

Status of SMB2 and SMB3 development in Samba

SDC 2012

Michael Adam

`obnox@samba.org`

Samba Team / SerNet

2012-09-17

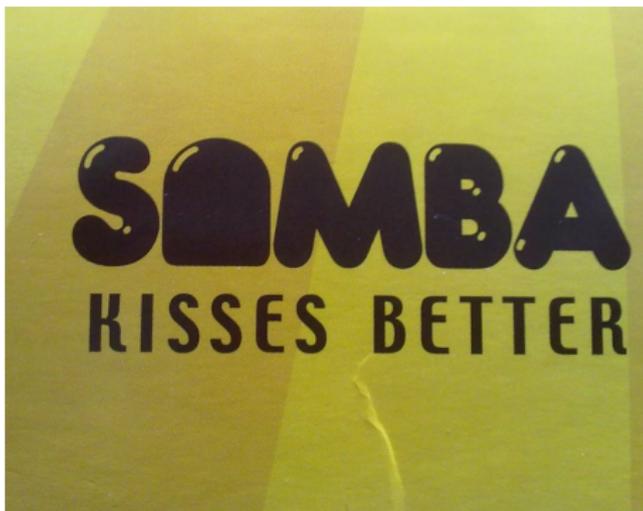
Hi there!



Oh ...

... please interrupt with questions!

A couple of introductory words about Samba...



SMB2 in Samba

- ▶ Only SMB 2.0 supported in currently released code
- ▶ experimental support in version 3.5
- ▶ SMB 2.0 officially supported in Samba 3.6
- ▶ Missing feature: durable file handles



SMB2 in Samba

- ▶ Only SMB 2.0 supported in currently released code
- ▶ experimental support in version 3.5
- ▶ SMB 2.0 officially supported in Samba 3.6
- ▶ Missing feature: durable file handles



SMB2 in Samba

- ▶ Only SMB 2.0 supported in currently released code
- ▶ experimental support in version 3.5
- ▶ SMB 2.0 officially supported in Samba 3.6
- ▶ Missing feature: durable file handles



SMB2 in Samba

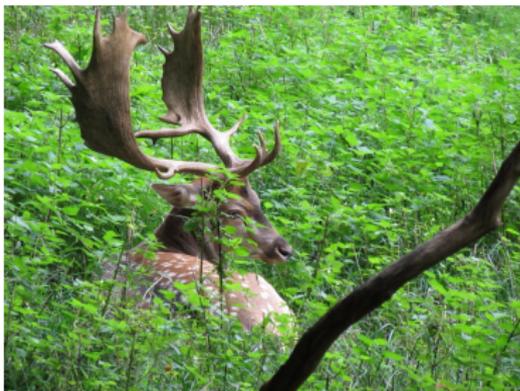
- ▶ Only SMB 2.0 supported in currently released code
- ▶ experimental support in version 3.5
- ▶ SMB 2.0 officially supported in Samba 3.6
- ▶ Missing feature: durable file handles



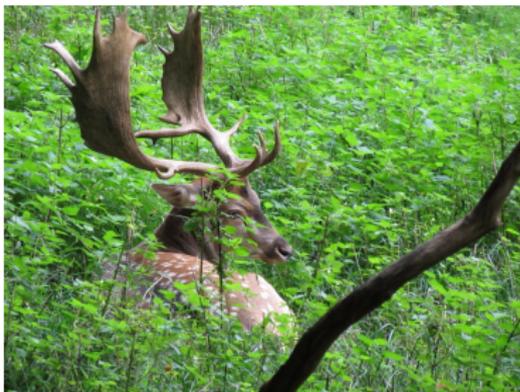
SMB2 in Samba

- ▶ Only SMB 2.0 supported in currently released code
- ▶ experimental support in version 3.5
- ▶ SMB 2.0 officially supported in Samba 3.6
- ▶ Missing feature: durable file handles





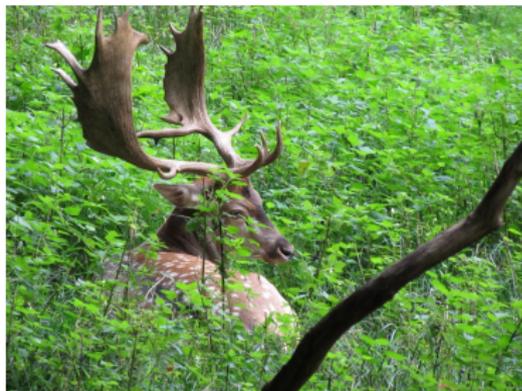
- ▶ SMB 2.0:
 - ▶ durable file handles
- ▶ SMB 2.1:
 - ▶ multi-credit / large mtu
 - ▶ dynamic reauthentication
 - ▶ leasing
 - ▶ resilient file handles
- ▶ SMB 3.0 (tpfka SMB 2.2):
 - ▶ new crypto (sgp/ahmap)
 - ▶ server negotiation
 - ▶ durable handles v2
 - ▶ persistent file handles
 - ▶ multi-credit
 - ▶ SMB direct
 - ▶ server leases



- ▶ SMB 2.0:
 - ▶ durable file handles
- ▶ SMB 2.1:
 - ▶ multi-credit / large mtu
 - ▶ dynamic reauthentication
 - ▶ leasing
 - ▶ resilient file handles
- ▶ SMB 3.0 (tpfka SMB 2.2):
 - ▶ new crypto (sign/encrypt)
 - ▶ secure negotiation
 - ▶ durable handles v2
 - ▶ persistent file handles
 - ▶ multi-channel
 - ▶ SMB direct
 - ▶ cluster features



- ▶ SMB 2.0:
 - ▶ durable file handles
- ▶ SMB 2.1:
 - ▶ multi-credit / large mtu
 - ▶ dynamic reauthentication
 - ▶ leasing
 - ▶ resilient file handles
- ▶ SMB 3.0 (tpfka SMB 2.2):
 - ▶ new crypto (sign/encrypt)
 - ▶ secure negotiation
 - ▶ durable handles v2
 - ▶ persistent file handles
 - ▶ multi-channel
 - ▶ SMB direct
 - ▶ cluster features
 - ▶ ...



- ▶ SMB 2.0:
 - ▶ durable file handles [(almost)DONE]
- ▶ SMB 2.1:
 - ▶ multi-credit / large mtu [DONE]
 - ▶ dynamic reauthentication [DONE]
 - ▶ leasing [TODO]
 - ▶ resilient file handles [TODO]
- ▶ SMB 3.0 (tpfka SMB 2.2):
 - ▶ new crypto (sign/encrypt) [DONE]
 - ▶ secure negotiation [DONE]
 - ▶ durable handles v2 [(almost)DONE]
 - ▶ persistent file handles [BEGUN]
 - ▶ multi-channel [TODO]
 - ▶ SMB direct [TODO]
 - ▶ cluster features [TODO]
 - ▶ ...

The Construction Squad ...

The Construction Squad ...



- ▶ Jeremy Allison
- ▶ Stefan Metzmacher
- ▶ Michael Adam
- ▶ Volker Lendecke
- ▶ Christian Ambach
- ▶ Ira Cooper
- ▶ Gregor Beck
- ▶ Björn Baumbach
- ▶ + ...

Durable Handles



Durable Handles



- ▶ target: short network outages
- ▶ client reconnects session (cleanup)
- ▶ then reconnects durable handle
- ▶ file server keeps disconnected handle open

Durable Handles



- ▶ target: short network outages
- ▶ client reconnects session (cleanup)
- ▶ then reconnects durable handle
- ▶ file server keeps disconnected handle open

Durable Handles



- ▶ target: short network outages
- ▶ client reconnects session (cleanup)
- ▶ then reconnects durable handle
- ▶ file server keeps disconnected handle open

Durable Handles



- ▶ target: short network outages
- ▶ client reconnects session (cleanup)
- ▶ then reconnects durable handle
- ▶ file server keeps disconnected handle open

Durable Handles And Samba



Durable Handles And Samba



- ▶ need to find old session by session-ID
- ▶ need to find file handle by persistent file ID
- ▶ threaded vs. multi-process:
keep files open vs. reopen files
- ▶ need to serialize state that had before been
in memory only
- ▶ new structures in samba:
separate smb-layer and file system layer
- ▶ ⇒ foundation for all further SMB2 work

Durable Handles And Samba



- ▶ need to find old session by session-ID
- ▶ need to find file handle by persistent file ID
- ▶ threaded vs. multi-process:
keep files open vs. reopen files
- ▶ need to **serialize state that had before been in memory only**
- ▶ new structures in samba:
separate smb-layer and file system layer
- ▶ ⇒ **foundation for all further SMB2 work**

Durable Handles And Samba



- ▶ need to find old session by session-ID
- ▶ need to find file handle by persistent file ID
- ▶ threaded vs. multi-process:
keep files open vs. reopen files
- ▶ need to serialize state that had before been
in memory only
- ▶ new structures in samba:
separate smb-layer and file system layer
- ▶ ⇒ foundation for all further SMB2 work

Durable Handles And Samba



- ▶ need to find old session by session-ID
- ▶ need to find file handle by persistent file ID
- ▶ threaded vs. multi-process:
keep files open vs. reopen files
- ▶ need to **serialize state that had before been in memory only**
- ▶ new structures in samba:
separate smb-layer and file system layer
- ▶ ⇒ foundation for all further SMB2 work

Durable Handles And Samba



- ▶ need to find old session by session-ID
- ▶ need to find file handle by persistent file ID
- ▶ threaded vs. multi-process:
keep files open vs. reopen files
- ▶ need to **serialize state that had before been in memory only**
- ▶ new structures in samba:
separate smb-layer and file system layer
- ▶ ⇒ foundation for all further SMB2 work

Durable Handles And Samba



- ▶ need to find old session by session-ID
- ▶ need to find file handle by persistent file ID
- ▶ threaded vs. multi-process:
keep files open vs. reopen files
- ▶ need to **serialize state that had before been in memory only**
- ▶ new structures in samba:
separate smb-layer and file system layer
- ▶ ⇒ **foundation for all further SMB2 work**

Preparations: Tests and Client Libraries

Preparations: Tests and Client Libraries

- ▶ tests to explore protocol details: use client libraries
- ▶ *before*: 4 independent client libraries:
[smb1, smb2] × [source3, source4]
(each incomplete and with its own problems)
- ▶ *now*: one low level library for smb1 *and* smb2
(the others are just wrappers now)
`libcli/smb/smbXcli_base.h`
- ▶ ⇒ we have written a lot of new tests:
reauth, multi-credit, multi-channel, durable/persistent handles, ...
- ▶ TODO: unify higher level libs to one library used in tests and client

Preparations: Tests and Client Libraries

- ▶ tests to explore protocol details: use client libraries
- ▶ *before*: 4 independent client libraries:
[smb1, smb2] × [source3, source4]
(each incomplete and with its own problems)
- ▶ *now*: one low level library for smb1 *and* smb2
(the others are just wrappers now)
`libcli/smb/smbXcli_base.h`
- ▶ ⇒ we have written a lot of new tests:
reauth, multi-credit, multi-channel, durable/persistent handles, ...
- ▶ TODO: unify higher level libs to one library used in tests and client

Preparations: Tests and Client Libraries

- ▶ tests to explore protocol details: use client libraries
- ▶ *before*: 4 independent client libraries:
[smb1, smb2] × [source3, source4]
(each incomplete and with its own problems)
- ▶ *now*: one low level library for smb1 *and* smb2
(the others are just wrappers now)
`libcli/smb/smbXcli_base.h`
- ▶ ⇒ we have written a lot of new tests:
reauth, multi-credit, multi-channel, durable/persistent handles, ...
- ▶ TODO: unify higher level libs to one library used in tests and client

Preparations: Tests and Client Libraries

- ▶ tests to explore protocol details: use client libraries
- ▶ *before*: 4 independent client libraries:
[smb1, smb2] × [source3, source4]
(each incomplete and with its own problems)
- ▶ *now*: one low level library for smb1 *and* smb2
(the others are just wrappers now)
libcli/smb/smbXcli_base.h
- ▶ ⇒ we have written a lot of new tests:
reauth, multi-credit, multi-channel, durable/persistent handles, ...
- ▶ TODO: unify higher level libs to one library used in tests and client

Preparations: Tests and Client Libraries

- ▶ tests to explore protocol details: use client libraries
- ▶ *before*: 4 independent client libraries:
[smb1, smb2] × [source3, source4]
(each incomplete and with its own problems)
- ▶ *now*: one low level library for smb1 *and* smb2
(the others are just wrappers now)
libcli/smb/smbXcli_base.h
- ▶ ⇒ we have written a lot of new tests:
reauth, multi-credit, multi-channel, durable/persistent handles, ...
- ▶ TODO: unify higher level libs to one library used in tests and client

Server: Improve Structures and Protocol Layer Mixup

Server: Improve Structures and Protocol Layer Mixup

- ▶ Old structures mix SMB1/2/3 layer with filesystem layers.
([MS-CIFS] [MS-SMB] [MS-SMB2]) ↔ [MS-FSA] ↔ SMB_VFS/posix)



- ▶ Problem: structures are used by different layers
⇒ can't be changed easily to fix a problem in just one layer
- ▶ plan: split layers:
 - ▶ SMB
 - ▶ ntfsa vfs layer
 - ▶ posix vfs layer as backend

Server: Improve Structures and Protocol Layer Mixup

- ▶ Old structures mix SMB1/2/3 layer with filesystem layers.
([MS-CIFS] [MS-SMB] [MS-SMB2]) ↔ [MS-FSA] ↔ SMB_VFS/posix)



- ▶ Problem: structures are used by different layers
⇒ can't be changed easily to fix a problem in just one layer
- ▶ plan: split layers:
 - ▶ SMB
 - ▶ ntfsa vfs layer
 - ▶ posix vfs layer as backend

Server: Improve Structures and Protocol Layer Mixup

- ▶ Old structures mix SMB1/2/3 layer with filesystem layers.
([MS-CIFS] [MS-SMB] [MS-SMB2]) ↔ [MS-FSA] ↔ SMB_VFS/posix)



- ▶ Problem: structures are used by different layers
⇒ can't be changed easily to fix a problem in just one layer
- ▶ plan: split layers:
 - ▶ SMB
 - ▶ ntfsa vfs layer
 - ▶ posix vfs layer as backend

old vs. new structures and dbs

OLD

structures

- ▶ `smbd_server_connection`
- ▶ `user_struct`
- ▶ `connection_struct`
- ▶ `files_struct`

databases

- ▶ `sessionid.tdb` (`smbstatus`)
- ▶ `connections.tdb` (`smbstatus`)
- ▶ `locking.tdb`
- ▶ `brlock.tdb`

old vs. new structures and dbs

OLD

structures

- ▶ `smbd_server_connection`
- ▶ `user_struct`
- ▶ `connection_struct`
- ▶ `files_struct`

databases

- ▶ `sessionid.tdb` (smbstatus)
- ▶ `connections.tdb` (smbstatus)
- ▶ `locking.tdb`
- ▶ `brlock.tdb`

NEW

new structures and dbs:

- ▶ `smbXsrv_connection`
- ▶ `smbXsrv_session`
`smbXsrv_session_global.tdb`
- ▶ `smbXsrv_tcon`
`smbXsrv_tcon_global.tdb`
- ▶ `smbXsrv_open`
`smbXsrv_open_global.tdb`

old vs. new structures and dbs

OLD

structures

- ▶ `smbd_server_connection`
- ▶ `user_struct`
- ▶ `connection_struct`
- ▶ `files_struct`

databases

- ▶ `sessionid.tdb` (`smbstatus`)
- ▶ `connections.tdb` (`smbstatus`)
- ▶ `locking.tdb`
- ▶ `brlock.tdb`

NEW

new structures and dbs:

- ▶ `smbXsrv_connection`
- ▶ `smbXsrv_session`
`smbXsrv_session_global.tdb`
- ▶ `smbXsrv_tcon`
`smbXsrv_tcon_global.tdb`
- ▶ `smbXsrv_open`
`smbXsrv_open_global.tdb`

moved to VFS:

- ▶ `connection_struct`
- ▶ `files_struct` / `locking.tdb`

gone:

- ▶ `session`

old vs. new structures and dbs

OLD

structures

- ▶ `smbd_server_connection`
- ▶ `user_struct`
- ▶ `connection_struct`
- ▶ `files_struct`

databases

- ▶ `sessionid.tdb` (`smbstatus`)
- ▶ `connections.tdb` (`smbstatus`)
- ▶ `locking.tdb`
- ▶ `brlock.tdb`

NEW

new structures and dbs:

- ▶ `smbXsrv_connection`
- ▶ `smbXsrv_session`
`smbXsrv_session_global.tdb`
- ▶ `smbXsrv_tcon`
`smbXsrv_tcon_global.tdb`
- ▶ `smbXsrv_open`
`smbXsrv_open_global.tdb`

moved to VFS:

- ▶ `connection_struct`
- ▶ `files_struct` / `locking.tdb`

gone:

- ▶ `sessionid.tdb` / `connections.tdb`

old vs. new structures and dbs

OLD

structures

- ▶ `smbd_server_connection`
- ▶ `user_struct`
- ▶ `connection_struct`
- ▶ `files_struct`

databases

- ▶ `sessionid.tdb` (`smbstatus`)
- ▶ `connections.tdb` (`smbstatus`)
- ▶ `locking.tdb`
- ▶ `brlock.tdb`

NEW

new structures and dbs:

- ▶ `smbXsrv_connection`
- ▶ `smbXsrv_session`
`smbXsrv_session_global.tdb`
- ▶ `smbXsrv_tcon`
`smbXsrv_tcon_global.tdb`
- ▶ `smbXsrv_open`
`smbXsrv_open_global.tdb`

moved to VFS:

- ▶ `connection_struct`
- ▶ `files_struct` / `locking.tdb`

gone:

- ▶ `sessionid.tdb`, `connections.tdb` ↻ ↺ ↻

Low Hanging Fruit



LHF1: session reconnect (SMB 2.0)

- ▶ when a SMB2 client reconnects to a server (after a network problem) it tries to recreate the user sessions, tree connects and durable open file handles
- ▶ SMB2/3 session setup: clients sends **previous_session_id**
⇒ the server closes all opens on the old session in case the server didn't notice the network problem of the client.
- ▶ implementation in samba was relatively easy using the new smbXsrv structures and the new helpers

LHF1: session reconnect (SMB 2.0)

- ▶ when a SMB2 client reconnects to a server (after a network problem) it tries to recreate the user sessions, tree connects and durable open file handles
- ▶ SMB2/3 session setup: clients sends **previous_session_id**
⇒ the server closes all opens on the old session in case the server didn't notice the network problem of the client.
- ▶ implementation in samba was relatively easy using the new smbXsrv structures and the new helpers

LHF1: session reconnect (SMB 2.0)

- ▶ when a SMB2 client reconnects to a server (after a network problem) it tries to recreate the user sessions, tree connects and durable open file handles
- ▶ SMB2/3 session setup: clients sends **previous_session_id**
⇒ the server closes all opens on the old session in case the server didn't notice the network problem of the client.
- ▶ implementation in samba was relatively easy
using the new smbXsrv structures and the new helpers

LHF1: session reconnect (SMB 2.0)

- ▶ when a SMB2 client reconnects to a server (after a network problem) it tries to recreate the user sessions, tree connects and durable open file handles
- ▶ SMB2/3 session setup: clients sends **previous_session_id**
⇒ the server closes all opens on the old session in case the server didn't notice the network problem of the client.
- ▶ implementation in samba was relatively easy using the new smbXsrv structures and the new helpers

LHF2: dynamic reauthentication (SMB 2.1)

- ▶ SMB1 and SMB 2.0: reauthentication was designed to only happen when a kerberos ticket expired
⇒ when the server returns NT_STATUS_USER_SESSION_EXPIRED
- ▶ SMB 2.1: clients can reauthenticate a session at anytime
⇒ we *have* to implement it.
(The **one** missing mandatory feature for SMB 2.1.)
- ▶ implementation was made relatively easy
by new gensec-based session setup code
and the new smbXsrv structures

LHF2: dynamic reauthentication (SMB 2.1)

- ▶ SMB1 and SMB 2.0: reauthentication was designed to only happen when a kerberos ticket expired
⇒ when the server returns NT_STATUS_USER_SESSION_EXPIRED
- ▶ SMB 2.1: clients can reauthenticate a session at anytime
⇒ we *have* to implement it.
(The **one** missing mandatory feature for SMB 2.1.)
- ▶ implementation was made relatively easy
by new gensec-based session setup code
and the new smbXsrv structures

LHF2: dynamic reauthentication (SMB 2.1)

- ▶ SMB1 and SMB 2.0: reauthentication was designed to only happen when a kerberos ticket expired
⇒ when the server returns `NT_STATUS_USER_SESSION_EXPIRED`
- ▶ SMB 2.1: clients can reauthenticate a session at anytime
⇒ we *have* to implement it.
(The **one** missing mandatory feature for SMB 2.1.)
- ▶ implementation was made relatively easy
by new gensec-based session setup code
and the new `smbXsrv` structures

LHF2: dynamic reauthentication (SMB 2.1)

- ▶ SMB1 and SMB 2.0: reauthentication was designed to only happen when a kerberos ticket expired
⇒ when the server returns `NT_STATUS_USER_SESSION_EXPIRED`
- ▶ SMB 2.1: clients can reauthenticate a session at anytime
⇒ we *have* to implement it.
(The **one** missing mandatory feature for SMB 2.1.)
- ▶ implementation was made relatively easy by new gensec-based session setup code and the new `smbXsrv` structures

LHF3: multi-credit (SMB2.1)

- ▶ Reduce number of SMB2 packets by allowing multiple *credits* to be consumed for large reads/writes.
- ▶ Implementation was also relatively easy and straight forward with the new smbXsrv architecture.

LHF3: multi-credit (SMB2.1)

- ▶ Reduce number of SMB2 packets by allowing multiple *credits* to be consumed for large reads/writes.
- ▶ Implementation was also relatively easy and straight forward with the new smbXsrv architecture.

LHF3: multi-credit (SMB2.1)

- ▶ Reduce number of SMB2 packets by allowing multiple *credits* to be consumed for large reads/writes.
- ▶ Implementation was also relatively easy and straight forward with the new smbXsrv architecture.

Durable Handles - smbXsrv extensions

```
git show 5e63494508ade5da00ad5ab9db139efe03d39c2e
```

```
diff --git a/source3/librpc/idl/smbXsrv.idl b/source3/librpc/idl/smbX
index 90572e5..2a6d7b3 100644
--- a/source3/librpc/idl/smbXsrv.idl
+++ b/source3/librpc/idl/smbXsrv.idl
@@ -267,12 +268,19 @@ interface smbXsrv
        hyper                open_volatile_id;
        dom_sid              open_owner;
        NTTIME              open_time;
+       GUID                create_guid;
+       GUID                client_guid;
+       GUID                app_instance_id;
        /* ... */
+       NTTIME              disconnect_time;
+       uint32              durable_timeout_msec;
+       boolean8            durable;
+       DATA_BLOB         backend_cookie;
    } smbXsrv_open_global0;
```

Durable Handles - smbXsrv extensions

```
git show 5e63494508ade5da00ad5ab9db139efe03d39c2e
```

```
diff --git a/source3/librpc/idl/smbXsrv.idl b/source3/librpc/idl/smbX
index 90572e5..2a6d7b3 100644
--- a/source3/librpc/idl/smbXsrv.idl
+++ b/source3/librpc/idl/smbXsrv.idl
@@ -267,12 +268,19 @@ interface smbXsrv
        hyper                open_volatile_id;
        dom_sid               open_owner;
        NTTIME                open_time;
+
+   +   GUID                 create_guid;
+   +   GUID                 client_guid;
+   +   GUID                 app_instance_id;
        /* ... */
+
+   +   NTTIME                disconnect_time;
+   +   uint32                durable_timeout_msec;
+   +   boolean8              durable;
+   +   DATA_BLOB            backend_cookie;
    } smbXsrv_open_global0;
```

Durable Handles - VFS interface

New VFS operations for durable handles

- ▶ `SMB_VFS_DURABLE_COOKIE()`
called from the DHnQ / DH2Q create call
- ▶ `SMB_VFS_DURABLE_DISCONNECT()`
called when a client is disconnected ("shutdown close")
- ▶ `SMB_VFS_DURABLE_RECONNECT()`
called from the durable reconnect call (DHnC / DH2C create)

Durable Handles - VFS interface

New VFS operations for durable handles

- ▶ `SMB_VFS_DURABLE_COOKIE()`
called from the DHnQ / DH2Q create call
- ▶ `SMB_VFS_DURABLE_DISCONNECT()`
called when a client is disconnected ("shutdown close")
- ▶ `SMB_VFS_DURABLE_RECONNECT()`
called from the durable reconnect call (DHnC / DH2C create)

Durable Handles - VFS default implementation

from source3/librpc/idl/open_files.idl:

```
typedef [public] struct {
    [value(VFS_DEFAULT_DURABLE_COOKIE_MAGIC), charset(DOS)] \
        uint8 magic[0x30];
    [value(VFS_DEFAULT_DURABLE_COOKIE_VERSION)] uint32 version;
    boolean8 allow_reconnect;
    file_id id;
    [string, charset(UTF8)] char servicepath;
    [string, charset(UTF8)] char base_name;
    hyper initial_allocation_size;
    hyper position_information;
} vfs_default_durable_cookie;
```

Durable Handles - VFS default implementation

from source3/librpc/idl/open_files.idl:

```
typedef [public] struct {
    [value(VFS_DEFAULT_DURABLE_COOKIE_MAGIC), charset(DOS)] \
        uint8 magic[0x30];
    [value(VFS_DEFAULT_DURABLE_COOKIE_VERSION)] uint32 version;
    boolean8 allow_reconnect;
    file_id id;
    [string, charset(UTF8)] char servicepath;
    [string, charset(UTF8)] char base_name;
    hyper initial_allocation_size;
    hyper position_information;
} vfs_default_durable_cookie;
```

Durable Handles - Where are we?

Durable Handles - Where are we?



- ▶ DONE (Samba 4.0.0rc1):
 - ▶ basic smbXsrv infrastructure
 - ▶ session reconnect
 - ▶ durable open: v1 and v2
 - ▶ durable reconnect: v1 and v2 with reopening files
- ▶ LIMITATIONS:
 - ▶ no interop yet:
 - ▶ ⇒ disabled when "posix locking = yes"
 - ▶ ⇒ disabled when "kernel oplocks = yes"
 - ▶ ⇒ disabled when "kernel share modes = yes"
 - ▶ no reconnect for delete-on-close

Durable Handles - TODOs

Durable Handles - TODOs



- ▶ fully implement scavenger mechanism
- ▶ delete on close handling
- ▶ keep track of write time
- ▶ interoperability:
 - ▶ keep handle open, use fd-passing
 - ▶ create Linux kernel interface for cookie/disconnect/reconnect
- ▶ cluster improvements ...



- ▶ SMB 3.0 / Windows 8:
 - ▶ client is fully aware of clustering
 - ▶ scale out (SO) shares
 - ▶ continuously available (CA) shares
- ▶ Samba:
 - ▶ CTDB all-active clustering
 - ▶ Windows client is unaware of clustering
- ▶ TODOs (durable/persistent):
 - ▶ node failure/smbd crash?
 - ▶ replay/retry
 - ▶ channel sequence number
 - ▶ application instance ID

SMB3 - Clustering



- ▶ SMB 3.0 / Windows 8:
 - ▶ client is fully aware of clustering
 - ▶ scale out (SO) shares
 - ▶ continuously available (CA) shares
- ▶ Samba:
 - ▶ CTDB all-active clustering
 - ▶ Windows client is unaware of clustering
- ▶ TODOs (durable/persistent):
 - ▶ node failure/smbd crash?
 - ▶ replay/retry
 - ▶ channel sequence number
 - ▶ application instance ID

SMB3 - Clustering



- ▶ SMB 3.0 / Windows 8:
 - ▶ client is fully aware of clustering
 - ▶ scale out (SO) shares
 - ▶ continuously available (CA) shares
- ▶ Samba:
 - ▶ CTDB all-active clustering
 - ▶ Windows client is unaware of clustering
- ▶ TODOs (durable/persistent):
 - ▶ node failure/smbd crash?
 - ▶ replay/retry
 - ▶ channel sequence number
 - ▶ application instance ID

SMB3 - Clustering

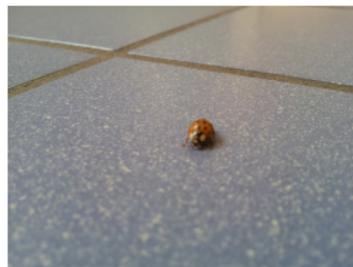


- ▶ SMB 3.0 / Windows 8:
 - ▶ client is fully aware of clustering
 - ▶ scale out (SO) shares
 - ▶ continuously available (CA) shares
- ▶ Samba:
 - ▶ CTDB all-active clustering
 - ▶ Windows client is unaware of clustering
- ▶ TODOs (durable/persistent):
 - ▶ node failure/smbd crash?
 - ▶ replay/retry
 - ▶ channel sequence number
 - ▶ application instance ID

SMB3 - Persistent Handles

we got:

- ▶ Hacked/faked proof of concept



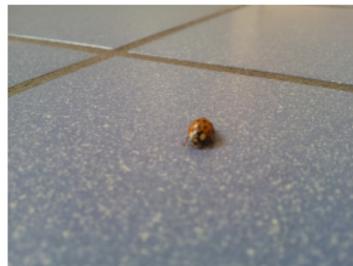
we need:

- ▶ really implement CA share (\leftrightarrow ctdb)
- ▶ implement server side of replay/retry
- ▶ provide guarantees associated with persistence, e.g.
- ▶ store more info, *persistently*, to survive server reboot
- ▶ *need* kernel interface (cookie) to keep files open

SMB3 - Persistent Handles

we got:

- ▶ Hacked/faked proof of concept



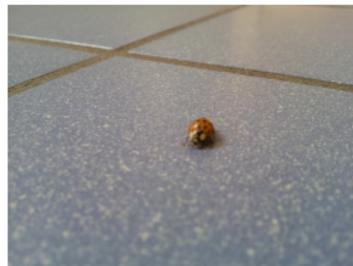
we need:

- ▶ really implement CA share (\leftrightarrow ctdb)
- ▶ implement server side of replay/retry
- ▶ provide guarantees associated with persistence, e.g.
- ▶ store more info, *persistently*, to survive server reboot
- ▶ *need* kernel interface (cookie) to keep files open

SMB3 - Persistent Handles

we got:

- ▶ Hacked/faked proof of concept



we need:

- ▶ really implement CA share (\leftrightarrow ctdb)
- ▶ implement server side of replay/retry
- ▶ provide guarantees associated with persistence, e.g.
- ▶ store more info, *persistently*, to survive server reboot
- ▶ *need* kernel interface (cookie) to keep files open

SMB3 - Multi Channel

- ▶ foundation laid in the smbXsrv structures
- ▶ will *need* kernel (cookie) interface to reopen file
- ▶ need to implement interface discovery

SMB3 - Multi Channel

- ▶ foundation laid in the smbXsrv structures
- ▶ will *need* kernel (cookie) interface to reopen file
- ▶ need to implement interface discovery

SMB3 - SMB direct (RDMA)

- ▶ Research using iwarp started (Metze)

SMB3 - SMB direct (RDMA)

- ▶ Research using iwarp started (Metze)

What is already working? - DEMO

What is already working? - DEMO

The image shows a terminal window on the left and a remote desktop connection on the right. The terminal window displays the output of several Samba-related commands:

```
[root@sambal source3]# ctdb status
Number of nodes:2
pnn:0 192.168.4.11 OK (THIS NODE)
pnn:1 192.168.4.21 OK
(Generation:1962446240
Size:2
hash:0 lmaster:0
hash:1 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1
[root@sambal source3]# bin/smbstatus -d 0

Samba version 4.0.0-GIT-949728f
PID Username Group Machine
-----
1:7097.0 CONTOSO\administrator CONTOSO\domain users 192.168.2.111 (192.168.2.111)

Service pid machine Connected at
-----
testdrive 1:7097.0 192.168.2.111 Fri Mar 2 16:30:40 2012

Locked files:
Pid Uid DenyMode Access R/W Oplock SharePath Name Time
---
1:7097.0 13000500 DENY_ALL 0x17019f RDWR BATCH /gifs0/share RHEL6.
0-20100922.1-Server-x86_64-DVD1.iso Fri Mar 2 16:47:18 2012
1:7097.0 13000500 DENY_NONE 0x100081 RDONLY NONE /gifs0/share . Fri
1 Mar 2 16:46:08 2012
No locked files

[root@sambal source3]# on
```

The remote desktop connection shows a Windows desktop environment. A file transfer progress window is visible, indicating that a file named "RHEL6-20100922.1-Server-x86_64-DVD1.iso" is being copied from Local Disk (C:) to testdrive (\\localhost\T). The progress is 1% complete, with a speed of 1.92 MB/s and approximately 15 minutes remaining. The file size is 1.015 GB.

Summary for Samba 4.0

- ▶ Samba 4.0.0rc1 released on 2012-09-13
- ▶ Samba 4.0.0 late this year (?!)
- ▶ SMB 3.0 as default
- ▶ basic durable handle support
- ▶ multi credit
- ▶ reauthentication
- ▶ SMB3 signing/encryption

Summary for Samba 4.0

- ▶ Samba 4.0.0rc1 released on 2012-09-13
- ▶ Samba 4.0.0 late this year (?!)
- ▶ SMB 3.0 as default
- ▶ basic durable handle support
- ▶ multi credit
- ▶ reauthentication
- ▶ SMB3 signing/encryption

Questions?

