The Road to the New VFS

Ralph Böhme, Samba Team, SerNet

2021-05-06

Road to a modern VFS for SMB2+

The Road to a modern VFS for SMB2+

The effort to modernize Samba's VFS interface has reached a major milestone with the release of Samba 4.14:

- This is an ongoing effort since a few years
- Initially driven by Jeremy Allison
  - standardizing path based filesystem syscalls on `*at()` variants
  - eg `openat()` instead of `open()`

We recently changed the fileserver code to use file handles instead of paths as often as possible

- eg `fstat()` instead of `stat()`

How did we get there?

SerNet

Some History

SMB1 Fallacies: Pervasive use of Paths

A path by any other name would smell as unpleasing.

Most metadata operation (get and set) in SMB1 can be done on paths:

- Path processing is complex and slow
  - one of the core function `unix_convert()` had more then 800 lines (before we refactored it last year)
  - plus several thousand lines of code in callees

So what's wrong with paths. Things to consider:

- Charset conversion
- Mangling non-Windows compatible paths to Windows compatible
- DFS paths
- Previous version paths (with "@GMT-..." tokens in the path)
- Case insensitive semantics
- Named streams support
- *Yuck!*

SerNet

Enter SMB2

By contrast, SMB2+ is a purely handle based protocol

- `SMB2 Create` request takes a pathname
- Everything else operates on a handle returned by `SMB2 Create`
- . . . with a few exceptions:
  - `QueryInfo(NormalizedNameInformation)` returns a full pathname
  - `QueryDirectory()` returns relative pathnames
  - `SetInfo(File{Link,Rename}Information)` takes a full target pathname

Deprecation of SMB1 in 4.11

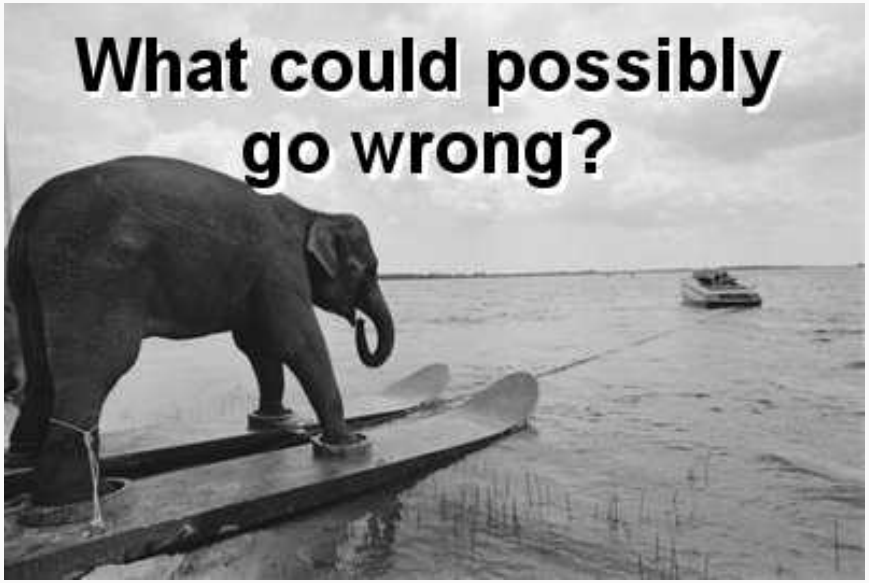
- The world has moved away from SMB1
- So did we, SMB1 is now disabled by default
- Not yet removed completely: used in tests

SerNet

The Idea

The idea: a (mostly) handle-based VFS for the SMB2+ World

- Streamline the VFS interface to be (mostly) handle-based
- No more `SMB_VFS_STAT()`, only `SMB_VFS_FSTAT()`
  - or `SMB_VFS_FGETXATTR()`, not `SMB_VFS_GETXATTR()`
  - or `SMB_VFS_FGET_DOS_ATTRIBUTES()`, not `SMB_VFS_GET_DOS_ATTRIBUTES()`
  - or `SMB_VFS_FGET_NT_ACL`, not `SMB_VFS_GET_NT_ACL()`
  - . . . and so on
- Perfect match for the SMB2+ protocol



SerNet

VFS Functions by Category

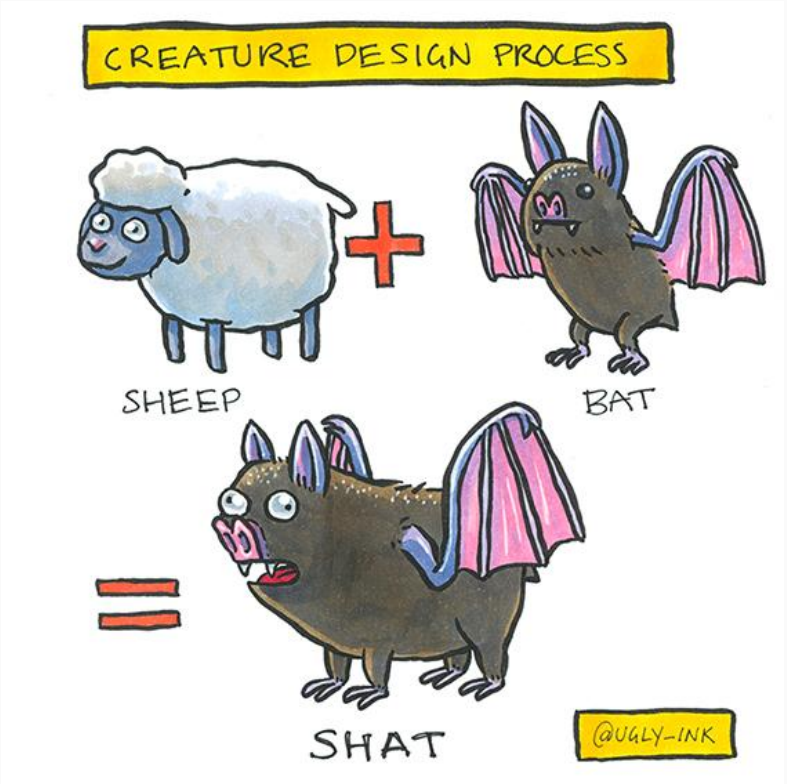
VFS Function Categories	Number
Path based	21
Path based namespace changing (create, delete, . . .)	8
Handle based	50
DFS-related	3
Disk operations	9
Pure path to path translation	4
Special cases (eg FileIDs)	6
Sum	101

Table 1: VFS interface functions grouped by category

SerNet

The Design Squad

Stefan Metzmacher
Volker Lendecke
Jeremy Allison
Ralph Böhme



SerNet

Challenge: Permissions

Opening a file handle requires at least `O_RDONLY`

- If you want a file handle on Linux, you call `open[at](file, mode)`
- You request an access mode of either `O_RDONLY`, `O_WRONLY` or `O_RDWR`
- path based `stat("file")` only needs only `"x"` on parent directory, but ...
- `fd = open("file", O_RDONLY)` in oder to `fstat(fd)` needs `"r"` on `"file"`
- Currently if the client only requests `READ_ATTRIBUTES` access
  - which is the access right corresponding to reading a file's metadata (ie `stat()`)
  - then Samba doesn't open a file handle but uses path based syscall (ie `stat()`)

Kernel oplocks

- `O_RDONLY` triggers a kernel oplock break

SerNet

Uh-oh!

SerNet

Oh path, oh path!

The fix: Linux `open()` flag `O_PATH`

- Available since since Linux 2.6.39 (May 2011), soon in FreeBSD
- Returns a file handle that acts as a mere path "reference"
  - I coined the term pathref for referring to these guys in Samba
- Doesn't need `"r"` on object, only `"x"` on the parent directory

Limitted number of syscalls are allowed

- the important one from Samba's perspective: `fstat()`
- Can read the inode metadata but not modify it
- Can't be used for any sort of IO
- Can also be used as dirfd for `*at()` syscalls

Fallback to open-as-root if `O_PATH` is not available

- root-opened fds are "guarded", access only via accessor functions
  - `fsp_get_pathref_fd(fsp)`, `fsp_get_io_fd(fsp)`
  - `fsp_get_pathref_fd(fsp)` must be auditted

SerNet

Oh path, oh path! Cont.

But wait, Samba needs more then `fstat()`:

- Samba needs to read ACLs and xattrs
- But both can't be retrieved via `O_PATH` handles
- Use the `/proc/self/fd/FD` trick:
  - use path based version with path "/proc/self/fd/%d"
  - replacing %d with the `O_PATH` fd

Example Code: Fallback to getxattr

```
if (somehow_figure_out_fd_is_opath_fd(fd)) {
        char buf[PATH_MAX];
        sprintf(buf, "/proc/self/fd/%d", fd);
        getxattr(buf, ...);
} else {
        fgetxattr(fd, ...);
}
```

Fine Print

- `/proc/self/fd` currently Linux only, elsewhere fallback to path based access
- Which is the same net result as in pre `O_PATH` Samba

SerNet

Challenge: SMB1 Heritage

Due to paths being used heavily in the protocol we have pervasive use of paths in the Samba codebase

- we want to convert 21 path based VFS functions, ...
- that are used at a few hundred places in the codebase and ...
- will we need a file handle in all those places

Samba high-level code "degrades" handles to path-based access in many places

- So in theory we have a handle (`fsp` in Samba parlance)
- But use path attached to `fsp` (`fsp->fsp_name`) with path based VFS function
- Or need to call a VFS function on the parent directory of `fsp->fsp_name`
- Sometimes paths get passed to functions, not a handle – even though we have one

SerNet

SMB1 Heritage: Turning pathnames into pathrefs

How to get a file handle? The old way

Samba's internal file handle structure is of type `struct files_struct` and all variable pointing to objects of such type are typically called `fsp`'s.

- `fsp`'s are returned by `SMB_VFS_CREATE_FILE()`
- this is the 1000 pounds Gorilla of the VFS functions zoo
- calls on to `SMB_VFS_OPENAT()` to open the low-level fd
- then goes through Samba's NTFS Windows emulation (eg `locking.tdb`)

New, additional way to get a file handle

We added new helper function `openat_pathref_fsp()` which skips the NTFS emulation logic and calls `SMB_VFS_OPENAT()` with `O_PATH`

- I called the resulting fsp's pathref fsps
- pathref fsps can be upgraded to "full" fsps
  - fd is reopened
  - NTFS Windows emulation code is run
  - This happens when passing a pathref fsp to `SMB_VFS_CREATE_FILE()`
    - embedded in the filename that gets passed to `SMB_VFS_CREATE_FILE()` (see next slide)

SerNet

SMB1 Heritage, cont.

Client supplied paths are processed by the core function `filename_convert()`

- Returs a pointer to an object of type `struct smb_filename`.
  - Variables are typically called `smb_fname`.
- `filename_convert()` is updated to call `openat_pathref_fsp()`
- storing the resulting pathref `fsp` inside `struct smb_filename`
  - `smb_fname->fsp`
- As a result the whole codebase has immediate access to a file handle.
- Which allows converting the whole codebase to use handle based VFS functions in a piecemeal fashion.

SerNet

VFS Functions by Category, todo

VFS Function Categories	Number	Todo
Path based	21	Use `O_PATH` pathrefs
Path based namespace changing (create, delete, ...)	8	-
Handle based but not allowed on `O_PATH` fds	8	Use /proc/fd
Handle based	42	-
DFS-related	3	-
Disk operations	9	-
Pure path to path translation	4	-
Special cases (eg FileIDs)	6	-
Sum todo	29	

Table 2: VFS interface functions by category needing changes

SerNet

The Construction Squad

Construction Squad

Noel Power
Samuel Cabrero
Jeremy Allison
Ralph Böhme

SerNet

VFS Conversion Status

Status

VFS Function Category	Done	Todo
Path based	6	15
Handle based but not allowed on O_PATH	8	0

Table 3: VFS Conversion Status

SerNet

Links

- https://wiki.samba.org/index.php/The_New_VFS
- The New VFS, long version of this presentation in the Samba sources

SerNet

Q&A

Thank you!
Questions?

Ralph Böhme
slow@samba.org
rb@sernet.de

SerNet