

# Samba Filesharing

Michael Adam

Samba Team / SerNet

September 25, 2014

# Samba File Serving Topics

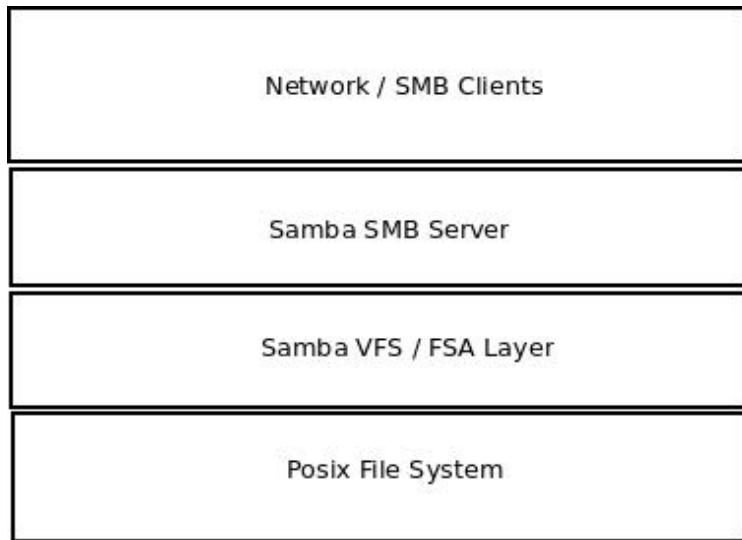
## Core File Server:

- ▶ SMB features (SMB3...)
- ▶ Performance
- ▶ Interop (Protocols, NFS, AFP, ...)
- ▶ ...

## Moreover:

- ▶ Auth/Domain Member
- ▶ RPC server
- ▶ ...

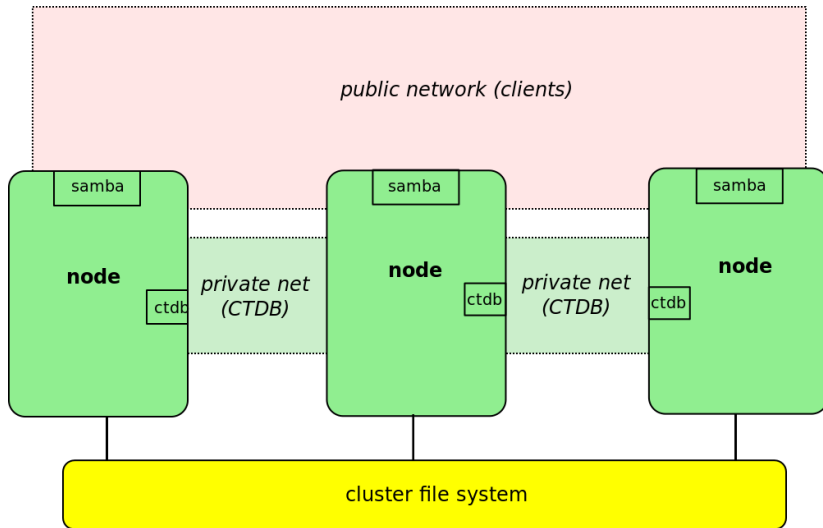
## File Server Layout/Scope



# SMB Features

- ▶ SMB 2.0:
  - ▶ durable file handles [4.0]
- ▶ SMB 2.1:
  - ▶ multi-credit / large mtu [4.0]
  - ▶ dynamic reauthentication [4.0]
  - ▶ leasing [WIP++]
  - ▶ resilient file handles [ever?]
- ▶ SMB 3.0:
  - ▶ new crypto (sign/encrypt) [4.0]
  - ▶ secure negotiation [4.0]
  - ▶ durable handles v2 [4.0]
  - ▶ persistent file handles [planning]
  - ▶ multi-channel [WIP+]
  - ▶ SMB direct [designed/starting]
  - ▶ cluster features [designing]
    - ▶ witness [WIP]
  - ▶ storage features [WIP]
- ▶ SMB 3.1: [WIP]

# Clusterd Samba / CTDB (SOFS since 2007)



Technical Details...

## Multi-Channel - Windows/Protocol

- ▶ find interfaces with interface discovery:  
FSCTL\_QUERY\_NETWORK\_INTERFACE\_INFO
- ▶ bind additional TCP (or RDMA) connection (channel) to established SMB3 session (session bind)
- ▶ bind (TCP) connections of same quality
- ▶ bind only to a single node
- ▶ replay / retry mechanisms, epoch numbers

# Multi-Channel - Samba

- ▶ samba/smbd: multi-process
  - ▶ process  $\Leftrightarrow$  tcp connection
  - ▶  $\Rightarrow$  transfer new connection to existing smbd
  - ▶ use fd-passing (sendmsg/recvmmsg)
- ▶ preparation: messaging rewrite using unix dgm sockets with sendmsg [DONE,Volker]
- ▶ add fd-passing [WIP]
- ▶ transfer connection already in negprot (ClientGUID) [TODO]
- ▶ implement channel epoch numbers [started]
- ▶ implement interface discovery [TODO]



# SMB Direct (RDMA)

- ▶ windows:
  - ▶ requires multi-channel
  - ▶ start with TCP, bind an RDMA channel
  - ▶ reads and writes use RDMA write/read
  - ▶ protocol/metadata via send/receive
- ▶ wireshark dissector: [DONE (Metze)]
- ▶ samba (TODO):
  - ▶ prereq: multi-channel / fd-passing
  - ▶ buffer / transport abstractions [TODO]
  - ▶ central daemon (or kernel module) to serve as RDMA "proxy"  
(libraries: not fork safe and no fd-passing)

## SMB Direct (RDMA) - Plan

- ▶ smbd-d (?) listens for RDMA connection
- ▶ main smbd listens for TCP connection
- ▶ main smbd listens (for RDMA) via unix socket connect to smbd-d
- ▶ client connects via TCP → smbd forks child smbd (c1)
- ▶ client connects via RDMA to smbd-d
- ▶ smbd-d notifies main smbd and transfers connection info
- ▶ smbd forks child (c2) that inherits connection to smbd-d
- ▶ c2 smbd passes [connection to smbd-d] to c1 (via ClientGUID) and exits
- ▶ c1 establishes mmap area with smbd-d
- ▶ client does rdma calls to smbd-d
  - ▶ metadata and protocol calls are transferred via socket to tcp-smbd
  - ▶ rdma read/write directly to tcp-smbd via mmap area

# Clustering Concepts (Windows)

- ▶ Cluster:
  - ▶ (“traditional”) failover cluster (active-passive)
  - ▶ protocol: SMB2\_SHARE\_CAP\_CLUSTER
  - ▶ Windows:
    - ▶ runs off a cluster (failover) volume
    - ▶ offers the Witness service
- ▶ Scale-Out (SOFS):
  - ▶ scale-out cluster (all-active!)
  - ▶ protocol: SMB2\_SHARE\_CAP\_SCALEOUT
  - ▶ no client caching
  - ▶ Windows: runs off a cluster shared volume (implies cluster)
- ▶ Continuous Availability (CA):
  - ▶ transparent failover, persistent handles
  - ▶ protocol: SMB2\_SHARE\_CAP\_CONTINUOUS\_AVAILABILITY
  - ▶ can independently turned on on any cluster share (failover or scale-out)
  - ▶ ⇒ changed client retry behaviour!
- ▶ SMB2\_SHARE\_CAP\_CLUSTER:
  - ▶ run witness service (DPC)

## Clustering – Client Behaviour (Win8)

- ▶ `SMB2_SHARE_CAP_CLUSTER`:
  - ▶ clients happily work if witness is not available
- ▶ `SMB2_SHARE_CAP_SCALEOUT`:
  - ▶ clients happily connect if `CLUSTER` is not set.
  - ▶ clients DO request oplocks/leases/durable handles
  - ▶ clients are not confused if they get these
- ▶ `SMB2_SHARE_CAP_CONTINUOUS_AVAILABILITY`:
  - ▶ clients happily connect if `CLUSTER` is not set.
  - ▶ clients typically request persistent handle with RWH lease

# Clustering with Samba/CTDB

- ▶ all-active SMB-cluster with Samba and CTDB...  
...since 2007! 😊
- ▶ transparent for the client
  - ▶ CTDB:
    - ▶ metadata and messaging engine for Samba in a cluster
    - ▶ plus cluster resource manager (IPs, services...)
  - ▶ client only sees one “big” SMB server
  - ▶ we could not change the client!...
  - ▶ works “well enough”
- ▶ challenge:
  - ▶ how to integrate SMB3 clustering with Samba/CTDB
  - ▶ good: rather orthogonal
  - ▶ ctdb-clustering transparent mostly due to management

# Witness Service

- ▶ an RPC service
  - ▶ monitoring of availability of resources (shares, NICs)
  - ▶ server asks client to move to another resource
- ▶ remember:
  - ▶ available on a Windows SMB3 share  $\Leftrightarrow$  `SMB2_SHARE_CAP_CLUSTER`
  - ▶ but clients happily connect w/o witness
- ▶ status in Samba [WIP (Metze, Gregor Beck)]:
  - ▶ async RPC: WIP, good progress ( $\Rightarrow$  Metze's talk)
  - ▶ wireshark dissector: essentially done
  - ▶ client: in `rpcclient` - done
  - ▶ server: dummy PoC / tracer bullet implementation done
  - ▶ CTDB: changes / integration needed

Questions?

obnox@samba.org  
ma@sernet.de

