Introduction to GIT on Linux – Install, Create Project, Commit Files

GIT is the most versatile distributed version control system.

The way GIT tracks and handles file changes is very efficient and different than how other version control software tracks the changes (including CVS and Subversion).

This article is for those who are new to GIT. This is a jump-start guide that will show you how to install GIT from source, create a new project, commit changes to the GIT repository.

GIT doesn’t use the client-server model. When you download a project from a remote GIT repository, you download everything, including the version history and changes of the individual files, and your local GIT acts as a server, where you can do check-in, check-out, and all other typical version control activities. Later when you are ready, you can merge your changes to the remote GIT repository.

If you are a developer, you might want to install GIT on your local machine for two reasons: 1) You like to manage your own project locally using a version control tool 2) You want to modify a code that is located in a remote central GIT repository.

If you are a sysadmin, you might want to install GIT on a server so that it acts as a central repository to hold all the source code for your company. From here, you can allow developers to download the projects to their local GIT repositories, make changes and they can check-in back to your central repository when they are done.

Irrespective of how you are planning to use GIT, the installation steps, and the basic commands mentioned below are exactly the same.

### 1. Download and Install GIT

First, download the GIT from [here](http://git-scm.com/download). Or, download it directly using wget as shown below.

cd

wget http://kernel.org/pub/software/scm/git/git-1.7.6.tar.bz2

Next, extract the downloaded file.

tar xvfj git-1.7.6.tar.bz2

cd git-1.7.6

Finally, install GIT as shown below using the default configure option. If you want to customize the installation, do “./configure –help” to view all available configuration options.

./configure

make

make install

### 2. Initial Configuration

Git is installed by default under /usr/local/bin. Once you’ve installed GIT, verify it as shown below.

$ whereis git

git: /usr/local/bin/git

$ git --version

git version 1.7.6

$ git --help.

The first step is to specify your username and email address to your GIT repository using “git config” as shown below.

git config --global user.name "GIT Admin"

git config --global user.email ramesh@thegeekstuff.com

Verify the git configuration information as shown below.

$ git config --list

user.name=GIT Admin

user.email=ramesh@thegeekstuff.com

core.repositoryformatversion=0

core.filemode=true

core.bare=false

core.logallrefupdates=true

This information is stored in the .gitconfig file under your home directory.

$ cat ~/.gitconfig

[user]

name = GIT Admin

email = ramesh@thegeekstuff.com

### 3. Create a Project

You can make any of your local directory as a GIT project (i.e repository). For example, if your project is located under /home/ramesh/projects/passworddragon, you can make that as your GIT project. First, cd to that directory, and execute git init as shown below.

$ cd /home/ramesh/projects/passworddragon

$ git init

Initialized empty Git repository in /home/ramesh/projects/passworddragon/.git/

This will create a .git directory under your project folder. Following is the content of the .git directory. GIT uses this directory to store information on how it is tracking the changes.

$ ls -altr .git

total 40

drwxrwxr-x 4 git git 4096 Aug 13 22:39 refs

drwxrwxr-x 4 git git 4096 Aug 13 22:39 objects

drwxrwxr-x 2 git git 4096 Aug 13 22:39 info

drwxrwxr-x 2 git git 4096 Aug 13 22:39 hooks

-rw-rw-r-- 1 git git 23 Aug 13 22:39 HEAD

-rw-rw-r-- 1 git git 73 Aug 13 22:39 description

-rw-rw-r-- 1 git git 92 Aug 13 22:39 config

drwxrwxr-x 2 git git 4096 Aug 13 22:39 branches

drwxrwxr-x 36 git git 4096 Aug 13 22:39 ..

drwxrwxr-x 7 git git 4096 Aug 13 22:39 .

Note: If you are sysadmin, who is trying to create a GIT central repository for your company, from where developers can download the projects, you may want to create a username called ‘git’ and organize all your projects under this account. For example: /home/git/project1, /home/git/project2, etc. Once you have the project organized, cd to the project directory, and do ‘git init’ from there as git user.

### 4. Add and Commit files to the Project

Once you’ve initialized the project using “git init”, add the files located under this project directory, using “git add”.

If there are different types of files under your project directory, and you want GIT to manage only certain types of files, add only those to the GIT as shown below. This example adds only the \*.java and \*.c files.

git add \*.java

git add \*.c

Typically you would like to add all the files under the project directory to the GIT project. Just do “git add .”, which will add all the files in the current directory and all the sub-directories to the GIT project.

git add .

Once you’ve added the files to the repository, you should commit those files, as shown below.

$ git commit -m 'Initial upload of the project'

create mode 100755 PasswordDragon.java

create mode 100755 pwm/ui/DataManager.java

create mode 100755 pwm/ui/PasswordFrame.java

create mode 100755 pwm/tools/StrongEncryption.java

create mode 100755 pwm/tools/PasswordStrength.java

..

If you didn’t specify your username and email address using “git config” as explained above, you’ll get the following error message.

$ git commit -m 'Initial upload of the project'

\*\*\* Please tell me who you are.

Run

git config --global user.email "you@example.com"

git config --global user.name "Your Name"

to set your account's default identity.

Omit --global to set the identity only in this repository.

fatal: empty ident not allowed

### 5. Make Changes and Commit the File

You’ve installed GIT, created a project repository, committed all the files to the GIT project.

Now it is time to start making some changes to a file and commit it to the repository.

vi PasswordDragon.java

Once you’ve modified a file locally, you can view the changes. i.e The difference between your local copy, and the copy already committed in the GIT project using “git diff” as shown below.

$ git diff

diff --git a/PasswordDragon.java b/PasswordDragon.java

index 6166ed1..fd82d32 100644

--- a/PasswordDragon.java

+++ b/PasswordDragon.java

@@ -2,7 +2,7 @@

- public counter=10

+ public counter=55

Once you’ve made modifications to it, reviewed the changes, and happy with it, you can commit the file to GIT repository. This is a two step process. First, you should add the file to the staging area, and commit to the GIT project as shown below.

git add PasswordDragon.java

When you perform commit, it will open your default editor, where you can enter the comment. Once you save your comment and exit the editor, it will commit the file to the GIT project and display the following message.

$ git commit

[master 80f10a9] Added password strength meter functionality

1 files changed, 56 insertions(+), 7 deletions(-)

Note: You can also do “git commit -a”, which will do the add and commit at the same time.

### 6. View Status and Commit Logs

From your local repository, when you perform “git status”, it will display the current status. When the local copy is not changed (or when all the files are committed), you’ll see the following message.

$ git status

# On branch master

nothing to commit (working directory clean)

If you’ve made changes to a file, and not committed yet, you’ll see the following message.

$ git status

# On branch master

# Changes not staged for commit:

# (use "git add ..." to update what will be committed)

# (use "git checkout -- ..." to discard changes in working directory)

#

# modified: PasswordDragon.java

#

no changes added to commit (use "git add" and/or "git commit -a")

You can also view the history of a file as shown below.

$ git log PasswordDragon.java

commit c919ced7f42f4bc06d563c1a1eaa107f2b2420d5

Author: GIT Admin

Date: Sat Aug 13 22:54:57 2011 -0700

Added password strength meter functionality

commit c141b7bdbff429de35e36bafb2e43edc655e9957

Author: GIT Admin

Date: Sat Aug 13 20:08:02 2011 -0700

Initial upload of the project