Samba and the road to 100,000 users

Presented by Andrew Bartlett
Samba Team - Catalyst // SambaXP 2017

Andrew Bartlett

- Samba developer since 2001
- Working on the AD DC since soon after the start of the 4.0 branch, since 2004!
  - Driven to work on the AD DC after being a high school Systems Administrator
- Working for Catalyst in Wellington since 2013
  - Now leading a team of 5 Catalyst Samba Engineers
- These views are mine alone
- Please ask questions during the talk

Samba is getting faster as an AD DC

- In a two-hour benchmark adding users and adding to four groups:
  - Samba 4.4: 26,000 users
  - Samba 4.5: 48,000 users
  - Samba 4.6: 55,000 users
  - Samba 4.7: 85,000 users!
    - The first 55,000 added in just 50mins
- This talk is about how we got there

Still a very long way to go

- Every user account implies a computer account also
  - Computers are domain joined and get ‘user’ objects
- Samba 3.x was deployed widely using OpenLDAP for the hard work
  - OpenLDAP scales really well
  - We need to match that scale to upgrade those domains
- We really want to remove barriers, both real and perceived to Samba’s use
  - Not reasonable to ask that Samba be deployed on the very edge of its capability

A year of incredible progress

- We have been told Samba's DB does not scale before
  - Nadezhda Ivanova presented the OpenLDAP Backend on that basis
- This is the year clients asked Catalyst to address Samba scale and performance
- A tale of small changes brining big results
  - Boil the kettle, not the ocean!

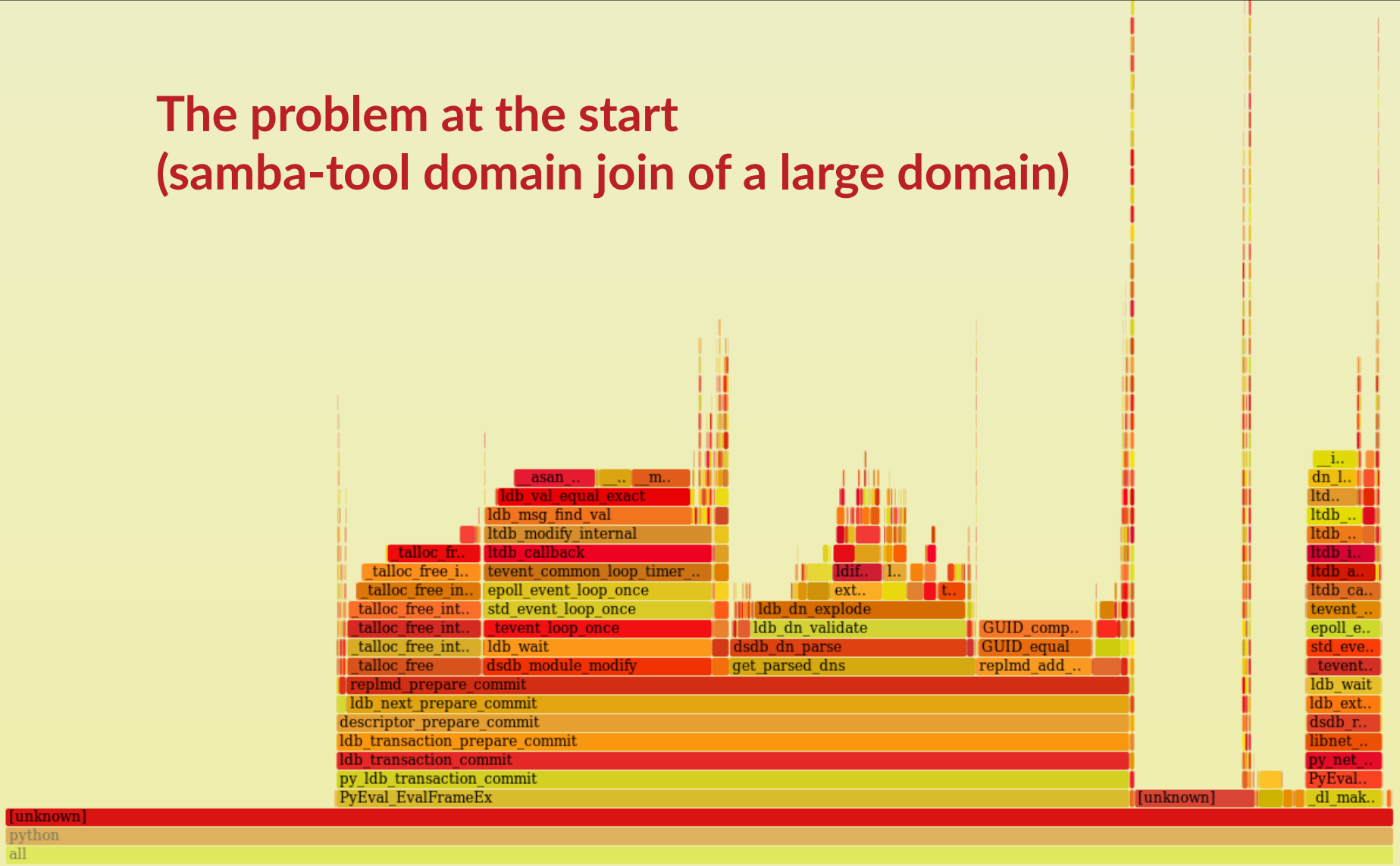
Rebuilding Samba for performance

- Once we started looking at performance, we quickly found things to fix
- Performance issues now the biggest area of our work!
  - Customers deploying Samba at scale
  - Customers growing and very keen to keep Samba
- Very glad to be the backbone of some multi-national corporate networks!

Replication as a performance bottleneck

- So what if it takes time to add 10,000 users or so?
  - Companies can't hire that fast anyway
- Biggest bottleneck is adding new DCs to Samba domains
  - e. g. opening a new office
- Growing pains: So many little inefficiencies
  - Everything is fast at < 5,000 users!
  - TODO: This loop is O(n^2)

The problem at the start
(samba-tool domain join of a large domain)



Linked attribute code had the perfect storm!

- Linked attributes are things like ‘member’ of a group.
- Each is replicated individually as a source / destination GUID pair
  - 1000 user means 100 pairs
- Before the new KCC, we had dense mesh replication
  - Changes broadcast to every DC

Over-replication of links (uptodateness ignored).

- Any change to any link caused all links to be replicated
  - To every partner (possibly all DCs)
  - And then replicated to each partner DC again!
- This could be 5000 link values for a large group!
  - Created load like each DC doing a join every time some groups changed
- This one issue make the other issues really prominent in multi-DC deployments
  - This changed the problems from bad to crippling
- Sadly we noticed this last!

Optimising the wrong things

- repl_meta_data has this lovely abstraction on link values
  - get_parsed_dns()
  - parsed_dn_find()
- A bisection search sounds good
  - Only useful if the data is sorted once, queried often
  - Instead the data was parsed, sorted and queried every time
- The most expensive cost was the parsing!

To find group members to support add/delete/modfy

- Previously, we had to parse every link
  - member: <GUID=a57fda98-631c-4897-8b2d-e3d8517d44f7>;
    <RMD_ADDTIME=1312841678300 00000>;
    <RMD_CHANGETIME=131284167830000000>;<RMD_FLAGS=0>;
    <RMD_INVOCID=a0a5a67 8-5114-4e30-bede-691df820b485>;
    <RMD_LOCAL_USN=3723>;<RMD_ORIGINATING_USN=3723 >;<RMD_VERSION=0>;
    <SID=S-1-5-21-734207269-1740946421-976543298-1103>;
    CN=testallowed,CN=Users,DC=samba,DC=example,DC=com
- Now we sort by GUID, and so can do a binary search

DN Parsing is still too costly

- Samba and LDB still parse DNs a lot
  - But without the previous fix, it was a dominant factor
- Parsing <SID=S-1-2-3-4> and <GUID=395643e5-35fb-442e-8c72-f4219e8c3070>
  - We now use the stack to parse these, not talloc memory
- libndr would allocate 1024 bytes for every context
  - So we added a variant that was told to use a fixed, passed-in buffer
- Inefficient sscanf() based parsing replaced with stricter direct C parser.

Checking for unique values (in a unique list)

- ldb_tdb needs to check that an ldb attribute value is not a duplicate
  - Currently this is an O(n^2) check
- But the repl_meta_data module has already prepared a sorted unique list
- We extended the meaning of LDB_FLAG_INTERNAL_DISABLE_SINGLE_VALUE_CHECK
- Douglas is currently working on improving the general case

How can GUID_cmp() be a hotspot?

- Linked lists are not cheap at scale
  - O(n) search time
  - Worse still if you search it n times
- The issue isn't the hot function, it is the caller
  - repl_meta_data was storing up the link changes to apply at the end of the transaction
- Code changed to apply changes right away, and avoid the list

talloc_free() is not free

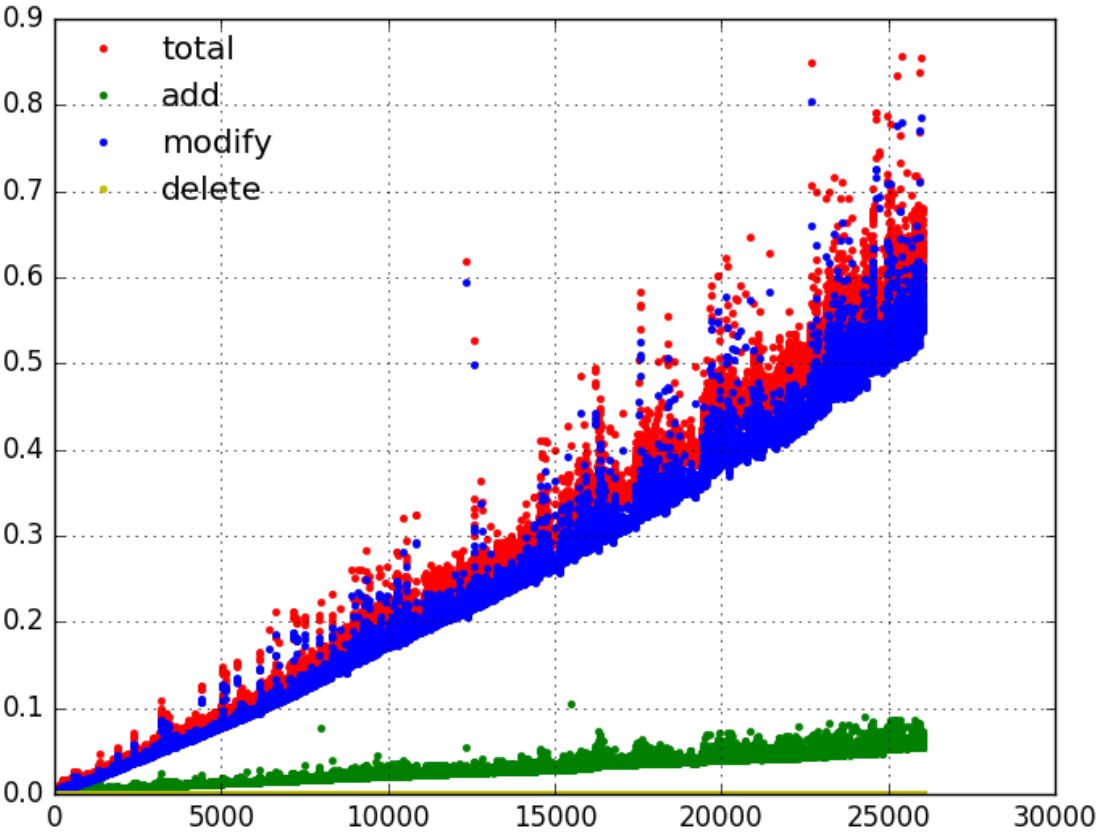
- I’ve spent quite some time making talloc_free() faster
- But the biggest gains came from not calling it
  - Once we sorted the link list, no need to allocate memory for every item

Next barrier to scale: Adding users

- The index code would check to see if the user:
  - just having been added
  - was already in the index.
- The index is currently an unsorted list of strings
  - so this was an O(n) search for each new user
- Additionally, the index code inefficiently allocated memory
  - We now do not allocate each string, just the entire index and use pointers

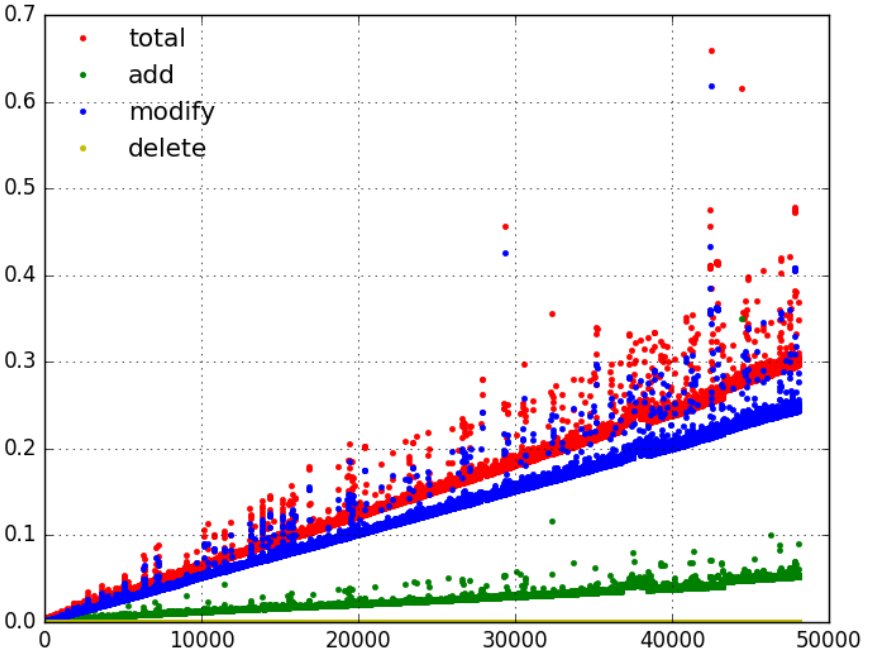
Before optimisation: Samba 4.4

- Adding a user and adding that user to four groups in a two-hour limit

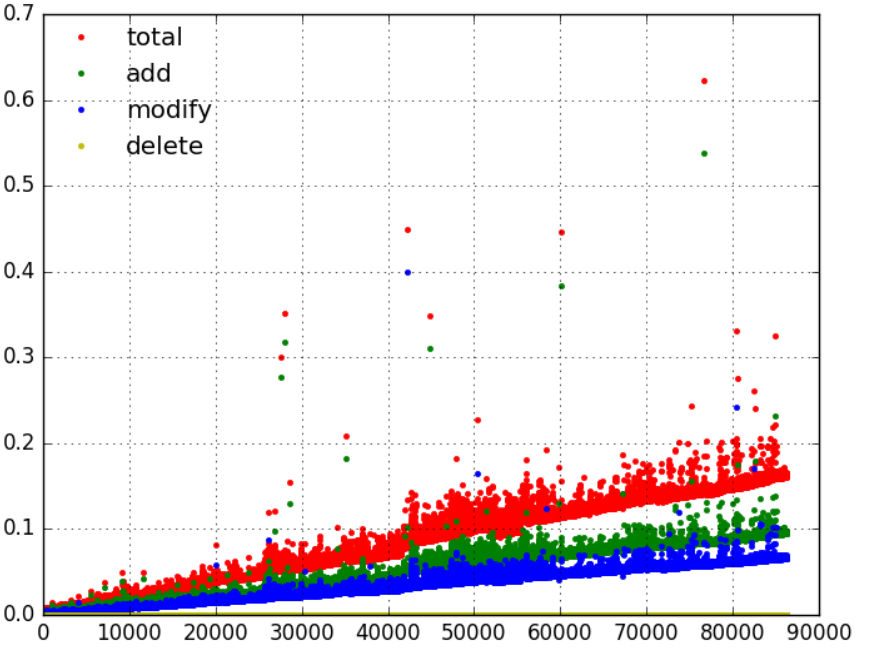


Much improved scale factors: two-hour limit

Samba 4.5



Samba 4.7



Another Issue: Search performance

- Some clients hit Samba really hard for search
- Zarafa came up on the list recently

ltdb_search now defers allocation

- Unpack of the result is as constant pointers to the buffer
  - Only allocate the buffer, and the array for any multi-valued attributes
- It is cheaper to copy the wanted results!
- Much less complex than Matthieu's approach of filtering at the unpack!

Too much locking

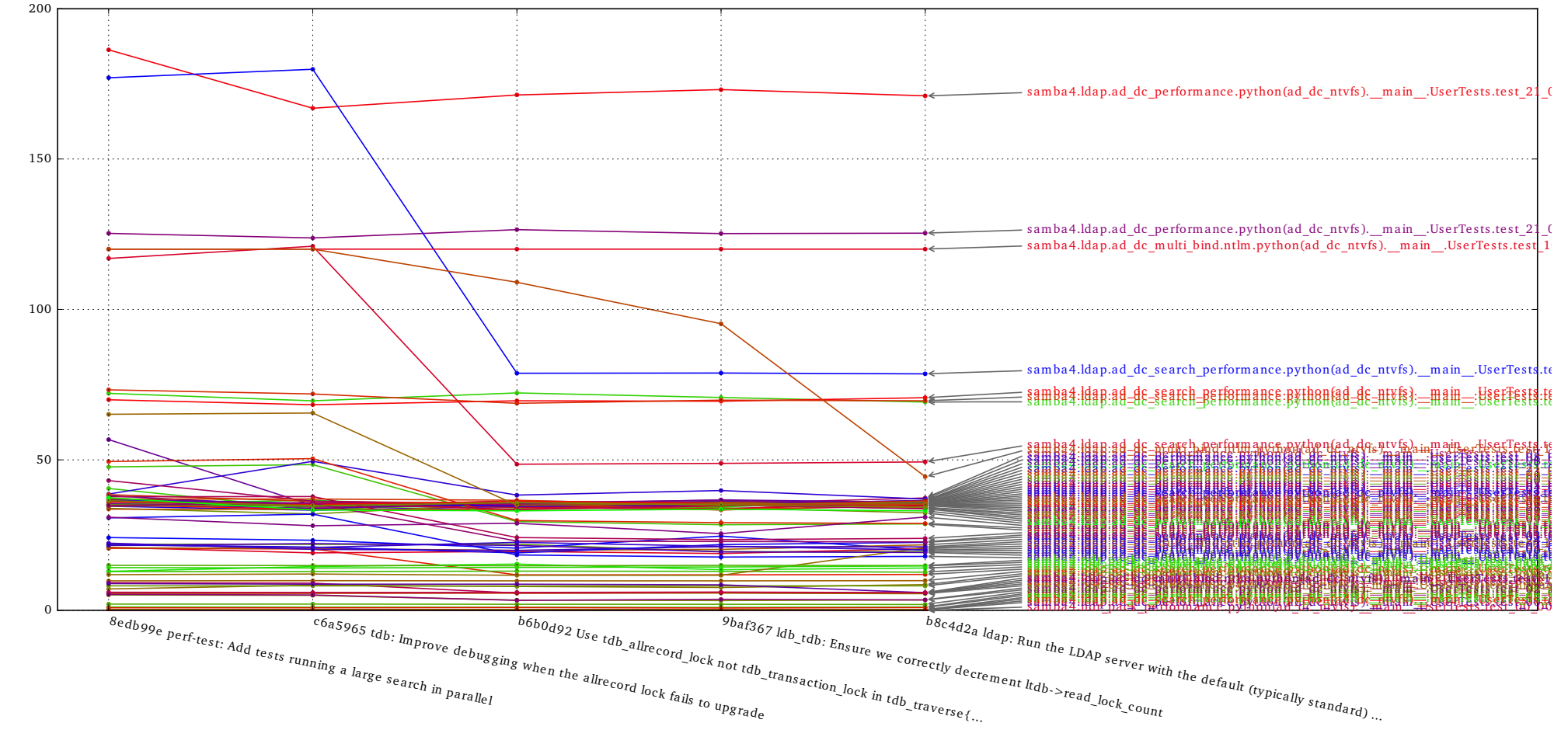
- A bug in the ldb_tdb search code meant we did walking lock during the traverse
- Very high kernel interaction for the fcntl() calls

Not enough (LDAP) processes

- Samba's LDAP server is a single process
- Historical decision
  - we just did not expect it to matter
- Will soon change to multi-process by default
  - Slower for serial bind/search/drop due to fork() cost
  - Faster for 5 or more concurrent operations

Poor un-indexed code made the index look good!

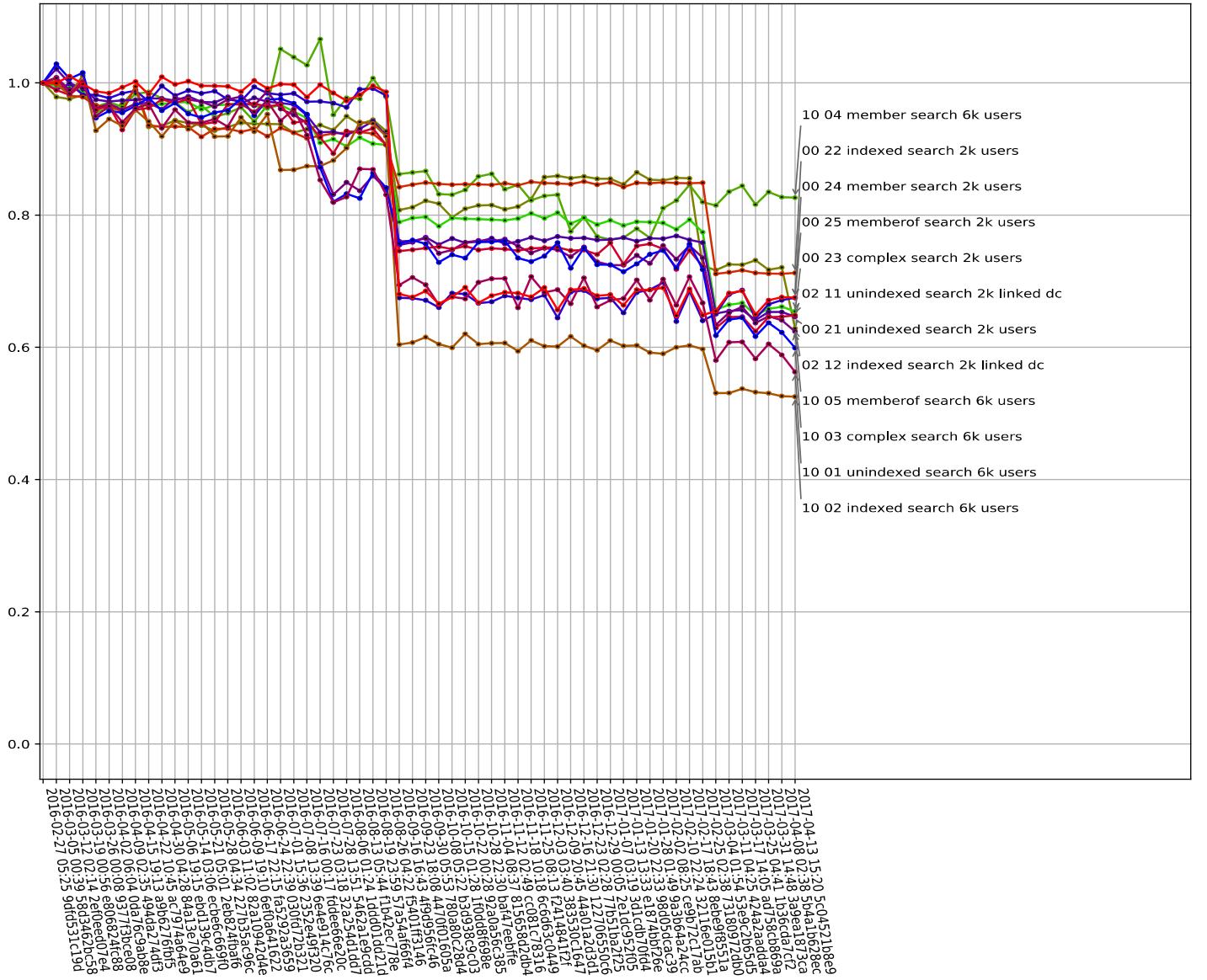
- Actually our ldb_tdb index scheme is very poor
- It only looked good when the unindexed code was hobbled!
- We need to re-design it to be faster to add/modify and intersect
  - Currently it is unordered strings that are not even the DB keys!


200
150
100
50
0
samba4.ldap.ad_dc_performance.python(ad_dc_ntvfs).__main__.UserTests.test_21_(
samba4.ldap.ad_dc_performance.python(ad_dc_ntvfs).__main__.UserTests.test_21_(
samba4.ldap.ad_dc_multi_bind.ntlm.python(ad_dc_ntvfs).__main__.UserTests.test_1
samba4.ldap.ad_dc_search_performance.python(ad_dc_ntvfs).__main__.UserTests.te
samba4.ldap.ad_dc_search_performance.python(ad_dc_ntvfs).__main__.UserTests.te
samba4.ldap.ad_dc_search_performance.python(ad_dc_ntvfs).__main__.UserTests.te
samba4.ldap.ad_dc_search_performance.python(ad_dc_ntvfs).__main__.UserTests.te
8edb99e perf-test: Add tests running a large search in parallel
c6a5965 tdb: Improve debugging when the allrecord lock fails to upgrade
b6b0d92 Use tdb_allrecord_lock not tdb_transaction_lock in tdb_traverse{...
9baf367 ldb_tdb: Ensure we correctly decrement ltdb->read_lock_count
b8c4d2a ldap: Run the LDAP server with the default (typically standard) ...

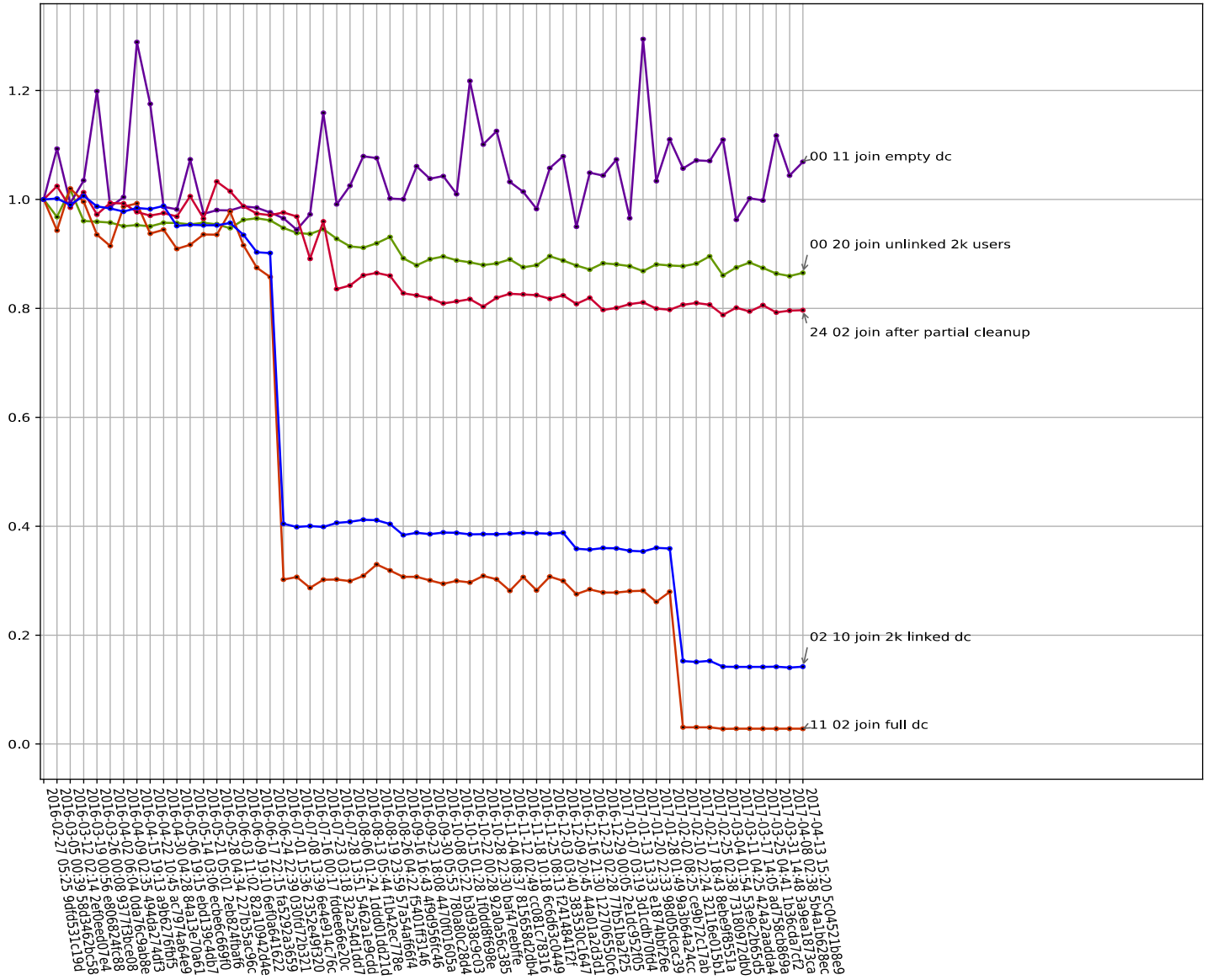
The good news

- Samba AD s getting faster, and each release is better
- We now monitor performance (see graph next slide)
- Each issue was solved individually
- Performance fixes build on each other

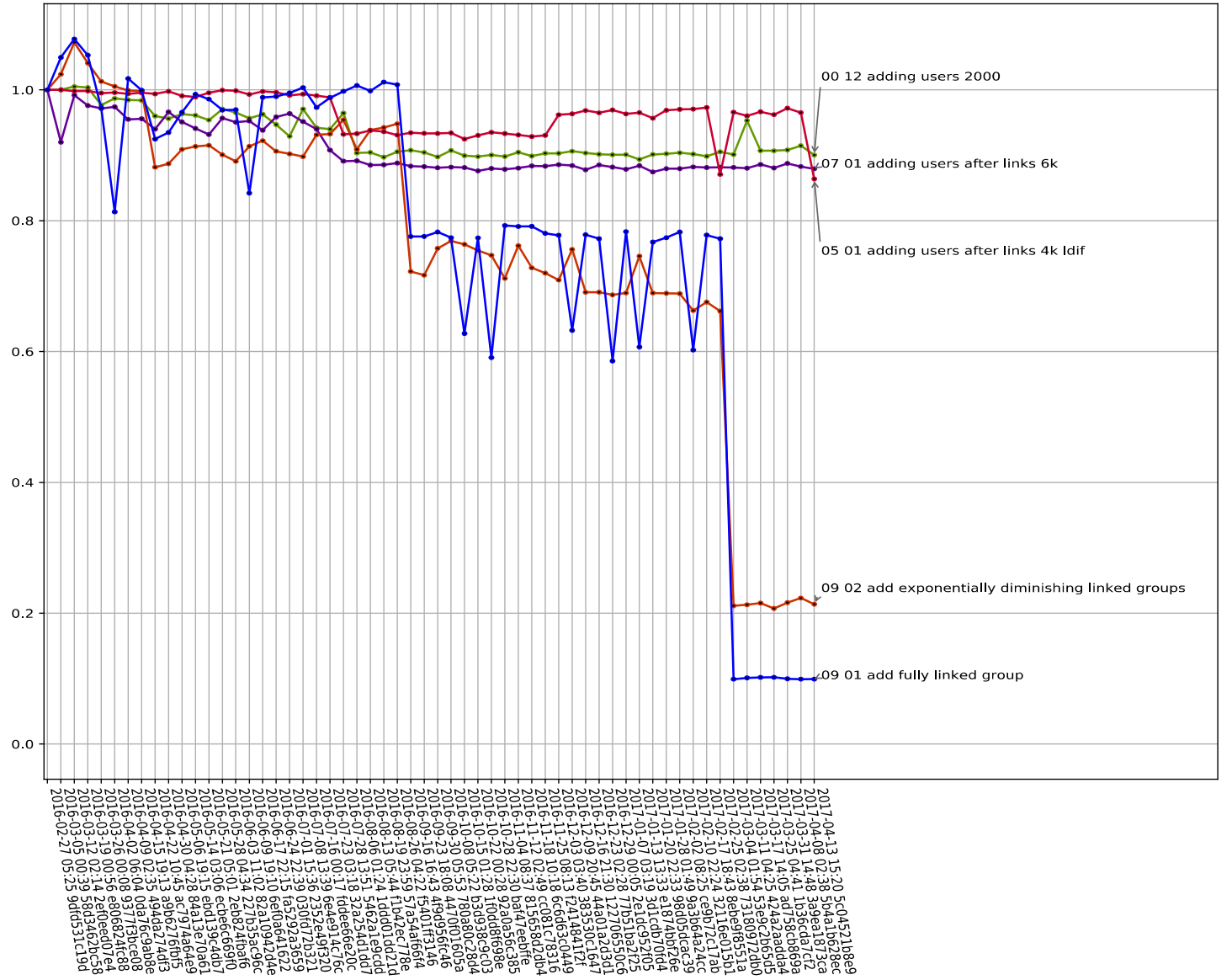
Performance graphs from March 2016 - Search



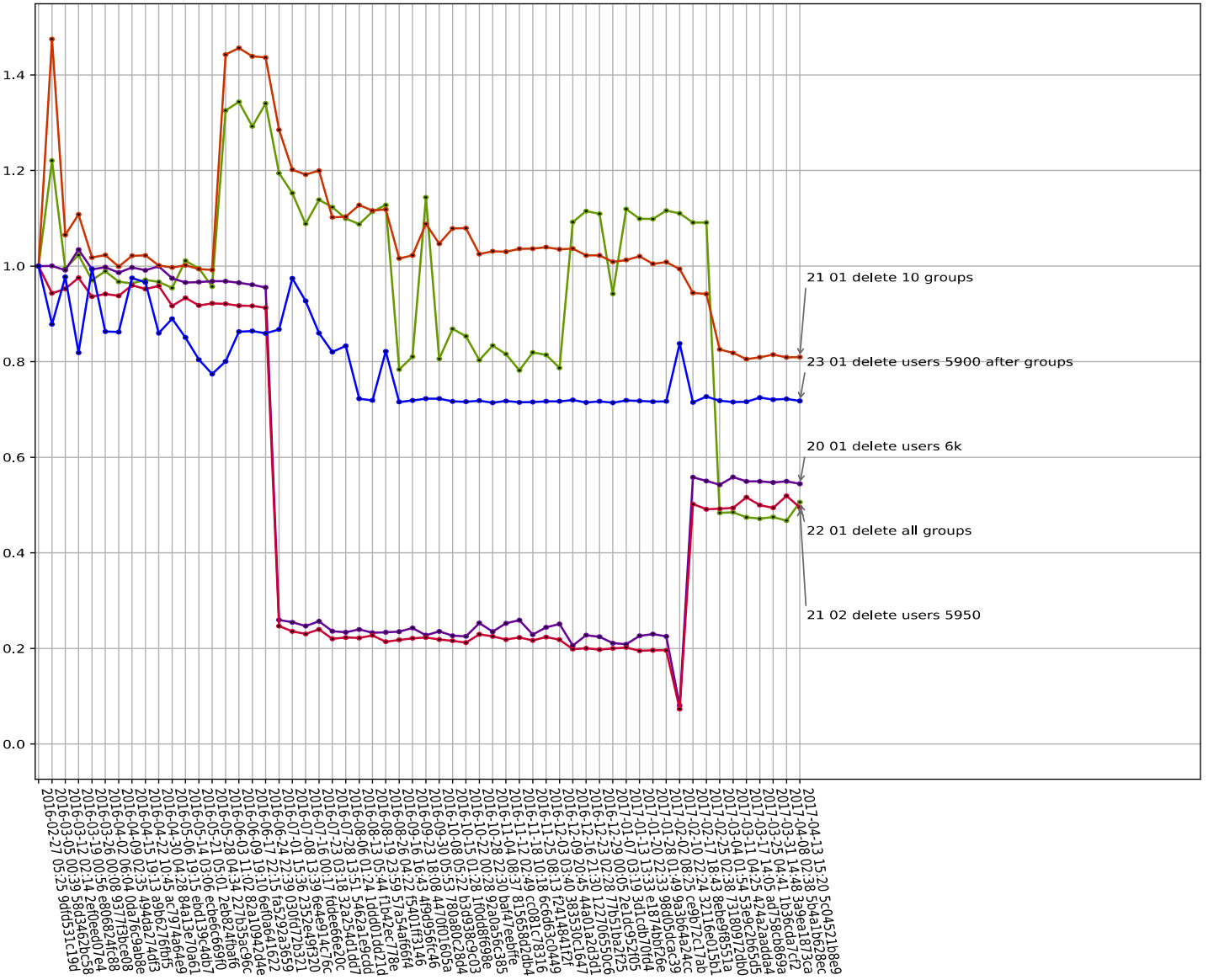
Performance graphs from March 2016 - Join



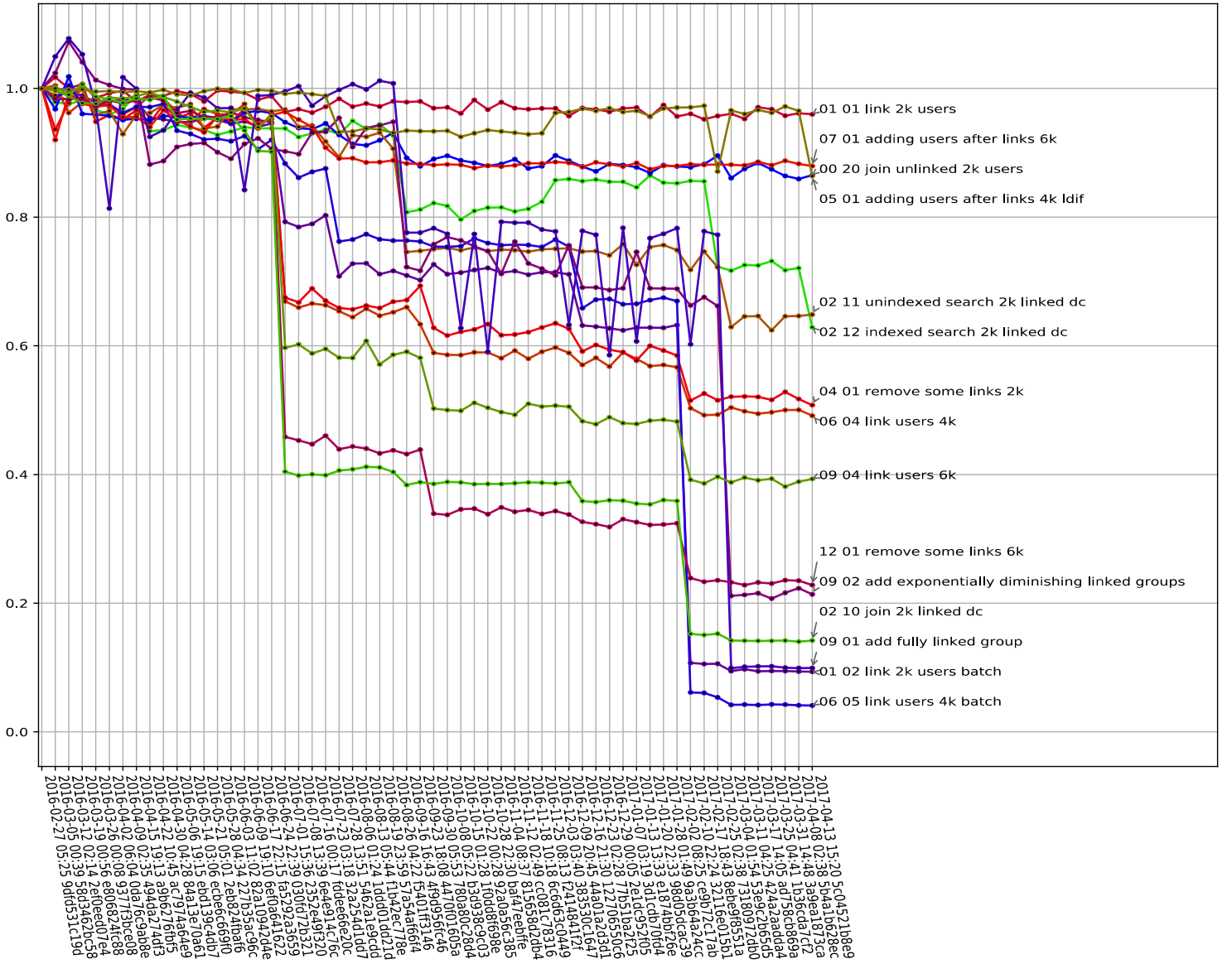
Performance graphs from March 2016 - Add user



Performance graphs from March 2016 - Delete user

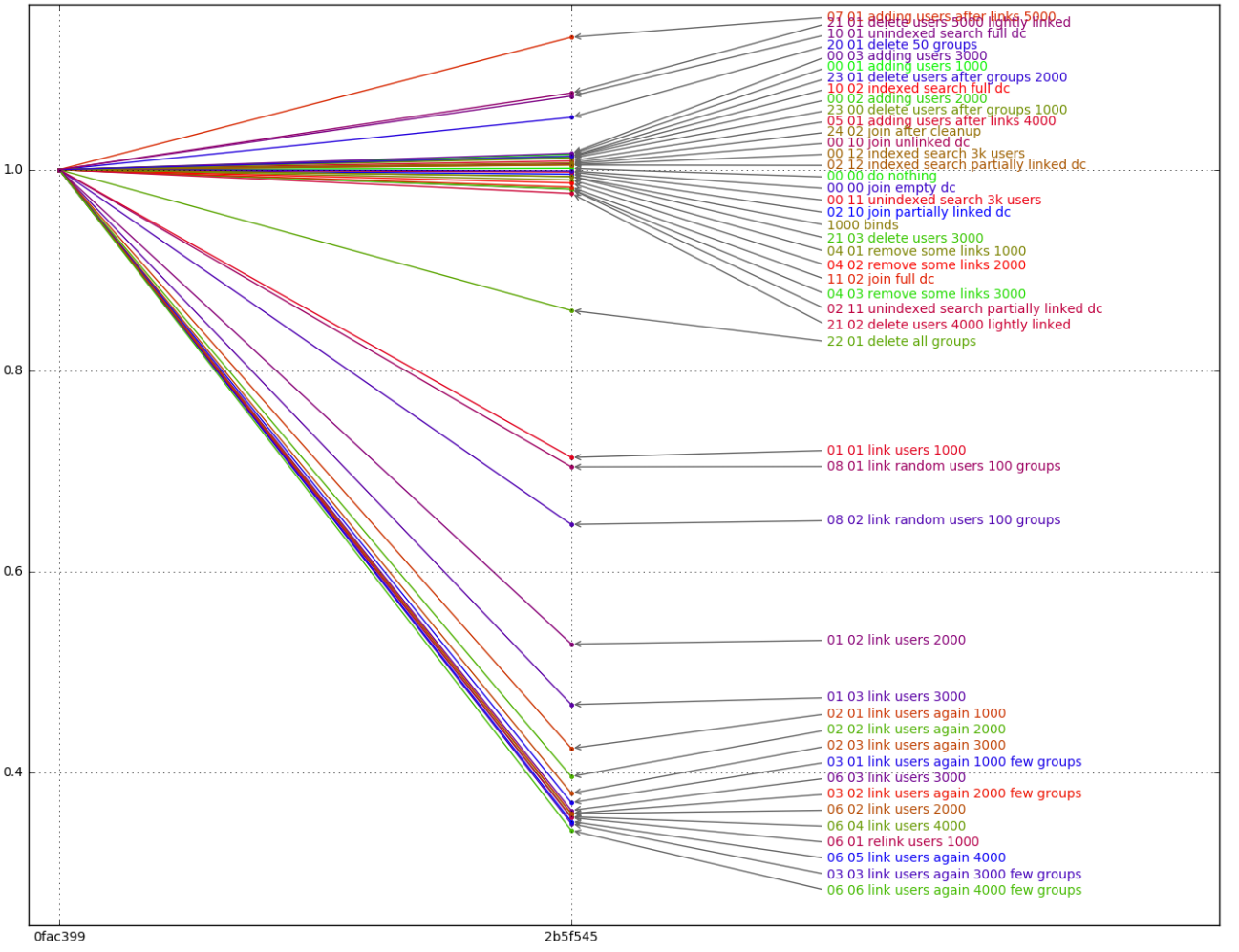


Performance graphs from March 2016 - linked attrs



Samba 4.7 so far!

- Over a 60% drop in time for some tests



Supporting more users on each DC

- Hoping to avoid needing to run extra DCs to spread the load
- Samba 4.6 removes single-process restrictions on NETLOGON
  - Really important for 802.1x backed wireless authentication
  - Unbreak the WiFi and watch the DC melt instead :-(
- Samba 4.7 will support a multi-process LDAP server
  - Easy to turn on in the code
  - Currently fork() and cleanup for exit() costs are too high

Should we still rewrite?

- A rewrites or rebase onto (say) OpenLDAP always looks attractive
- Samba4 was such a thing for the fileserver!
- I think we learnt that lesson, and have seen what it took to do MIT Kerberos
- I would rather still carve these issues off one-at-a-time
  - Bisectable changes are good!

The future for performance

- Remove other O(n) and O(n^2) operations
  - Multi-valued attribute handling
- Better index handling
  - Our current index code is still very much a first pass
  - Proposal to move to a GUID based index
- Reaching the limits for the current DB:
  - memcpy() and memmove() from ldb_tdb transactions are 20% of the time

Lightening Memory-mapped Database from Symas

- The company behind OpenLDAP
- Built by Howard Chu to make OpenLDAP fly
- LMDB backend prototyped by Jakub Hrozek of Red Hat for sssd
  - Appears to be 3 times faster for some operations
- Garming Sam has been working on reimplementation
  - Preparing it in a way that could be submitted
  - Based more tightly on the TDB LDB backend
- Still very much a WIP, but it successfully ran provision and tests!

Maintaining Performance and scale

- Large scale operation needs to be part of Samba's autobuild
- Project to develop a new performance metric for Samba domains
  - Currently under development
- Ongoing graphing of performance measurements
  - Try to spot regressions before they get too old

Help wanted!

- For the performance metric tool I need to calibrate it
- I need volunteers running AD willing to run a tshark script
  - Windows or Samba AD welcome
  - What does your busy hour look like?
  - What is the pattern of requests?
- E-mail abartlet@samba.org if you can help

Are we at 100k users?

- No
- But we now how to get there

Recap: Improvements in Samba 4.5

- Samba 4.5 addressed major issues with the client-side of replication
  - 3 of the 4 $O(n^2)$ loops removed
  - Critical as these were under the transaction lock
- Turned on graph (rather than all to all) replication by default
  - Previously every Samba DC would notify every other Samba DC about changes
  - This could trigger a short replication storm

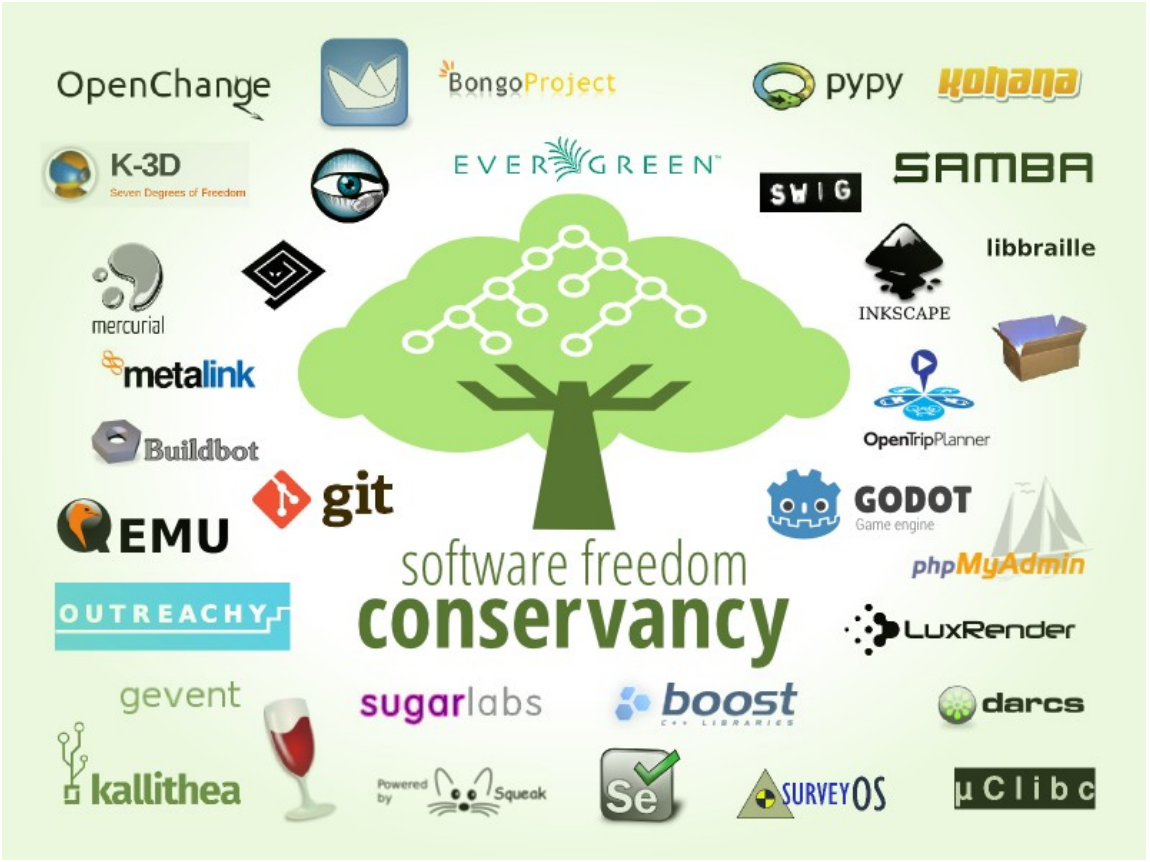
Recap: Some improvement in 4.6

- Samba 4.6 will avoid over-replication of links
  - When replicating from server A, we also ask is what changes it got from B
  - That means we don't need to ask B for changes directly
  - We did this for attributes, but didn't do this for links previously
- Faster parsing of links also improved performance around 20% for some tasks
  - Avoid sscanf() and malloc()

Recap: More improvements for 4.7

- Correct global locking will make un-indexed searches much faster
- Multi-process support will allow all CPUs to be used
- GUID-based index to be explored

Become an OFFICIAL CONSERVANCY SUPPORTER!



Catalyst's Open Source Technologies – Questions?