Introduction to Git

By Mulia Nasution (mul14@refactory.id)

Git is Version **C**ontrol **S**ystem

Git is not the only one VCS in our world

There are several popular VCS

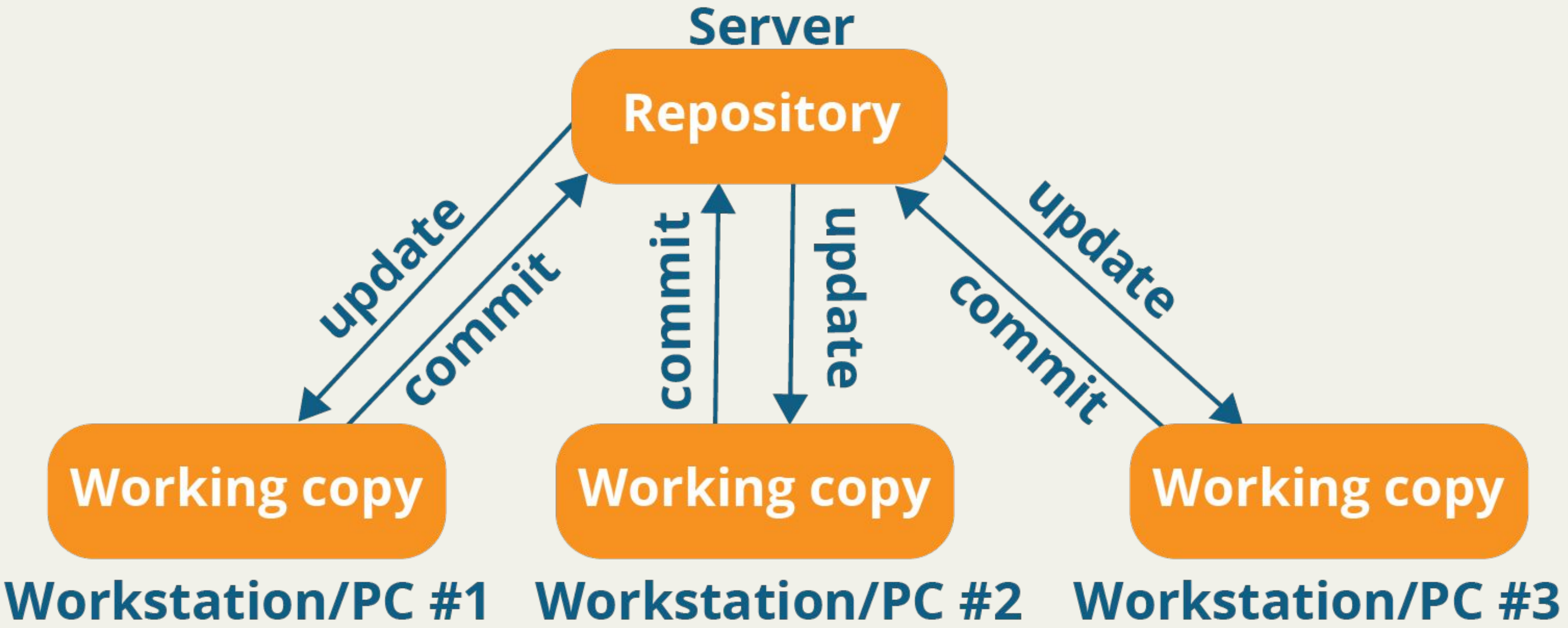
Centralized VCS

- Subversion (svn)
- CVS (cvs)
- Perforce

Distributed VCS

- Git (git)
- Mercurial (hg)
- Bazaar (bzr)

Centralized version control system



Server

Repository

update

commit

commit

update

commit

update

Working copy

Working copy

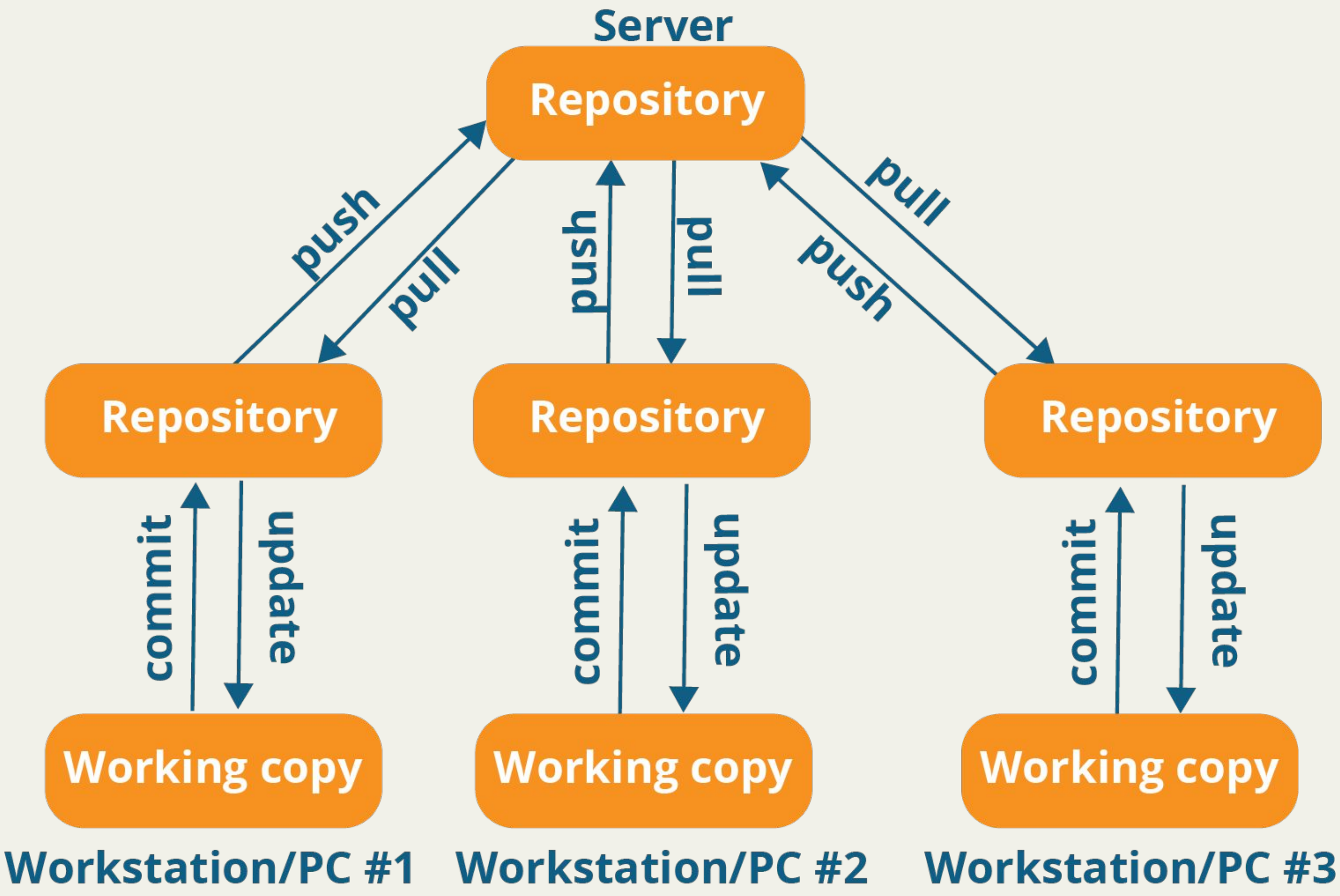
Working copy

Workstation/PC #1

Workstation/PC #2

Workstation/PC #3

Distributed version control system



Why we should use VCS (or git)?

To avoid something like this

skripsi.docx

skripsi-revisi.docx

skripsi-revisi-lagi.docx

skripsi-final.docx

skripsi-super-ultra-final-blaster.docx

Another reason to use VCS (or git)?

- Track the history of the app
- Rollback to specific version
- Blame someone
- Collaborations
- Easy to deploy applications
- Safer than without VCS

Git Most Common Commands (page 1)

git **init**	Initialize git repository
git **clone**	Clone a existing repository
git **status**	Check the status of git repository
git **add**	Add changed files in working folder to staging area
git **commit** -m "message"	Record the changes
git **log**	View the commit histories
git **pull**	Get the changes from remote repository
git **push**	Push the changes to remote repository
git **remote**	Configure the remote repository

Git Most Common Commands (page 2)

git **diff**	View the differences before and after changes
git **checkout**	Switch to another branch or restore a file (check switch and restore)
git **switch**	Switch to another branch (since git 2.23)
git **restore**	Restore files (since git 2.23)
git **reset**	Reset files
git **merge**	Merge a branch to other branch
git **branch**	Use branch in git
git **config**	Configure git
git **stash**	Stash the changes

Confusing?

Check out Git GUI

https://git-scm.com/downloads/guis

But, we **highly recommend** you use Git from the **command line** and not use a GUI interface.

Demo

Q & A

Exercise

Before we do some exercises, let's make sure you already have SSH key

Exercise: Basic Flow

1. Clone repository from remote repository
2. Initialize a project
3. Create .gitignore file
4. Add all files into staging
5. Commit with meaningful message
6. Push to remote repository

Exercise 2: Use Branch

1. Create new branch from master branch
2. Make some changes
3. Commit with meaningful message
4. Push
5. Create a merge request (or pull request)
6. Merge

Exercise 3: Handle Some Conflicts

1. Create new branch from master branch
2. Make some changes
3. Commit with meaningful message
4. Push
5. Create a merge request (or pull request)
6. Another person will do the same thing (repeat step 1 to 5)
7. Merge, and handle the conflicts

Thanks