ConVox SVN Documentation

Author: Randhir Prasad

Dare: 16-12-2021

SVN stands for Subversion

Apache Subversion (SVN) is a software versioning and revision control system distributed under an open source license. Subversion is part of a rich community of developers and users. This document provides you an understanding on SVN system that is needed to maintain the current and historical versions of files such as source code, encrypted code and documentations.

Version Control System (VCS) is a software that helps software developers to work together and maintain a complete history of their work.

Following are the goals of a Version Control System

- Allow developers to work simultaneously.
- Do not overwrite each other's changes.
- Maintain history of every version of everything.

A **VCS** is divided into two categories.

- Centralized Version Control System (CVCS), and
- Distributed/Decentralized Version Control System (DVCS).

Subversion falls under centralized version control system, meaning that it uses central server to store all files and enables team collaboration.

Important Terminologies

Repository: A repository is the central place where developers store all their work with history about changes. Repository is accessed over a network, acting as a server and version control tool acting as a client. Clients can connect to the repository, and then they can store/retrieve their changes to/from repository. By storing changes, a client makes these changes available to other people and by retrieving changes, a client takes other people's changes as a working copy.

Trunk: The trunk is a directory where all the main development happens and is usually checked out by developers to work on the project.

Tags : The tags directory is used to store named snapshots of the project. Tag operation allows to give descriptive and memorable names to specific version in the repository.

Branches: Branch operation is used to create another line of development. It is useful when you want your development process to fork off into two different directions. e.g. when you release version 5.0, you might want to create a branch so that development of 6.0 features can be kept separate from 5.0 bug-fixes.

Working copy: Working copy is a snapshot of the repository. The repository is shared by all the teams, but people do not modify it directly. Instead each developer checks out the working copy. The working copy is a private workplace where developers can do their work remaining isolated from the rest of the team.

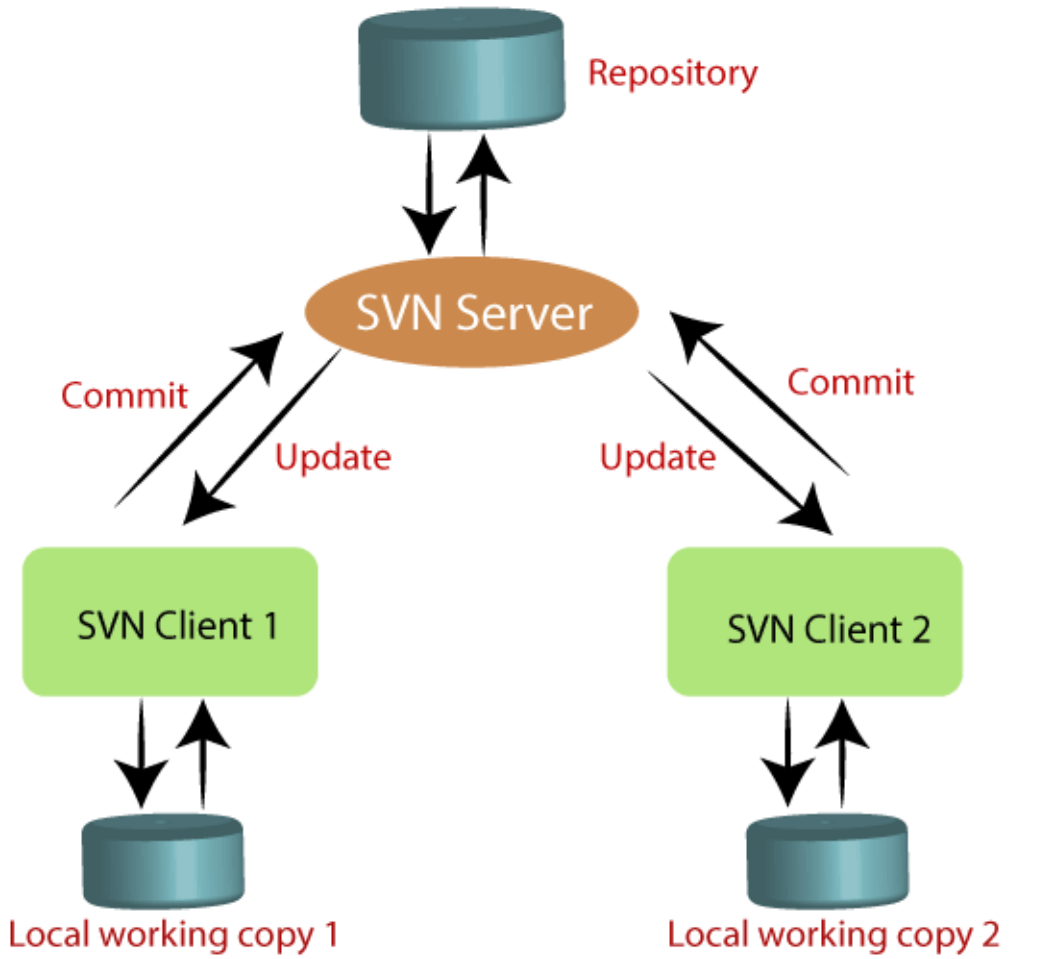
Commit changes: Commit is a process of storing changes from private workplace to central server. After commit, changes are made available to all the team. Other developers can retrieve these changes by updating their working copy. Commit is an atomic operation. Either the whole commit succeeds or is rolled back. Users never see half finished commit.

SVN Components

There are two core components of SVN; they are as follows:

- SVN server
- SVN client

These components can be considered as web servers and web browsers. The client acts as the browser that accesses the data on the server.



SVN Installation

To check whether it is installed or not use following command.

svn --version

svn --version --quiet

If you not getting any SVN version then use yum command and install SVN in your server

yum install svn*

yum install subversion*

If you wish to update your SVN then use command as:

yum update svn*

User Setup

htpasswd command is used to create and update the plain-text files which are used to store usernames and passwords for basic authentication of HTTP users. '-c' options creates password file, if password file already exists, it is overwritten. That is why use '-c' option only the first time. '-m' option enables MD5 encryption for passwords.

htpasswd -cm /etc/svn-users test_user

htpasswd -m /etc/svn-auth-accounts test_user

SVN Commands

usage: svn <subcommand> [options] [args]

To get help and to know SVN available subcommands we use following command

svn help

1. svn create is used to create a new repository. Most of the times this operation is done only once.

svn create <test_project>

2. The svn checkout command is used to create the working copy of the SVN project. The checkout operation is needed to be performed once after each change occurs in the directory structure. If the directory structure is changed, we may need to re-check out it. This command will be executed as follows:

svn checkout <URL Path>

svn co <URL Path>

e.g. svn co http://192.168.1.23/svn/ConVoxCCS3.2/tags/ConVoxCCS3.2.4_GoDigit/

3. The svn add command is used to add the files in the repository for the SVN. Whenever we create a new file in our working copy, we have to send it to the SVN server. This command will be executed as follows:

svn add <filename>

e.g. svn add doc/references/changes_data_upload_api_08092021.txt

Note: This file will be visible after an SVN commit only.

4. The svn delete command is used to remove the files from the repository. When we perform a delete operation, it removes the file from the working. To delete it from the repository, run a commit command after the delete command. The svn delete command will be executed as follows:

svn delete <filename>

svn del <filename>

e.g. svn del var/www/html/ConVoxCCS/TETS_ENC/index.php

Note: This file will be deleted from the repository after an SVN commit only.

5. The svn command is used to save the changes made on the repository. Whenever we made changes on our working copy and want to reflect it on the SVN server. In such a case, we have to make a commit operation. The commit command will be executed as follows:

svn commit -m "Commit message**"**

Note: Give meaningful and clear commit message inside "" only. The commit message is a message for the audience that we are making changes on the project.

For every cusomer we have two tags one is source and another is encrypted. we always commit source code/files to source tag and encrypted code/files to encrypted tag.

6. The svn diff command is used to display the differences between two versions of files. We can find the differences between the working copy and the remote (SVN) copy. We can also find the two revisions, paths, and more.

svn diff -Revision1: Revision2 <filename>

e.g. svn diff -r5512:r5324 http://192.168.1.23/svn/ConVoxCCS3.2/tags/encrypted/ConVoxCCS3.2.4_GoDigit_ENC/usr/share/convox/convox_customer_leads.pl

7. The svn status command displays the status of the working copy. It shows the status whether the repository is updated, added/deleted, or file is not under revision control and more.

svn status <path>

svn st <path>

e.g. svn st /var/www/html/ConVoxCCS/

8. The svn log command is used to display all the commits made on the repository or file.

svn log <Path>

e.g. svn log http://192.168.1.23/svn/ConVoxCCS3.2/tags/ConVoxCCS3.2.4_GoDigit/

svn log http://192.168.1.23/svn/ConVoxCCS3.2/tags/ConVoxCCS3.2.4_GoDigit/ |less

9. The svn move command is used to move the files from the working directory. However, these files can

be sent to the SVN server by commit operation.

svn move <src> <dest>

Note: The above command will move the file to the targeted destination. Commit the file to make the changes on the repository.

10. The svn rename command is used to rename the files.

svn rename CURR_PATH NEW_PATH

11. The svn list command is used to display the content of the repository. It is useful in the case; you want to view the details of the repository without making a working copy and without downloading it. The svn list command with the verbose option will provide more description of files. It will display the revision number of the last commit, Author, size, date, and time of the last commit.

svn list

e.g. svn list http://192.168.1.23/svn/ConVoxCCS3.2/tags/ConVoxCCS3.2.4_GoDigit/

svn list -v http://192.168.1.23/svn/ConVoxCCS3.2/tags/ConVoxCCS3.2.4_GoDigit/

12. The update command is used to update the working copy of the project. It brings the changes from the working copy to the repository. It matches the working copy with the HEAD by default. It is also used in the case when changes are made by the other users; we have to update the repository

svn update <path>

e.g. svn update /root/ConVoxCCS3.2.4_GoDigit/

13. The svn info command provides a quick look at the working copy. It is accessible in the local working copy, and it does not communicate with the SVN server.

svn info <path>

e.g. svn info /var/www/html/ConVoxCCS/

14. The svn merge is used to apply two differences between two sources to a working path.

svn merge SOURCE1[@N] SOURCE2[@M] [TARGET_PATH]

15. The svn copy command is used to copy the particular tag using the reference tag.

svn copy <reference tag file path> <new tag path>

e.g. svn copy http://192.168.1.23/svn/ConVoxCCS3.2/tags/ConVoxCCS3.2.4_GoDigit_MotorClaims/ http://192.168.1.23/svn/ConVoxCCS3.2/tags/encrypted/ConVoxCCS3.2.4_GoDigit_MotorClaims_Enc/ -m "Creating encrypted tag for Godigit General Insurance Ltd. (ACC6026) on 16-12-2021"

Note: Using svn copy command we create new svn tag for a new customer.

SVN Add Multiple Files:

svn st | grep "^?" | awk '{print $2}' | while read k; do svn add $k; done

SVN Delete Multiple Files:

svn st | grep "^!" | awk '{print $2}' | while read k; do svn del $k; done

To encrypt our files

Samba server

ssh 192.168.1.90

Command:

> **/usr/src/ioncube_encoder5_basic_7.0/ioncube_encoder5** Directory_Name/ **-o** Directory_Name_Enc/