

Becoming a Samba Developer

Tim Prouty
tprouty@samba.org

April 24, 2009



Question:

- Why is it beneficial to become an active developer in the samba community rather than just a consumer of free software?

Overview

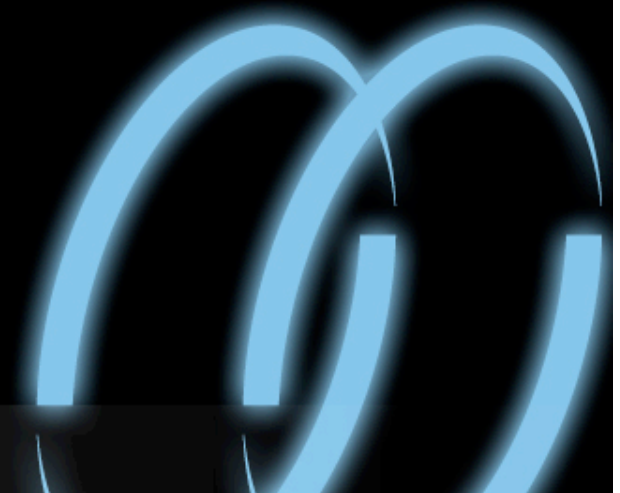
- What is Isilon?
- Our history with Samba
- A new Samba philosophy
- Moving to a new model
- Our final merge
- Going forward

What is Isilon?

- Software company that sells hardware
 - Innovative software running on commodity hardware
- Clustered storage
 - Fully symmetric architecture
 - Scalable
 - High performance



Our History With Samba



Free Solution for a Startup

- Isilon founded in 2001
- Customers primarily interested in nfs
- Started with Samba 2.x
- Demand for CIFS grew

Isilon Targets the Enterprise

- Larger Customers
- Needed cluster coherence (locking)
 - CTDB hadn't been invented yet
- New Features
 - Streams, change notification, ACLs
- Bugs uncovered
- Higher performance
 - Zero copy read/write, directory enumeration
- Complex domain topologies

Growing Pains

- CIFS is a drug
 - The more we gave customers, the more they wanted!
- Increasing levels of reliability and quality required
- Significant in-house Samba development
- Large diffs + merging = pain!

Merging Samba

- 3.0.9 to 3.0.11 to 3.0.24 to 3.4
- Merges take time
- Require rewriting code
- New bugs always introduced
- Merges only get more difficult
- Needed to get out of the merge business!

A New Samba Philosophy



Alternatives

- Continue current merge strategy
- Write our own CIFS stack
 - NetApp, EMC, Sun
 - Customer perception
- Actively participate in Samba community
 - Upstream our code
 - Invest in the long term future of Samba

Proprietary Free Software?

- Our code was available, but not consumable
- Samba is GPL
- Nothing to hide
- Upstream everything we can!
- It's better for:
 - Customers
 - Samba community
 - Us

Investing in open source

- Overall increased quality
- Mutual benefit
 - Additional dev resources
 - Bug fixes
 - New features
 - Additional QA resources
 - Code is stressed
 - Bugs found earlier

Moving to a New Model



Goals

- Eliminate need for costly future merges
- Significantly reduce diff against upstream
- Allow easy integration of upstream patches
- Utilize 'make test'
 - Excellent 'quicktest' for developers
- Become better members of the open source community

Creating a Patch Stack

- 35,000 lines changed in 55,000 line diff: Bad!
- Breaking up into patches: Good!
- Backtracked through two years of patches
 - 425 separate patches
 - Grouped patches into categories

Submitting Initial Patches

- Started years ago
- Months leading up to the merge:
 - Spend extra time to generalize bug fixes
 - Potentially rewrite for upstream
 - Small code cleanup projects
- Built relationships with Samba community

Becoming Samba Team Members

- Informed community of our intentions
- Approached a few people about commit access
- A week later we were on the team!

Our Final Merge



Strategy

- The best patch stack is no patch stack
 - Everything goes upstream!
- Modularization and APIs
 - VFS, kernel oplocks, refactoring
 - Everyone can contribute to the core, but still have system dependent code.
- Work directly from upstream git repo

Getting a Build Machine in the Farm

- Build farm with 'make test' is an excellent resource!
 - Continued benefit
- Added an Isilon VM to the build farm
 - Warning-free
 - Improved 'make test'
 - Share directory vs. tdb directory
 - Custom conf

Actual implementation

- Work on 1 feature at a time
- Mostly full rewrites
- Internal review
- Submission to samba-technical
- Regular refreshes from upstream into internal repository

Going Forward



Goals

- Higher quality upstream releases
- Improved release management
 - Stabilize code quickly
 - Build system
- Focus on improvements instead of parity
 - Better performance
 - Enhance architecture
 - New features

Practically

- Mostly developing from internal tree instead of directly upstream
- Patch stack management
 - 1 patch!
 - Pushing our patches up
 - Pulling bug fixes down
 - Future full refreshes from upstream
- Keeping build machine running on master

In closing...

- Our problem, our solution, implementing our solution
- Pay a cost in either:
 - Patch management or
 - Upstream interaction
- Upstream interaction has more benefits:
 - Cost is lower
 - Focus on improvements instead of merges
 - Higher quality code

Questions?

tprouty@samba.org



Isilon. Proven Leader in Scale-out NAS