在线实验，请到PC端体验

Linux系统安装git服务器

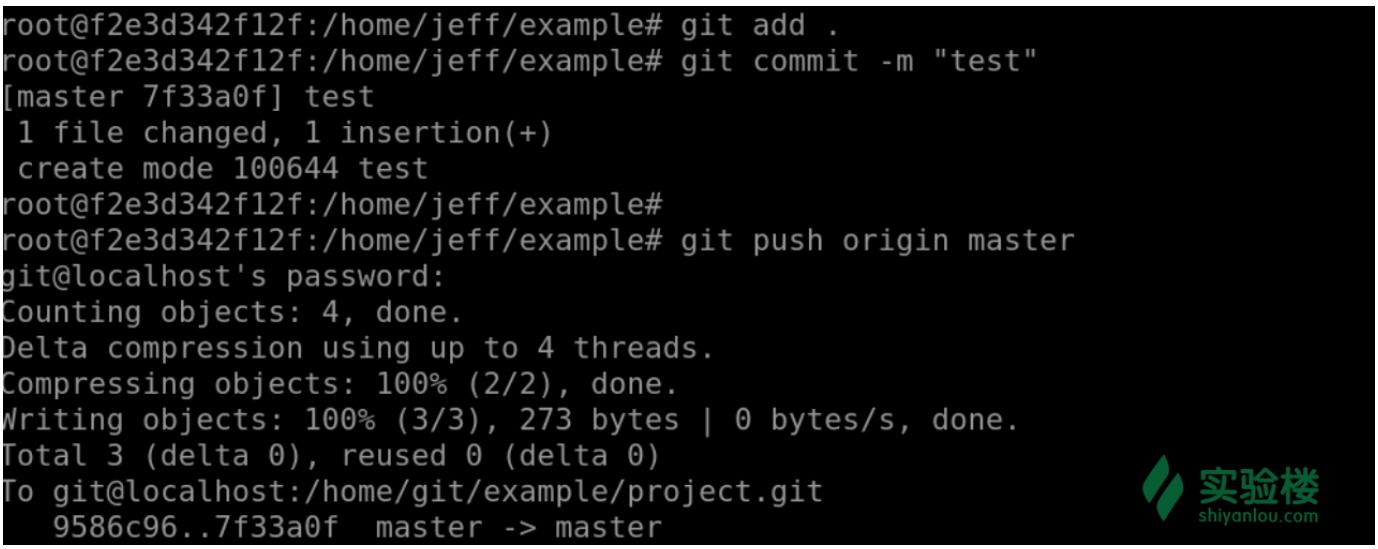
一、实验简介

Git 是一个开源的分布式版本控制软件。自诞生以来，Git 就以其开源、简单、快捷、分布式、高效等特点，应付了类似 Linux 内核源代码等各种复杂的项目开发需求。如今，Git 已经非常成熟，被广泛接受与使用，越来越多的项目都迁移到 Git 仓库中进行管理。学会使用Git,对于代码的版本控制，管理，无疑有很大的帮助。通过学习本课程，实现自己搭建一台 Git 服务器，虽然有很多现成的代码托管网站比如 Github，但是当你想保护代码的安全的时候，私有的 Git 服务器无疑是最安全的。

1.1 知识点

- 了解常见的代码托管软件
- 了解 Git 的一些常用的操作
- 安装配置 Git 服务器，提交自己写的代码

1.2 效果截图



1.3 常见的版本控制软件介绍

VSS（Visual Source Safe)：为微软的产品，较为易学易用，使用微软的产品进行开发通常都用这个，但是 VSS 只能在 windows 平台上运行，并且不是免费软件，安全性也不高，因此应用并不太广泛。

CVS（Concurrent Version System)：是免费开源的配置管理工具，其源代码和安装文件都可以免费下载。

由于其简单易用、功能强大，跨平台，支持并发版本控制，而且免费，它在全球中小型软件企业中得到了广泛使用。不过 CVS 的部署比VSS要复杂一些。

SVN（Subversion)：作为 CVS 的重写版和改进版，其目标就是作为一个更好的版本控制软件，取代 CVS 。总的来说，CVS 在发展的过程中逐渐失去优势，已经不再适合现代开发，目前，绝大多数 CVS 服务已经改用 SVN。

Git：Git是一款免费、开源的分布式版本控制系统，用于敏捷高效地处理任何或小或大的项目。与 CVS、Subversion 一类的集中式版本控制工具不同，它采用了分布式版本库的作法，不需要服务器端软件，就可以运作版本控制，使得源代码的发布和交流极其方便。Git 的速度很快，这对于诸如 Linux 内核这样的大项目来说自然很重要。

> 分布式和集中式：简单地说，“分布式”就是每一个客户端都有数据的副本，查询等的数据操作都使用副本进行；并定期或不定期的与数据交换中心进行交换，以获得最新的数据；“集中式”是指整个整个系统中只使用一份数据（只在服务器上），所有客户端必须连接上服务器才能进行数据查询等操作。

二、安装 Git

在本次实验里我们的虚拟机既做本地端，又做远程端，所以安装 Git 只需要一次。学完了本实验后，你可以尝试在自己的电脑里搭建 Git 服务器，实现局域网内的代码托管。

```
$ sudo apt-get update
$ sudo apt-get install git  #安装Git
```

2.1 添加 git 用户

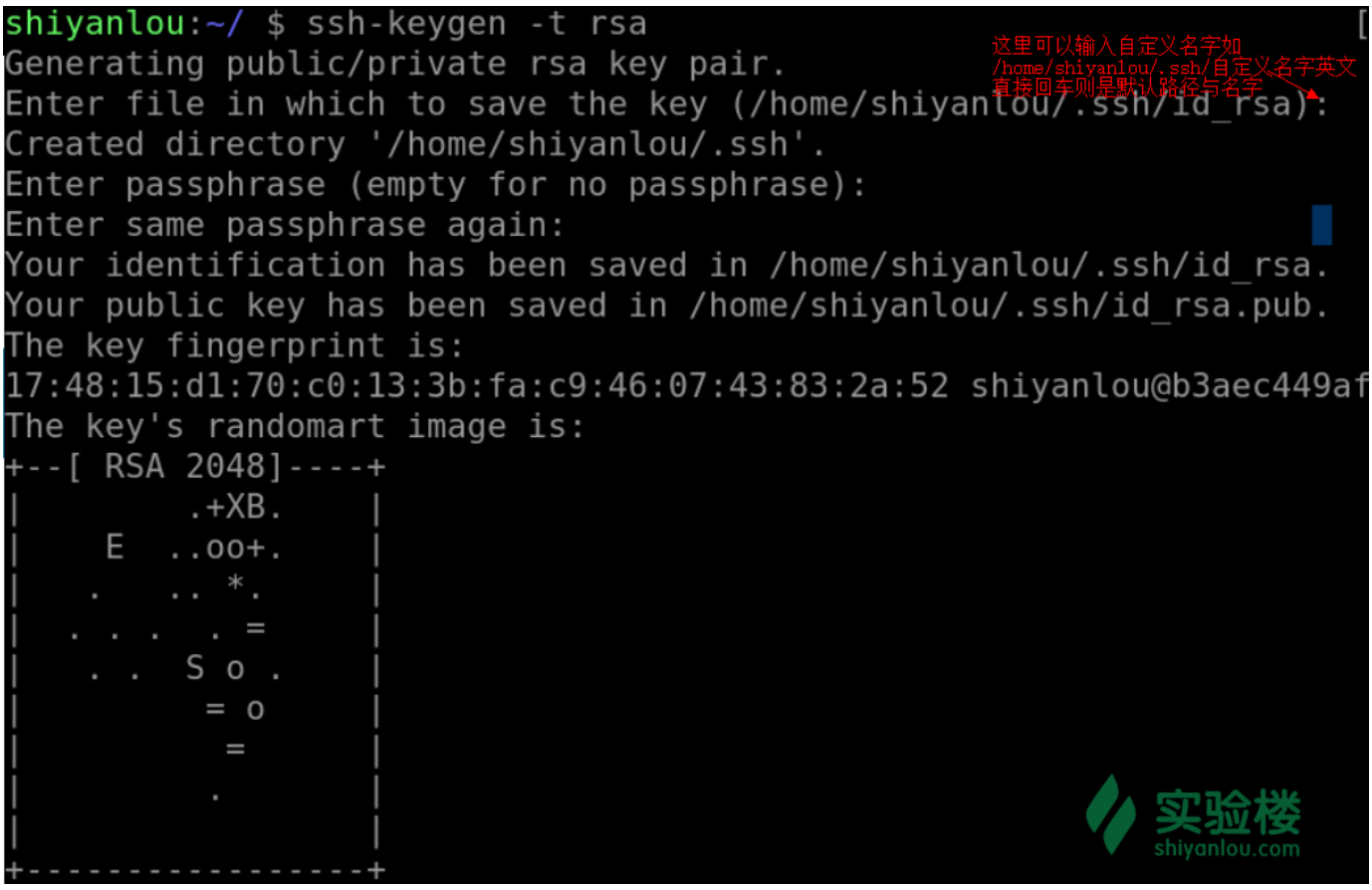
为了方便管理和操作，我们添加一个用户 git ,并给他设置工作目录：

```
$ sudo useradd git
$ sudo passwd  git  # 设置密码,为了方便操作，密码较为的简单
$ sudo mkdir /home/git   # 设置 git 用户的工作目录
$ sudo chown -R  git  /home/git  #将工作目录的权限给git用户
```

2.2 在本地端生成密钥

在管理 Git 项目上，有 HTTPS 和 SSH 两种方式，使用 HTTPS 每次提交都要输入用户名和密码，使用 SSH 需要先配置和添加好 SSH key，我们这次实验选择在本地生成 SSH key，并将公钥发送给远程端。

```
$ ssh-keygen -t rsa #生成密钥，默认将放在家目录的 .ssh 文件夹中，并且不修改名字的话，默认密钥对名为 id_rsa
$ cat ~/.ssh/id_rsa.pub | ssh git@localhost "mkdir -p ~/.ssh && cat >>  ~/.ssh/authorized_keys"   # 将公钥发送给远程端
```



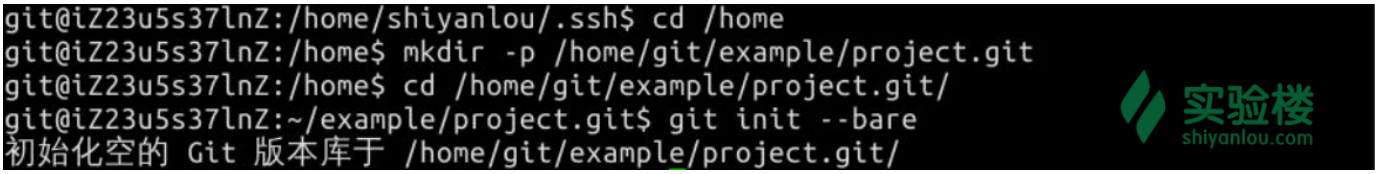
2.3 在远程端建立仓库

由于在我们的实验里，本地端和远程端是一体的，所以对远程端的操作可以直接执行，无需

使用 ssh 到远程端去操作，现在让我们来设置 Git 远程端的配置吧。

```
$ su git    # 切换用户到git,在本实验中用户权限的使用很重要，之后会讲解
$ mkdir -p /home/git/example/project.git  #创建仓库文件夹，-p 是指建立上层目录。
$ cd /home/git/example/project.git
$ git init  --bare  #初始化仓库
```

好的，到此为止，远程端就配置好了。



三、在本地端建立仓库

在远程端配置好了以后，就需要在本地端完成仓库的创建和配置，这一步操作和代码托管网站的操作一致，

没有学过 Git 命令的可以看看，用过的可以再练习一遍。

3.1 初始化仓库

先配置个人的用户名和电子邮件地址：

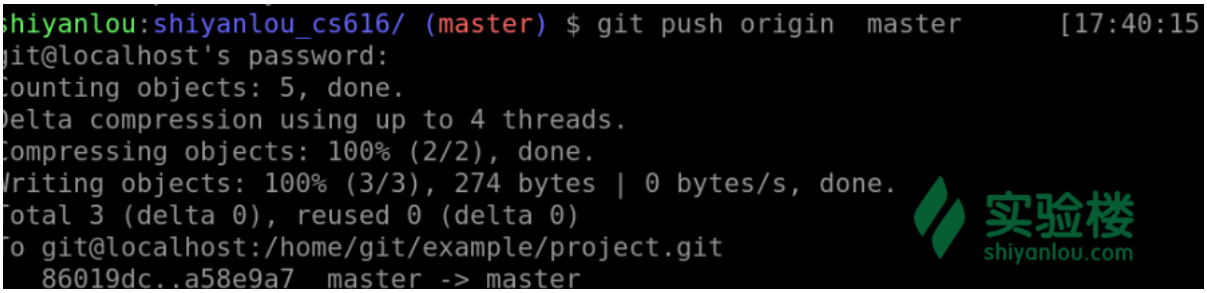
```
$ exit #退出 git 这个用户
$ mkdir -p /home/shiyanlou/example/shiyanlou_cs616
$ cd /home/shiyanlou/example/shiyanlou_cs616
$ git init  #初始化  git  仓库
$ git config --global user.name "shiyanlou"      # 设置用户名
$ git config --global user.email  shiyanlou.localhost     # 设置用户的邮件
$ vim readme      #创建一个文档来测试提交的效果
```

3.2 提交自己的代码

在初始化之后，我们就开始尝试提交自己的代码了，在3.1小节我们写了一个 readme ，现在来提交它。

```
$ git add .
$ git commit -m "test"
$ git remote add origin git@localhost:/home/git/example/project.git
$ git push origin master
```

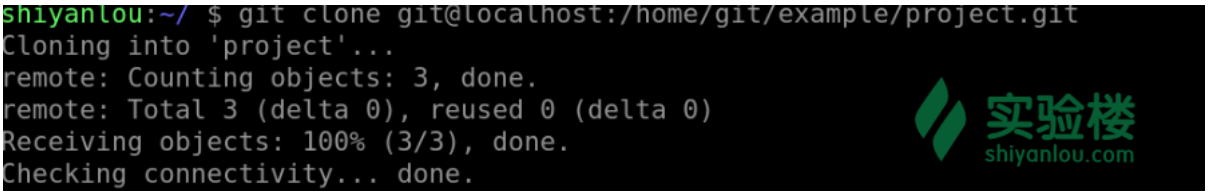
可以看到代码成功地提交了，到此为止一个简单适用的 Git 服务器搭建就完成了。



3.3 检验效果

我们把刚才提交的代码克隆下来，看看效果如何，远程仓库是不是托管了代码。

```
$ cd /home
$ git clone git@localhost:/home/git/example/project.git
```



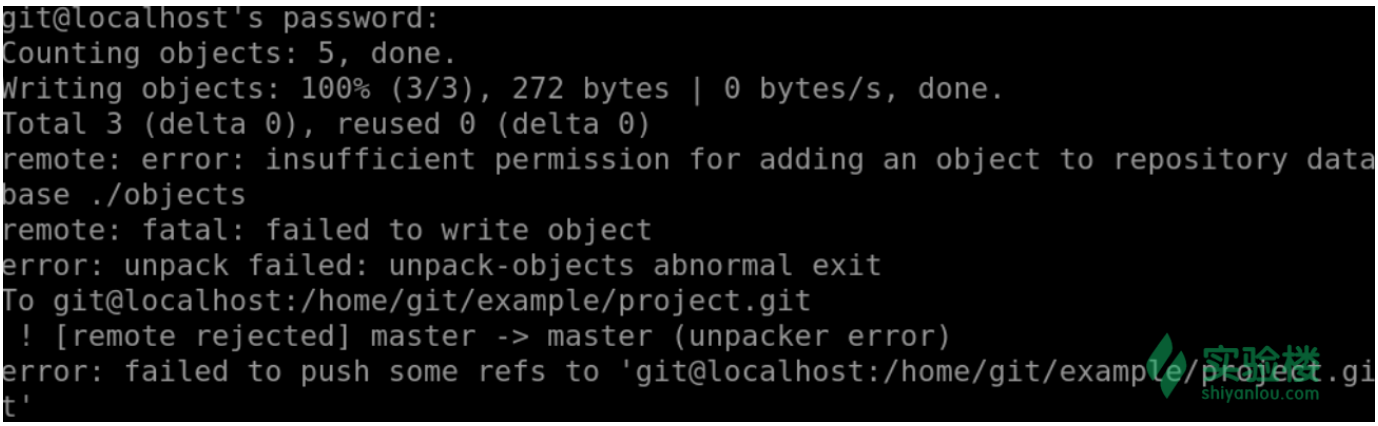
可以看到，效果很好，文件成功的下载了。

四、遇到权限错误怎么办?

现在我们把 `/home/git` 的权限改为其他用户，我们来看看会怎么样

```
$ sudo chown root:root /home/git
```

那就会在 `git push` 的时候出现了如图的问题：



之所以会这样，是因为 Git 服务是由 git 这个用户来操作的，现在改变了 `/home/git` 的所有权， `git` 这个用户无法操作里面的文件，所以报错了，解决这个问题的办法就是将

它的权限交给 `git` 这个用户，

```
$ sudo chown -R git:root /home/git
```

现在再提交就可以了。

五、实验总结

从这个实验我们了解到了 Git 服务器的搭建和配置，以及一些常见的 Git 命令，需要注意的是用户权限的问题。

如果你想试试添加界面，就像 Github，可以试试GitLab (https://about.gitlab.com/downloads/#ubuntu1404),

GitLab 是一个用于仓库管理系统的开源项目,基于 Ruby on Rails。

参考资料

- https://www.linux.com/learn/how-run-your-own-git-server (https://www.linux.com/learn/how-run-your-own-git-server)
- https://about.gitlab.com/downloads/#ubuntu1404 (https://about.gitlab.com/downloads/#ubuntu1404)
- https://zh.wikipedia.org/wiki/Git (https://zh.wikipedia.org/wiki/Git)

课程教师

jefflee

共发布过**14**门课程

查看老师的所有课程 > (/teacher/59274)