

Subversion

Network Monitoring & Management

Contents

- What is version control?
- Introduction to SVN
- Basic principles
- Differences with CVS
- Commands
- Examples
- Configuring and accessing a repository

What is version control?

Three basic principals:

- Keep a record and history of changes
- Give public access to the information
- To maintain different versions from the same data set

What types of data?

Source code,

- Documentation
- Configuration files
- Generally, any type of data

Terminology

repository

- A centralized copy of all files being tracked, structured in directory trees

Working copy

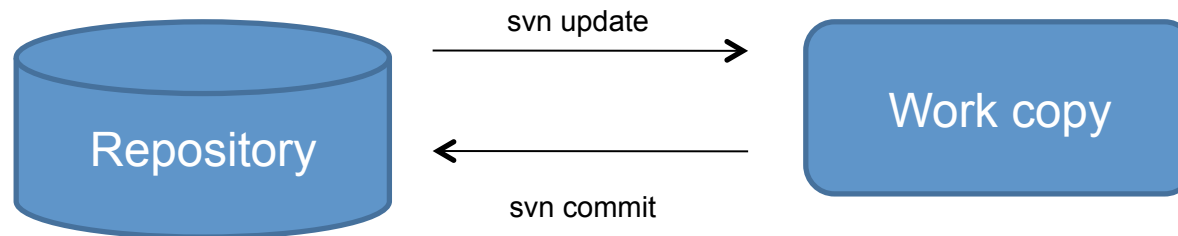
- A local copy of data that can be changed and synchronized with the repository. Contains special information that allows for synchronization.

Version

- A group of directories and files that reflect the repository state at a determined moment.

Basics

- The repository is the master copy
- All work is done in a work copy
- Changes are reflected and appear in the repository (using the *commit* command)



Change control: states

Without change and updated

- Copy is identical to the repository
- A *commit* or *update* does nothing

Local changes and updated

- The local copy has changed and the repository has not received the changes.
- A *commit* will update the repository. *update* does nothing.

Without changes and not updated

- Local copy has not changed, but the repository has changed.
- *update* will change local state, *commit* won't work.

Local change and not up-to-date

- Conflict! Need to run *update*
- If SVN cannot resolve the conflict automatically, then a manual resolution will be required.

Sample session

- Initial repository checkout
 - `svn checkout <project>`
 - `vi <myfile.conf> (...changes ...)`
 - `svn commit <myfile.conf> (reflects changes)`
- More changes:
 - `svn update`
 - `vi <myfile.conf>`
 - `svn commit <myfile.conf>`

SVN and the repository

Clients can access locally or via the network

SVNROOT environment variable:

SVNROOT=

- /svn/myproject # local disk
- svn://svnserver/svn/myproject # via svnserve
- svn+ssh:// svnserver/svn/myproject # via SSH

Creating a repository

Installation

- #apt-get install subversion
- #svncreate <repository>
- Edit <repository>/

Create as a “service”

- Create /etc/init.d/subversion, which basically includes
 - svnserve -d -r <repository>
- #chkconfig --add subversion
- #chkconfig --level 2345 subversion on

Edit permissions

- Edit <repositorio>/conf/svnserve.conf
- Specify the password file:
 - [general]
 - password-db = <userfile>
 - realm = example realm
- Create users:
 - [users]
 - pedro = foopassword
 - sandra = barpassword

SVN - clients

- There are clients for most operating systems:
 - svn (UNIX)
 - TortoiseSVN (Windows)
 - ...
- Local access or via the network

SVN Commands

- **import**
 - Import a new project from a repository
- **checkout (co)**
 - Copy the repository to a local directory
- **update (up)**
 - Update the local copy from the repository
- **add**
 - Add or new file or directory to the local copy
- **delete**
 - Remove a file from the local copy
- **commit**
 - Update the repository from the local copy

Other useful commands

- **mkdir**
 - Add a directory to the local copy
- **status**
 - File version and status
- **diff**
 - Show the differences between a local element (file, directory) and the item in the repository.
- **log**
 - Show the change history for one or more files
- Many others: copy, export...

Work cycle

- Update the work copy
 - svn update
- Make changes to your local copy
 - svn add
 - svn delete
 - svn copy
 - svn move
- Check your changes
 - svn status
 - svn diff
 - svn revert
- Combine your changes with others
 - svn merge
 - svn resolve
- Complete your changes and place them in the repository
 - svn commit

Advantages & differences w/ CVS

- CVS only controls changes to files
- SVN creates a virtual file system that includes directories
- CVS cannot control name changes or copies of files
- The way in which SVN controls directories allows for name changes and copies of files.
- SVN permits “atomic” change control: all changes or no changes will be accepted
- CVS does not provide similar functionality
- In general SVN provides more flexibility for access, such as HTTP via Apache and the advantages this provides.

Conclusions

- A sophisticated version control system
- Very useful for programmers
- For network administrators many of the higher-level functions are not necessary
- In reality, CVS and Subversion can both be used to assist with network administration.
- However, one cannot ignore:
 - The most popular tool is the tool that receives the most support
 - Many of us give support to programming teams in our daily work

References

- “Version Control with Subversion” – O’Reilly
- Online and free at <http://svnbook.red-bean.com>