CIFS Protocol Extensions Update

Steve French
CIFS maintainer and Senior Engineer IBM LTC

Jeremy Allison
Senior Engineer Samba 3/Novell

Legal Statement

This work represents the views of the authors and does not necessarily reflect the views of IBM Corporation or Novell Corporation.

A full list of U.S. trademarks owned by IBM may be found at http://www.ibm.com/legal/copytrade.shtml.

Linux is a registered trademark of Linus Torvalds.

Other company, product, and service names may be trademarks or service marks of others.

Who are we ... server and client maintainers

- Steve French
  - Author and maintainer of Linux cifs vfs (for accessing Samba, Windows and various SMB/CIFS based NAS appliances)
  - Member of the Samba team, coauthor of CIFS Technical Reference and former SNIA CIFS Working Group chair
  - Architect: Filesystems/NFS/Samba IBM LTC
- Jeremy Allison
  - One of the original authors of Samba 3 server
  - Novell/SuSE Samba lead

Outline

- Why SMB/CIFS ... 22 years and counting?
- Unix Extensions ... good enough?
  - Why were they developed?
  - What and where are they?
- But something was missing ...
  - What about MS SFU? Or SMB2?
  - What about more Extensions ...?

Outline (continued)

- CIFS POSIX Extensions
  - Basics
  - ACLs
  - POSIX Locking
  - Other new feature
- status of current implementations ... are they available?
- And looking toward the future ...

22 years ago

- The birth of SMB/CIFS: Dr. Barry Feigenbaum et al of IBM (published 1984 IBM PC Conf), continued by Intel, 3Com, Microsoft and others

Then

- IBM PC LAN Program (PCLP)
- MS-Net

And Now...

- Windows goes on and on - sees new Vistas
- Other servers from many companies
  - Samba 3.0.23 and 4 (Novell, RedHat, IBM ...)
  - NetApp ...
- And many clients
  - Smbclient
  - Linux CIFS VFS
  - JCIFS, MacOS ...

But Why CIFS?

- CIFS is a surprisingly broad, rich protocol
- Existing CIFS servers and clients need fewer changes to achieve functional and performance goals than alternative approaches
- Reasonable performance for certain workloads already, no unnecessary intermediate RPC layer, and straightforward caching model
- Broad support for many platforms including all of most common ones
- Synergy with large installed base of CIFS clients and servers

But Still ... Why CIFS?

- CIFS is the defacto standard network filesystem for hundreds of millions of machines (and not just for Windows).
- CIFS clients and servers exist for most or all major platforms
- And the alternatives have problems ...

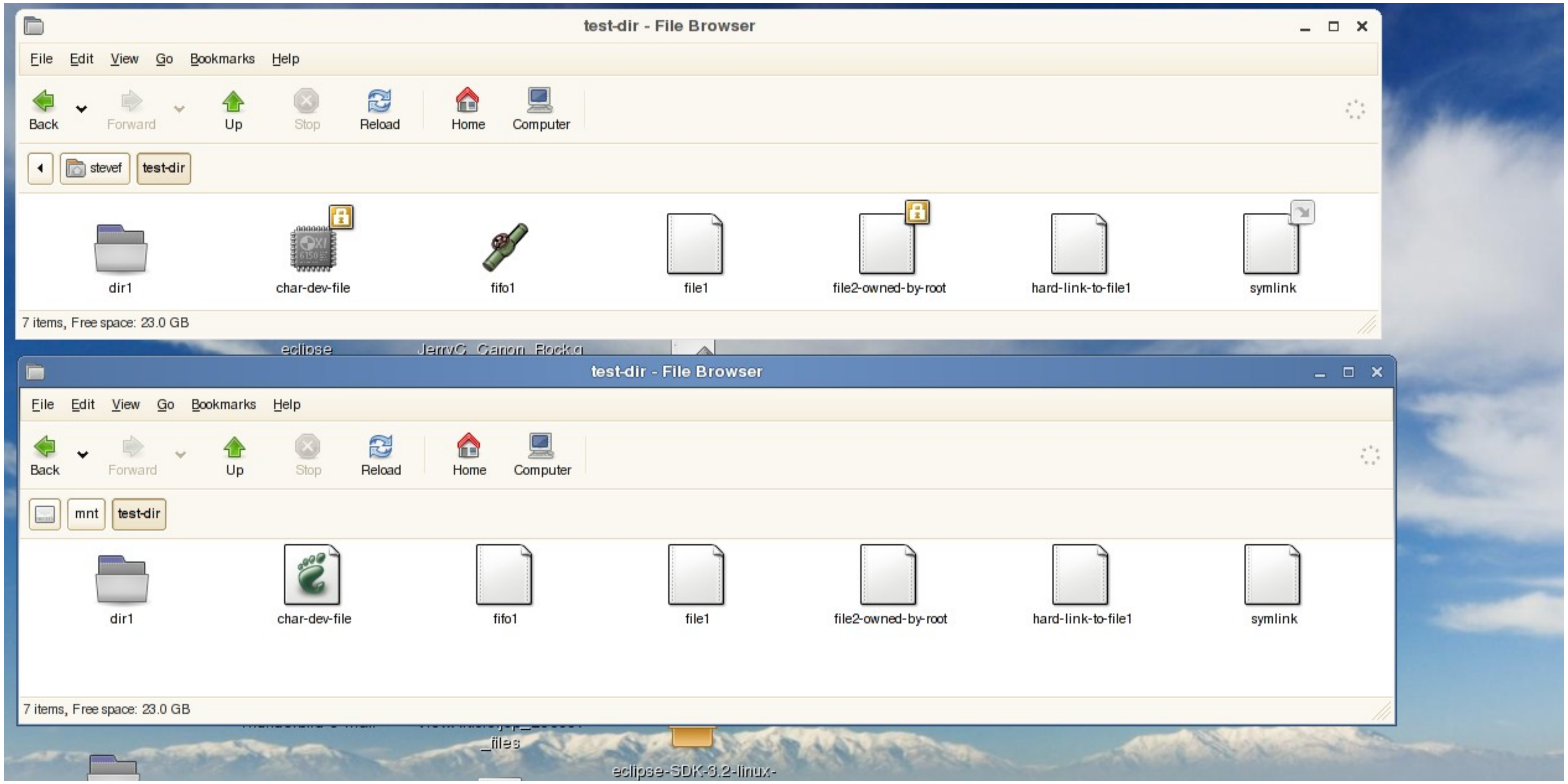
And the alternatives?

- NFS v3 or v4
- AFS/DFS
- HTTP/WebDav
- Cluster Filesystem Protocols

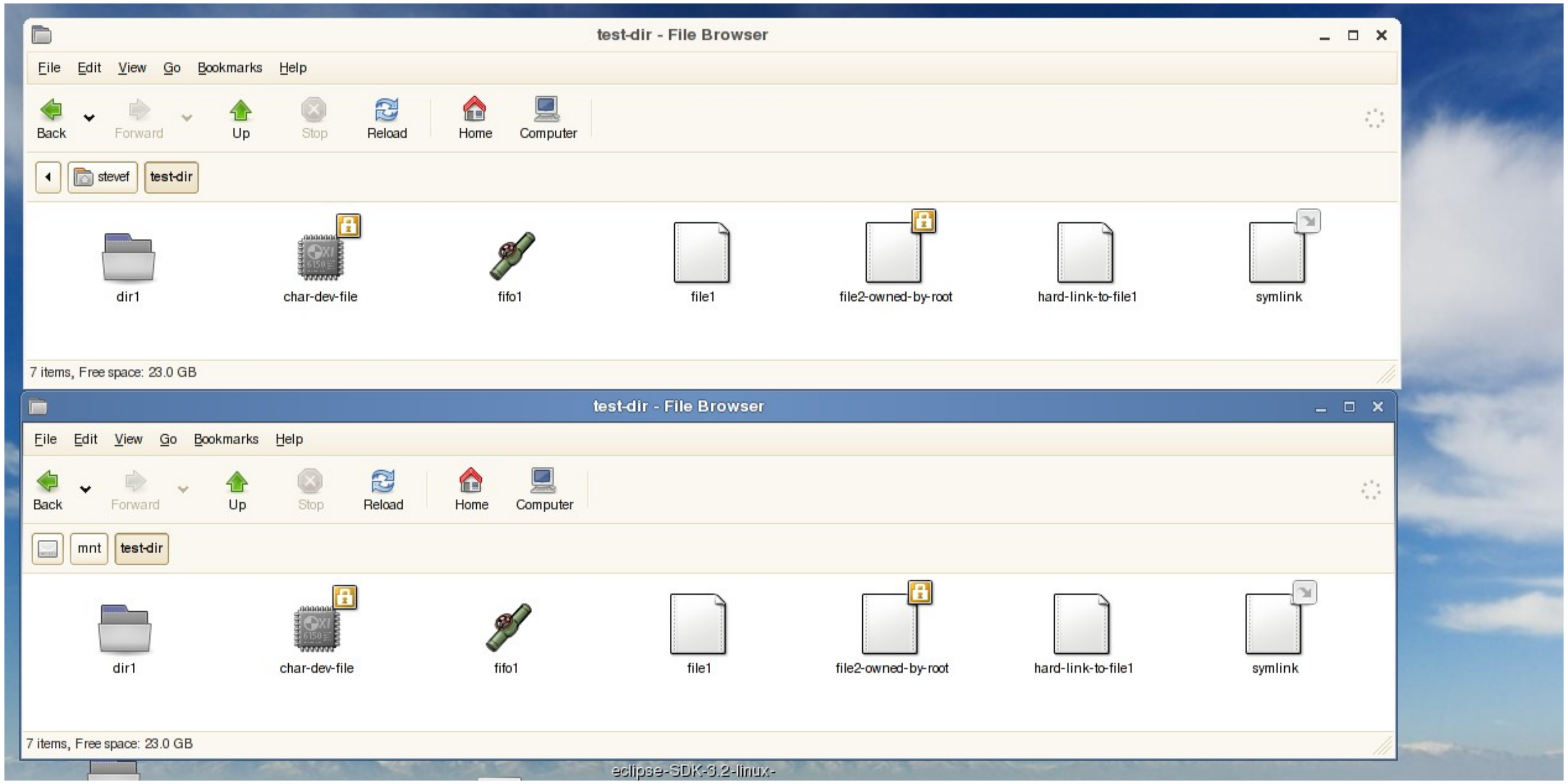
CIFS Unix Extensions

- Developed/Documented by HP (extending early work by SCO) and others then documented by SNIA in the CIFS Technical Reference
  - Required only modest extensions to server
  - Solved key problems for POSIX clients including:
    - How to return: UID/GID, mode
    - How to handle symlinks
    - How to handle special files (devices/fifos)

Without CIFS extensions, less local/remote transparency...



Much improved with CIFS Extensions



What about SFU approach?

- Lessons from SFU:
  - Map mode, group and user (SID) owner fields to ACLs
  - Do hardlinks via NT Rename
  - Get inode numbers
  - Remap illegal characters to Unicode reserved range
  - FIFOs and device files via OS/2 EAs on system files
- OK, but not good enough …
  - Some POSIX byte range lock tests fail
  - Semantics are awkward for symlinks, devices
  - UID mapping a mess
  - Performance slow
  - Operations much less atomic and not robust enough
  - Rename/delete semantics are hard to make reliable

CIFS Unix Extensions

- Problem ... a lot was missing:
  - Way to negotiate per mount capabilities
  - POSIX byte range locking
  - ACL alternative (such as POSIX ACLs)
  - A way to handle some key fields in statfs
  - Way to handle various newer vfs entry points
    - lsattr/chattr
    - Inotify
    - New xattr (EA) namespaces

Original Unix Extensions Missing POSIX ACLs and statfs info

```
smf-t41p:/home/stevef # getfacl /mnt/test-dir/file1
# file: mnt/test-dir/file1
# owner: root
# group: root
user::rwx
group::rw-
other::rwx

smf-t41p:/home/stevef # stat -f /mnt1
  File: "/mnt1"
    ID: 0        Namelen: 4096    Type: UNKNOWN
(0xff534d42)
Block size: 1024       Fundamental block size: 1024
Blocks: Total: 521748     Free: 421028     Available:
421028
```
Inodes: Total: 0   Free: 0

With CIFS POSIX Extensions, ACLs and statfs better

```
smf-t41p:/home/stevef # getfacl /mnt/test-dir/file1
# file: mnt/test-dir/file1
# owner: stevef
# group: users
user::rw-
user:stevef:r--
group::r--
mask::r--
other::r--

smf-t41p:/home/stevef # stat -f /mnt1
  File: "/mnt1"
    ID: 0        Namelen: 4096    Type: UNKNOWN (0xff534d42)
Block size: 4096       Fundamental block size: 4096
Blocks: Total: 130437     Free: 111883     Available: 105257
Inodes: Total: 66400      Free: 66299
```

POSIX Locking

- Locking semantics differ between CIFS and POSIX at the application layer.
  - CIFS locking is mandatory, POSIX advisory.
  - CIFS locking stacks and is offset/length specific, POSIX locking merges and splits and the offset/lengths don't have to match.
  - CIFS locking is unsigned and absolute, POSIX locking is signed and relative.
  - POSIX close destroys all locks.

Protocol changes

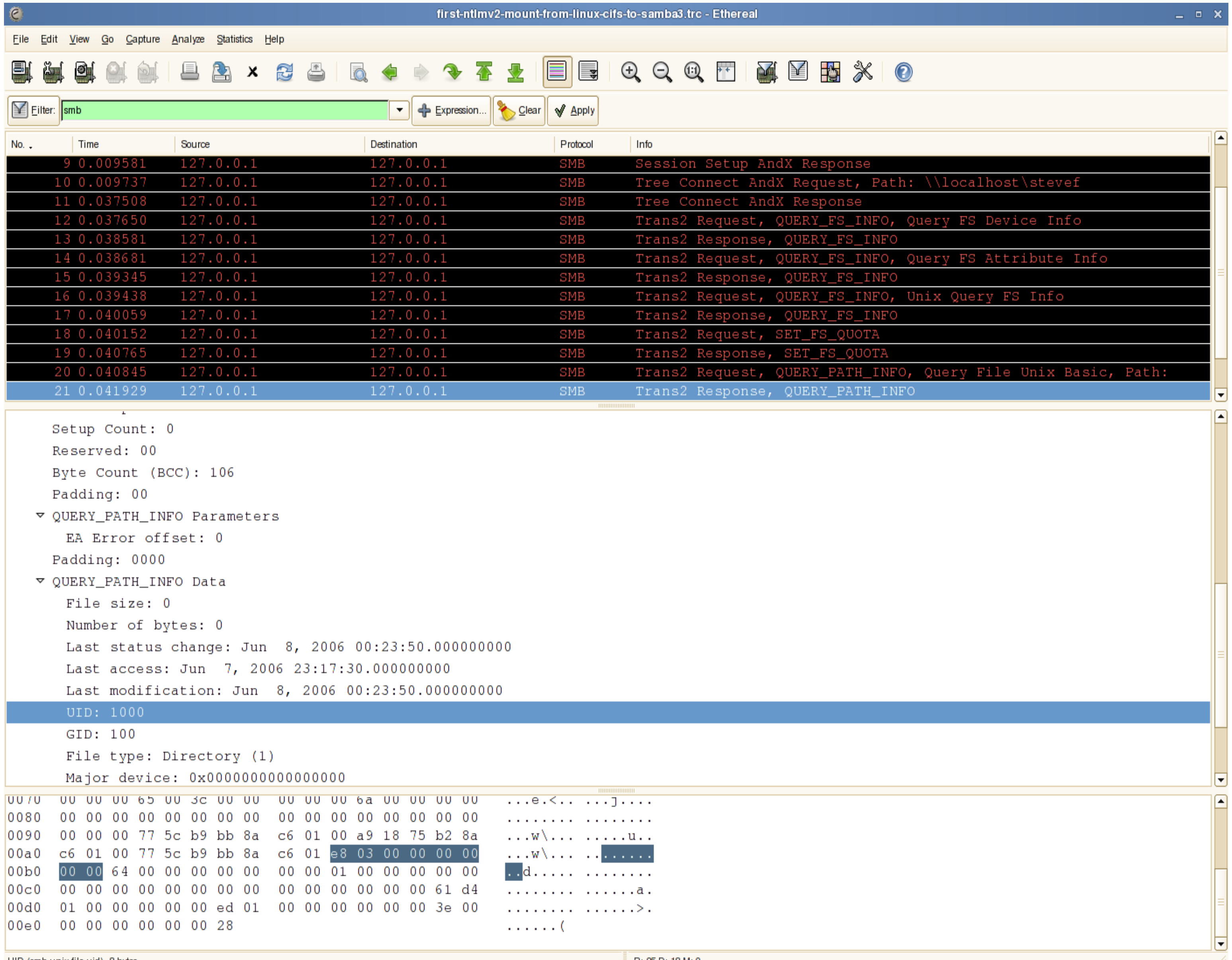
- The mandatory/advisory difference in locking semantics has an unexpected effect.
- READX/WRITEX semantics must change when POSIX locks are negotiated.
  - Once POSIX locks are negotiated by the SETFSINFO call, the semantics of READ/WRITE CIFS calls change – they ignore existing read/write locks.
  - POSIX-extensions aware clients probably *want* these semantics.
    - It's a side effect, but a good one !

Status

- Clients
  - CIFS client
    - Version 1.45 (Linux 2.6.18) includes the much improved POSIX locking
    - Version 1.32 included POSIX ACLs, statfs, lsattr
  - Smbclient
    - Samba 3.0.23 includes client test code for POSIX locking
- Server
  - Samba 3.0.23 includes POSIX Locking (POSIX ACLs, QFSInfo, Unix Extensions implemented before)
  - HP/UX and a few other servers also support original Unix Extensions

Roadmap

- Client
  - 2.6.19 will include new mkdir/open
- Server
  - Samba 3.0.24 will better map onto local posix locks
  - Samba 4 Unix/POSIX Extensions started with new POSIX CIFS client backend
- In discussions with other client and server vendors about feature needs


first-ntlmv2-mount-from-linux-cifs-to-samba3.trc - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: smb
Expression...
Clear
Apply

```
No.   Time        Source      Destination  Protocol  Info
  9 0.009581    127.0.0.1   127.0.0.1    SMB       Session Setup AndX Response
 10 0.009737    127.0.0.1   127.0.0.1    SMB       Tree Connect AndX Request, Path: \\localhost\stevef
 11 0.037508    127.0.0.1   127.0.0.1    SMB       Tree Connect AndX Response
 12 0.037650    127.0.0.1   127.0.0.1    SMB       Trans2 Request, QUERY_FS_INFO, Query FS Device Info
 13 0.038581    127.0.0.1   127.0.0.1    SMB       Trans2 Response, QUERY_FS_INFO
 14 0.038681    127.0.0.1   127.0.0.1    SMB       Trans2 Request, QUERY_FS_INFO, Query FS Attribute Info
 15 0.039345    127.0.0.1   127.0.0.1    SMB       Trans2 Response, QUERY_FS_INFO
 16 0.039438    127.0.0.1   127.0.0.1    SMB       Trans2 Request, QUERY_FS_INFO, Unix Query FS Info
 17 0.040059    127.0.0.1   127.0.0.1    SMB       Trans2 Response, QUERY_FS_INFO
 18 0.040152    127.0.0.1   127.0.0.1    SMB       Trans2 Request, SET_FS_QUOTA
 19 0.040765    127.0.0.1   127.0.0.1    SMB       Trans2 Response, SET_FS_QUOTA
 20 0.040845    127.0.0.1   127.0.0.1    SMB       Trans2 Request, QUERY_PATH_INFO, Query File Unix Basic, Path:
 21 0.041929    127.0.0.1   127.0.0.1    SMB       Trans2 Response, QUERY_PATH_INFO
```

```
    Setup Count: 0
    Reserved: 00
    Byte Count (BCC): 106
    Padding: 00
  ▽ QUERY_PATH_INFO Parameters
      EA Error offset: 0
    Padding: 0000
  ▽ QUERY_PATH_INFO Data
      File size: 0
      Number of bytes: 0
      Last status change: Jun  8, 2006 00:23:50.000000000
      Last access: Jun  7, 2006 23:17:30.000000000
      Last modification: Jun  8, 2006 00:23:50.000000000
      UID: 1000
      GID: 100
      File type: Directory (1)
      Major device: 0x0000000000000000
```

```
0070  00 00 00 65 00 3c 00 00  00 00 00 6a 00 00 00 00   ...e.<.. ...j....
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0090  00 00 00 77 5c b9 bb 8a  c6 01 00 a9 18 75 b2 8a   ...w\... .....u..
00a0  c6 01 00 77 5c b9 bb 8a  c6 01 e8 03 00 00 00 00   ...w\... ........
00b0  00 00 64 00 00 00 00 00  00 00 01 00 00 00 00 00   ..d..... ........
00c0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 61 d4   ........ ......a.
00d0  01 00 00 00 00 00 ed 01  00 00 00 00 00 00 3e 00   ........ ......>.
00e0  00 00 00 00 00 00 28                               ......(
```

Gory details

Minimal changes to negotiation ...

New capability for Session Setup

```
        #define CAP_UNIX   0x00800000
```

Optional Dialect for Negprot

        “POSIX 2”

New SMB Commands

        None

New Infolevels

        Total # Defined: 12 (“POSIX Extensions”)
        Implemented by Linux CIFS VFS: 10
        Implemented by Samba server: 9
        Original CIFS “Unix Extensions”: 5

More gory details

```
New File/PathInfo levels (Set and Get):
#define SMB_QUERY_FILE_UNIX_BASIC       0x200
#define SMB_QUERY_FILE_UNIX_LINK        0x201
#define SMB_SET_FILE_UNIX_HLINK         0x203 /* set only */
#define SMB_QUERY_POSIX_ACL             0x204
#define SMB_QUERY_XATTR                 0x205
#define SMB_QUERY_ATTR_FLAGS            0x206
#define SMB_QUERY_POSIX_PERMISSION      0x207 /* query only */
#define SMB_QUERY_POSIX_LOCK            0x208

New FindFirst/FindNext level (readdir)
#define SMB_FIND_FILE_UNIX                0x202

New QFSInfo level
#define SMB_QUERY_CIFS_UNIX_INFO    0x200 (set/query)
#define SMB_QUERY_POSIX_FS_INFO     0x201 (query only)
```

How to negotiate Unix/POSIX Capabilities

```
typedef struct {
        __le16 MajorVersionNumber;
        __le16 MinorVersionNumber;
        __le64 Capability;
} __attribute__((packed)) FILE_SYSTEM_UNIX_INFO; /* Unix extensions, level 0x200 *

/* Version numbers for CIFS UNIX major and minor. */
#define CIFS_UNIX_MAJOR_VERSION 1
#define CIFS_UNIX_MINOR_VERSION 0

/* Linux/Unix extensions capability flags */
#define CIFS_UNIX_FCNTL_CAP             0x00000001 /* support for fcntl locks */
#define CIFS_UNIX_POSIX_ACL_CAP         0x00000002 /* support getfacl/setfacl */
#define CIFS_UNIX_XATTR_CAP             0x00000004 /* support new namespace   */
#define CIFS_UNIX_EXTATTR_CAP           0x00000008 /* support chattr/chflag   */
#define CIFS_UNIX_POSIX_PATHNAMES_CAP   0x00000010 /* Allow POSIX path chars  */
```

Wire specifics

- Trans2 SETFSINFO call (0x4) with info level of SMB_SET_CIFS_UNIX_INFO (0x200) used to set capabilities bitmask.
  - CIFS_UNIX_FCNTL_LOCKS_CAP (0x1) turns on POSIX lock semantics – changes read/write semantics.
- Trans2 QFILEINFO (0x7) call has one new level, SMB_QUERY_POSIX_LOCK (0x208) whose parameters map to the POSIX F_GETLK fcntl() call.

Wire specifics (continued)

- Trans2 SETFILEINFO (0x8) call has one new level, SMB_SET_POSIX_LOCK (0x208) whose parameters map to the POSIX F_SETLK fcntl() call.
- Lock offsets and ranges must be translated by the client from the POSIX signed relative values to CIFS 64-bit unsigned absolute values.
  - [2 bytes] lock_type
    [2 bytes] lock_flags
    [4 bytes] pid = locking context.
    [8 bytes] start = unsigned 64 bits.
    [8 bytes] length = unsigned 64 bits.

API / Protocol interaction

- Common POSIX programming idiom is to set a SIGALRM to cancel a blocked lock.
  - This means cancellation of blocking locks.
  - Protocol request for blocking lock doesn't return until request succeeds (no timeout in POSIX locking).
  - Locks must be able to be canceled.
    - Re-used NTCANCEL (0xA4) call.
    - Causes lock request to return NT_STATUS_LOCK_NOT_GRANTED.
  - Close FID drops all locks on that dev/inode pair (treats as cancel).

Windows client/POSIX interaction

- POSIX clients read/write requests conflict with Windows locks, but not POSIX locks (Windows locks are mandatory for POSIX clients).
- Windows clients read/write requests conflict with both Windows and POSIX locks (both lock types are mandatory for Windows clients).
- Windows locks are set, unlocked and canceled via LOCKINGX (0x24) call.
- POSIX locks are set and unlocked via the Trans2 SETFILEINFO call, and canceled via the NTCANCEL call.

A few Extensions still needed

- inotify
- A few ioctls such as lsattr/chattr/chflags (currently implemented only in cifs client) e.g. To make a file immutable, or append-only, or to zero blocks on delete.

```
stevef@smf-t41p:~/test-dir> lsattr /boot/append-only-file
-----ad------ /boot/append-only-file
stevef@smf-t41p:~/test-dir> lsattr /mnt1/append-only-file
lsattr: Inappropriate ioctl for device While reading
flags on /mnt1/append-only-file
```

Unfinished features for full POSIX

- POSIX open/mkdir
  - Should take POSIX mode_t argument, and return the mode_t argument on create.
  - Should open with FILE_SHARE_READ/WRITE/DELETE.
- POSIX rename
  - If POSIX open should allow rename of open file.
- POSIX delete
  - If POSIX open should allow delete of open file.
  - File should disappear from directory listing.

New Infolevels

- #define SMB_POSIX_OPEN 0x209
    - MKDIR will be flag on open rather than distinct level

POSIX Errors

- NT Status codes (16 bit error nums) already has a reserved range
  - 0xF3000000 + POSIX errnum
  - POSIX errnum vary in theory, but not much in practice for common ones use
  - POSIX errnums fixed
  - New capability(will probably be)
    - #define CIFS_UNIX_POSIX_ERRORS 0x20
  - Do we need to define new errmapping SMB for client to resolve unknown POSIX errors backs to NT Status?

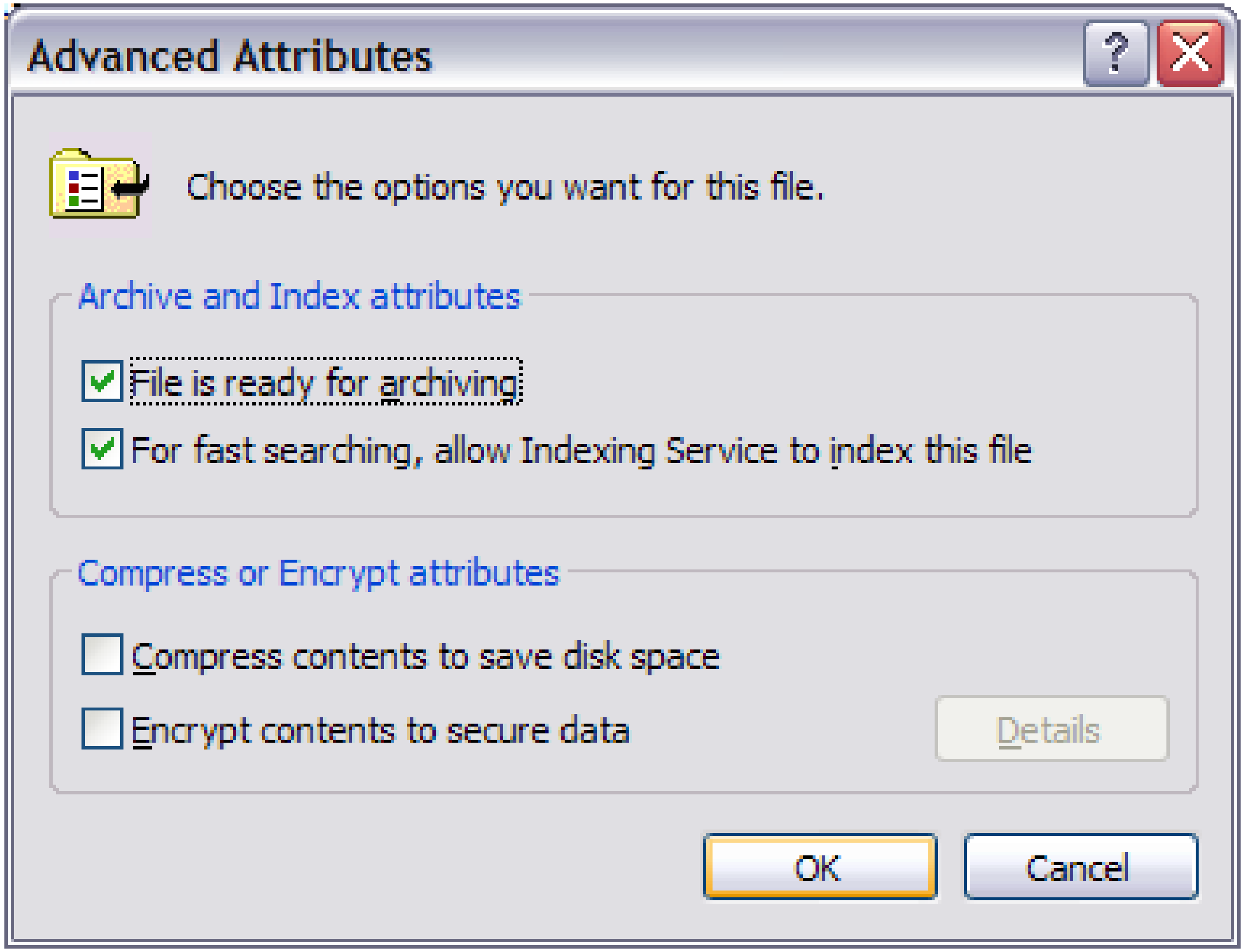
Beating the competition - NFSv4

- NFSv4 has sign+sealed data transport, using GSS-API sign/seal with krb5 encryption.
- CIFS needs something similar – we already have SMB signing, we just need to add the “sealing” component.
- Discussions are ongoing as to the best way to do this for UNIX to UNIX CIFS.
  - Please take part on samba-technical.
  - Remember working code trumps elegant design....

More general improvements still needed in our aging protocol

- These changes were not really Unix or Linux specific but POSIX apps may have stricter assumptions
- Full local/remote transparency desired
- Need near perfect POSIX semantics over cifs
- Newer requirements
  - Better caching of directory information
  - Improved DFS (distributed name space)
  - Better Performance
  - Better recovery after network failure
  - QoS

File Encryption / Compression as easy as local


Advanced Attributes

Choose the options you want for this file.

Archive and Index attributes

[✓] File is ready for archiving

[✓] For fast searching, allow Indexing Service to index this file

Compress or Encrypt attributes

[ ] Compress contents to save disk space

[ ] Encrypt contents to secure data

Session Encryption (seal vs. sign)

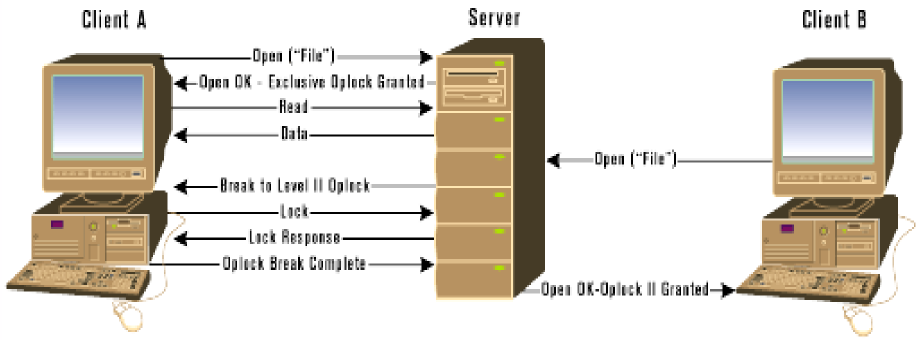
- SMB/CIFS signing almost a decade old
- There are sealed RPC pipes, but not sealed SMB sessions
- Per file encryption can be done (e.g. EcryptFS or IE to IIS)
- per-SMB sess encryption needed (NFSv4 gss sealing rqmnt similar) for perf reasons & also easier to admin

CIFS Encryption requirements

- Better performing and/or easier to configure than “encrypt everything” approach of ipsec
- Leverage cifs authentication context (not require 2nd login)
- Encrypt (at least) file data and file/directory names
- Don't repeat original SMB signing mistakes

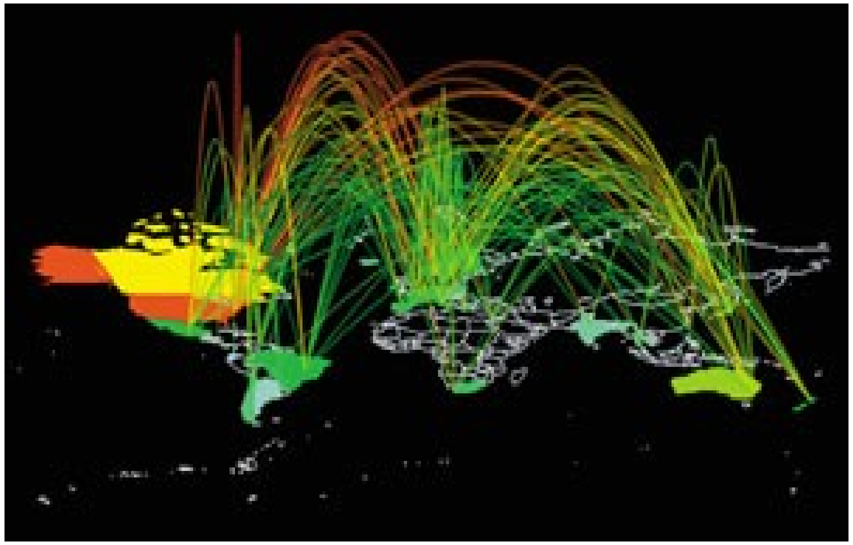
Caching improvements

- FCNTLs already defined/reserved for this
  - #define FSCTL_REQUEST_OPLOCK_LEVEL_1 0x00090000
  - #define FSCTL_REQUEST_OPLOCK_LEVEL_2 0x00090004
  - #define FSCTL_REQUEST_BATCH_OPLOCK 0x00090008
  - #define FSCTL_REQUEST_FILTER_OPLOCK 0x0009008C
- Current work going on to test this



Source: http://www.microsoft.com/mind/1196/cifs.asp

DFS (Global Namespace) improvements



- We need to improve ability to find nearest replica, and recover after failure
- And also to hint “least busy” server for load balancing

New Transports

- To adapt to larger writes
- Reduced latency
- Quality of Service

Where to go from here?

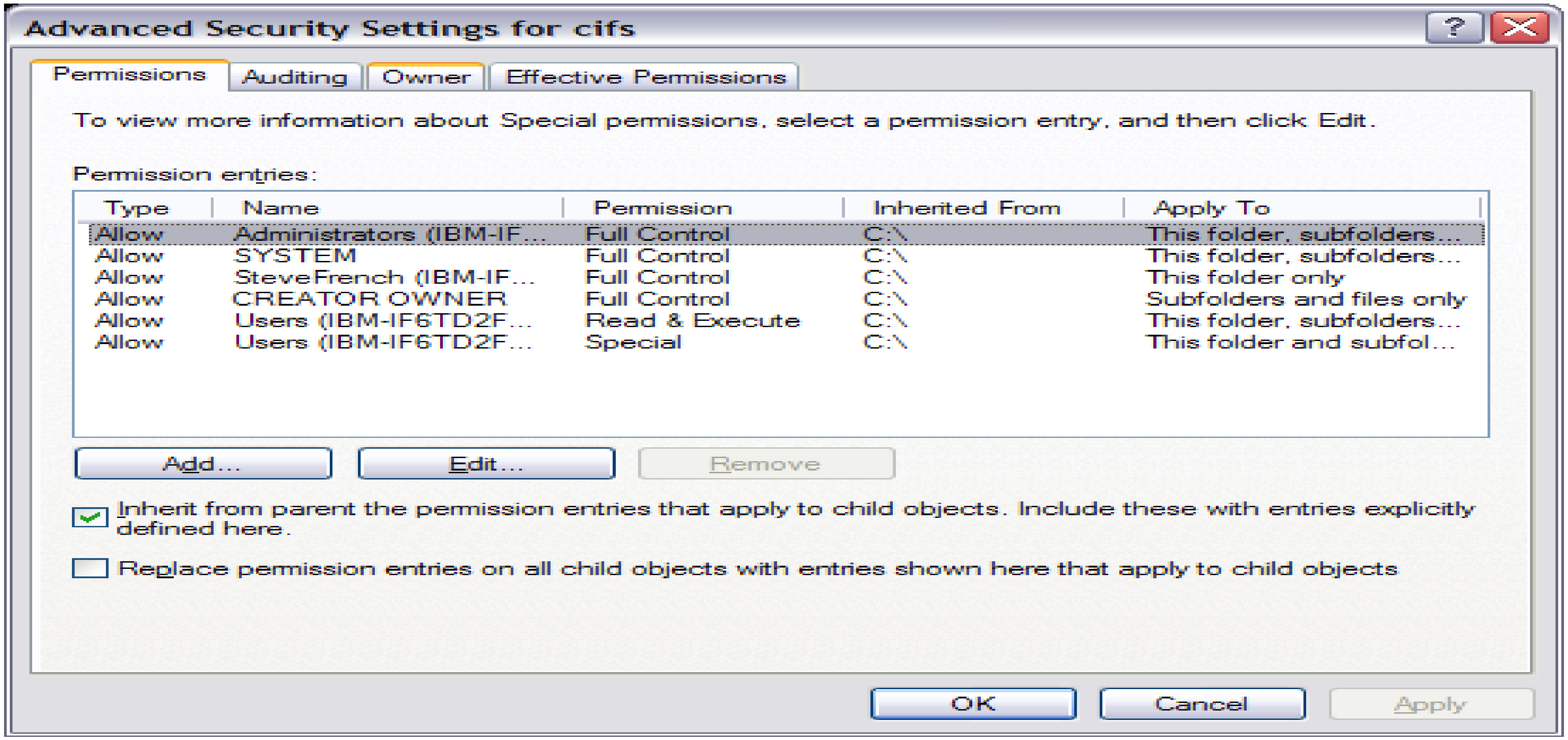
- Discussions on samba-technical and linux-cifs-client mailing lists
- Wire layout is visible in fs/cifs/cifspdu.h
- Working on updated draft reference document for these cifs protocol extensions
- See http://samba.org/samba/CIFS_POSIX_extensions.html

Thank you for your time!

Backing material

- CIFS Protocol surprisingly rich, already has support for rich ACLs, auditing, quotas

Security already functionally rich enough



To view more information about Special permissions, select a permission entry, and then click Edit.

Permission entries:

Type	Name	Permission	Inherited From	Apply To
Allow	Administrators (IBM-IF...	Full Control	C:\	This folder, subfolders...
Allow	SYSTEM	Full Control	C:\	This folder, subfolders...
Allow	SteveFrench (IBM-IF...	Full Control	C:\	This folder only
Allow	CREATOR OWNER	Full Control	C:\	Subfolders and files only
Allow	Users (IBM-IF6TD2F...	Read & Execute	C:\	This folder, subfolders...
Allow	Users (IBM-IF6TD2F...	Special	C:\	This folder and subfol...

✓ Inherit from parent the permission entries that apply to child objects. Include these with entries explicitly defined here.

☐ Replace permission entries on all child objects with entries shown here that apply to child objects