

Gitify your life

web, blog, configs, data, and backups

Richard Hartmann,
RichiH@{freenode,OFTC,IRCnet},
richih.mailinglist@gmail.com

2012-03-11

Intro o o	ikiwiki ooo ooo	etckeeper o	vcsh oooo oooo	git-annex oooooo ooo	bup o	Zsh o	mr oo	Outro o oo o
-----------------	-----------------------	----------------	----------------------	----------------------------	----------	----------	----------	-----------------------

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr
- 9 Outro

Intro ○ ○	ikiwiki ○○○ ○○○	etckeeper ○	vcsh ○○○○ ○○○○	git-annex ○○○○○○ ○○○	bup ○	Zsh ○	mr ○○	Outro ○ ○○ ○
-----------------	-----------------------	----------------	----------------------	----------------------------	----------	----------	----------	-----------------------

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr
- 9 Outro

Who am I?

- Richard "RichiH" Hartmann
- Backbone and project manager at a German ISP
- freenode & OFTC staff
- Passionate about FLOSS
- Author of vcsh

What is git?

- Version control system
- Distributed
 - No need for central repository
 - Allows you to commit while offline
- Does not have a concept of files, only of diffs, internally
- Light-weight branches
- pre-/post-action hooks
- Full history in every checkout

Intro o o	ikiwiki ooo ooo	etckeeper o	vcsh oooo oooo	git-annex ooooooo ooo	bup o	Zsh o	mr oo	Outro o oo o
-----------------	-----------------------	----------------	----------------------	-----------------------------	----------	----------	----------	-----------------------

Outline

- 1 Intro
- 2 **ikiwiki**
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr
- 9 Outro

What is ikiwiki?

- Written in Perl
- Can use git or subversion as back-end
- Offers web-based editing and CLI push/pull
- Parses various markup languages
- Extensive templating and CSS support
- Acts as Wiki, CMS, and blog
- RSS and Atom feed for whole site, per page, per tag, etc
- Supports OpenID

Supported markup languages

- Markdown, extended to support
 - WikiLink ([[LinkToArticle]])
 - directive ([[!tag talk/gitify]], [[!author RichiH]], etc)
- WikiText
- HTML
- reStructuredText
- Textile

How does it work?

- User commits and pushes source files
- Partial/Full rebuild triggered by commit hook or web commit
- Parses input files
- Compiles into HTML, create new pages, updates RSS, etc
- Commits Markdown source for autogenerated/-changed pages into repository
- User then pulls changes to local repository

Common uses

- Public Wiki
- Private notes
- Blog
- CMS

Adding/editing content

- Web-based (useful, but boring)
- CLI-based (awesome!)
- GUI-based

Advanced usage

- Interface with source files, only
- Maintain wiki and docs in the same repository as your source code
- Separate staging or even preview branches

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper**
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr
- 9 Outro

In a word

- Implemented in POSIX shell
- Auto-commits /etc prior to and after all actions by package manager
- Hooks into apt, yum, pacman-g2, and cron
- Allows manual commits
- Various back-ends
 - bzip
 - darcs
 - git
 - mercurial
- Easy way to recover from failures, misconfiguration or to clone machines

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh**
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr
- 9 Outro

What is vcsh?

- Implemented in POSIX shell
- "version control shell" or "version control system \$HOME"
- Based on git
 - git unable to maintain several working copies in one directory
 - Sucks if you want to keep your configs in git
- vcsh uses fake bare git repositories to work around this
- Simple but powerful hook system
- Think of it as an extension to git

fake bare.. what?

- Normal git repo:
 - working copy in `$GIT_WORK_TREE`
 - git data in `$GIT_WORK_TREE/.git` aka `$GIT_DIR`
- Bare git repo:
 - git data in `$GIT_DIR`
 - no `$GIT_WORK_TREE`
- Fake bare git repo:
 - working copy in `$GIT_WORK_TREE`
 - git data in `$GIT_DIR`
 - `$GIT_WORK_TREE == $HOME`
 - `$GIT_DIR == $XDG_CONFIG_HOME/vcsh/repo.d/$repo.vcsh`
 - `core.bare = false`

Problems with fake bare git repos

- Fake bare repos are messy to set up and use
- Reason why git disallows shared \$GIT_WORK_TREE: complexity due to context-dependency
- Mistakes lead to confusion or data loss; imagine \$GIT_WORK_TREE set and
 - `git add`
 - `git reset --hard HEAD~1`
 - `git checkout -- *`
 - `git clean -f`

Solution: vcsh

- Wraps around git
- Hides complexity and does sanity checks
- Several git repos checked out into \$HOME at once
 - One repo for zsh, Vim, mplayer, etc
 - Enables specific subsets of repos per host
- Manages complete repo life-cycle

Create new repo

```
# create new repo
vcsh init vim
# add files to it
vcsh run vim git add .vim .vimrc
# commit using shorthand form
vcsh vim commit
# push using longhand form
vcsh run vim git push
```

Made-up life-cycle

```
# clone repo into new name zsh
vcsh clone git://github.com/RichiH/zshrc.git zsh
# optionally update legacy repos
vcsh setup zsh
# display all files managed by this repo
vcsh run zsh git ls-files
# rename repo just because
vcsh rename zsh zshrc
# delete repo
vcsh delete zshrc
```

run vs enter

```
# do everything from outside
vcsh run zsh git add .zshrc
vcsh run zsh git commit
vcsh run zsh git push
# the same, but from within
vcsh enter zsh
git add .zshrc
git commit
git push
exit
```

Advanced usage

- Have your prompt display vcsh information
- git-annex within vcsh to manage non-configuration files in \$HOME
- Floating backups in arbitrary working copies
 - .git/
 - Working copy
 - Complete repository, including objects, etc
- Use git on top of or in parallel to other VCSs

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex**
- 6 bup
- 7 Zsh
- 8 mr
- 9 Outro

What is git-annex?

- Based on git
- No need to check files into git
- Still able to check files into git if you want
- Able to maintain complete data history; does not do so by default
- Written with low bandwidth and flaky connections in mind
- Various work-flows

Internal workings 1/2

- Written in Haskell, so strong typing etc, internally
- Uses rsync to transfer data
- Moves files into `.git/annex/objects`
- Makes files read-only
- Stores location data in branch `git-annex`
- Puts symlink in place of file
- User adds and commits symlinks to master branch

Internal workings 2/2

- Read-only files force you to `git annex unlock` prior to changing them
- Ensures that you will `git annex add` all unlocked files
- `git-annex` can then discard or keep old data, depending on setup

Data integrity

- Set minimal number of required copies per suffix, directory, etc
- SHA1, SHA2- $\{224,256,384,512\}$ for integrity
- All remotes and special remotes can be verified
 - remotes verify locally and transmit the result
 - special remotes transfer all data to verify
- Verification takes required amount of copies into account
- `git fsck`; `git annex fsck`

Special remotes 1/2

- Store data in non-git-annex remotes
- Track all data stored in special remotes
- Support encryption for storage on untrusted machines/media
- Hook system lets you write to and read from arbitrary remotes

Special remotes 2/2

- bup
- directory
- rsync
- S3, Swift, etc
- Tahoe-LAFS
- web (media.ccc.de, Project Gutenberg, archive.org, etc)

The Archivist

- Put data into git-annex
- Distribute data among any number of drives, tapes, remotes, etc
- Store offline media in a safe place
- Maintain full information about number and location of all copies

Media consumption

- Import podcasts, videos, and slides
- Sync or export to consumption devices
- Consume media
- Drop consumed media from annex
- Deletion propagates through all annexes over time

The Nomad

- Keep copies of data on www
- Optionally sync between several local devices for backup
- Add data locally and/or remotely while on the road
- Sync data between local and remote once at an Internet café or similar
- Perfect for photos while travelling

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup**
- 7 Zsh
- 8 mr
- 9 Outro

In a word...

- Written in Python
- Fast
- Very space-efficient (reduced 120 GiB (rsnapshot) to 45 GiB)
- Built-in de-duplication
- Can be mounted via FUSE
- Can not drop old data (there is a branch that supports this)

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh**
- 8 mr
- 9 Outro

That's not git-based?

No, it's not...

- Extremely powerful tab completion for the tools in this talk (and others!)
- Versatile left *and right* prompts
- `vcs_info`
 - Displays information about the working copy in `$PWD` in prompt
 - Lots of customization options
 - Supports bazaar, codeville, cvs, darcs, fossil, git, GNU arch, mercurial, monotone, Perforce, svn, svk
- Too many other reasons to list (literally...)
- Can mimic Bash, Ksh, tcsh, etc; try it!

Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr**
- 9 Outro

Tying it all together

- Written in Perl
- Bulk pull, push, etc and custom commands all, some, or one of your repositories
- Supports subversion, git, cvs, mercurial, bzip, darcs, cvs, vcsh, fossil, veracity, unison, and git-svn
- Trivial to extend to support more VCSs
- If you want to try all this, why not `vcsh clone my mr repository template` and run `mr up` to pull my zsh config via vcsh?

Too many repos...?

Suggested mr layout

```
% cat /.mrconfig
include = cat /.config/mr/config.d/*
% ls .config/mr/available.d
mr.vcsh
zsh.vcsh
...
% ls -l .config/mr/config.d
mr.vcsh -> ../available.d/mr.vcsh
zsh.vcsh -> ../available.d/zsh.vcsh
...
%
```


Outline

- 1 Intro
- 2 ikiwiki
- 3 etckeeper
- 4 vcsh
- 5 git-annex
- 6 bup
- 7 Zsh
- 8 mr
- 9 **Outro**

The final pitch...

I need literally less than five minutes of Internet access to sync my entire digital life while on the road.

Project websites

Most of these are packaged for the major distributions

- ikiwiki: <http://ikiwiki.info/>
- etckeeper: <http://joey.kitenet.net/code/etckeeper/>
- vcsh: <https://github.com/RichiH/vcsh>
- git-annex: <http://git-annex.branchable.com/>
- bup: <https://github.com/apenwarr/bup>
- mr: <http://kitenet.net/~joey/code/mr/>
- Wiki around this topic:
<http://vcs-home.branchable.com/>

Previous talks

Previous talks, a bit more in-depth than this one and available as video download:

- vcsh: <http://fosdem.org/2012/schedule/event/vcsh>
- git-annex:
<http://fosdem.org/2012/schedule/event/gitannex>

The End!

Thanks!

Thanks for listening!

Questions? Ask now or follow me outside once my time-slot is over.

See slide footer for further contact information.

#vcs-home @ irc.oftc.net
vcs-home@lists.madduck.net