LinuxCon 2010

Svn Scaling & GVN

http://marc.merlins.org/linux/talks/SvnScalingGvn/

Marc MERLIN
marc_soft@merlins.org

Quick Outline

- Quick overview of svn client & svn fsfs backend
- Why SVN vs P4, git, etc...
- Svn server scaling considerations
- Performance tuning
- Google enhancements to svn server
- Googlecode svn related work: http://gvn.googlecode.com/
- gvn client
- Googlecode svn server hooks
- Google svnserve / SASL / userdb

Svn Client

- Svn client keeps lot of local state
- Repository state stored in client: 2x space usage
- No load required for diff or status
- Unlimited clients with virtually no server state
- Scales virtually infinitely in number of files and clients vs p4 which has ever growing state databases on a single server
- Downside is local client slowness with no db state
- svn commit needs to scan all your files (no svn edit)

Svn FSFS

- Svn has two backends: a berkeley db and fsfs
- Fsfs is the scalable option which works over NFS
- In this case NFS doesn't have to be a dirty word :)
- 1 file per revision (not per file in the repository)
- Skip-deltas avoid rebuilding files back to rev 1
- Svn 1.6 has repository pack to reduce file count.
- New revision built in local transaction and renamed to last revision
- Lock time is very short, low contention (db/write-lock before updating db/current)

Accessing the Server

- file:// local access to repository
- Svn over ssh
- Svnserve (akin to cvs pserver)
- Apache module (webdav), works across http proxies
- Server methods allow for multiple svn server frontends that share the same fsfs filesystem

SVN vs P4

- P4 requires an ever bigger single homed server
- Once you maxed out the RAM on your single homed server and moved your huge db files to a Ramdisk SAN, pray that hardware grows quicker than you
- If your P4 server dies, you need to failover to a replica with downtime and possible lost revisions and client sync issues
- Svn can diff and status while disconnected, p4 cannot
- Price: licenses $$ vs free + admin price in both cases
- Merge Tracking broken in svn, working in p4 (p4 integrate)
- Svn client slow, especially over nfs homedirs (many stats)

Svn vs Distributed SCS

- Centralized for regulatory reasons (ISO9000...)
- Regulated offsite backups
- Centralized for per revision hooks/triggers
- Centralized for access control
- Distributed for disconnected access
- Distributed for mesh development
- They are not incompatible: distributed can be done on top of centralized (svk / git-svn or equivalent for perforce).

Svn vs Git

- If you are a happy git user, stay with git :)
- Svn does allow only checking out a subdirectory of a huge repository (git does not handle repos with tens of gigabytes of data, where you often only need a piece at a time).
- Svn also has better supported windows client for those who care :)
- More details in the comments here: http://lwn.net/Articles/381794/

Svn Server Scaling

- Use a reliable replicated nfs server (NAS appliance)
- Use multiple svn server frontends: 2-4
- Turn off atime updates (relatime on recent linux)
- atime may need to be disabled on nfs server (not client)
- Switch to sharded revisions in svn 1.5+
- Make sure your fsync() are fast (raid/memory fronted drives/ack nfs early)
- Transactions can be built (write intensive) on local disk before being promoted to a revision on nfs

Svn Server Scaling: Local Transactions

- Transactions require a lot of read/write ops before they are ready to become new revisions
- For repositories on nfs, build them on local disk before they are promoted as a revision on nfs
- db/transactions is scratch data on local disk and constantly read/written by webdav and db/txn-protorevs is on nfs and contains actual file diff data in one file before becoming a new revision in the repo

```
# stop all servers that can write to the repository
cd REPO_PATH/db
mv transactions  /LOCAL/DISK/PATH/transactions
ln -s /LOCAL/DISK/PATH/transactions  transactions
# start the servers
```

Svn Server Scaling: NFS Caching

- Revisions are write only and can be cached forever
- nocto on linux allows for mounting without cache coherency for existing files
- Increase metadata caching to more than 60 secs
- Very nfs client implementation specific

```
mount -t nfs nfs_server:/mount_point /mnt/svn -o rw,nosuid,tcp,rsize=32768,wsize=32768
mount -t nfs nfs_server:/mount_point /mnt/svn-nocto -o
                                    rw,nosuid,tcp,rsize=32768,wsize=32768,nocto,actimeo=3600
cd /mnt/svn/repo_path
mv revs revs-nocto
mv txn-protorevs txn-protorevs-nocto
ln -s /mnt/svn-nocto/repo_path/db/revs-nocto revs
ln -s /mnt/svn-nocto/repo_path/db/txn-protorevs-nocto txn-protorevs
```

Apache-DAV Setup

- While svnserve is faster, if you have to use apache-dav, consider these tips:
  - For load balancing, you need to use IP affinity
  - Tell apache to use a single TCP connection per client

```
# 1. Enable HTTP persistent connections so a single transaction can
#    be built up over a single connection.
KeepAlive            on
# 2. Allow as many KeepAlives as required (0 => infinite) to keep
#    the same connection alive.
MaxKeepAliveRequests  0
# 3. Limit a child to serving only this 1 connection (counter-intuitive, but see the "Note" at
# http://httpd.apache.org/docs/2.2/mod/mpm_common.html#maxrequestsperchild )
# This allows you to kill a resource intensive/hanging client connection without affecting
# other people
MaxRequestsPerChild   1
```

Svnserve

- apache/webdav is 4 times slower on some ops like tree diff compared to svnserve
- Svnserve + SASL has less latency than svn over ssh
- svn over ssh has a limit of 10 incoming auths before children are forked
- Svnserve lacked logging, but we added that (was integrated in 1.6)
- Make sure you watch entropy problems with /dev/random vs /dev/urandom (recompile SASL for /dev/urandom as necessary)

gvn.googlecode.com

- gvn: svn frontend
- gvn/hook userdb auth lib
- Svnserve / webdav userdb auth code
- Hooks/triggers for svn commits
- Unmaintained, but available for those who are interested

GVN UserDB

- Restrict repository access via webdav or svnserve
- Easily maintain a user/group list without editing a single passwd file
- Use different passwords for svn than for unix auth
- Users get autogenerated svn password via a web page
- Userdb is also used by hooks like check_owners hook
- Or for group perms for bypass hook for admins

GVN UserDB (2)

- Hashed per user directories with svn specific data
- Synchronized with your regular unix accounts

```
discostu:/opt/googlesvn/tools$ ./admin-userdb create user merlintest
merlintest:B8GfEI_4:Sq1aEdkqoC/LE
discostu:/opt/googlesvn/tools$ ./admin-userdb delete user merlintest
merlintest deleted
# reset a password
discostu:/opt/googlesvn/tools$ ./admin-userdb reset user merlintest2
merlintest2:7f{6SI#S:M/0dva58LezR2

# After this, here's what a sync does:
www-data@discostu:$ /opt/svn/scripts/create-owners-groups
deleting former user merlintest2
www-data@discostu:/opt/svn/userdb/mer/merlintest$ ls
groups
password
user_is_svn_only;not_autosynced  # ← marker file
```

Repository authentication against UserDB

- You need to add an authentication module to apache or svnserve so that they can authenticate against users & passwords in the userdb tree
- Get apache/webdav or svnserve auth module:

https://gvn.googlecode.com/svn/trunk/contrib/userdb/mod_authn_gvn_userdb.c

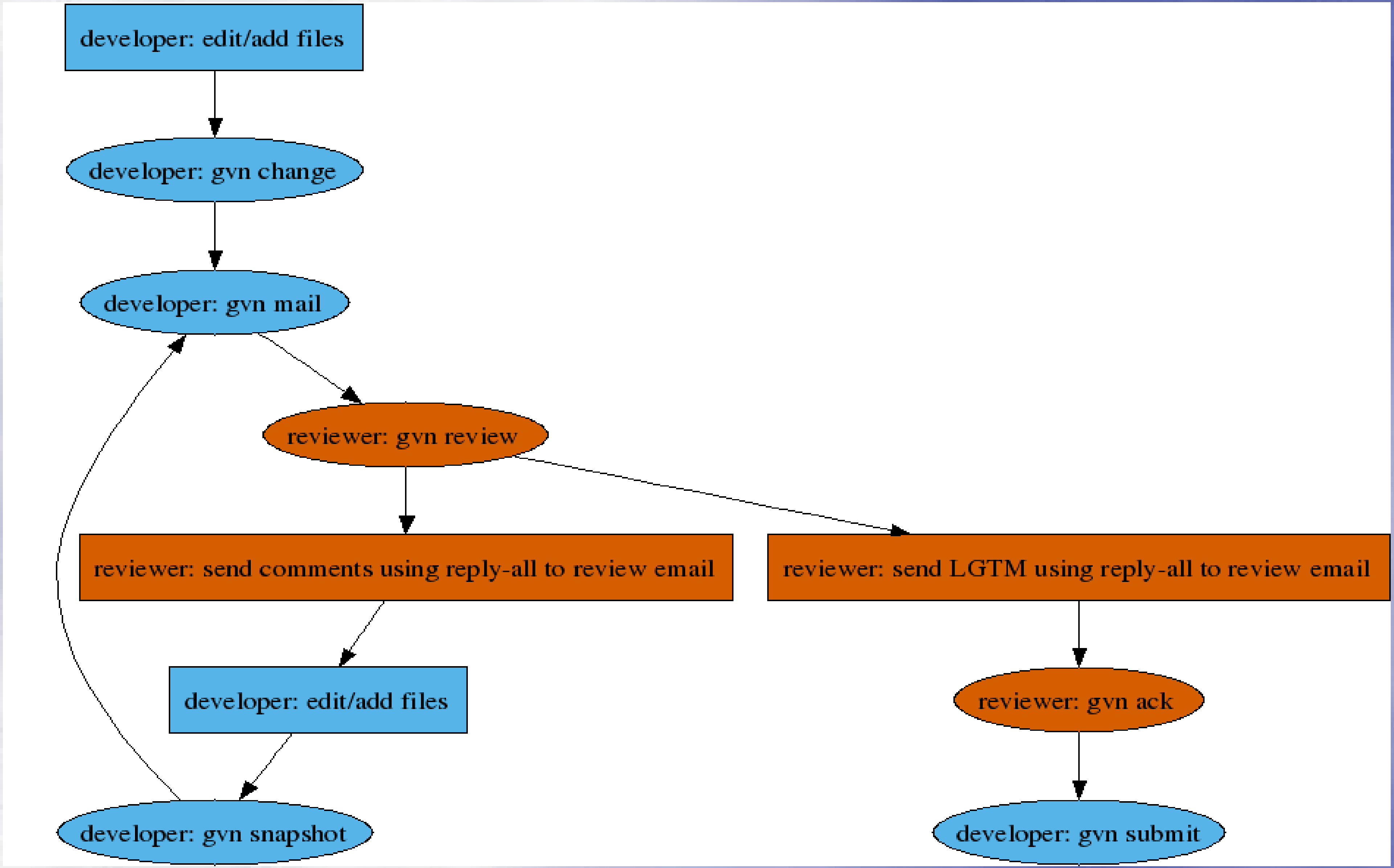
https://gvn.googlecode.com/svn/trunk/contrib/userdb/cyrus_gvn_userdb.c

```
For apache/webdav:
LoadModule  authz_svn_module      "modules/mod_authz_svn.so"
  # ... authn
  AuthType basic
  AuthName svn_repos
  AuthBasicProvider dir
  AuthUserDirectory /opt/svn/userdb
  Require valid-user

For svnserve/sasl2, install:
/usr/lib/sasl2/libgvn_userdb.so.0.0.0
and /usr/lib/sasl2/svn.conf:
pwcheck_method: auxprop
auxprop_plugin: gvn_userdb
gvn_userdb_path: /opt/svn/userdb
mech_list: ANONYMOUS CRAM-MD5
```

Gvn client

- Gvn is google's svn wrapper with changelist and code review support


developer: edit/add files
developer: gvn change
developer: gvn mail
reviewer: gvn review
reviewer: send comments using reply-all to review email
reviewer: send LGTM using reply-all to review email
developer: edit/add files
reviewer: gvn ack
developer: gvn snapshot
developer: gvn submit

Gvn Client

- gvn mail
- Reviewer does: gvn ack / gvn approve
- Sender can now submit branch thanks to gvn:approve
- gvn keeps track of which branch is submitted as what

```
saruman:~/svn/tools$ svn proplist --revprop -r 122139
Unversioned properties on revision 122139:
  gvn:approve:alexbooth
  svn:log
  svn:author
  gvn:submitted
  svn:date
saruman:~/svn/tools$ svn propget --revprop -r 122139 gvn:submitted
122149
saruman:~/svn/tools$ svn proplist --revprop -r 122149
Unversioned properties on revision 122149:
  svn:log
  svn:author
  gvn:change
  svn:date
saruman:~/svn/tools$ svn propget --revprop -r 122149 gvn:change
masri/AddedBoxInLI@122139
```

Gvn Svn Hooks

- Svn provides good hook infrastructure
- Little infrastructure exists to access it easily from python or other languages with a hook chain
- Gvn hook infrastructure fills that void
- Each hook invokes a list of “hooklets”, any can succeed, fail, or succeed and bypass other hooks
- Hooklets do not have to be written in python

```
https://gvn.googlecode.com/svn/trunk/bin/gvn-hook-runner: wrapper that svn hooks
are symlinked to
https://gvn.googlecode.com/svn/trunk/lib/gvn/hooks/runner.py eventually runs
gvn.hooks.info.HookInfo(argv, sys.stdin) which queues all the scripts for that
hook type (start-commit, post-commit, ..., post-unlock)

repos/hooks/pre-commit -> /opt/googlesvn/bin/gvn-hook-runner: hook dispatcher
repos/hooks/pre-commit.d/check_owners.py: example hook
```

Gvn & Hook Overview

- Gvn submits changes in /branches for reviews until they get acked and promoted to the main tree
- Pre-commit: check files as well as revision properties as well as node properties on new commit
- Pre-revprop-change: change a rev prop on an already committed revision
- gvn ack sets gvn:submitted and gvn:change
- gvn:superuser is a list of groups/users set as a node property on node “/”.
- svn propset "gvn:superusers" "group:svn-team" /
- gvn:superusers then allows
  gvn commit --with-revprop gvn:bypass-hooks filename

Pre-commit hooks

- block_bad_path_names.py: reject all files/directories with whitespace & forbidden windows characters
- block_monitor_commit_tests.py: hook to test svn and hooks but block the commit before a revision is created (used for monitoring, run last in the chain)
- block_some_prop_mods.py: block properties with bad names or gvn:superusers node prop
- bypass.py: allow skipping all remaining hook gvn:bypass-hooks if committing user is in the superusers list
- check_changebranch.py: block users from mucking about with other users' changebranch areas

Pre-commit hooks (2)

- check_gvn_revprops.py: block gvn:approve* & gvn:submitted, and check gvn:change edits
- check_owners.py: prevents commits in directory trees, unless submitter is allowed in OWNERS file, or was allowed through a gvn:approve:submitter revprop in the branch revision

Post-commit hooks

- mailer.py: gvn hook port of collabnet mailer
- verify.py: paranoid check of each revision after it's submitted

pre-revprop-change hooks

- block_bad_revprop_names.py: prevent adding rev properties with bad names.
- check_gvn_revprops.py: allow gvn to add gvn:approve:submitter and gvn:submitted, gvn:change, but no subsequent changes (users cannot delete their approval tracks).
- check_svn_revprops.py: block anything related to a svn:* rev property (including log entries).

Gvn Svn Hooks: Unittests

- Code that wasn't peer reviewed and doesn't have unittests is not that likely to work the day you wrote it, and even less so later in the future. Unittests are good [tm] :)
- Gvn and gvn hooks have many unittests

```
//gvn/testing/hooks/common.py:

def BuildPathCopyTxn(fs, path_from, txn_path, author, logmsg, pool, rev=None):
  """Create a test transaction from files in a directory path.

  Args:
    fs: svn_fs reference
    path_from: where on disk, to mirror a directory tree from
    txn_path: where in the depot to graft those files in, as a transation
    author: who owns that transaction
    logmsg: log message to link to the transaction
    pool: svn_pool reference
    rev: which revision this transaction is based on (None = HEAD)

  Returns:
    svn txn object

  This duplicates a file tree from path_from (without trailing slash)
  into a transaction inside txn_path (without a trailing slash)
  In many cases, txn_path should likely be empty to add those files at the
  top of the svn tree”””
```

Gvn Svn Hooks: Unittests

```
def BuildPropTxn(fs, prop_name, prop_val, path, author, logmsg, pool, rev=None):
  """Create a test transaction with a file attribute on an existing path.

  Args:
    fs: svn_fs reference
    prop_name: which property to create
    prop_val: which property value to set
    path: which file or directory to attach the property to, or "REV" for
          rev property
    author: who owns that transaction
    logmsg: log message to link to the transaction
    pool: svn_pool reference
    rev: which revision this transaction is based on (None = HEAD)

  Returns:
    svn txn object

  This adds the attaches prop_name: prop_value to path.
  txn_path is required to point to an existing file or directory"""

class HookTestCase(unittest.TestCase):
  def setUp(self):
    """create temp repo, test group file, and logger object."""

  # We provide 4 methods to test hooks:
  # _TestPaths allows to create a txn with files for a pre_commit hook
  # _TestProp does the same with a node prop or a revprop
  # _ChangeRevProps creates/changes/deletes a node or rev prop for a
  #                 pre_revprop hook
  # _TestFileDelete deletes a file from an existing repo
```

Svn Future

- Git easily beats svn in local state updates. Svn's local client code is its weakest point (quite slow), so SVN's local client code is being rewritten finally
- Maintainers have made it clear they have no intention to make SVN a DVCS but to maintain it for its current userbase and users interested in centralized systems
- Merge tracking is still being improved
- More details:

  http://lwn.net/Articles/381794/

References

- This talk and documents: http://marc.merlins.org/linux/talks/SvnScalingGvn/
- Tuning Document:

  http://www.orcaware.com/svn/mediawiki/index.php?title=Server_performance_tuning_for_Linux_and_Unix
- Sample svn tree with hook symlinks:

  http://marc.merlins.org/linux/talks/SvnScalingGvn/svn-tree/

Thanks

- Erik Kline: Initial Svn work at google
- Eric Gillespie: Svn project/tech lead/programmer
- David Glasser: Svn hacking/performance tuning
- Dan Christian: Svn benchmarking & perf tuning notes
- The rest of the Svn developer community

Questions?

- This talk and documents: http://marc.merlins.org/linux/talks/SvnScalingGvn/