SM Lab 12: Setting Up data repository and version control using github

Objective: The aim of this lab is to learn how to set up a data repository and version control system in Ubuntu using github.

A Data Repository is a logical (and sometimes physical) partitioning of data where multiple databases which apply to specific applications or sets of applications reside. Version control applies to any any kind of practice that tracks and provides control over changes to source code. Software developers sometimes use revision control software to maintain documentation and configuration files as well as source code. In this lab, we will:

a) Install and create a github account
b) Authenticate with GitHub by connecting using SSH keys
c) Create a repository
d) Fork a repository

A. Install and create a github account

It is easy to install using apt-get, type

$ sudo apt-get update
$ sudo apt-get install git

If there are some missing packages or have unfulfilled dependencies for installing git, typing sudo apt-get update will take care of it.

Once you have installed git, sign up on git from the sign-up page - https://github.com/
Pick a username. Pick your preferred email and a password. Check your email for github account confirmation.
open your command line again to label your commits and to associate your email address with your git-commits. Type:

$ git config --global user.name “Your Name”   # this will be the username you used for signingup
$ git config --global user.email “Your email address” #this will be the email you used for signingup

B. Authenticating github using SSH keys – this will be similar to cloning your working directory using SSH keys i.e. provide access to a Git repository via SSH, which is a secure protocol. You must have an SSH keypair generated on your computer, and attached to your GitHub account.

First, we need to check for existing SSH keys on your computer, type

$ ls -al ~/.ssh

Check the directory listing to see if you already have a public SSH key. By default, the filenames of the public keys are one of the following:

- *id_dsa.pub*
- *id_ecdsa.pub*
- *id_ed25519.pub*
- *id_rsa.pub*

If you receive an error that ~/.ssh does not exist, then we will create a ssh directory and generate a pair of SSH keys.

To generate a public/private rsa key pair using the provided email as a label

$ ssh-keygen -t rsa -b 4096 -C "*your_email address”*

You will be prompted to enter a file in which to save the key, just press **Enter** to continue.

```
Enter file in which to save the key (/Users/you/.ssh/id_rsa): [Press enter]
```

You will be asked to enter a passphrase:

```
Enter passphrase (empty for no passphrase): [Type a passphrase]
# Enter same passphrase again: [Type passphrase again]
```

After you enter a passphrase, you'll be given the fingerprint, or *id*, of your SSH key. It will look something like this:

```
Your identification has been saved in /Users/you/.ssh/id_rsa.
# Your public key has been saved in /Users/you/.ssh/id_rsa.pub.
# The key fingerprint is:
# 01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db your_email address
```

Now you need to add the key generated to the ssh-agent

Ensure ssh-agent is enabled before adding the ssh key
To enable ssh agent /start the ssh agent in the background
$ eval “$(ssh-agent -s)”

It will print the Agent ID on the terminal

Now, to add SSH key to ssh-agent
$ ssh-add ~/.ssh/id_rsa

Note: If you didn't generate a new SSH key and used an existing SSH key instead, you will need to replace *id_rsa* in the above command with the name of your existing private key file.

Now, we need to add SSH key to your github account and test the connection, for this you need to copy paste the public key.
$ gedit ~/.ssh/id_rsa.pub
Copy the contents on clipboard (right click after selecting the key) with all spaces and white markers.
Now, **here are steps to add the key in your github account:**

1. In the top right corner of your github page, click your profile photo, then click **Settings**.
2. In the user settings sidebar, click **SSH keys**
3. Click **Add SSH key**
4. In the Title field, add a descriptive label for the new key. For example, if you're using a personal Lenovo Yoga 13 series, you might call this key "Personal Lenovo Yoga 13 series"
5. Paste your key into the "Key" field
6. Click **Add key**
7. Confirm the action by entering your GitHub password

Now to attempt to SSH into github server type

1. type, $ssh -T git@github.com
2. You may see this warning,
   The authenticity of host 'github.com (207.97.227.239)' can't be established.
   ```
   # RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
   # Are you sure you want to continue connecting (yes/no)?
   ```

3. Verify the fingerprint in the message you see matches the following message, then type yes

4. You might get something like
   ```
   Hi username! You've successfully authenticated, but GitHub does not
   # provide shell access.
   ```

   If your username on github matches with the username in the above message, then you have successfully set-up your SSH keys.

C. Create a repository
In this step, we will create our first project, add a README file, make changes to the README file, preview changes and commit these changes by making a new branch. Kindly follow step by step instructions given here:
https://help.github.com/articles/create-a-repo/

D. Forking a repository – One would fork a repository for two reasons:
1) To get a starting point for a working idea
2) To provide bug fixes/editions to an existing project

Link below provides step by step process to first fork a project into local repository, then how to clone a repository into local workspace, how to pull requests to commit changes in the original repository
https://help.github.com/articles/fork-a-repo/

The important commands to understand are git clone, git remote -v, git remote add upstream

To understand how to create pull requests in the original repository, follow the step by process from the link below:
https://help.github.com/articles/using-pull-requests/

Now you can try working in teams for projects that need version control and experience the ease of using github!