git for Sysadmins

St. Louis Unix User's Group

10 July 2019
L V Lammert

Why git?

- Distributed Version Control System
- More capable than CVS, SVN, et al
- Used for major projects – a la Linux Kernel
- Extremely powerful and flexible
- Distributed = designed for teams
- BUT ***most*** of the time all we need for Admin work is a history of previous version,
  so KISS rules!

Concept

- Git creates a *repository* in the working directory
- This *repository* is stored in a .git directory at the level it was created, e.g.
  /etc/.git
- git automagically tracks files under the repository directory according to .gitignore patterns

Benefits

- Save each and every file version by commit
- Allow reviewing and diff’ing any committed version
- Show status of every file in the repository with a single command

The Task Tonight

- Initialize a repository for /etc
- Decide what files should be kept in the repository
- Setup the template (.gitignore)
- Add files
- Change files
- Review changed files

Goals

- Manage the *project* directory (i.e. /etc)
- Keep a log of any recorded file changes
- Recover an older version of a file if required
- Ignore cruft that is ***not*** important
- Show file status at any time
- Option: Maintain a remote repository with the *project (imagine that, a* **backup**!*)*

Installation

- Available on almost any system
  hint: ‘Nix’s and ***others***
- Install with any package manager
- Which version is installed?
  $ git –version

Global Configuration

- git repositories – user owns local copy
- Global Username

```
git config --global user.name \
"git somebody"
```

- Global Email

```
git config --global user email “lvl@sluug.org”
```

Admin Issues

- git Global Configuration::
  - User *owns* repository
  - Not *multi-user!*
- Suggestion:
  - Use a commit message format of:
    “<Initials> What was done”
    “LVL: Created repository”

Step 1 - Create repository

- In the top directory (e.g. /etc)

```
git init
```

- What was created?

```
ls -al .git
<.git directory tree>
```

- How to *see* it from git?

```
git status
```

Step 2 – Set file pattern(s)

- .gitignore
- Ignore vim backups
  *~
- Ignore a directory
  <directory name>

Step 3 – Initial Status

- git status
  - All files
  - All directories
  - Logical pruning
- Rename “dist” files, [re]move

Step 3 – Add files

- Everything

  ```
  git add .
  ```

- By directory

  ```
  git add postfix/*
  ```

- By file

  ```
  git add passwd
  ```

- git commit <files> -m “LVL: Created repository”

Step 4 – Check Status

- git status
- Resolve any issues
- Change a file and watch the result

Step 5 – Differences

- Compare a changed file to the current *checked-in* version
  git diff <filename>
- Compare a changed file to a different revision
  git diff <filename> <md5>

Peace of mind!

But wait! There’s More!!

- /etc is done, Yeah!
- What about a backup?
  Save to a ***Remote*** repository
- What about an entire system?
  Tweak .gititnore

Remotes

- push to a remote repository
  - Imagine that – a ***backup***!
- git remote add backup \ lvl@apollo.omnitec.net:/<path>
- git push backup
- git pull backup
- git remote rm backup

More intelligent .gitignore

- Ignore everything

```
*
```

- Include specific directories

```
!/etc
!/home/lvl
!…..
```

Results

- Specific directories on the entire system!

Different Versions

- Compare a changed file to the current *checked-in* version

  ```
  git diff <filename>
  ```

- Compare a changed file to a different revision

  ```
  git diff <filename> <md5>
  ```

Resources

- git-scm.com
- gitref.org
- Top 10 git tutorials https://www.webfx.com/blog/web-design/git-tutorials-beginners/
- From Andrew: https://lathama.net/Home_Directory_Dot-files_in_Git

Thanks!

Lee Lammert
Omnitec Corporation
lvl@omnitec.net