Samba/Microsoft alignment: possible future directions

Tom Talpey

Microsoft

Outline

- This talk was going to be about Durable Remote Memory
  - Until I was informed it was the keynote ☺
- So it’s first going to cover things from Samba and Microsoft that, together, we:
  - (have) **D**elivered
  - (are) **D**eveloping
  - (will further) **D**iscuss
- All of which, I hope, may lead to our continued collaboration.

Delivered

Observations on the state of Microsoft/Samba SMB3

SMB3 from Microsoft and Samba

- SMB3 is the key storage protocol for Windows Interoperability
- Servers
  - Windows – SMB 3.1.1
    - In use for file *and block* storage services, with RDMA
  - Azure – SMB 3.0
    - Azure File service for VMs and applications, at scale
  - Samba – SMB 3.1.1
    - File service including rich related enterprise features
- Clients
  - Windows – SMB 3.1.1
  - Samba/Linux – SMB 3.1.1

Samba / Windows Interop

- Samba interoperation with Windows
  - Samba Server works with Windows SMB clients
  - Samba Client works with Windows SMB servers
  - Some features (RDMA, persistent handle recovery) slow to appear
- But, Windows is now only one aspect of the SMB3 world

SMB3 in the Cloud

- Linux guests are the dominant presence in Azure Cloud
- Azure Files service supports application file access
- To access Azure Files, Linux CIFS Client requires
  - Robust persistent handle recovery
  - Scalable performance at cloud service latencies
  - True Posix semantics expected by applications
    - SMB3 and Azure Files need these too!
- Strong desire to close this gap

Azure Files

- "net use //thecloud" – SMB3.0 service on cloud port 445 (SMB2.1 also)
- In support of cloud Windows and Linux guests
  - https://docs.microsoft.com/en-us/azure/storage/storage-dotnet-how-to-use-files
  - https://docs.microsoft.com/en-us/azure/storage/storage-how-to-use-files-linux
- Windows guests – working
- Linux guests – some assembly required
- SMB3 readiness for Posix in Linux client problematic for apps
- NFS support a popular request

And oh by the way

- SMB1-based attacks are increasing
- It’s fast-approaching the time to go beyond deprecation
- Many Windows SKUs already disable SMB1 by default
- No supported Windows version requires it
  - Windows XP ended the SMB1-only era

- It’s time for Samba to discuss this again
  - Server – drop SMB1 support in future 4.x?
  - Client - refactor cifs.ko to SMB1 legacy, and innovate with a new smb3.ko?

Microsoft Linux (also FreeBSD)

- Linux Integration Services (Enlightenment drivers)
  - Efficient guests on Hyper-V (and in Cloud)
- HVSocket
  - Efficient, secure guest access to Hyper-V host partition
  - AF_HYPERV socket type
  - https://docs.microsoft.com/en-us/virtualization/hyper-v-on-windows/user-guide/make-integration-service
- Samba specifically leverage these features?
  - Open to a discussion

Developing

Development at Microsoft with direct benefits to Samba

Linux Subsystem for Windows

- "Ubuntu on Windows"
  - Actual Ubuntu (Xenial) binaries, on modified Xenial kernel (4.4.0)
  - Uses Windows personality support to emulate system calls
    - Roots in former NT Posix Subsystem, SUA/SFU, etc
  - Exposes local NTFS filesystems
- Relationship to Samba: "apt install samba"
  - Which mostly works!
  - Server issues
    - Port 445 collision, privileges
  - Client issues
    - Kernel module support

LSW Samba client

Cifs-utils ☺

```
ttalpey@TTALPEY1:~$ sudo apt install cifs-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  keyutils libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libldb1 libpython-stdlib libpython2.7
  libpython2.7-minimal libpython2.7-stdlib libtalloc2 libtdb1 libtevent0 libwbclient0 python python-crypto python-ldb
  python-minimal python-samba python-talloc python-tdb python2.7 python2.7-minimal samba-common samba-
common-bin
  samba-libs
Suggested packages:
  smbclient winbind cups-common python-doc python-tk python-crypto-dbg python-crypto-doc python2.7-doc
binutils
  binfmt-support heimdal-clients
The following NEW packages will be installed:
  cifs-utils keyutils libavahi-client3 libavahi-common-data libavahi-common3 libcups2 libldb1 libpython-stdlib
  libpython2.7 libpython2.7-minimal libpython2.7-stdlib libtalloc2 libtdb1 libtevent0 libwbclient0 python
  python-crypto python-ldb python-minimal python-samba python-talloc python-tdb python2.7 python2.7-minimal
  samba-common samba-common-bin samba-libs
0 upgraded, 27 newly installed, 0 to remove and 0 not upgraded.
Need to get 12.7 MB of archives.
After this operation, 56.3 MB of additional disk space will be used.
Do you want to continue? [Y/n]
…
```

Client ☹

```
ttalpey@TTALPEY1:~$ sudo mount.cifs //nas.local/public ./mnt
Password for root@//nas.local/public:
mount error: cifs filesystem not supported by the system
mount error(19): No such device
Refer to the mount.cifs(8) manual page (e.g. man mount.cifs)
```

LSW Samba server

Server ☺

```
root@TTALPEY1:~# apt install samba
Selecting previously unselected package samba-vfs-modules.
Preparing to unpack .../samba-vfs-modules_2%3a4.3.11+dfsg-0ubuntu0.16.04.6_amd64.deb ...
Unpacking samba-vfs-modules (2:4.3.11+dfsg-0ubuntu0.16.04.6) ...
Processing triggers for man-db (2.7.5-1) ...
Processing triggers for libc-bin (2.23-0ubuntu7) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for ufw (0.35-0ubuntu2) ...
Setting up python-dnspython (1.12.0-1) ...
Setting up tdb-tools (1.3.8-2) ...
update-alternatives: using /usr/bin/tdbbackup.tdbtools to provide /usr/bin/tdbbackup (tdbbackup) in auto mode
Setting up libfile-copy-recursive-perl (0.38-1) ...
Setting up update-inetd (4.43) ...
Setting up samba (2:4.3.11+dfsg-0ubuntu0.16.04.6) ...
Adding group `sambashare' (GID 116) ...
Done.
invoke-rc.d: could not determine current runlevel
invoke-rc.d: could not determine current runlevel
invoke-rc.d: could not determine current runlevel
Setting up attr (1:2.4.47-2) ...
Setting up libaio1:amd64 (0.3.110-2) ...
Setting up samba-dsdb-modules (2:4.3.11+dfsg-0ubuntu0.16.04.6) ...
Setting up samba-vfs-modules (2:4.3.11+dfsg-0ubuntu0.16.04.6) ...
Processing triggers for libc-bin (2.23-0ubuntu7) ...
Processing triggers for systemd (229-4ubuntu16) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for ufw (0.35-0ubuntu2) ...
root@TTALPEY1:~#
```

Service ☹

```
ttalpey@TTALPEY1:~$ sudo service samba start
* Starting NetBIOS name server nmbd
[ OK ]
 * Starting SMB/CIFS daemon smbd
[ OK ]
root@TTALPEY1:~# cat /var/log/smbd.log
[2017/05/03 07:19:57.885986,  0]
../lib/util/become_daemon.c:124(daemon_ready)
  STATUS=daemon 'smbd' finished starting up and ready to serve connections
[2017/05/03 07:19:57.917769,  0]
../source3/lib/util_sock.c:357(open_socket_in)
  open_socket_in(): setsockopt: SO_REUSEPORT = true on port 445 failed with
error = Protocol not available
[2017/05/03 07:19:57.917972,  0]
../source3/lib/util_sock.c:396(open_socket_in)
  bind failed on port 445 socket_addr = ::.
  Error = Permission denied
[2017/05/03 07:19:57.918129,  0]
../source3/smbd/server.c:709(smbd_open_one_socket)
  smbd_open_once_socket: open_socket_in: Permission denied
[2017/05/03 07:19:57.918498,  0]
../source3/lib/util_sock.c:357(open_socket_in)
  open_socket_in(): setsockopt: SO_REUSEPORT = true on port 139 failed with
error = Protocol not available
[2017/05/03 07:19:57.918935,  0]
../source3/smbd/server.c:709(smbd_open_one_socket)
  smbd_open_once_socket: open_socket_in: Permission denied
```

Samba LSW Opportunities

- Any opportunities for Samba in the above?
  - No commitments below, just ideas for discussion
- Samba Server on LSW – maybe not?
  - Conflicts with Windows SMB3, at a minimum
  - But it sure is an interesting question!
- Samba Client on LSW – maybe yes?
  - New remote filesystem support for LSW apps?
  - Needs SMB3/CIFS module with kernel networking support, or…
  - SMB3/CIFS module with Hyper-V Socket
  - SMB3/CIFS module with guest RDMA!?

SMB Direct Linux

- Microsoft prototyping SMB Direct support for Linux!
- Client-only
  - Not contemplating doing a server implementation
- Simplifying initial principles:
  - Connections are made via RDMA directly – no TCP, no multichannel
  - Send/receive transfers only (no direct data placement via RDMA Read / RDMA Write)
    - Transfers up to ~1MB are supported with SMB Direct fragmentation

SMB Direct

- Initial implementation
  - Connects, and negotiates SMB3.1.1 on Windows Server RDMA connection
  - Transfers data successfully
  - Currently, fails on sustained file copy (server detects MID out-of-range)
- Not (yet) supported:
  - SMB Direct placement (RDMA Read/Write)
    - Requires explicit memory registration and care with RDMA verbs (completions)
  - Full multichannel, with fallback/forward
    - Requires significant client transport architecture work

Discussions

The future of SMB3 Unix interop

Unix (Posix) Extensions

- The key to tying it all together
- And, long-overdue
- Do we now have sufficient understanding of requirements?
- Of a protocol?
- It’s time to move forward.

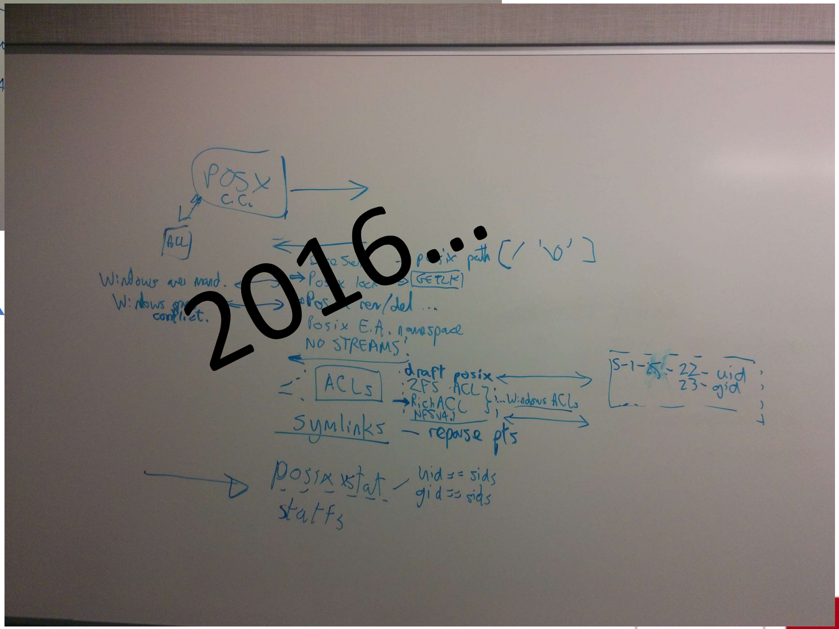
History - SMB2 Unix Extensions

- SMB1 Extensions – SNIA 2002
- SMB2 Unix Extensions "Project" – 2010
  - https://www.snia.org/sites/default/orig/sdc_archives/2010_presentations/wednesday/TomTalpey-Unix_ExtensionsForSMB2.pdf
- ~6 years of discussion
  - Many valid reasons for the delay, but it's probably time
  - Apple, other extensions also relevant
  - Goal: a single standard (ideally)

Whiteboards of earlier discussions

- From Tom's phone @ previous plugfests
  - IMG*20151002*
  - IMG*<u>2016</u>*0622
- Note: the lists pretty much fit on a single board
- Are we ready to write it down?
  - Next month at the Redmond plugfest?
  - This Fall at the SMB Plugfest / SDC?
- The lights are still on at unixsmb2.org
  - Thanks, Chris!

2015…
CAPABILITIES (SERVER + SHARE)
Y CASE SENSITIVITY / SHARE LEVEL
N POSIX ADVISORY LOCKING
N POSIX OPEN/UNLINK (OTHER THAN LOCKING) + RENAME
N NATIVE UTF8 PATH W/O MAPPING
Y UID/GID/MODE
SOME POSIX INFO
N NATIVE POSIX A
APPLE STUFF
DISCOVERY
CREATE OPTIONS
SET/GET INFO PER HANDLE



2017!

Durability

Another suggestion for future engagement

Persistent Memory

- Presence on current and future server motherboards
- PMEM/DAX emergence
- RDMA standardization
- Push Mode
- RDMA Extensions
- SNIA NVMP

Windows PMEM Support

- Persistent Memory is supported in Windows 10 and Windows Server 2016
  - PM support is foundational in Windows and is SKU-independent
- Support for JEDEC-defined NVDIMM-N devices available in
  - Windows Server 2016
  - Windows 10 (Anniversary Update – Fall 2016)
- Access methods:
  - ✓ Direct Access (DAX) Filesystem
    - Mapped files with load/store/flush paradigm
    - Cached and noncached with read/write paradigm
  - ✓ Block-mode ("persistent ramdisk")
    - Raw disk paradigm
  - ✓ Application interfaces
    - Mapped and traditional file
    - NVM Programming Library
  - "PMEM-aware" open coded

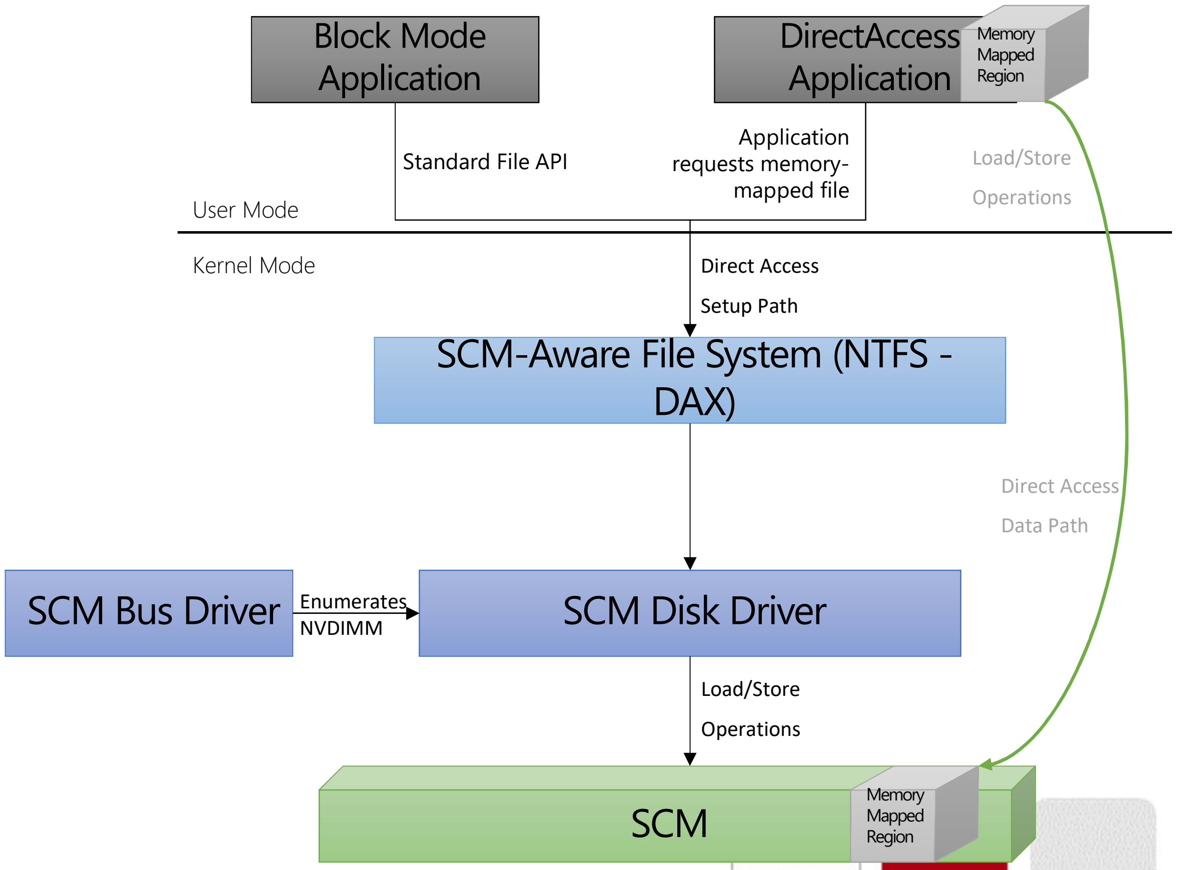
Direct Access Architecture (Windows)

Overview

- Support in Windows Server 2016 and Windows 10 Anniversary Update (Fall 2016)
- App has direct access to Storage Class Memory (SCM/Pmem) via existing memory-mapping semantics
- Updates directly modify SCM, Storage Stack not involved
- DAX volumes identified through new flag

Characteristics

- True device performance (no software overhead)
- Byte-Addressable
- Filter Drivers relying on I/O may not work or attach – no I/O, new volume flag
- AV Filters can still operate (Windows Defender already updated)



IO in DAX mode (Windows)

- Memory Mapped Access
  - This is true zero-copy access to storage
    - An application has direct access to persistent memory
  - Important → No paging reads or paging writes will be generated
- Cached IO Access
  - The cache manager creates a cache map that maps directly to PM hardware
  - The cache manager copies directly between user's buffer and persistent memory
    - Cached IO has one-copy access to persistent storage
  - Cached IO is coherent with memory mapped IO
  - As in memory mapped IO, no paging reads or paging writes are generated
    - No Cache Manager Lazy Writer thread
- Non-Cached IO Access
  - Is simply converted to cached IO by the file system
    - Cache manager copies directly between user's buffer and persistent memory
  - Is coherent with cached and memory mapped IO

Backward App Compatibility on PM Hardware

- Block Mode Volumes
  - Maintains existing storage semantics
    - All IO operations traverse the storage stack to the PM disk driver
    - Sector atomicity guaranteed by the PM disk driver
    - Has shortened path length through the storage stack to reduce latency
      - No storport or miniport drivers
      - No SCSI translations
  - Fully compatible with existing applications
  - Supported by all Windows file systems
  - Works with existing file system filters
  - Block mode vs. DAX mode is chosen at format time

Performance Comparison (WS2016)

4K random writes

1 Thread, single core

	IOPS	Avg Latency (ns)	MB / Sec
NVMe SSD	14,553	66,632	56.85
Block Mode NVDIMM	148,567	6,418	580.34
DAX Mode NVDIMM	1,112,007	828	4,343.78

Similar Linux Facilities

- DAX
  - Same name, different implementation
- NOVA
  - UCSD project http://nvsl.ucsd.edu/
- NVML
  - Same open source library http://pmem.io/nvml/
- Significant industry convergence

Going Remote

- One local copy of storage isn't storage at all
  - Basically, temp data
- Enterprise-grade storage requires replication
  - Multi-device quorum
  - In addition to integrity, privacy, manageability, … (requirements vary)
- Remote access is required

- Pmem value is all about LATENCY
  - Single digit microsecond remote latency goal
  - Which btw is 2-3 orders of magnitude better than today's block storage
    - We can take steps to get there, with great benefit at each

➢ Use RDMA
  - Full latency benefit motivates an RDMA protocol extension

RDMA Protocols

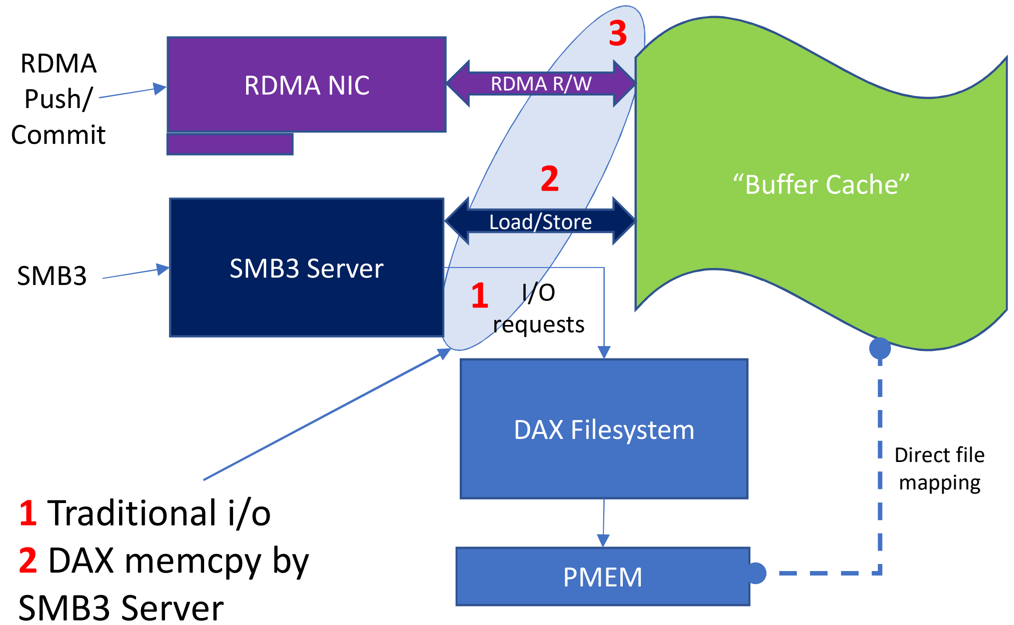
- Need a remote guarantee of Durability
- RDMA Write alone is not sufficient for this semantic
- An extension is required
  - Proposed “RDMA Commit”, a.k.a. “RDMA Flush”
- Executes like RDMA Read
  - Ordered, Flow controlled, acknowledged
  - Initiator requests specific byte ranges to be made durable
  - Responder acknowledges only when durability complete
  - Strong consensus on these basics
- Being discussed in IBTA, SNIA and other venues
  - Details being worked out
  - Scope of durability: region-based, region-list-based, connection, all under discussion
    - Connection scope seems most efficient for implementations
  - Additional semantics possible (signaling, ordering, integrity, …)

RDMA-Aware Storage Protocol Use

- SMB3/SMB Direct
  - With future potential “Push Mode”
- NFS/RDMA
  - And future pNFS/RDMA layout (Christoph Hellwig proposing)
- Other
  - Commit to any remotely-mappable device, e.g. NVMe with a PCIe BAR
  - Anything that can be memory-registered and accessed via RDMA

Example: Going Remote – SMB3

- SMB3 RDMA and "Push Mode" discussed at previous SNIA Storage Developers Conferences
- Enables zero-copy remote read/write to DAX file
  - Ultra-low latency and overhead
- 2, 3 can be enabled even *before* RDMA Commit extensions become available, with slight extra cost



1 Traditional i/o
2 DAX memcpy by SMB3 Server
3 Push Mode direct from RDMA NIC

The (Near?) Future

- I'd like to see Samba support DAX on Linux
  - Initially, as an ordinary filesystem ("1" above)
    - A fast one since it's RAM, albeit block mode
  - Ideally, with r/w access via memory-mapped files ("2" above)
    - Way faster since via memcpy not block driver
- Eventually, via push-mode RDMA ("3" above)
  - SMB3 implementation potentially simple (one FSCTL and some leasing rules)
  - https://www.snia.org/sites/default/files/SDC/2016/presentations/persistent_memory/Tom_Talpey_Low_Latency_Remote_Storage_A_Full-stack_View.pdf

THANK YOU