Samba, CIFS and Linux Network Filesystems

Steve French

Senior Engineer
IBM Linux Technology Center
sfrench@us.ibm.com

VFS

"The Virtual File System (otherwise known as the Virtual Filesystem Switch) is the software layer in the kernel that provides the filesystem interface to userspace programmes. It also provides an abstraction within the kernel which allows different filesystem implementations to co-exist."

Richard Gooch in documentation/filesystems/vfs.txt

VFS is quite different from the very complex, rich "IFS" interface that Windows NT, 2000 and XP provide. Many Unixes support a VFS like interface to filesystem driver software

Linux Filesystems

- 40 Filesystems in 2.4.18, mostly local
- 2 Added by 2.5.7: driverfs and IBM's JFS (and expand_fs is gone)
- RAMFS in Linux 2.4 is a good example of a minimal, cleanly written filesystem
- Network filesystems unfortunately are much more complicated for Linux see include/linux/fs.h to see a complete list of the Linux filesystems
  - NFS, NCPFS, SMBFS, Coda, Intermezzo
- I am writing a CIFS VFS (smbfs overlaps slightly)
- AFS and GFS (not in the main tree) are also important

Filesystems are still critical

- In this Internet era
- … despite databases
- … despite new storage paradigms
- “Traditional” Filesystems are still critical and are being actively developed, enhanced improved
- NAS (and hybrid NAS/SAN) products and management software is increasing in importance
- Storage (especially network storage requirements) is growing faster than processor requirements
- The goal: make network storage so reliable, fast, secure, and easy to operate that users hardly ever think about it. (the industry has a long way to go, fortunately for future grads)

… and they keep changing. Note: Linux 2.5 Filesystem changes

- See Documentation/filesystems/porting for a complete list
- BKL (Big Kernel Lock) taken in fewer places for improved performance
- Read_super (called eventually from do_mount) is gone replaced by get_sb (big help for network filesystems that do not want mount helpers)
- Filesystem declaration changed and a few flags
- New inode allocation/destroy routines
- Seven new filesystem helper routines

Why not just one Network Filesys?

What about HTTP/WebDAV?

- IFS for WebDAV included in Windows XP (but VFS client not in Linux)
- Apache, IIS and others have support available for WebDAV serving
- WebDAV has some network filesystem characteristics

- Although destined to increase in importance for the Internet, problematic to rely only on WebDAV for all network file I/O
  - Slow - not as suitable for network file I/o within the server room (or within the intranet) as for slower, bandwidth constrained Internet network file I/o
  - Requires awkward compensations to map to Windows operating systems
  - Feature-poor compared to CIFS and NFSv4 alternatives
  - CIFS is well entrenched on the vast majority of clients already and coexists with WebDAV in XP

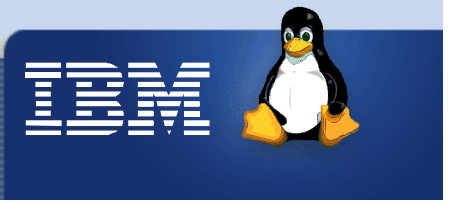
Why are Network Filesystems hard?

- Two views of the same data/meta-data (local and server)
  - Distributed cache coherence
    - File data
    - Directory data
  - Redundant locks on client and server
  - Differing inode numbering on client and server
  - Differing namespace
- Compensation for holes in protocol and/or server OS
- Network Security across potentially hostile routers
  - Distributed authorization
  - Access Control
  - PDU integrity
- Exotic failure scenarios and file & server “migration”

Major Network Filesystem “Families”

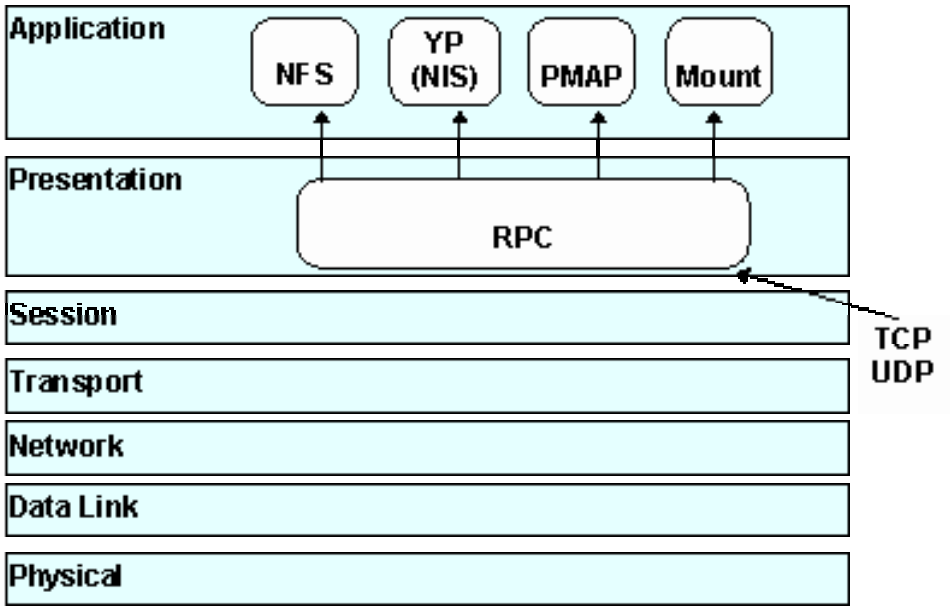
(most widely distributed “families” listed at top)

- HTTP->WebDAV
- SMB->CIFS
- NCP (Netware Core Protocol)
- NFSv2->NFSv3->WebNFS->NFSv4->DAFS
- AFS->DFS, AFS->Coda->Intermezzo
- GFS



Where does NFS fit in?

An OSI Model view



Source: http://www.protocols.com/pbook/sun.htm

Where does CIFS Fit In?

An OSI Model view

7 Application	Higher level protocols e.g. SMB / CIFS		e.g. Browser Service
6 Presentation			
5 Session	Name Service	datagram service	Session Service
4 Transport	UDP		TCP
3 Network	IP		
2 Datalink	e.g. IEEE 802.2		
1 Physical	Token Ring / Ethernet etc		

Source: http://ourworld.compuserve.com/homepages/timothydevans/osi.htm

CIFS

The Common Internet File System (CIFS) is a file sharing protocol. Client systems use this protocol to request file access services from server systems over a network. It is based on the Server Message Block protocol widely in use by personal computers and workstations running a wide variety of operating systems… CIFS has become a key file sharing protocol due to its widespread distribution and its inclusion of enhancements that improve its suitability for internet authoring and file sharing. CIFS assumes even more importance due to the indirect use of CIFS as a transport protocol for various higher level NT and Windows9x communication protocols, as well as for network printing, resource location services, remote management/administration, network authentication (secure establishment services) and RPC (Remote Procedure Calls).

Source: SNIA CIFS Technical Reference version 1.0

Scope of CIFS

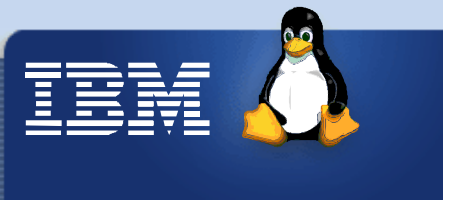
- SMB/CIFS is often thought of in a broader sense of the related collection of protocols providing:
  - Network file and print sharing
  - Logon handling and user/group authentication
  - ACL management and authorization
  - User, password and group management
  - Server management
  - Print Queue management
  - Server/Resource location
  - Various Network IPC mechanisms
  - “Single system image” of network resources

CIFS – Common Internet File System

- People think of Microsoft when they think of CIFS since they coined the new name for SMB protocol in 1996, soon after Sun announced WebNFS extensions to NFSv3.
- And CIFS is critical for interoperability with Microsoft systems
  - "CIFS is the basis of everything that we do" and
  - "The Underlying storage fabric is CIFS … for a long time." [Rob Short MS VP 8/2001]
- But Dr. Barry Feigenbaum (IBM) actually invented CIFS's predecessor SMB (originally called "BAF" protocol) in the mid-1980s and multiple companies contributed
- SMB is the X/Open (OpenGroup) "Standard for PC Interworking" (1992)
- SMB/CIFS is the main network filesystem on OS/400, OS/2, DOS and other operating systems and implementations are available on most every major operating system of the past 10 years.
- Storage Network Industry Association just released CIFS Technical Ref
- Unix and Macintosh extensions to CIFS are documented by SNIA and implemented

What is Samba?

- … The most visible open source CIFS Server implementation (which has grown to include tools, utilities and even ftp like clients)
- Samba, started in 1991 by Dr. Andrew Tridgell, is one of the larger open source projects (almost 200,000 Lines of Code) – certainly one of the largest included in all Linux distributions and one of the most visible
- Which is quite impressive since Samba development effort is focused on protocol analysis and interoperability testing and maximal portability rather than coding of new features
- And The implementation is fairly “terse” – ie smaller in code size than one would expect for the amount of function implemented and some key portions have been rewritten multiple times
- Samba team is somewhat conservative



Samba (continued)

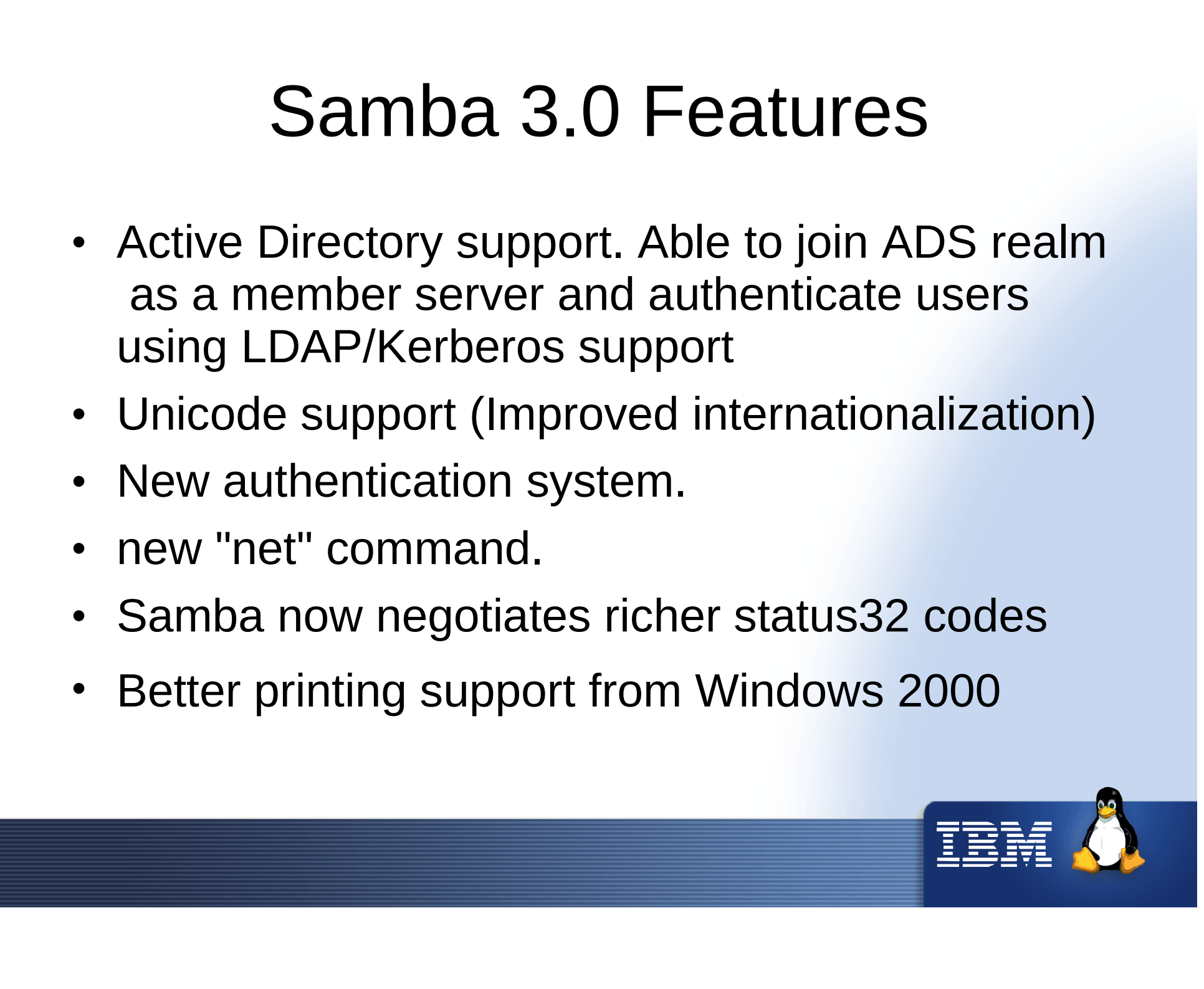
- Samba's target environment is networks with either Windows servers or Windows clients but it is broadening due to spread of CIFS capable devices and server appliances
- Samba implements a superset of the various loosely related protocols documented by SNIA (CIFS T/R version 1), X/Open (PC Interworking specs), OpenGroup (DCE/RPC) and IETF (RFC 1001/1002 and various draft RFCs, some expired). These include extensions to the protocol for Macintosh and Unix systems.
- It aims at equal or better function than the corresponding function in Windows 2000/XP/Active Directory (in the long run)
- It is quite fast and highly configurable and implements some interesting features
- See http://www.samba.org

Samba/CIFS benefits

- New technology being brought into Linux
  - Marries much of what is good in Windows with what is good in Linux
- Standards (both open and defacto) being adopted
  - SNIA CIFS Technical Reference spec is improving. New CIFS related RFCs are in process (e.g. CIFS URL) with others at early design stages
  - Keeping up with important parts of Defacto standard CIFS client implementation of Windows 2000->XP, although moving target, causes constant improvement
  - Perception of playing Catchup (with Microsoft) is interesting. Samba is more functional than Win2K in some key areas and does not need full functional parity with Win2K in all areas in order to greatly enhance “Linux” much as Linux NFS does not need to perfectly match Solaris NFS related function.
- (Note NFSv4 borrows many CIFS concepts)

Samba 3.0 Features

- Active Directory support. Able to join ADS realm as a member server and authenticate users using LDAP/Kerberos support
- Unicode support (Improved internationalization)
- New authentication system.
- new "net" command.
- Samba now negotiates richer status32 codes
- Better printing support from Windows 2000

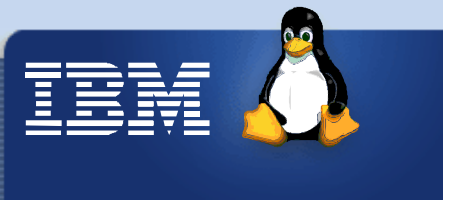


CIFSFS Target Enhancements (not in existing 2.4 SMBFS module)

- Locking support
- Client side caching (oplock) support
- “native TCP/IP” (port 445) not just RFC1001/1002 transport supported
- Internationalization (Unicode)
- Microsoft/Samba “dfs” support (globally rooted namespace and server side replication share)
- Synergy with WinBIND(PAM logon and NSS modules)
- Support for Unix Extensions to CIFS (added to standards document in approx. 1998)
- (optional) Use LDAP to find global root of SMB/CIFS namespace
- (optional) Remote boot support (for network booted clients such as kiosks and appliances)

Reliability features

- POSIX semantics honored where possible (improvement over smbfs and current nfs) – should reduce data loss problem with network filesystems (e.g. most don't handle locks correctly, and cache data on client incorrectly)
- Transparent reconnect after server/network failure
- Directory notification support (allowing safe caching of directory entries)
- Large file support – can handle files larger than 2GB
- Improved error handling (CIFSFS will understand those additional errors in CIFS spec that are not handled by SMBFS today)



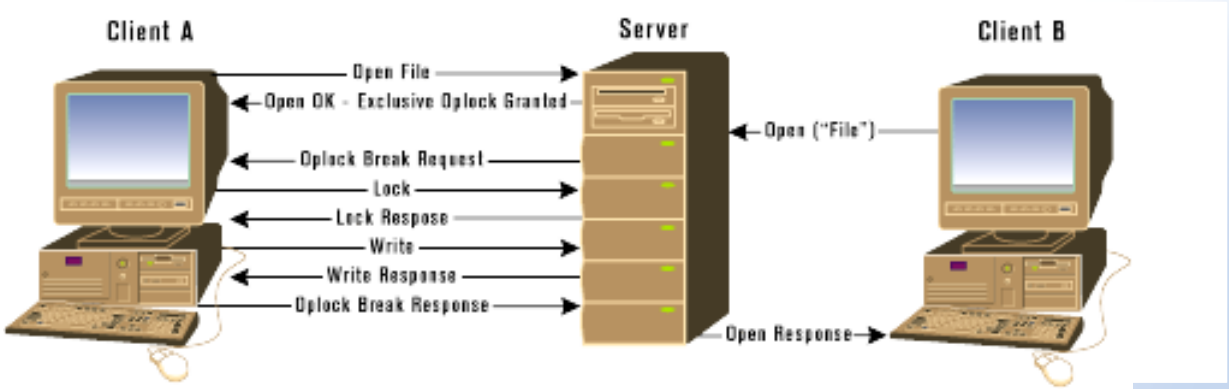
Security Features

- SMB signing (ensures data integrity on public networks). Will allow feature to be enabled/disabled at mount time.
- Will support (at least) NTLM++ authentication
- (Optional) Enhanced open authentication - Kerberos v.5/SPNEGO SMB session authentication (ala Leach presentation and the paper from Craig Russ at Unisys).
- (Optional) SSL support for confidentiality of data
- Configuration settings for security negotiation (e.g. whether to allow plain text passwords)
- (Optional) per-file signing
- Multi-user mounts
- Quota and ACL support
- And more …

Performance

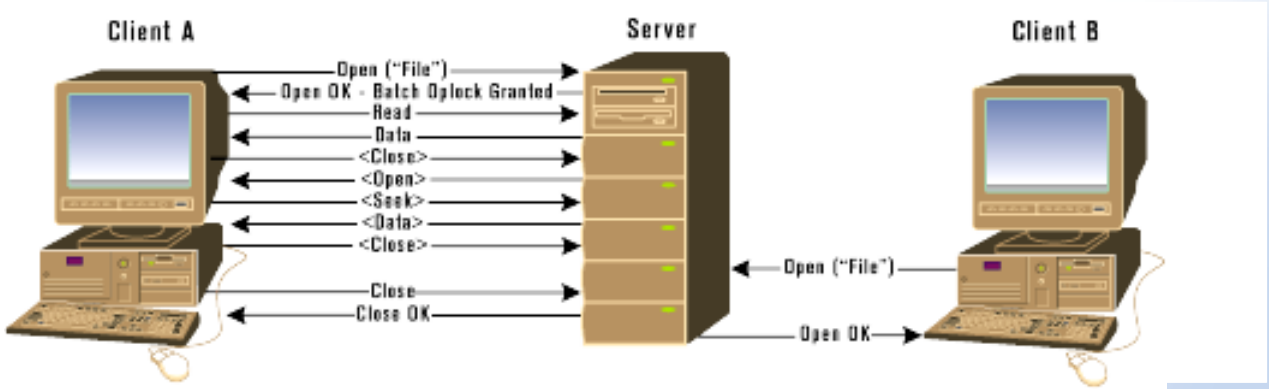
CIFS – Exclusive Oplock

Opportunistic Locks - mechanism for simple
Distributed Token Management (ie client cache coherence)

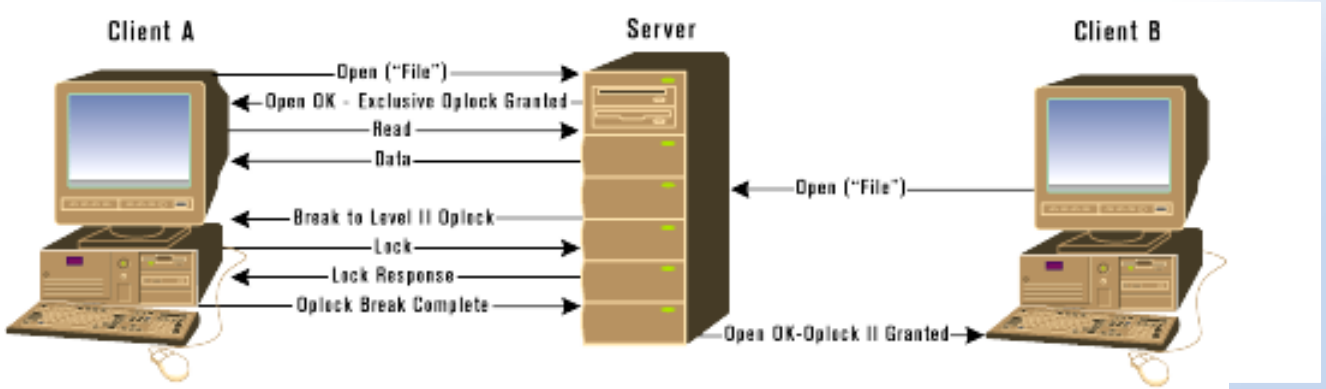


Source (oplock figures): www.microsoft.com/Mind/1196/CIFS.htm

CIFS – Batch oplock



CIFS – level II oplock



Performance points to consider for client

- For common cases in (at least) a typical intranet - let's show (and fix if need)
  - Maximize parallelism from each client
  - Maximize client caching opportunities
  - Minimize roundtrips from clt to srv
    - Command chaining - does it need improvements ala NFS v4?
  - Minimize protocol overhead (frame hdrs)
  - Latency when lightly loaded (timers ...)
  - Session establishment and auth overhead

Performance points - what is missing?

- Change open type - rerequest oplock (on long opened file) without close

(Breaking news … ioctl discovered that seems to be for this purpose)

- Do we need a new token/oplock manager to better handle replicas?
- Secondary session (maxvs > 1) - rawVC or new RDMA secondary connection
- Client timing/observations of server - autotuning read-ahead and oplock/lazy close behavior

Performance points - what is missing?

- Lots of distinct infrequently used long paths are big problem for Linux
- ... but could be tough to skip due to referrals
- Vnode invalidation tricky on client due to kernel caching
- Are access hints on file sufficient? Is something similar for directories needed?

Compatibility

VFS Operations (file ops)

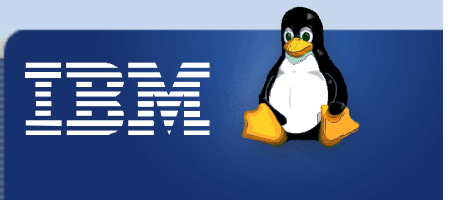
- Llseek
- Read
- Write
- Readdir
- Poll
- Ioctl
- Mmap
- Open
- Flush
- Release
- Fsync
- Fasync

VFS Operations (file ops part 2)

- Lock
- Readv
- Writev
- Sendpage
- Get_unmapped_area

VFS Operations (part 3) inode ops

- Create
- Lookup
- Link and Symlink
- Unlink
- Mkdir
- Rmdir
- Mknod
- Rename
- readlink



VFS Operations (part 4) inode ops

- Follow_link
- Truncate
- Permission
- Revalidate
- Setattr
- Getattr

- And superblock operations
  - Statfs
  - Unlockfs
  - Write_super
  - Inode_from_fh

Key Data Structures

- Superblock matches ok (connected to CIFS servers)
- Inode is easy match except for 777 permissions (vs. SecurityDescriptors) and to lesser extent FileAttributes
- File structure is ok match but missing AccessFlags
- Dentry structure is mostly transparent
- Lock structure is reasonable match (behavior differences though) – mandatory vs. advisory

Error codes

- Lists of error codes by SMB PDU are nowhere near complete
- Error mapping, especially from (large list of) NT STATUS codes to (smaller) errno.h Posix errors is adhoc
- Out of band alerts, management API, CIFS client MIB etc. is possibility for future documentation and/or standardization

Subtle OS specific features

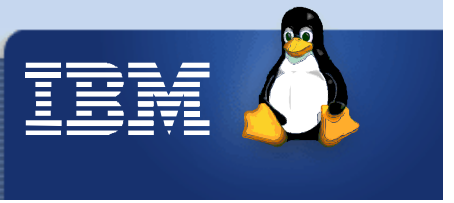
- Underdocumented and less understood features that affect the client are:
  - Reparse points (e.g. copy on write like links used by SIS on Win2K among others).
  - Symbolic links – Unix extensions can be used
  - Sparse file attribute bit
  - Compressed and encrypted file attribute bits
  - Extended attributes and streams (tend to be application/desktop specific). Stream mechanism itself is understood though.
  - Ever growing ioctl/fsctl calls & T2 Info levels

Reliability

Reliability Issues to Consider

1. Logon reliability, redundancy
2. Ancillary Logon time actions

- Redundancy of DFS root
- File migration to new locations
- Redundancy of (especially r/o) resources using DFS replicas
- Reconnection after transient TCP session failure without user intervention – how transparent to an application?
- What if TGT has expired? Can KDE or GNOME prompt the user to re-enter the password or is a local service necessary?



Reliability Issues to Consider

- It is a given that clients should leverage DFS when available for reconnection -
  - But how long can we keep info on replicas safely without asking for a referral again?
  - What order should clients try to connect to replicas? Round robin in same subnet 1st?
  - And the mgmt API for DFS (AddRoot, etc.) is only partially implemented ...

Data integrity in presence of data cached by client kernel

- Invalidating Pinned mmaped pages are tricky invalidate_inode_pages2() may help but not demonstrated since not directly invoked by any filesystems in 2.4
- What having a distinct lock protocol help?
- How to handle directory entry caching?
  - One option is the notify SMBs

    Performance issues unproven so ignored by most

Security

Security features

- Minimum
  - NTLM authentication
- But we should be offering
  - Kerberos authentication
  - NTLMv2 & packet signing
  - Per file (server decides) signing and encryption (ala NFSv4 proposal) – requires addition of new RC
- The CIFS and Kerberos protocol should add support for
  - Optional AES encryption (mostly a Kerberos issue)
  - Optional per-packet privacy or mode widespread support for SSL or dual transport SSL and TCP
  - Are changes needed for Hardware assist for session establishment

Security features

- For Linux filesysystems in particular a choice must be made early on –
  - Do all users get the same SMB UID when going to the server (relying on local ACLs to prevent users from modifying server data)?
  - Do you do a new SMB SessSetupX for each new user of the mount point?
- Complications – where do you get the password (especially on implicit dfs connections)? Desktop to FS interactions not strong suite of Unixes today
- And … what about password during reconnect? Store in cifs fs?

Some interesting security references

For more information - Security

- Kerberos and interoperable authentication
  - Project Pismere/MIT http://web.mit.edu/pismere/M
- MIT kerberos -
  - http://web.mit.edu/kerberos/www/
- Heimdal Kerberos implementation
  - http://www.pdc.kth.se/heimdal/L
- Luke Leighton's Book on DCE/RPC & SMB
  - **DCE/RPC over SMB: Samba and Windows NT Domain Internals**

For more information - Security

- Westerlund, A and Danielsson, J, "Heimdal and Windows 2000 Kerberos -- How to get them to play together", USENIX 2001
- Swift, M and Brezak, J, "The Windows 2000 RC4-HMAC Kerberos Encryption Type" Work in progress, draft-brezak-win2k-krb-rc4-hmac-02.txt

Name Resolution

CIFS Name Resolution issues

- Skipping NBT (I.e. RFC 1001 name service) – using "pure TCP" on port 139. Is this always ok?
- In pure TCP - what should go in UNC path in tConX? IP address works but what about multiple servers on one port? *SMBSERVER does not work for Win2K
- Does ipv6 address work in SMB tConx path?
- Are there cases in which Unicode translation should be disabled even though both server and client support it? (same codepage on each - how can the client detect the server's code page)
- Performance of kernel to user space transitions for access to DNS address helper routines when given “implicit” mounts (e.g. “dfs” junctions)
- How effective is skipping NBT in reducing WINS related denial of service attacks?

More CIFS Name Resolution issues

- Is there a recommend order for trying address resolution? Big performance hit
  - Probably not - many Unix/Linux clients may be satisfied with pure DNS resolution
- Need for DFS support in more clients is a given but ...
  - How do you locate the global root?
    - DNS lookup of reserved domain specific name
    - LDAP query (Win2K seems to store it in both)
    - UDDI query
    - Local config file (worst case for most)
- Need for Equivalent to old IBM concept of logon domain specific “aliases” (short names that map to hard to remember long network path names created by an administrator)
- UNC naming is foreign to some Linux/Unix users

Options for the Future

Status

Here is a sample of the fun problems involved

CIFS challenges in Linux - Contributions welcome

- Mount does not pass UNC name until 2.5:
  - Mount //server/share /mnt –o user=name,pass=
- Hard to detect mandatory vs. advisory locks (SGID/GroupExec hack)
- Invalidating mmap pages
- Network mounts from multiuser systems with correct per-user credentials
- Open and create are small subset of Windows equivalent function creating compensation issues for options (e.g. impersonation)
- Many missing or hard to set flags (such as sparse file, access pattern hints)
- AccessFlags
- Security Descriptors and access control mapping (and 7**7**7 group perm)
- Missing security libraries and DNS resolution helper code in kernel
- Efficiency of large scale directory change notification
- A few of the O_ flags map poorly
- AD Integration, Winbind/NSS integration
- Unicode/Internationalization issues

CIFS Linux VFS Module Status

New project, soon to be released to the Samba web site

- Working:
  - Mount in kernel at NTLM level
  - Unicode or ASCII
  - Most Current level of SMBs following CIFS T/R
  - Open, Read, Write, Release (close)
  - Get and Set Attributes
  - Readdir (for small to medium sized directories)
  - Hardlinks
  - Large file (>2GB) support
  - Passes 5 of first 7 tests attempted
- Not implemented:
  - Chmod (777 returned), Native CIFS ACLs/SecurityDescriptors, chown integr
  - Locking, DFS hierarchical filesystem/referrals, symbolic links
  - Distrubuted caching/oplock, MMAP
  - Kerberos/SPNEGO Authentication
  - Network Boot via CIFS