirements can not deploy AFS.

The following is a prioritized list of Windows specific projects that Secure Endpoints Inc. believes must be implemented to meet the needs of the community. The goal is to complete these projects over a two year period:

- Design and implement new user interface tools:
  - New AFS Control Panel tool for end-user configuration including GUI equivalents for all command line tools that do not require a file path or a volume to complete.
  - New AFS Administration tool for AFS Client Service configuration based upon the Microsoft Management Console.
- Implement SMB/CIFS Unicode support. This will remove character-set and name length restrictions. Implementation is a pre-requisite for completing the SMB/CIFS Remote Administration Protocol.
- Implement Windows DFS referrals.
- Implement support for multiple data streams per file.
- Finish implementing the SMB/CIFS Remote Administration Protocol which provides integration with the Windows Explorer Shell.
- Design and implement a Disconnected Mode (Off-line folder) functionality [possibly covered by U.S. Patent 6,125,388]
- Design and implement a mechanism for the storage of DOS and Extended Attributes within the AFS file system. (model on OS/2 FAT extensions if multiple data streams have not been implemented.)
- Design and Implement a native File System driver.
- Integrate the AFS Client with Windows File Access Security model

The following is a list of platform generic OpenAFS projects which must be accomplished:

- Design and implement strong data confidentiality and integrity protection based on GSS-API Kerberos 5 mechanism. (rxgk) [prototype 1st Qtr 2007]
- Directory format extensions to support Unicode filenames. http://www.afsig.se/snipsnap/space/AFS+directory+format+extensions.
- Protection Server database extensions and RPCs to support multiple Security Class Authentication Name to AFS ID associations

- Directory format extensions to support multiple data streams.
- Mandatory file locking and byte range locks within the AFS file server.
- Design and implement an AFS RPC mechanism for use on TCP (and other stream based) connections
- Design and implement support for IPv6

Estimated completion dates for these projects are highly dependent upon resource availability.

7. OpenAFS for Windows Needs Your Support

The future of AFS is dependent on a successful future for the Windows client. Without a first class client that seamlessly integrates with Windows, the pressure to migrate to a Microsoft based technologies will be overwhelming to many IT support organizations. In order for OpenAFS.org to meet the needs of the community the projects listed in the road map must be completed. As an open source project, it is the contributions of the community that determine what will or will not be achieved. As the efforts of the last year have demonstrated, great things can happen when the community comes together to provide the necessary resources.

7.1. Financial Contributions

Although financial contributions are not the only way to support the development of OpenAFS for Windows, they are the most useful. Financial contributions can be made in a variety of ways.

7.1.1. Secure Endpoints Inc.

Secure Endpoints Inc. provides development and support services for OpenAFS for Windows and MIT Kerberos for Windows. Secure Endpoints Inc. is owned by the Windows Gatekeeper for OpenAFS. Donations provided to Secure Endpoints Inc. for the development of OpenAFS are used to cover the OpenAFS gatekeeper responsibilities and providing support to the OpenAFS community via the OpenAFS mailing lists.

Secure Endpoints Inc. accepts software development agreements from organizations who wish to fund a well-defined set of bug fixes or new features.

Secure Endpoints Inc. provides contract based support for the OpenAFS for Windows and the MIT Kerberos for Windows products.

7.1.2. The USENIX OpenAFS Fund

USENIX, a 501c3 non-profit corporation, has formed the USENIX OpenAFS Fund in order to accept tax deductible donations on behalf of the OpenAFS Elders. The donated funds will be allocated by the OpenAFS Elders to fund OpenAFS development, documentation, project management, and maintaining openafs.org.
Donations can be made by sending a check, drawn on a U.S. bank, made out to the USENIX OpenAFS Fund to:

USENIX OpenAFS Fund
USENIX Association
2560 Ninth St., Suite 215
Berkeley, CA 94710

or by making a donation online.

7.2. Direct contributions of code and/or documentation

Organizations that use OpenAFS in house and have development staffs are encouraged to contribute any code modifications they make to OpenAFS.org via openafs-bugs@openafs.org. Contributions of documentation are highly desired.

Interested parties should contact the OpenAFS Gatekeepers at openafs-gatekeepers@openafs.org. Architectural designs should be discussed on the OpenAFS for Windows Development mailing list: openafs-win32-devel@openafs.org.