Proofpoint Git Access

Recommended Help Links

URL	Description
Show GIT repos	*Note: This is not real time information*
FAQ for admin	
FAQ for users	
Git	Engineering resource for setting up and using git
Git vs SVN	Good links to get yourself familiar with the concepts and benefits of using Git
http://cheat.errtheblog.com/s/git	Cheat sheet 1 (Favorite)
http://help.github.com/git-cheat-sheets/	Cheat sheet 2
https://git.wiki.kernel.org/index.php/GitCheatSheet	Cheat sheet 3
http://help.github.com/ignore-files/	Help on .gitignore file

Getting Started

What is it?

Git is a distributed revision and source code management system with a primary emphasis on speed versus other version control systems. Git was initially designed and developed by Linus Torvalds for Linux kernel development. Every Git working directory is a full-fledged repository with complete history and full revision tracking capabilities, not dependent on network access or a central server. Git is free software distributed under the terms of the GNU General Public License version 2.

Getting access to Proofpoint Repositories

Generate RSA key

These are the commands for generating your public/private key. If you are on Windows platform, see the GitHub instructions which walks you through some pre-setup steps.

Create a new ssh key with your proofpoint email account.

```
$ ssh-keygen -t rsa -C "your_email@proofpoint.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/your_user_directory/.ssh/id_rsa):<press enter>
```

Enter a passphrase for extra security.

```
Enter passphrase (empty for no passphrase:<enter a passphrase>
Enter same passphrase again:<enter passphrase again>
```

You should end up with two files created in .ssh directory:

1. id_rsa
2. id_rsa.pub

Email public key to x-gitosis-admin@proofpoint.com

Send your public key to x-gitosis-admin@proofpoint.com. You should also include the following information:

1. The organization or group that you belong to
2. The projects that you need access to
3. Type of access you need to a project. This may be applicable if read only access is required (e.g. for an external contractor)

You should **never** share your private key to anyone. This is private and only you as a user should have this key. It is recommended that you keep your private key safe and backed-up. If you lose your key, then you will need to regenerate a new one and send it to the Proofpoint gitosis admins again.

Set Up Your Info

Git tracks who makes each commit by checking the user's name and email. In addition, we use this info to associate your commits with your GitHub account. To set these, enter the code below, replacing the name and email with your own. The name should be your actual name, not your GitHub username.

```
$ git config --global user.name "Firstname Lastname"
$ git config --global user.email "your_email@proofpoint.com"
```

Checking out a Proofpoint Project

Once your key has been uploaded and access granted to your projects by the Proofpoint gitosis admins, then you are ready to checkout your project using the clone command.

```
$ git clone git@git.us.proofpoint.com:<YOUR_PROJECT>.git
```

If you are prompted to enter in the password for the git user, then something has gone wrong. Send an email to x-gitosis-admin@proofpoint.com. Otherwise, it should have checked-out your project under the same name.

Setting up a new Repo

Send a message to x-gitosis-admin@proofpoint.com for your new repo

You should send a message to x-gitosis-admin@proofpoint.com. Similar information as above should be sent:

1. name of the repo (e.g. pps-gui-tests)
2. Which group should have access to this?
3. What is the general purpose of this repo (one may already exist under a different name)

Add post push notification

- inform x-gitosis-admin@proofpoint.com, so the notification hook will be put in place.
- add recipients by:
    - git clone git@git:git-commit-hooks.git
    - add recipients to config/mailing-list in the repo

- commit the change and do git push
- wait until next 5 minute interval of the hour to get your change kicked in

Initialize your Repo

Your repo **must** have a README.txt file. Other than that, you should follow standard git initialization commands:

```
$ cd $MY_NEW_GIT_PROJECT
$ git init
$ git add README.txt
$ git add .gitignore
$ git add <any other files and directories>
$ git commit -m "Initial commit"
```

Note: You can skip these steps if you already have a local repo that you want to push to a remote repository

.gitignore file
The .gitignore file allows you to specify files that no user should checkin to this repository. Typical exclusions include your IDE files or temporary files that the application creates during runtime in the project directories. You can find more information about the .gitignore file on GitHub.

Attached is a typical .gitignore file we use for the Platform WebServices. If you download this file, then make sure to rename it from "gitignore" to ".gitignore" and commit this to your repository.

Add remote tracking to your project

Execute the following replacing <*PROJECT_NAME*> with the name that you had requested from the Proofpoint gitosis admins:

```
$ git remote add origin git@git.us.proofpoint.com:<PROJECT_NAME>.git
```

Push your changes to remote server

```
$ git push -u origin master
```

Reset your local repo (Optional)

This step should not be necessary if you have correctly followed the instructions above. However, in the event that something isn't working correctly (e.g. unable to push/pull to/from the remote), I have found it easier to reset my links to the remote repository by deleting my local repo and re-cloning it:

```
$ cd ..
$ rm -rf <PROJECT_NAME>
$ git clone git@git.us.proofpoint.com:<PROJECT_NAME>.git
```