SMB3 Multi-Channel in Samba

SNIA SDC 2015

Michael Adam

Red Hat / Samba Team

September 23, 2015

SMB3 features in Samba

1. SMB 3.0 (Win8 / 2012):
   - new crypto (sign/encrypt) [4.0]
   - secure negotiation [4.0]
   - durable handles v2 [4.0]
   - **Multi-Channel** [WIP+]
   - SMB direct [design]
   - cluster features [design/WIP]
     - witness [WIP+]
   - persistent file handles [design/tracer]
   - storage features [WIP]
2. SMB 3.0.2 (Win8.1 / 2012R2): [4.3]
3. SMB 3.1.1 (Win10 / 2014):
   - negotiate contexts, preauth: [4.3]

Multi-Channel

Multi-Channel - General

multiple transport connections in one SMB(3) session

- **channel**: transport connection bound to a session
- client decides which connections to bind and to use
- session is valid as long as at least one channel is intact

two purposes

1. increase throughput:
   - use multiple connections of same type
2. improve fault tolerance:
   - channel failure: replay/retry detection

Multi-Channel - General

use case: channels of different type/quality

- use only the channels of best quality
- fall back to inferior channels if superior ones fail
- e.g.: laptop switching between WiFi and LAN (?)

Multi-Channel - Windows/Protocol

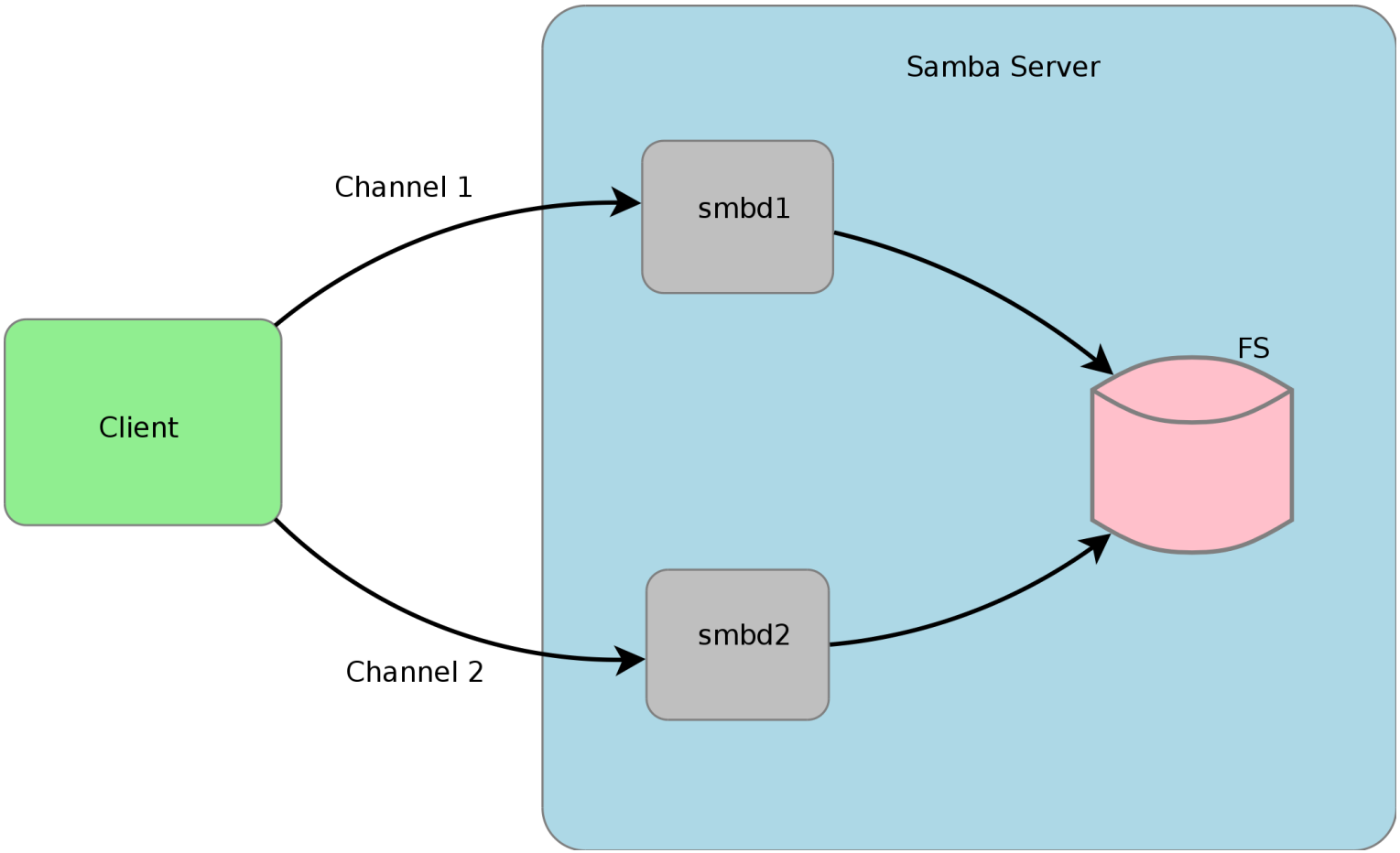
1. establish initial session on TCP connection
2. find interfaces with interface discovery: `FSCTL_QUERY_NETWORK_INTERFACE_INFO`
3. bind additional TCP (or later RDMA) connection (channel) to established SMB3 session (*session bind*)
4. Windows: uses connections of same (and best) quality
5. Windows: binds only to a single node
6. replay / retry mechanisms, epoch numbers

Multi-Channel ∈ Samba

samba/smbd: multi-process

- **Currently:** process ⇔ TCP connection
- **Idea:** transfer new TCP connection to existing smbd
- **How?** ⇒ use fd-passing (sendmsg/recvmsg)
- **When?**
  - *Natural choice*: at SessionSetup (Bind)
  - Idea: as early as possible, based on ClientGUID
    ⇒ per ClientGUID single process model
- But: This may not work! ☹

Multi-Channel ∈ Samba

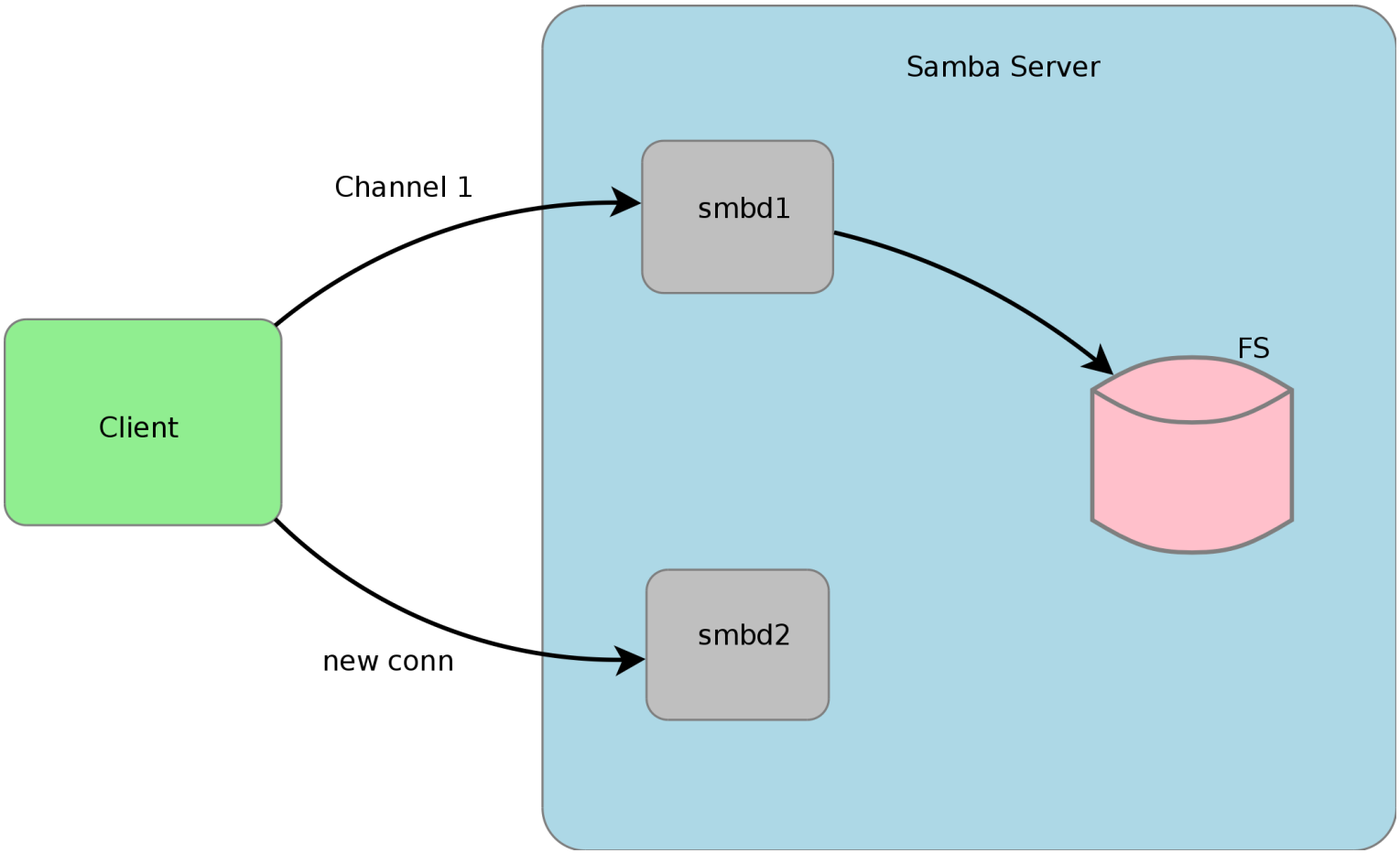


Multi-Channel ∈ Samba

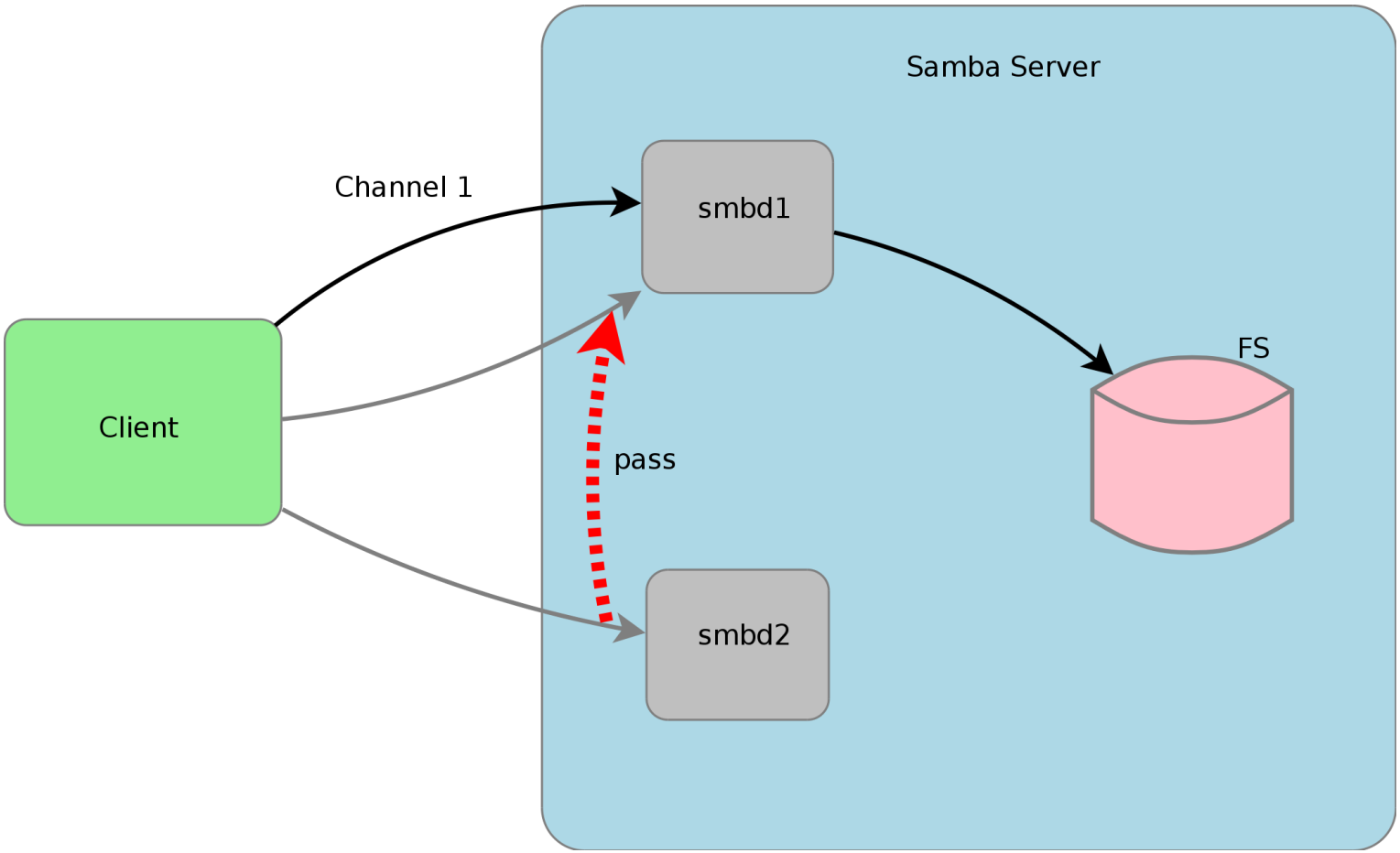
samba/smbd: multi-process

- **Currently:** process ⇔ TCP connection
- **Idea:** transfer new TCP connection to existing smbd
- **How?** ⇒ use fd-passing (sendmsg/recvmsg)
- **When?**
  - *Natural choice*: at SessionSetup (Bind)
  - Idea: as early as possible, based on ClientGUID
    ⇒ per ClientGUID single process model
- But: There may be problems! ... ☹

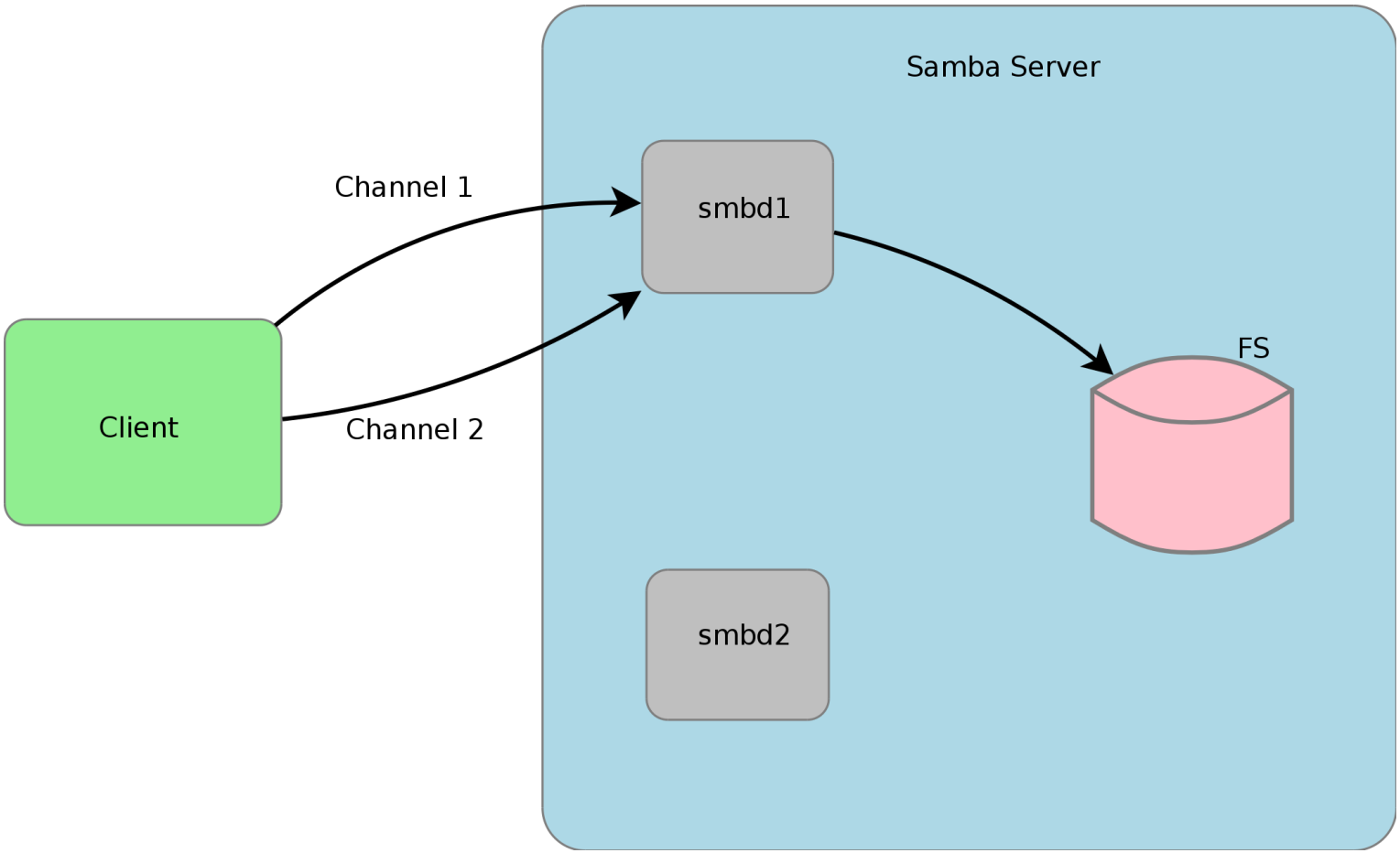
Multi-Channel ∈ Samba



Multi-Channel ∈ Samba



Multi-Channel ∈ Samba



Multi-Channel ∈ Samba

samba/smbd: multi-process

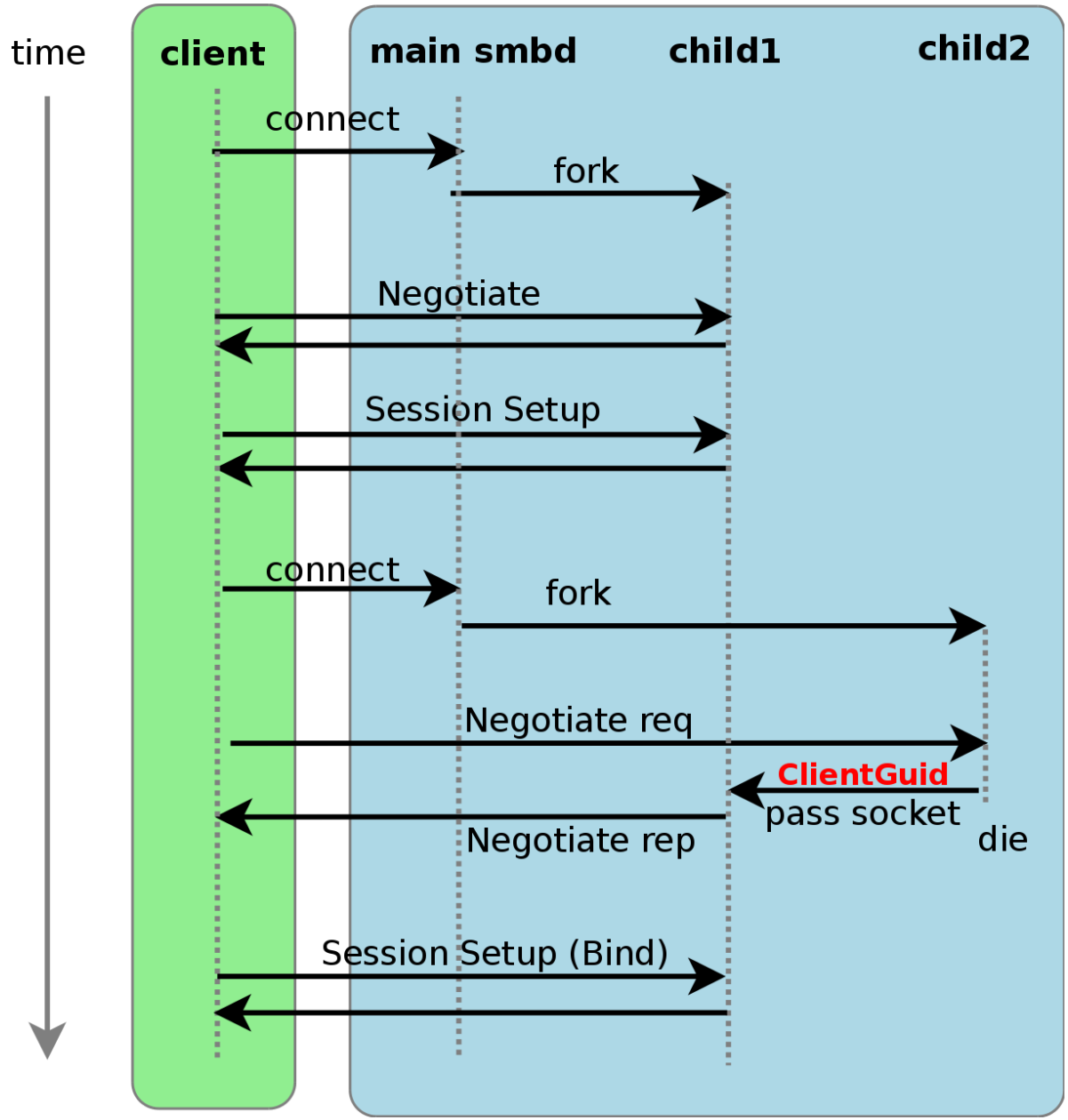
- **Currently:** process ⇔ TCP connection
- **Idea:** transfer new TCP connection to existing smbd
- **How?** ⇒ use fd-passing (sendmsg/recvmsg)
- **When?**
  - *Natural choice*: at SessionSetup (Bind)
  - Idea: as early as possible, based on ClientGUID
    ⇒ per ClientGUID single process model
- But: There may be problems! ... ☹

Multi-Channel ∈ Samba

samba/smbd: multi-process

- **Currently:** process ⇔ TCP connection
- **Idea:** transfer new TCP connection to existing smbd
- **How?** ⇒ use fd-passing (sendmsg/recvmsg)
- **When?**
  - *Natural choice*: at SessionSetup (Bind)
  - Idea: as early as possible, based on ClientGUID
    ⇒ per ClientGUID single process model
- But: There may be problems! ... ☹

Multi-Channel ∈ Samba : pass by ClientGUID



Multi-Channel ∈ Samba : pass by ClientGUID

Wait a minute - what about performance?

- Single process...
- But we use short-lived worker-pthreads for I/O ops!
- Benchmarks and tunings still to be done.

Multi-Channel ∈ Samba

samba/smbd: multi-process

- **Currently:** process ⇔ TCP connection
- **Idea:** transfer new TCP connection to existing smbd
- **How?** ⇒ use fd-passing (sendmsg/recvmsg)
- **When?**
  - *Natural choice*: at SessionSetup (Bind)
  - *Idea*: as early as possible, based on ClientGUID
    ⇒ per ClientGUID single process model
- But: There may be problems! ... ☹

The Relevance of the ClientGUID

Assumption was:

- All channels in a session have the same ClientGUID
- The server enforces this

Evidence from [MS-SMB2]:

- 3.3.5.9 Receiving an SMB2 CREATE Request:
  - sets Open.ClientGuid to Connection.ClientGuid
  - replay detection checks Open.ClientGuid == Connection.ClientGuid
- 3.3.5.9.7/12 Durable (v2) Reconnect Create Context:
  - check Open.ClientGuid == Connection.ClientGuid

The truth is...

The Windows server does not enforce it!

The Relevance of the ClientGUID

Assumption was:

- All channels in a session have the same ClientGUID
- The server enforces this

Evidence from [MS-SMB2]:

- 3.3.5.9 Receiving an SMB2 CREATE Request:
  - sets Open.ClientGuid to Connection.ClientGuid
  - replay detection checks
    Open.ClientGuid == Connection.ClientGuid
- 3.3.5.9.7/12 Durable (v2) Reconnect Create Context:
  - check Open.ClientGuid == Connection.ClientGuid

The truth is...

The Windows server does not enforce it!

The Relevance of the ClientGUID

Assumption was:

- All channels in a session have the same ClientGUID
- The server enforces this

Evidence from [MS-SMB2]:

- 3.3.5.9 Receiving an SMB2 CREATE Request:
  - sets Open.ClientGuid to Connection.ClientGuid
  - replay detection checks
    Open.ClientGuid == Connection.ClientGuid
- 3.3.5.9.7/12 Durable (v2) Reconnect Create Context:
  - check Open.ClientGuid == Connection.ClientGuid

The truth is...

The Windows server does not enforce it!

The Relevance of the ClientGUID

Windows behaviour according to MS

- The server does NOT enforce same ClientGUID in a session.
- But clients can be expected to do it.
- But it is not explicitly documented like this.

The good news:

There will be documentation notes:

- Things will not work as expected when clients behave differently.
- It is OK for a server to enforce equality of ClientGUID within session.

The Relevance of the ClientGUID

Windows behaviour according to MS

- The server does NOT enforce same ClientGUID in a session.
- But clients can be expected to do it.
- But it is not explicitly documented like this.

The good news:

There will be documentation notes:

- Things will not work as expected when clients behave differently.
- It is OK for a server to enforce equality of ClientGUID within session.

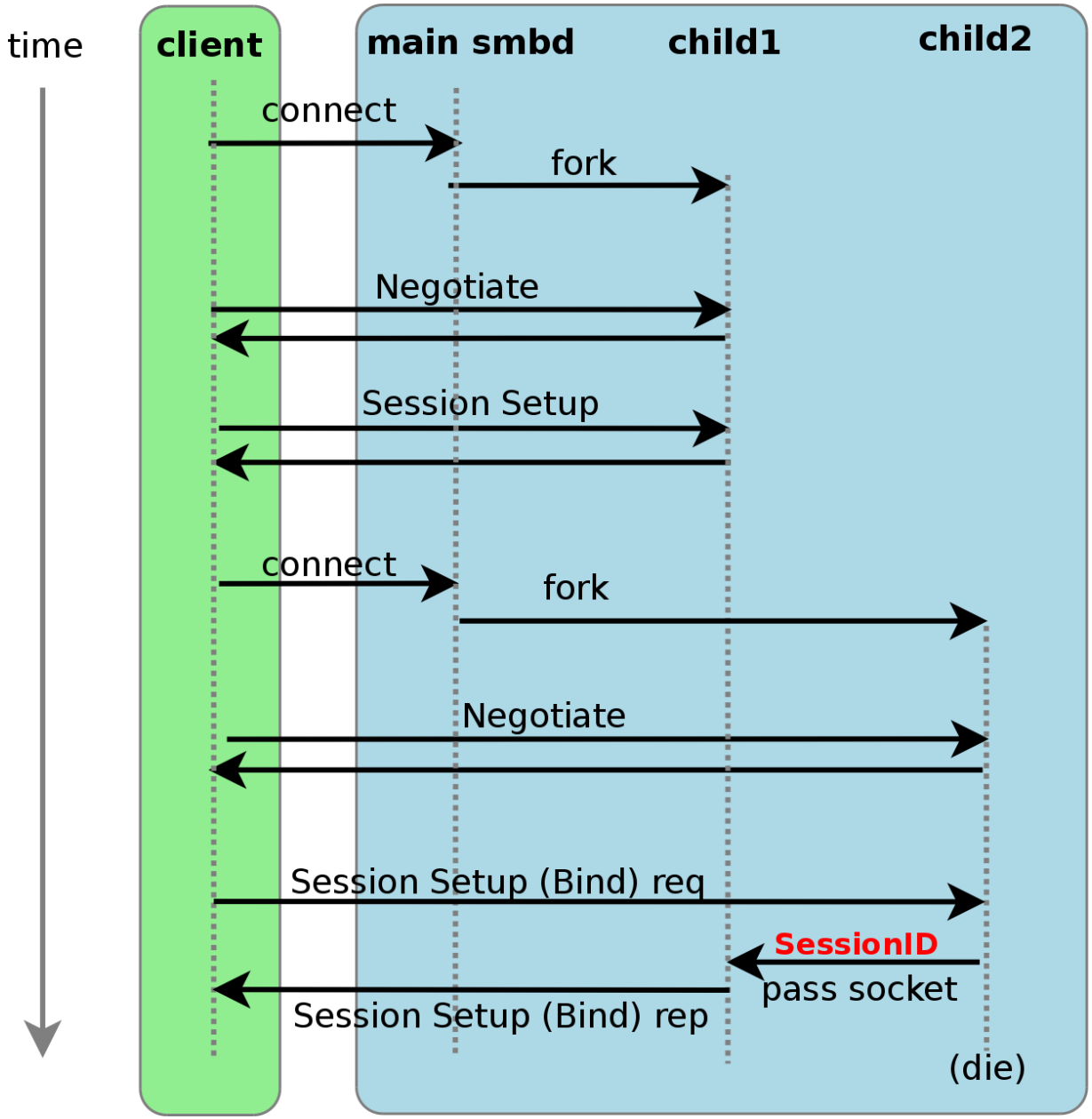
More digression on ClientGUID : Leases

According to [MS-SMB2]

- Server Global data Structures:
  - GlobalLeaseTableList indexed by ClientGuid (3.3.1.5)
    - LeaseTable indexed by LeaseKey (3.3.1.11)
- Requesting a lease (3.3.1.4):
  - Object store takes an abstract ClientLeaseId
    - Win7: combination of ClientGuid and LeaseKey
    - Win8+: LeaseKey
- Object store indicates a lease break (3.3.4.7):
  - smb server uses ClientGuid and LeaseKey given by ObjectStore

$\Rightarrow$ Inconsistent! – What to do?

Multi-Channel ∈ Samba : pass by SessionID (plan B)



Multi-Channel ∈ Samba : Status

1. messaging rewrite using unix dgm sockets with sendmsg [DONE,4.2]
2. add fd-passing to messaging [DONE,4.2]
3. preparations in internal structures [DONE,master]
4. prepare code to cope with multiple channels [DONE,master]
5. implement smbd message to pass a tcp socket [ess.DONE]
6. transfer connection in Negotiate (by ClientGUID) [ess.DONE]
7. implement session bind [ess.DONE]
8. transfer connection in Session Bind (by SessionID) [thinking..]
9. implement channel epoch numbers [WIP]
10. implement interface discovery [WIP]
11. implement test cases [WIP(isn't it always?... ☺)]
12. implement fd-passing in socket-wrapper [WIP]

Multi-Channel ∈ Samba : Status

1. messaging rewrite using unix dgm sockets with sendmsg [DONE,4.2]
2. add fd-passing to messaging [DONE,4.2]
3. preparations in internal structures [DONE,master]
4. prepare code to cope with multiple channels [DONE,master]
5. implement smbd message to pass a tcp socket [ess.DONE]
6. transfer connection in Negotiate (by ClientGUID) [ess.DONE]
7. implement session bind [ess.DONE]
8. transfer connection in Session Bind (by SessionID) [thinking..]
9. implement channel epoch numbers [WIP]
10. implement interface discovery [WIP]
11. implement test cases [WIP(isn't it always?... ☺)]
12. implement fd-passing in socket-wrapper [WIP]

Multi-Channel ∈ Samba : Status

WIP code

- `git://git.samba.org/obnox/samba/samba-obnox.git`
- branch: `master-multi-channel-obnox`

Multi-Channel ∈ Samba : Clustering/CTDB

Special considerations

- channels of one session only to one node
- do not bind connections to CTDB public IPs (can move)!
- ⇒ add static IPs on public interfaces
  use these for interface discovery

Multi-Channel ∈ Samba : Clustering/CTDB

Special considerations

- channels of one session only to one node
- do not bind connections to CTDB public IPs (can move)!
- ⇒ add static IPs on public interfaces
  use these for interface discovery

Multi-Channel ∈ Samba : When?

When will we get it?

In the next major release.
i.e. Samba 4.4.0
i.e. March 2016

Multi-Channel ∈ Samba : When?

When will we get it?

In the next major release.
i.e. Samba 4.4.0
i.e. March 2016

Multi-Channel ∈ Samba : Details from smbXsrv.idl

for MSG_SMBXSRV_CONNECTION_PASS

```
typedef struct {
    NTTIME                  initial_connect_time;
    GUID                    client_guid;
    hyper                   seq_low;
    DATA_BLOB               negotiate_request;
} smbXsrv_connection_pass0;
```

Multi-Channel ∈ Samba : Details from `smbXsrv.idl`

layering before

```
smbXsrv_session
  ->smbXsrv_connection
```

layering now

```
smbXsrv_session
  ->smbXsrv_client
    ->smbXsrv_connections
```

Multi-Channel Demo

Outlook: SMB Direct

SMB Direct : SMB3 over RDMA

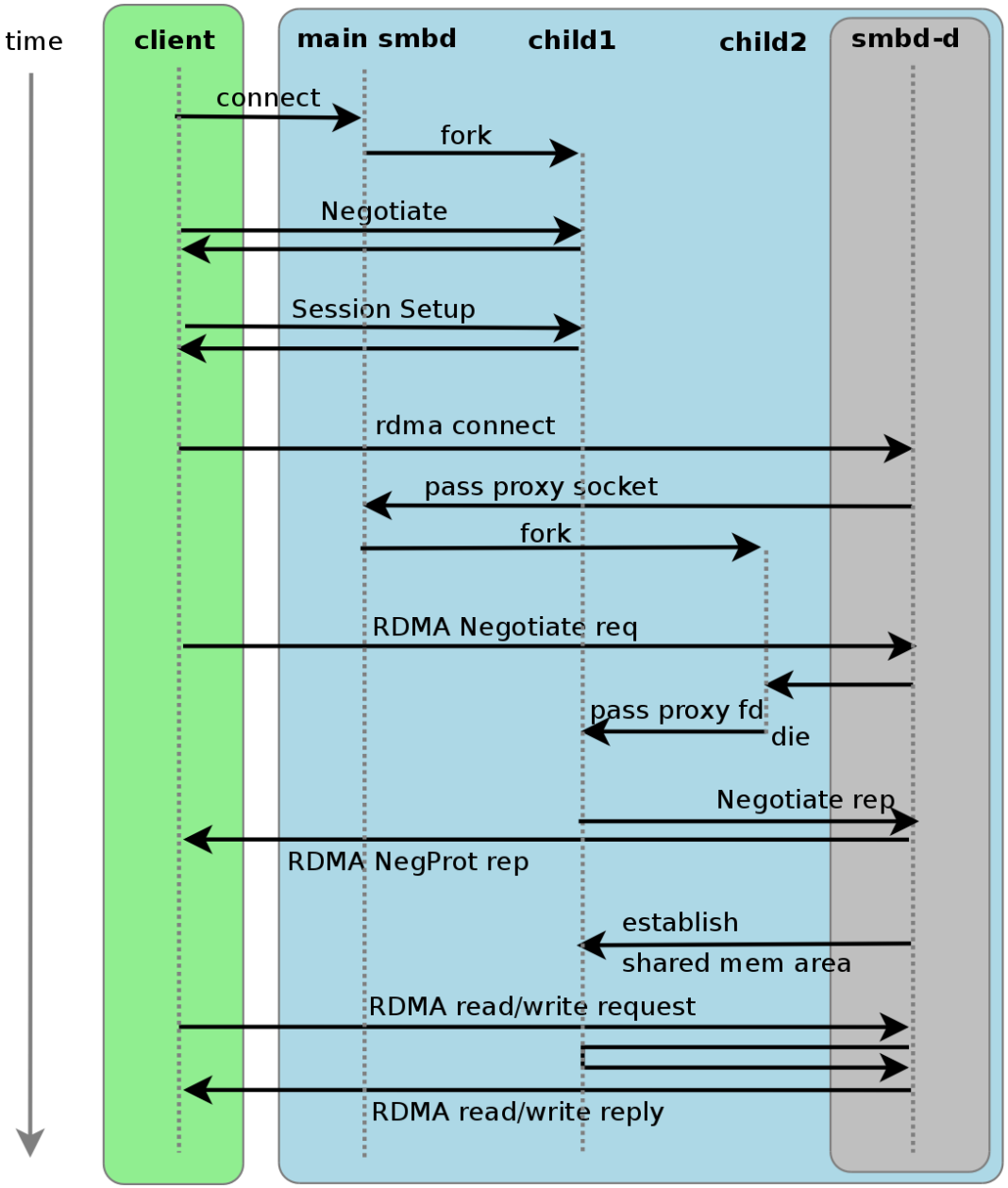
Windows/Protocol

- requires multi-channel
- start with TCP, bind an RDMA channel
- SMB Direct: small wrapper protocol to put SMB into rdma
- reads and writes use RDMA write/read
- protocol/metadata via send/receive

SMB Direct ∈ Samba

- wireshark dissector: [DONE]
- Samba:
  - prereq: multi-channel [WIP]
  - buffer / transport abstractions [WIP]
- problem with RDMA libraries:
  - not fork safe
  - no fd-passing
- ⇒ central RDMA proxy
  - PoC/dev: user space daemon
  - production: kernel module

SMB Direct ∈ Samba



SMB features in Samba

https://wiki.samba.org/index.php/Samba3/SMB3

Thanks for your attention!

Questions?

obnox@samba.org

obnox@redhat.com