SMB3.1.1 POSIX/Linux Extensions

Steven French
Principal Software Engineer (Azure Storage Microsoft)
and member of the Samba team
Outline

- What is POSIX?
- POSIX vs. Linux
- Why SMB3.1.1?
- What are the SMB3.1.1 POSIX/Linux Extensions?
- What works today? And how to try them out?
- Wireshark status
- Improving the documentation ...
- Where do we go from here?
SMB3.1.1 POSIX Extensions status

- For the Linux client POSIX extensions were available experimentally starting in the 5.8 Linux kernel (in 2020) and much improved by the 5.15 kernel (released in October 2021)
- Samba’s “smbclient” tool has support for POSIX extensions
- Ksmbd server has had support for the POSIX extensions for a few years
- Samba server now has partial support for the POSIX extensions
- And there are more ...
What is POSIX?

- POSIX was created in 1988
- Later standardized via “The Austin Group” in the late 90s
- Current is https://pubs.opengroup.org/onlinepubs/9699919799/
- SMB was created four years earlier, in 1984 by Dr. Barry Feigenbaum at IBM
- SMB3.1.1 was created in 2015 and has been extended multiple times
• POSIX is old and the file system calls haven’t changed much

• But Linux keeps evolving
Linux > POSIX

- Linux API is so much bigger than POSIX!
  - Currently huge number of syscalls!
    - Try “git grep SYSCALL_DEFINE”
      - currently 879!
      - 18 more than when we checked at SDC!
      - 500+ are even documented “man syscalls”
    - Linux FS layer alone has 223!
    - Vs. only about 100 POSIX API calls
Linux File System API grows

Recently e.g.: very important feature: “folios” added and many changes to internal APIs as well (netfs, fscache ...) and io_uring continues to improve

<table>
<thead>
<tr>
<th>Syscall name</th>
<th>Kernel Version introduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>epoll_pwait2</td>
<td>5.11</td>
</tr>
<tr>
<td>mount_setattr</td>
<td>5.12</td>
</tr>
<tr>
<td>faccessat2</td>
<td>5.8</td>
</tr>
<tr>
<td>close_range</td>
<td>5.9</td>
</tr>
</tbody>
</table>
Goals: Fast! Easy! Transparent!

- Repeating an older slide about goals of SMB3.1.1:
  - Fastest, most secure general purpose way to access file data, whether cloud or on premises or virtualized
  - Implement all reasonable Linux/POSIX features - so apps don’t know they run on SMB3 mounts (vs. local)
  - As Linux evolves, and needs new features, quickly add to Linux kernel client and Samba and ksmbd
Why Not Other Protocols?

- SMB3.1.1 is easily extensible
- SMB3.1.1 works tightly with a set of protocols which can do more than any other file system protocol
- SMB3.1.1 has the best, most exhaustive set of testcases (not just smbtorture …)
- SMB3.1.1 and related protocols have more documentation (and documentation that has been tested and verified)
- SMB3.1.1 is proven across multiple client types, OS, architectures (and POSIX extensions have been a moving target, done before …) and has been extended before …
- (And don’t forget … SAMBA rocks! And multiple open source server choices that support SMB3.1.1 (including ksmbd and Samba) And cifs.ko is one of most active FS)
Why are extensions needed?

SMB2 and later (including SMB3.1.1) default file semantics are largely based on Windows behavior which differs from POSIX (and Linux) in some ways.
What about SMB1/CIFS? It had Unix Extensions...

- **DO NOT USE OLD INSECURE DIALECTS** (lookup Ned Pyle’s presentations …)
- SMB3.1.1 is faster, more secure, simpler, better
- And broadly implemented by modern clients and servers
- And the SMB3.1.1 POSIX extensions are simpler than the old CIFS Unix Extensions and intended to completely replace their use
What works today without POSIX Extensions

- Normal file and directory operations (open, read, write, fsync, close) to all servers, and hardlinks and even client handled symlinks (“mfsymlinks”), case preserving file name behavior, mapping almost all problematic characters in filenames (“\” is the one exception)

- To most servers:
  - Sparse file operations: setsparse, query allocated ranges, punch hole
  - copy_range and clone_range (clone range is less commonly supported)
  - Special file handling via reparse points (or xattrs ala “sfu”)
  - Xattrs

- Emulation of mode bits via various alternatives (cifsacl, modefromsid)
What can be emulated today without POSIX Extensions

- fcollapse and finsert
- Most delete and rename scenarios (some exceptions is where the rename fails with access denied with rename onto an existing file)
- Most byte range (easier with OFD rather than “posix” BRLs) and whole file lock scenarios
- Most special mode bits (e.g. sticky bit)
But POSIX apps expect

- Case sensitive file and directory names
- Primitive mode bits returned ("0777" perms) rather than a "rich ACL"
- Delete of open files (and these not seen by readdir)
- Rename of open file
- Advisory (rather than mandatory) byte range locks
- 2 additional fields returned by statfs (total/free inodes)
What if POSIX and Windows semantics collide?

If the same file is being used by Windows and Linux clients – how do we deal with semantic differences?

- Lessons from the CIFS (SMB1) Unix Extensions
- What about RichACLs and mode bits?
What about the Apple Mac extensions to SMB3.1.1?

... They only address part of what Linux needs
• Implement AAPL context
  - Improved Mac interop is another benefit
  - Samba even has a vfs_fruit module that adds other interesting features (spotlight integration e.g.)
• Subset of POSIX requirements can be solved
  • kAAPL_SERVER_CAPS = 0x01,
  • kAAPL_SUPPORTS_READ_DIR_ATTR = 0x01,
    • kAAPL_SUPPORTS_OSX_COPYFILE = 0x02,
  • kAAPL_UNIX_BASED = 0x04
  • kAAPL_SUPPORTS_NFS_ACE = 0x08
• kAAPL_VOLUME_CAPS = 0x02,
  • kAAPL_SUPPORT_RESOLVE_ID = 0x01,
  • kAAPL_CASE_SENSITIVE = 0x02
• kAAPL_MODEL_INFO = 0x04 (pad, length, model string)
How can I try the SMB3.1.1 POSIX extensions?

- Client implementations include: Linux kernel (cifs.ko), especially 5.8 kernel and later, and newer versions of Samba’s smbclient (thanks Volker)
- Server versions include Ksmbd server (thanks Namjae) as well as newer Samba server (thanks JRA, Volker, David etc.) …
- Third party implementations also were tested at SNIA SDC SMB3 plugfest
Demo

Some examples of where POSIX Extensions are needed ...
Case Sensitivity – without extensions can open the wrong file ... and owner and mode bits usually set to defaults (not real owner).

```
root@smfrench-ThinkPad-P52:/home/smfrench# ls /mnt1/small-dir -l
total 12
-rwxr-xr-x 1 root root 0 Sep 13 02:42 747-file
-rwxr-xr-x 1 root root 19 Sep 13 02:44 casesensitiveexample
-rwxr-xr-x 1 root root 11 Sep 13 02:44 CaseSensitiveExample
-rwxr-xr-x 1 root root 06 Sep 13 02:44 CASESENSITIVEEXAMPLE
-rwxr-xr-x 1 root root 0 Sep 13 02:12 file1-root
-rwxr-xr-x 1 root root 0 Sep 13 02:41 file-as-smfrench
root@smfrench-ThinkPad-P52:/home/smfrench# cat /mnt1/small-dir/CaseSensitiveExample
mixed case
root@smfrench-ThinkPad-P52:/home/smfrench# cat /mnt1/small-dir/casesensitiveexample
mixed case
root@smfrench-ThinkPad-P52:/home/smfrench# cat /mnt1/small-dir/CASESENSITIVEEXAMPLE
mixed case
root@smfrench-ThinkPad-P52:/home/smfrench# ```
SMB3.1.1 POSIX Extensions are Easy to Understand

- A simple negotiate context, an open context, a new file info level and a new fsinfo level
- Everything else relies on existing SMB3.1.1 features
Demo

Some examples to Samba and KSMBD (kernel server) using the SMB3.1.1 POSIX Extensions
Setting up Samba and ksmbd shares are easy.

NB: Samba requires "smb3 unix extensions = yes" in smb.conf.
Additional Samba server configuration advice

- Build Samba with ".configure.developer"
- To Support for reporting mode bits currently Samba server requires disabling "acl_xattr" (do not enable saving ACLs to xattr this in "vfs objects" on the share if you want to test returning mode bits)
- Consider relaxing the "create mask"
  - smb.conf parameters "create mask = 0777" and "directory mask = 0777"
Mounting from the Linux kernel client

- Remember to add “posix” on mount command
- Also consider “mfsymlinks” if want client only symlinks

```
root@smfrench-ThinkPad-P52:/home/smfrench/mulder-posix/samba# mount -t cifs //local/mnt1 -o username=testuser,password=testpass,mfsymlinks,posix
```

Display Internal CIFS Data Structures for Debugging

CIFS Version 2.38
Features: DFS,FSCACHE,STATS,DEBUG,ALLOW_INSECURE_LEGACY,CIFS_POSIX,UPCALL(SPNEGO),WITNESS
CIFSMaxBufSize: 16384
Active VFS Requests: 0

Servers:
1) ConnectionId: 0x4 Hostname: localhost
Number of credits: 443 Dialect 0x311 posix
TCP status: 1 Instance: 1
Local Users To Server: 1 SecMode: 0x1 Req On Wire: 0
In Send: 0 In MaxReq Wait: 0

Sessions:
1) Address: 127.0.0.1 Uses: 1 Capability: 0x300047 Session Status: 1
Security type: RawNTLMSSP SessionId: 0x35ada477
User: 0 Cred User: 0

Shares:
Can display owner and mode bits

(bug in group owner vs. user owner – fixed now, see next slide)
Query FS Info – includes additional posix fields

```
root@smfrench-ThinkPad-P52:/home/smfrench# mount -t cifs //local -o username=testuser,password=testpass,mfsymlinks,posix
root@smfrench-ThinkPad-P52:/home/smfrench# stat -f /mnt1
File: "/mnt1"
  ID: c7df5aa0f1e89eff Namelen: 255  Type: smb2
  Block size: 4096  Fundamental block size: 4096
  Blocks: Total: 139092115  Free: 48993190  Available: 48993190
  Inodes: Total: 278320128  Free: 273211966
root@smfrench-ThinkPad-P52:/home/smfrench# ```
Query FS Info – additional POSIX fields

Samba server works now too!
More details: Local vs non-POSIX SMB3.1.1 (example to Samba server)
More details: Local vs new POSIX SMB3.1.1 (example to Samba server)
Better performance (POSIX QFS Info now compounded)

Before:

Now:
SMB3.1.1 POSIX Extensions details
Additional information can be found at:

- For the SMB3.1.1 POSIX Extensions see the Samba wiki at: [SMB3-Linux – SambaWiki](https://wiki.samba.org/index.php/SMB3-Linux). Contributions to improve this page are welcome.
- Information on the older SMB1 extensions can be found at [UNIX Extensions – SambaWiki](https://wiki.samba.org/index.php/UNIX_Extensions) e.g. and in the older SNIA doc.
NetBIOS Session Service

SMB2 (Server Message Block Protocol version 2)

SMB2 Header

Negotiate Protocol Request (0x00)

- StructureSize: 0x0024
  - Dialect Count: 4
- Security Mode: 0x01, Signing enabled
  - Reserved: 0000
- Capabilities: 0x00000077, DFS, LEASING, LARGE MTU, PERSISTENT HA
  - Client Guid: 032f6ff0-493c-44d-8b01-425c86949469
  - NegotiateContextOffset: 0x0070
  - NegotiateContextCount: 4
    - Reserved: 0000
    - Dialect: 0x0210
    - Dialect: 0x0300
    - Dialect: 0x0302
    - Dialect: 0x0311
- Negotiate Context: SMB2_PREAUTH_INTEGRITY_CAPABILITIES
- Negotiate Context: SMB2_ENCRYPTION_CAPABILITIES
- Negotiate Context: Unknown Type: (0x5)
  - Negotiate Context: SMB2_POSIX_EXTENSIONS_CAPABILITIES
    - Type: SMB2_POSIX_EXTENSIONS_CAPABILITIES (0x0100)
    - DataLength: 16
    - Reserved: 00000000
    - POSIX Reserved: 0x5025ad93
NetBIOS Session Service

SMB2 (Server Message Block Protocol version 2)

- SMB2 Header
  - Create Request (0x05)
    - StructureSize: 0x0039
      - Oplock: No oplock (0x00)
      - Impersonation level: Impersonation (2)
      - Create Flags: 0x0000000000000000
      - Reserved: 0000000000000000
      - Access Mask: 0x00000100
      - File Attributes: 0x00000000
      - Share Access: 0x00000000, Read, Write, Delete
      - Disposition: Create (if file exists fail, else create it) (2)
      - Create Options: 0x00000001
  - Filename: 0750
    - Blob Offset: 0x00000078
      - Blob Length: 8
      - Blob Offset: 0x00000088
      - Blob Length: 40
  - ExtraInfo SMB2_POSIX_CREATE_CONTEXT
    - Chain Element: SMB2_POSIX_CREATE_CONTEXT "5025ad93-b49c-e711-"
      - Chain Offset: 0x00000000
      - Tag: 5025ad93-b49c-e711-b423-03de960bcd7c
        - Blob Offset: 0x00000010
          - Blob Length: 16
        - Blob Offset: 0x00000020
          - Blob Length: 4
          - Data: POSIX Create Context request
            - POSIX perms: 0740
What Next?
Missing Features and More Linux optimizations ...
What Next?

- Examine the xfstest skips (and failures) in much detail and add small incremental changes
  - “xfstests” is the standard Linux fs functional test suite and no one file system can pass all tests due to various fs optional features.
  - Some can be emulated some need new flags
- Where that is not possible, consider adding new POSIX extensions version (simply adding additional uuid to the POSIX negotiate context)
What Next?

- What about minor extensions to reduce roundtrips and provide better/safer emulation?
  
  - Fcollapse and finsert are two examples
  - NTFS fsctls like FSCTL REARRANGE_FILE and SHUFFLE_FILE could help if available over SMB3
  - What about exposing Windows’s FILE_FLAG_POSIX_SEMANTICS
  - More compounding can help too
  - Add fsctl for rename swap (rename exchange)?
Improving server side symlink handling

- When encountering client only symlinks (mfsymlinks) this is not an issue.
- But for server side symlinks rely on server supporting reparse points to report symlinks.
- This can be improved by simply improving the handling of “stopped on symlink” error (which includes sufficient information in most cases to avoid needing to query reparse points).
Examples from xfstest investigations

- Add support for renameat2 and rename exchange
- POSIX ACLs (can be emulated and there is pushback on implementing primitive POSIX ACLs)
- Support for additional chattr flags ("immutable" and "noatime" updates e.g.)
- `fallocate` --collapse-range
- Dedupe support
- Defragmentation support (may require VFS changes)
Examples from xfstest investigations

- Richacl support (tests 362 through 370) ??
- O_TMPFILE support (emulatable, but VFS changes would help)
- FITRIM support (may be emulatable)
- Quota support (may be emulatable already)
- Support for NFS export (nfs server on smb3 mounts)
- Case sensitive xattrs (EAs)
- SELinux support
Examples from xfsTest investigations

- Support for online ‘label manipulation’ (see e.g. xfsTest generic/492)
- Support for casefolding ("chattr +F")
- Would native (rather than emulated) BSD flock (whole file lock) support help?
More details (with example xfstest #)

- atime options irrelevant (test 003)
- O_TMPFILE (generic/004)
- Defragmentation (018)
- Renameat2 (025)
- POSIX ACLs (026)
- FITRIM (038)
- Metadata journaling (049)
- Freezing fsctl (068) - https://lwn.net/Articles/287435/
More details (continued)

- Chattr +ia (079) (“immutable”, “append only”)
- Chattr +A (277) (“no atime updates”)
- Linux disk quotas (082)
- Security (093) and trusted (097) xattr namespaces
- preallocated extent not marked with FIEMAP_EXTENT_UNWRITTEN (094)
- Dedupe (121)
- Advisory locks (131)
More details (continued)

- suid/sgid bits are cleared after direct write (test 355)
- Richacls (362)
- Encryption support (395)
- Timestamp bounds unknown (402)
- chattr +d (“nodump”) (424)
- Information about fiemap of attribute fork (425)
- NFS export (open by inode #) (426)
- Backslash in name (“Key urkmoo does not exist for FAKESLASH test??” in test 453)
More details (continued)

- Conflicting xattrs (test 454)
- XATTR_REPLACE ( test 486 )
- xfs_io label (492)
- Lsattr -d (508)
- Xattrs with slashes in name (523)
- Casefolding support (556)
- Dupremove utility (559), actton utility (596)
- Fsverity (571)
Thank you for your time

- A very exciting time for ...