

# SUBVERSION

And how to manage  
multiple source repositories

2006-02-25

Matteo Redaelli

## Table of Contents

Introduction.....	3
Features.....	3
Requirements.....	4
Installation.....	4
Initial setup.....	4
Adding new users.....	5
Adding a new repository.....	5
Upgrade subversion and repositories to a new release.....	5
References.....	6

## Introduction

“The goal of the Subversion project is to build a **version control system** that is a compelling replacement for CVS in the open source community. The software is released under an Apache/BSD-style open source license” and used in many worldwide projects (Apache, kde, gcc, python, samba, mono, zope, GnuPG, netfilter,... See <http://subversion.tigris.org/testimonials.html>)

## Features

- **Most current CVS features.**

Subversion is meant to be a better CVS, so it has most of CVS's features. Generally, Subversion's interface to a particular feature is similar to CVS's, except where there's a compelling reason to do otherwise.

- **Directories, renames, and file meta-data are versioned.**

Lack of these features is one of the most common complaints against CVS. Subversion versions not only file contents and file existence, but also directories, copies, and renames. It also allows arbitrary metadata ("properties") to be versioned along with any file or directory, and provides a mechanism for versioning the 'execute' permission flag on files.

- **Commits are truly atomic.**

No part of a commit takes effect until the entire commit has succeeded. Revision numbers are per-commit, not per-file; log messages are attached to the revision, not stored redundantly as in CVS.

- **Apache network server option, with WebDAV/DeltaV protocol.**

Subversion can use the HTTP-based WebDAV/DeltaV protocol for network communications, and the Apache web server to provide repository-side network service. This gives Subversion an advantage over CVS in interoperability, and provides various key features for free: authentication, path-based authorization, wire compression, and basic repository browsing.

- **Standalone server option.**

Subversion also offers a standalone server option using a custom protocol (not everyone wants to run Apache 2.x). The standalone server can run as an inetd service, or in daemon mode, and offers basic authentication and authorization. It can also be tunnelled over ssh.

- **Branching and tagging are cheap (constant time) operations**

There is no reason for these operations to be expensive, so they aren't.

Branches and tags are both implemented in terms of an underlying "copy" operation. A copy takes up a small, constant amount of space. Any copy is a tag; and if you start committing on a copy, then it's a branch as well. (This does away with CVS's "branch-point tagging", by removing the distinction that made branch-point tags necessary in the first place.)

- **Natively client/server, layered library design**

Subversion is designed to be client/server from the beginning; thus avoiding some of the maintenance problems which have plagued CVS. The code is structured as a set of modules with well-defined interfaces, designed to be called by other applications.

- **Client/server protocol sends diffs in both directions**

The network protocol uses bandwidth efficiently by transmitting diffs in both directions whenever possible (CVS sends diffs from server to client, but not client to server).

- **Costs are proportional to change size, not data size**

In general, the time required for a Subversion operation is proportional to the size of the *changes* resulting from that operation, not to the absolute size of the project in which the changes are taking place. This is a

property of the Subversion repository model.

- **Choice of database or plain-file repository implementations**

Repositories can be created with either an embedded database back-end (BerkeleyDB) or with normal flat-file back-end, which uses a custom format.

- **Versioning of symbolic links**

Unix users can place symbolic links under version control. The links are recreated in Unix working copies, but not in win32 working copies.

- **Efficient handling of binary files**

Subversion is equally efficient on binary as on text files, because it uses a binary diffing algorithm to transmit and store successive revisions.

- **Parseable output**

All output of the Subversion command-line client is carefully designed to be both human readable and automatically parseable; scriptability is a high priority.

- **Localized messages**

Subversion uses gettext() to display translated error, informational, and help messages, based on current locale settings.

## Requirements

Linux Debian Sarge.

## Installation

With the user “root”:

- apt-get update
- apt-get install libapache2-svn subversion-tools subversion
- Create a user “svnadmin” with the primary group “www-data” and a secondary “svnadmin”
- set “umask 002” in \$HOME/.bash\_profile
- install sudo with “apt-get install sudo” and configure it to allow “svnadmin” to start and stop apache2 ( i.e “/etc/init.d/apache2”)
- chgrp svnadmin /etc/apache2/mods-available/dav\_svn.conf

## Initial setup

With the user “svnadmin”:

- mkdir \$HOME/projects
- mkdir \$HOME/etc
- create a password file and an initial user (user “myuser” and password “mypassword”) with “htpasswd2 -c \$HOME/etc/svn.passwd myuser mypassword”

## Adding new users

With the user “svnadmin” you can add the new user “pippo” (password “pluto”) with “htpasswd2 \$HOME/etc/svn.passwd pippo pluto”

## Adding a new repository

With the user “svnadmin”:

- `svnadmin create $HOME/projects/pirellicom`
- Add the following rows to the file “/etc/apache2/mods-available/dav\_svn.conf”

```
<Location /svn_pirellicom>
  DAV svn
  SVNPath /home/svnadmin/projects/pirellicom

  AuthType Basic
  AuthName "Subversion Repository PIRELLICOM"
  AuthUserFile /home/svnadmin/etc/svn.passwd
  ErrorDocument 404 default

  <LimitExcept GET PROPFIND OPTIONS REPORT>
    Require valid-user
  # Require user pirellicom
</LimitExcept>
</Location>
```
- `sudo /etc/init.d/apache restart`
- Test with a browser the anonymous login to the new WebDav folder using the url “`http://server/dva_pirellicom`”

## Upgrade subversion and repositories to a new release

In Sarge the standard subversion release is 1.1.3. In order to manage lock of file we had to upgrade to the release 1.2.3 with the following steps:

With “svnadmin”:

- `svnadmin dump $HOME/projects/pirellicom > /tmp/pirellicom.dmp`

With “root”:

- add to the file “/etc/apt/sources.list” the row

```
deb http://ftp.estpak.ee/backports.org sarge-backports main contrib non-free"
```
- `apt-get update`
- `apt-get install libapache2-svn subversion-tools subversion`
- remove the previous entry in file “/etc/apt/sources.list”
- `apt-get update`

With “svnadmin”:

- `mv $HOME/projects/pirellicom $HOME/projects/pirellicom.old`
- `svnadmin create $HOME/projects/pirellicom`
- `svnadmin load $HOME/projects/pirellicom < /tmp/pirellicom.dmp`

## References

[1] <http://subversion.tigris.org/>

[2] <http://www.nongnu.org/cvs/>

[3] <http://www.apache.org/>