Version Control System GIT

JANUARY **01**

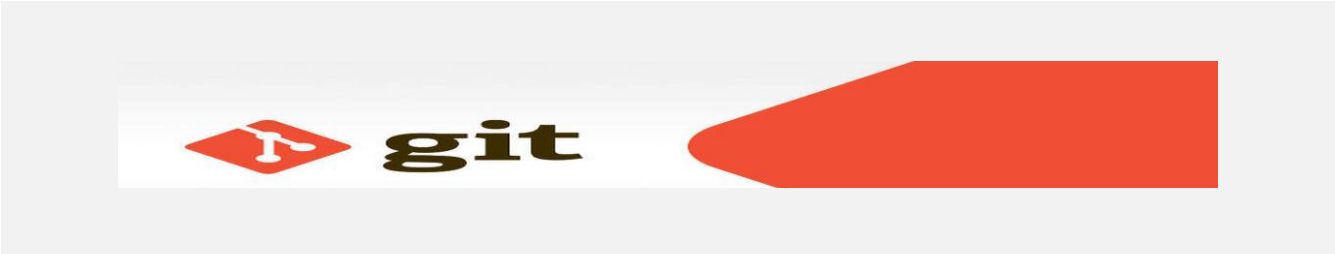
The Dark Web OSEye
Authored by: Shailesh Gupta

The Dark Web OSeye Technical & Education

GIT

Version Control System

Git is a distributed version-control system for tracking changes in any set of files, originally designed for coordinating work among programmers cooperating on source code during software development.



Benefit:

- Easily Recover File
- Who Introduce Issue and When
- Roll Back To Previous Working State

Type of Version Control:

- Local Version Control system
- Centralized version control system
- Distributed Version Control system

Local Version Control System:

The version control system in which user can create and install their own local version control system on there own personal system or environment.

{**Bidirectional DATA Base**}

DATA BASE **USER**

Benefit:

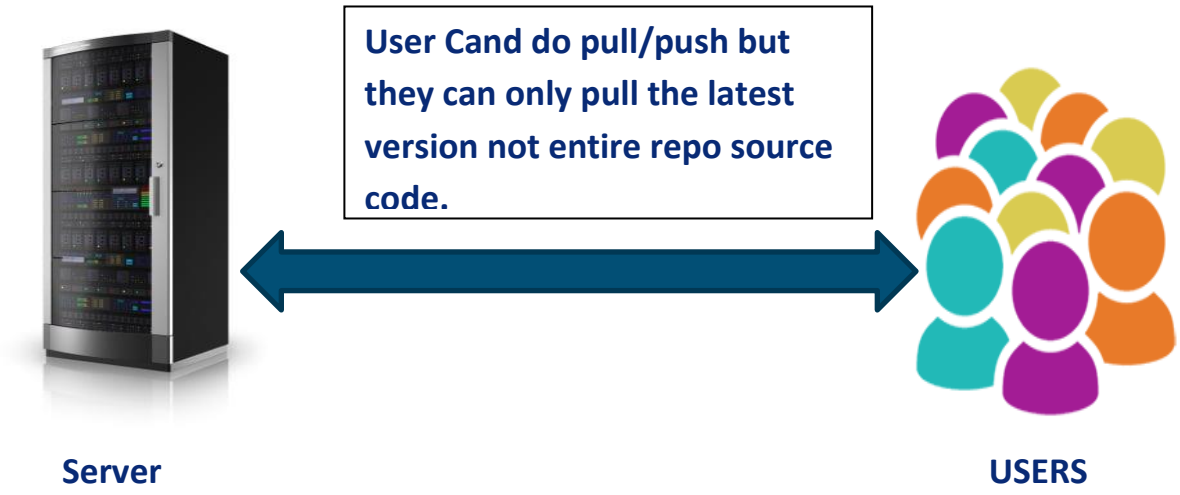
- Something is better that nothing

Drawback:

- Only accessible for one user
- Share is possible only manually
- If we lose the system, we can loss all data

Centralized Version Control System:

This is the version control system in which we can create a central data base on server and multiple users can access that version control system. But while access (Pull & Push) version control system it only pulls the latest version of software and changes. And in case if user loss their data they will fetch it from server but is server is damage entire data or software are vanished.



Distributed Version Control System:

This is the version control system in which we can pull/push the data from the server. But in this when we are going to pull the any repository it pull entire source code of the repository (pull entire repository). So, if the global repository will not work or damage. We can easily recover the entire data and if some time server is damage, we can also configure and restore the backup easily.

For example… GitHub, Bitbucket etc.

GIT

This is the distributed version control system. By using this we can manage the version change history. In general word we can say that it track the version change.

Benefit:

- Easily Recover File
- Who Introduce Issue and when
- Roll-back to previous working stat

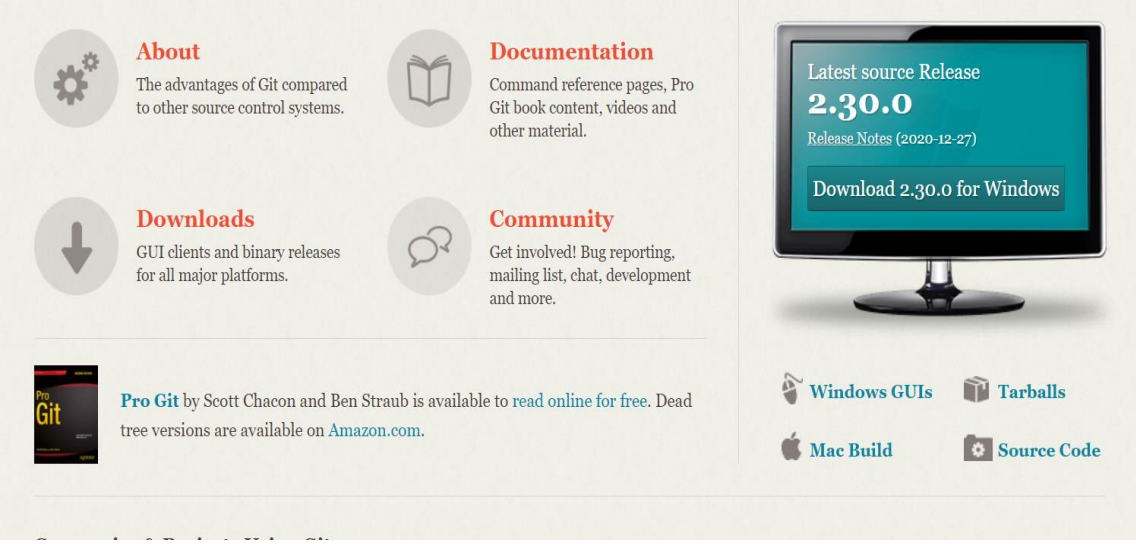
Features:

- Snapshot capture not difference
- Almost every operation is local
- Git has integrity {Show the checksum via tool to check the actual data}
- Git generally only add data

Installation of GIT

- Installation on Windows

To install on windows, we can use the git official website
https://git-scm.com/



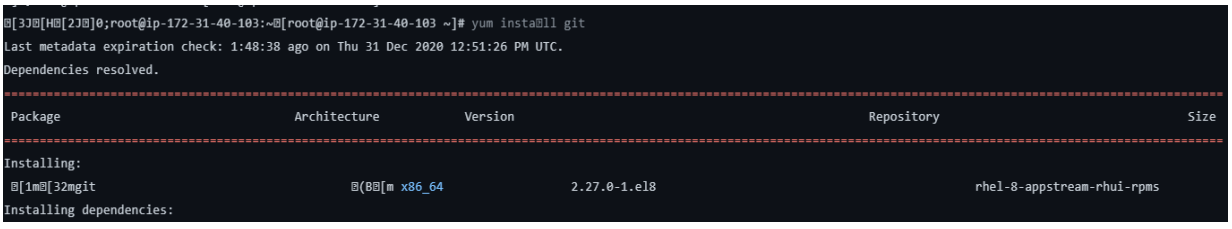
After .exe download, we can follow the normal app installation file process on windows. After installation we can get two things

i. GIT Command Line Tool
ii. GIT Bash

Installation on Linux:

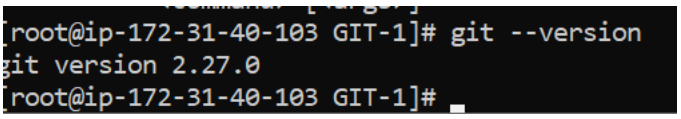
To install we can use below command for linux..

#yum install git



To check Git is installed or not we can just check the Git version by using below command.

#git –version



Git Command: -

To make a directory as a working directory of git

#git init

```
root@ip-172-31-40-103 GitComand]# git init
nitialized empty Git repository in /root/GitComand/.git/
root@ip-172-31-40-103 GitComand]#
```

To Configure the Git

#git config --global user.name “ShailAdmin”

#git config –global user.email example@example.com

```
[root@ip-172-31-40-103 GitComand]# git config --global user.name "ShailAdmin"
[root@ip-172-31-40-103 GitComand]# git config --global user.email "example@example.com"
```

To check the git status

#git status

```
[root@ip-172-31-40-103 GitComand]# git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

To add file into staging area

#git add filename {For single file}

#git add . { Add all available file in one go

#git add –a

```
root@ip-172-31-40-103 GitComand]# touch 1 2 3 4 5
root@ip-172-31-40-103 GitComand]# git status
n branch master

o commits yet

ntracked files:
 (use "git add <file>..." to include in what will be committed)
       1
       2
       3
       4
       5

othing added to commit but untracked files present (use "git add" to track)
root@ip-172-31-40-103 GitComand]# git add 1
root@ip-172-31-40-103 GitComand]# git status
n branch master

o commits yet

hanges to be committed:
 (use "git rm --cached <file>..." to unstage)
       new file:   1

ntracked files:
 (use "git add <file>..." to include in what will be committed)
       2
       3
       4
       5

root@ip-172-31-40-103 GitComand]# git add --a
root@ip-172-31-40-103 GitComand]# git status
n branch master

o commits yet

hanges to be committed:
 (use "git rm --cached <file>..." to unstage)
       new file:   1
       new file:   2
       new file:   3
       new file:   4
       new file:   5

root@ip-172-31-40-103 GitComand]#
```

To git commit :

#git commit -m “Commit message should be here”

```
[root@ip-172-31-40-103 GitComand]# git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   1
        new file:   2
        new file:   3
        new file:   4
        new file:   5

[root@ip-172-31-40-103 GitComand]# git commit -m "This is The First Commit Command"
[master (root-commit) d734e1a] This is The First Commit Command
 5 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1
 create mode 100644 2
 create mode 100644 3
 create mode 100644 4
 create mode 100644 5
[root@ip-172-31-40-103 GitComand]# git status
On branch master
nothing to commit, working tree clean
[root@ip-172-31-40-103 GitComand]#
```

Git Logs:-

To check the commit, we can check the git logs.

#git log

```
[root@ip-172-31-40-103 GitComand]# git log
commit d734e1af156ea9fb290ef055f61b83029c76799a (HEAD -> master)
Author: ShailAdmin <example@example.com>
Date:   Sat Jan 2 16:29:41 2021 +0000

    This is The First Commit Command
[root@ip-172-31-40-103 GitComand]#
```

To check the change in details we can use below command

#git log -p

```
root@ip-172-31-40-103 GitComand]# git log -p
commit d734e1af156ea9fb290ef055f61b83029c76799a (HEAD -> master)
Author: ShailAdmin <example@example.com>
Date:   Sat Jan 2 16:29:41 2021 +0000

    This is The First Commit Command

diff --git a/1 b/1
new file mode 100644
index 0000000..e69de29
diff --git a/2 b/2
new file mode 100644
index 0000000..e69de29
diff --git a/3 b/3
new file mode 100644
index 0000000..e69de29
diff --git a/4 b/4
new file mode 100644
index 0000000..e69de29
diff --git a/5 b/5
new file mode 100644
index 0000000..e69de29
root@ip-172-31-40-103 GitComand]#
```

#git log --stat

#git log --pretty=online/short/full

#git log --since=2.day/month/week

#git log --pretty=formate “%h -- %an”

To Download the entire gir repository

#git clone repo-url

```
[root@ip-172-31-40-103 GitComand]# cd
[root@ip-172-31-40-103 ~]# git clone https://github.com/onebirdrocks/geektime-ELK.git
Cloning into 'geektime-ELK'...
remote: Enumerating objects: 898, done.
remote: Total 898 (delta 0), reused 0 (delta 0), pack-reused 898
Receiving objects: 100% (898/898), 155.34 MiB | 18.08 MiB/s, done.
Resolving deltas: 100% (280/280), done.
Updating files: 100% (218/218), done.
[root@ip-172-31-40-103 ~]#
```

To ignore any file or directory

I. Create .gitignore file
II. And add the name of file or directory which we want to ignore in our project

To check the difference between staging area and last commit

#git diff

#git diff –staged

To delete file or directory

#git rm -rf filename/DirectoryName

To rename file or Directory

#git mv filename/DirectoryName

To make file or directory untracked

#git rm –cached filename/DirectoryName

Git Remote Repository..

To manage the remote repository we can use multiple hosting source like github, bitbuket etc

GitHub__ https://github.com/

BitBucket__https://bitbucket.org/product/

To Add git remote repository to terminal

#git remote add origin(we can give any Name) repo_url

```
[root@ip-172-31-40-103 GitComand]# git remote add origin1 https://github.com/ShailAdmin/Jenkins.git
[root@ip-172-31-40-103 GitComand]# git remote
origin1
[root@ip-172-31-40-103 GitComand]# git remote -v
origin1 https://github.com/ShailAdmin/Jenkins.git (fetch)
origin1 https://github.com/ShailAdmin/Jenkins.git (push)
[root@ip-172-31-40-103 GitComand]#
```

To push date to global git repository.

#git push origin master

```
[root@ip-172-31-40-103 GitComand]# git remote
origin1
[root@ip-172-31-40-103 GitComand]# git push origin1 master
Username for 'https://github.com': ShailAdmin
Password for 'https://ShailAdmin@github.com':
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (8/8), 641 bytes | 641.00 KiB/s, done.
Total 8 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/ShailAdmin/Jenkins/pull/new/master
remote:
To https://github.com/ShailAdmin/Jenkins.git
 * [new branch]      master -> master
[root@ip-172-31-40-103 GitComand]#
```

To Create new branch

#git checkout -b branchName

```
[root@ip-172-31-40-103 GitComand]# git checkout -b TestBranch
Switched to a new branch 'TestBranch'
[root@ip-172-31-40-103 GitComand]#
```

To check available branch

```
[root@ip-172-31-40-103 GitComand]# git branch
* TestBranch
  master
[root@ip-172-31-40-103 GitComand]#
```

To switch to branch

#git checkout branchName

```
[root@ip-172-31-40-103 GitComand]# git checkout master
D       1
Switched to branch 'master'
[root@ip-172-31-40-103 GitComand]#
```

To merge branch

#git merge BranchName

```
[root@ip-172-31-40-103 GitComand]# git merge TestBranch
Already up to date.
[root@ip-172-31-40-103 GitComand]#
```

To push local branch to remote repository

#git push origin BranchName

```
[root@ip-172-31-40-103 GitComand]# git branch
  TestBranch
* master
[root@ip-172-31-40-103 GitComand]# git push origin1 TestBranch
Username for 'https://github.com': ShailAdmin
Password for 'https://ShailAdmin@github.com':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'TestBranch' on GitHub by visiting:
remote:      https://github.com/ShailAdmin/Jenkins/pull/new/TestBranch
remote:
To https://github.com/ShailAdmin/Jenkins.git
 * [new branch]      TestBranch -> TestBranch
```

To delete branch

#git -d BranchName

#git -D BranchName

To Delete branch from remote repository

#git -d origin BranchName

For output of test command on linux terminal follow the below git repository

Repo Name : GitDocs
URL :
For any other Help follow the official Git website

Git Cheat Sheet url :