The CTDB Report

Martin Schwenke <martin@meltin.net>
Amitay Isaacs <amitay@samba.org>

Samba Team
IBM (Australia Development Laboratory, Linux Technology Center)

SambaXP 2019
Overview

- Progress in the past year
- Plans presented at SambaXP 2017/2018
- Design ideas
- New daemons
- Way forward
Progress in the past year
### Committers

<table>
<thead>
<tr>
<th>Name</th>
<th>Contributions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alexander Bokovoy</td>
<td>4</td>
</tr>
<tr>
<td>Amitay Isaacs</td>
<td>82</td>
</tr>
<tr>
<td>Andreas Schneider</td>
<td>11</td>
</tr>
<tr>
<td>Andrew Bartlett</td>
<td>3</td>
</tr>
<tr>
<td>Carlos O’Donell</td>
<td>1</td>
</tr>
<tr>
<td>Christof Schmitt</td>
<td>3</td>
</tr>
<tr>
<td>David Disseldorp</td>
<td>8</td>
</tr>
<tr>
<td>Douglas Bagnall</td>
<td>1</td>
</tr>
<tr>
<td>Martin Schwenke</td>
<td>315</td>
</tr>
<tr>
<td>Noel Power</td>
<td>4</td>
</tr>
<tr>
<td>Olly Betts</td>
<td>1</td>
</tr>
<tr>
<td>Rafael David Tinoco</td>
<td>1</td>
</tr>
<tr>
<td>Ralph Boehme</td>
<td>1</td>
</tr>
<tr>
<td>Ralph Wuerthner</td>
<td>1</td>
</tr>
<tr>
<td>Samuel Cabrero</td>
<td>1</td>
</tr>
<tr>
<td>Stefan Metzmacher</td>
<td>5</td>
</tr>
<tr>
<td>Swen Schillig</td>
<td>17</td>
</tr>
<tr>
<td>Volker Lendecke</td>
<td>6</td>
</tr>
<tr>
<td>Zhu Shangzhong</td>
<td>1</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>466</strong></td>
</tr>
</tbody>
</table>
## Progress in the past year

### Commits by area

<table>
<thead>
<tr>
<th>Category</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration changes for 4.9</td>
<td>23</td>
</tr>
<tr>
<td>Add eventd (including preparation + fixes)</td>
<td>64</td>
</tr>
<tr>
<td>Portability</td>
<td>32</td>
</tr>
<tr>
<td>Portability - Packet handling</td>
<td>35</td>
</tr>
<tr>
<td>Recovery lock reliability</td>
<td>20</td>
</tr>
<tr>
<td>Vacuuming improvements</td>
<td>11</td>
</tr>
<tr>
<td>Scripts - NFS fixes for systemd</td>
<td>13</td>
</tr>
<tr>
<td>Test - <code>local_daemons.sh</code></td>
<td>53</td>
</tr>
<tr>
<td>Test - generic improvements</td>
<td>52</td>
</tr>
<tr>
<td>Build/WAF 2.0/Py3</td>
<td>22</td>
</tr>
<tr>
<td>Generic Samba clean-ups</td>
<td>13</td>
</tr>
<tr>
<td>Other</td>
<td>128</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>466</strong></td>
</tr>
</tbody>
</table>
Plans presented at SambaXP 2017/2018
Separate daemons

- event daemon
- service daemon
- failover daemon + connection tracking daemon
- cluster daemon
- database daemon
- transport
- smbd proxy
eventd

serviced

failoverd + conntrackd

clusterd + databased

transport
Plans presented at SambaXP 2017/2018

- eventd
  - In Samba 4.9

- serviced

- failoverd + conntrackd

- clusterd + databased

- transport
eventd
  - In Samba 4.9
serviced
  - Initial version finished before 4.9
failoverd + conntrackd

clusterd + databased

transport
eventd
- In Samba 4.9

serviced
- Initial version finished before 4.9

failoverd + conntrackd
- Hurriedly, nearly finished before 4.9
- A lot of copy & paste from serviced
- Could have gone into 4.9...
- ...but required lots of integration work

clusterd + databased

transport
eventd

- In Samba 4.9

serviced

- Initial version finished before 4.9

failoverd + conntrackd

- Hurriedly, nearly finished before 4.9
- A lot of copy & paste from serviced
- Could have gone into 4.9...
- ...but required lots of integration work

clusterd + databased

- Not implemented

transport
eventd
- In Samba 4.9

serviced
- Initial version finished before 4.9

failoverd + conntrackd
- Hurriedly, nearly finished before 4.9
- A lot of copy & paste from serviced
- Could have gone into 4.9...
- ...but required lots of integration work

clusterd + databased
- Not implemented

transport
- In design phase
Plans presented at SambaXP 2017/2018

Status

Conclusions
Status

- Components not mature enough for 4.9, not merged

Conclusions
Status

- Components not mature enough for 4.9, not merged

Conclusions

- Lots of boilerplate code for each daemon and client tool
- Each daemon with a unix domain socket
- Separate protocol for each daemon
  - Client — Server
  - Server — Server ?
Status
  • Components not mature enough for 4.9, not merged

Conclusions
  • Lots of boilerplate code for each daemon and client tool
  • Each daemon with a unix domain socket
  • Separate protocol for each daemon
    • Client — Server
    • Server — Server ?
  • sock_daemon
Status

- Components not mature enough for 4.9, not merged

Conclusions

- Lots of boilerplate code for each daemon and client tool
- Each daemon with a unix domain socket
- Separate protocol for each daemon
  - Client — Server
  - Server — Server?
- sock_daemon
- Testing becomes easier
  - No need for fake daemons
Status

- Components not mature enough for 4.9, not merged

Conclusions

- Lots of boilerplate code for each daemon and client tool
- Each daemon with a unix domain socket
- Separate protocol for each daemon
  - Client — Server
  - Server — Server ?
- sock_daemon
- Testing becomes easier
  - No need for fake daemons
- …and complicated
  - serviced → eventd
  - failoverd → eventd, transport
  - Need multiple daemons for setup
Design ideas
Topics

- Reduce copy/paste code
- Simplify testing
- Unify protocol
- Too many sockets
Design ideas

Reduce copy/paste code
Design ideas

Reduce copy/paste code

- `sock_daemon` was good for abstracting
- Forces boilerplate code for each daemon
- Avoids handling protocol, but not very effective
Design ideas

Reduce copy/paste code

- `sock_daemon` was good for abstracting
- Forces boilerplate code for each daemon
- Avoids handling protocol, but not very effective
- Enter `tdaemon`
Design ideas

Reduce copy/paste code

- `sock_daemon` was good for abstracting
- Forces boilerplate code for each daemon
- Avoids handling protocol, but not very effective
- Enter `tdaemon`
- And possibly `tclient`
Design ideas

Simplify testing
Design ideas

Simplify testing

- Unit testing of ctdb daemon is impossible!
- Separate daemons are easier to unit test
- How to handle dependencies?
- Can we combine multiple daemons?
Design ideas

Simplify testing

- Unit testing of ctdb daemon is impossible!
- Separate daemons are easier to unit test
- How to handle dependencies?
- Can we combine multiple daemons?
- Each daemon is a `tevent_req` computation ...
- One daemon to rule them all?
Design ideas

Simplify testing

- Unit testing of ctdb daemon is impossible!
- Separate daemons are easier to unit test
- How to handle dependencies?
- Can we combine multiple daemons?
- Each daemon is a `tevent_req` computation ...
- One daemon to rule them all?
- `masterd`
Design ideas

Unify protocol
Design ideas

Unify protocol

- Each daemon needs some common “controls”
- Should Client — Server be different from Server — Server?
- New protocol?
- Design it right from beginning – endian neutral
- Common “controls” can be implemented once
Design ideas

Unify protocol

- Each daemon needs some common “controls”
- Should Client — Server be different from Server — Server?
- New protocol?
- Design it right from beginning – endian neutral
- Common “controls” can be implemented once
- tdaemon
Design ideas

Too many sockets
Too many sockets

- Each daemon with unix domain socket
- Easy to test, . . .
- . . .but gets messy to manage many sockets
Design ideas

Too many sockets

- Each daemon with unix domain socket
- Easy to test, . . .
- . . . but gets messy to manage many sockets
- Messaging server?
- . . . Unix databagram messaging
Too many sockets

- Each daemon with unix domain socket
- Easy to test, ...
- ... but gets messy to manage many sockets
- Messaging server?
- ... Unix databagram messaging
- transportd
- Every daemon now uses common transport client code
Design ideas

Too many sockets

- Each daemon with unix domain socket
- Easy to test, ...
- ... but gets messy to manage many sockets
- Messaging server?
- ... Unix databagam messaging
- transportd
- Every daemon now uses common transport client code
- Works very well for tdaemon abstraction
New daemons
New daemon

Topics

- Master daemon
- Transport daemon
New daemons

Master daemon
New daemons

Master daemon

- Start and monitor multiple daemons
New daemons

Master daemon

- Start and monitor multiple daemons
  - Multiple process model
  - Single process model

No, this is not systemd :-)

Martin Schwenke, Amitay Isaacs
The CTDB Report
New daemons

Master daemon

- Start and monitor multiple daemons
  - Multiple process model
  - Single process model
- Bundle all dependencies for testing in one daemon
New daemons

Master daemon
- Start and monitor multiple daemons
  - Multiple process model
  - Single process model
- Bundle all dependencies for testing in one daemon
- No, this is not systemd :-(
New daemons

Transport daemon
New daemons

Transport daemon

- All daemons talk to transport
- Routes packets between daemons
- Routes packets between nodes
- Understands just enough protocol for routing
- Keep it light and blazing fast!
New daemons

Transport daemon

- All daemons talk to transport
- Routes packets between daemons
- Routes packets between nodes
- Understands just enough protocol for routing
- Keep it light and blazing fast!

Further ideas

- Minimise dynamic memory allocation…
- …to zero?
Way forward
Way forward

Incremental development
Way forward

Incremental development

- To make best long-term progress, avoid churn

Martin Schwenke, Amitay Isaacs

The CTDB Report
Incremental development

- To make best long-term progress, avoid churn
- To avoid churn, we need to develop against transportd API
Way forward

Incremental development

- To make best long-term progress, avoid churn
- To avoid churn, we need to develop against transportd API
- Either need to develop new database daemon against transportd API...
Way forward

Incremental development

- To make best long-term progress, avoid churn
- To avoid churn, we need to develop against transportd API
- Either need to develop new database daemon against transportd API...
- ...or retrofit existing ctdbd against transportd API
Way forward

Incremental development

- To make best long-term progress, avoid churn
- To avoid churn, we need to develop against transportd API
- Either need to develop new database daemon against transportd API...
- ...or retrofit existing ctdbd against transportd API
- The latter involves significant churn
Way forward

Fake transportd client
Way forward

Fake transportd client

- Implement alternative transportd client code that uses current ctdbd as transport?
Fake transportd client

- Implement alternative transportd client code that uses current ctdbd as transport?
- Implement new components using this API
Way forward

Fake transportd client

- Implement alternative transportd client code that uses current cttdbd as transport?
- Implement new components using this API
- Implement new database daemon and transportd
Way forward

Fake transportd client

- Implement alternative transportd client code that uses current ctdbd as transport?
- Implement new components using this API
- Implement new database daemon and transportd
- However, first step involves churn
Way forward

Recovery scalability
Way forward

Recovery scalability

- Recovery master node is (probably) a bottleneck for recovery
Recovery scalability

- Recovery master node is (probably) a bottleneck for recovery
- Recovery master could distribute recovery of individual databases across nodes
Way forward

Recovery scalability

- Recovery master node is (probably) a bottleneck for recovery
- Recovery master could distribute recovery of individual databases across nodes
- Could implement in current code
Way forward

Recovery scalability

- Recovery master node is (probably) a bottleneck for recovery
- Recovery master could distribute recovery of individual databases across nodes
- Could implement in current code
- Churn!

Martin Schwenke, Amitay Isaacs

The CTDB Report
Problem

- Every time we churn we delay progress towards new design...
CTDB developers needed

- Samba Team has one full time CTDB developer
CTDB developers needed

- Samba Team has one full time CTDB developer
- Some amount of burnout...
CTDB developers needed

- Samba Team has one full time CTDB developer
- Some amount of burnout...
- Any volunteers?
This work represents the view of the authors and does not necessarily represent the view of IBM.

IBM is a registered trademark of International Business Machines Corporation in the United States and/or other countries.

Linux is a registered trademark of Linus Torvalds.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.
Questions?