JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Git-ify Your (digital) Life

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de> | JSC Internal Seminar

Member of the Helmholtz-Association

Overview

Git	*a short review*
etckeeper	*keep your system's configs*
vcsh	*version your $HOME*
mr	*my / multiple repositories*
git-annex	*so meta!*
bup	*backup with Git*
gcrypt	*GPG-encrypted Git repositories*

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part I: Git

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

Version Control System *Git*

A short overview

- decentralized / distributed

  alike Mercurial/hg or Bazar in contrast to CVS or Subversion
- works on deltas (diffs, patches) instead of whole files
- non-linear history
- cryptological verification of revisions

  each revision (*commit*) has a unique SHA-1 hash computed from diff + meta info
- no need for a server / everything is locally available

  because of first point

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part II: etckeeper

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

etckepper – Keep Your System's Configurations

[1]

- creates a Git (or Mercurial/Bazaar/Darcs) repo for `/etc`
- uses additional meta-file for remembering permissions for each file
  DVCSs usually don't track file owner info; only executable bit
- uses *pre-* and *post-commit* hooks to fix file permissions
- hooks itself into package managers (e.g. *apt*, *zypper*) to auto-commit `/etc` before and after package changes
- manual commits also possible

[1] Windows users: sleep or think of moving to Linux

Member of the Helmholtz-Association

etckepper – Keep Your System's Configurations

Example 1

Initialization

```
etckeeper init
# after some time
cd /etc && git log --oneline
> 5bb2977 daily autocommit
> cdd9c8c yast update
> 9b76558 I added some cron jobs
> 711446f initial commit
```

Member of the Helmholtz-Association

etckepper – Keep Your System's Configurations

Examples 1 & 2

Initialization

```
etckeeper init
# after some time
cd /etc && git log --oneline
> 5bb2977 daily autocommit
> cdd9c8c yast update
> 9b76558 I added some cron jobs
> 711446f initial commit
```

Switching setup

```
# on April first
git checkout april_first_joke_etc
etckeeper init
# day later
git checkout master
etckeeper init
```

Member of the Helmholtz-Association

etckepper – Keep Your System's Configurations

Example 3

Get difference between two system's configs

```
git remote add my-other-host ssh://my-other-host/etc
git fetch my-other-host
git diff my-other-host/master group | head
> diff --git a/group b/group
> index 0242b84..b5e4384 100644
> --- a/group
> +++ b/group
> @@ -5,21 +5,21 @@ sys:x:3:
> adm:x:4:joey
> tty:x:5:
> disk:x:6:
> -lp:x:7:cupsys
> +lp:x:7:
```

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part III: vcsh

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

vcsh – Version Control System for (your) $HOME

version `.profile`, `.{bash,zsh,vim}rc`, `...` — without pollution

- separate Git repositories for dotfiles

  without polluting $HOME with `.git` directories
- easily migrate your personalized environment to other hosts

  clone your `.vim` repository on new host to have it synchronized
- allows for different branches for different hosts

  e.g. “`tklatt-zamws`”, “`myself-laptop`”, “`su-myserver`”
- *vcsh* is a single Shell script

Member of the Helmholtz-Association

vcsh – Version Control System for (your) $HOME

Examples

One repository for your Vim config . . .

```
vcsh init vim
vcsh vim add ~/.vimrc ~/.vim
vcsh vim commit -m "Initial commit of my Vim configuration"
vcsh vim remote add origin git@my-server.net:vim-repo
vcsh vim push -u origin master
```

Member of the Helmholtz-Association

vcsh – Version Control System for (your) $HOME

Examples

One repository for your Vim config . . .

```
vcsh init vim
vcsh vim add ~/.vimrc ~/.vim
vcsh vim commit -m "Initial commit of my Vim configuration"
vcsh vim remote add origin git@my-server.net:vim-repo
vcsh vim push -u origin master
```

. . . another for Zsh

```
vcsh init zsh
vcsh zsh add ~/.zsh ~/.zshrc ~/.zshenv
vcsh zsh commit -m "Initial commit of my Zsh configuration"
vcsh zsh remote add origin git@my-server.net:zsh-repo
vcsh zsh push -u origin master
```

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part IV: mr

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

mr – my / multiple repositories

One command to rule them all

- Problem: a bunch of *vcsh* repos are not very handy
- iterates over list of repos and runs same command on each
- can handle *Git*, *git-svn* and *vcsh* repos equally
- provides `bootstrap` command to setup/clone an environment on new host
- integrates well with *vcsh* (*mr* config directory can be a *vcsh* repo itself)
- a single Perl script

Example 1: Cloning and Bootstrapping

```
vcsh list
> vim zsh git ssh bin
mr update    # runs ‘git pull‘ or ‘git clone‘ for each
# downloads named .mrconfig and clones all repos in there
mr bootstrap https://my-server.net/.mrconfig
```

Member of the Helmholtz-Association

mr – my / multiple repositories

Examples

Status, Commit & Push

```
mr status
> mr status: /home/t.klatt/.config/vcsh/repo.d/git.git
> mr status: /home/t.klatt/.config/vcsh/repo.d/mr.git
> mr status: /home/t.klatt/.config/vcsh/repo.d/ssh.git
> mr status: /home/t.klatt/.config/vcsh/repo.d/vim.git
>  M .vim/bundles.vim
> mr status: /home/t.klatt/.config/vcsh/repo.d/zsh.git
>  M .zshrc
> mr status: finished (5 ok)

mr commit
# ‘git commit -a‘ is called on every dirty repo

mr push
# ‘git push‘ is called on every repo
```

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part V: git-annex

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

git-annex – Version Files Without Their Contents

So meta! [2]

- saves meta info (i.e. name, size, SHA-1) of files without their contents
- saves actual files read-only in `.git/annex/objects`

  symlinks them to original/real location
- keeps track of which remote has which files

  each remote identified by UUID
- designed for flaky connections

  uses *rsync* for data transfer

[2] Windows users: Wake up!

git-annex – Version Files Without Their Contents

I mean, so really meta!

- written in Haskell

  available for Linux, OSX, Android (beta), Windows (alpha)
- allows for special remotes
  - Amazon S3 / Glacier
  - WebDAV
  - rsync
  - the web (`http(s)://`, `ftp://`, archive.org, arxiv.org/[format]/[ID], etc.)
  - podcast feeds
  - XMPP
  - simple directories
- example collection of some conference proceedings (slides + video recordings)

  `https://github.com/RichiH/conference_proceedings`

Member of the Helmholtz-Association

git-annex – Version Files Without Their Contents

Example Szenario: The Archivist [3]

- annex all files
- actual files offline in special remotes on USB drives, tapes, etc.
- having full info about name, size and location of all files in one place at hand

Example

```
git annex whereis
> whereis my_cool_big_file (1 copy)
>     7570b02e-15e9-11e0-adf0-9f3f94cb2eaa  -- backup drive
> whereis other_file (3 copies)
>     0c443de8-e644-11df-acbf-f7cd7ca6210d  -- here (laptop)
>     62b39bbe-4149-11e0-af01-bb89245a1e61  -- usb drive
>     7570b02e-15e9-11e0-adf0-9f3f94cb2eaa  -- backup drive
```

[3] taken from official website

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part VI: bup

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

bup – Git for LARGE Files

- recap: Git is designed for plaintext files
  binary files are just a huge blob for Git; no diff possible
- uses Git object trees and replaces hashing and packing algorithms
- designed for space-saving incremental backups
- backups can be FUSE mounted
- can be a special remote for *git-annex*
- `bup web`: browse backup trees in web browser
- written in Python

Member of the Helmholtz-Association

bup – Git for LARGE Files

Examples

Initialization

```
bup init
> Initialized empty Git repository in /backup/homebck
bup index /home/t.klatt                                    # took a few seconds
bup save --name zamws-home-tklatt /home/t.klatt            # took about 3min
> Reading index: 203502, done.
> Saving: 100.00% (14743111/14743111k, 203502/203502 files), done.
```

Member of the Helmholtz-Association

bup – Git for LARGE Files

Examples

Initialization

```
bup init
> Initialized empty Git repository in /backup/homebck
bup index /home/t.klatt                                    # took a few seconds
bup save --name zamws-home-tklatt /home/t.klatt            # took about 3min
> Reading index: 203502, done.
> Saving: 100.00% (14743111/14743111k, 203502/203502 files), done.
```

Browsing

```
# after some time (and daily cron job for backup)
bup ls zamws-home-tklatt | grep -ce '^2014-.*'
> 71
du -sh /home/t.klatt $BUP_DIR
> 18G    /home
> 24G    /backup/homebck
```

Member of the Helmholtz-Association

The Crash

or stupid action (`cd ~ && rm -rf *`) *while drunk*

```
[t.klatt@zam]>_
```

bup – Git for LARGE Files

Example

Backup Recovering

```
cd /home/t.klatt
bup restore -C . zamws-home-tklatt/latest/home/t.klatt/
> Restoring ...  # takes some time; much longer than ‘bup init & bup save‘
```

Member of the Helmholtz-Association

JÜLICH
FORSCHUNGSZENTRUM

Git-based tools to ease your life

Part VII: gcrypt

May, 7th 2014 | Torbjörn Klatt <t.klatt@fz-juelich.de>

Member of the Helmholtz-Association

gcrypt – GPG-encrypted Git remotes

- implements a git-remote-handler to deal with `gcrypt::` remotes
  transport via *rsync*, *sftp* or *git*
- remote repository is GPG-encrypted for one or multiple participants
- each pack is encrypted with a symetric key stored in a asymetric encrypted manifest file
- can be a special remote for *git-annex*
- Hint: use it as a remote for your *etckeeper*'s repo
- Remark: You might want to use Joey "joeyh" Hess' fork of *gcrypt* [4]

[4] `https://github.com/joeyh/git-remote-gcrypt` because it has some bugfixes not yet merged upstream

gcrypt – GPG-encrypted Git remotes

Example

Initialization and Committing

```
git add my_secret_file & git commit -m "secret file"
git remote add secret-server gcrypt::git@my-server.net:secret-repo
git push secret-server master
```

Cloning as another Participant

```
git clone git@my-server.net:secret-repo
ls -lA secret-repo
> -rw------- 1 t.klatt users  303 Jan 15 09:24 0153f2b0...ea5f861d
> -rw------- 1 t.klatt users 1.4K Jan 15 09:24 91bd0c09...4881aa0a
> drwx------ 1 t.klatt users  138 Jan 15 09:25 .git
gpg -d 91bd0c09...4881aa0a
> fc564bef...94c3ff80 refs/heads/master
> pack :SHA256:0153f2b0...ea5f861d w+bxes2v...1MCkGi8+
> repo :id:3lmzxTGoXJVmHPtfaOTf
```

Member of the Helmholtz-Association

Project Links

etckeeper	https://github.com/agimenez/etckeeper
vcsh	https://github.com/RichiH/vcsh
mr	https://github.com/joeyh/myrepos
git-annex	https://git-annex.branchable.com/
bup	https://github.com/bup/bup
gcrypt	https://github.com/blake2-ppc/git-remote-gcrypt

Member of the Helmholtz-Association

Sources

- This talk is heavily inspired by Richard "*RichiH*" Hartman's talk at *Linuxtag 2013* [5]
- official and unofficial documentation of named tools
- (long-term) experiments with named tools

[5] `http://www.linuxtag.org/2013/de/program/mittwoch-22-mai-2013.html?eventid=147`

Member of the Helmholtz-Association

Thank you for your interest and time!

Questions?

(now or later)

Building 16.3
Room 022
Tel 96452
eMail t.klatt@fz-juelich.de

PGP-Key 0x64216AF3
Fingerprint 7176 4979 01E4 C412 BCCC
B403 F4E5 CF72 6421 6AF3

Member of the Helmholtz-Association