在远程仓库一节中，我们讲了远程仓库实际上和本地仓库没啥不同，纯粹为了 7x24 小时开机并交换大家的修改。

GitHub 就是一个免费托管开源代码的远程仓库。但是对于某些视源代码如生命的商业公司来说，既不想公开源代码，又舍不得给 GitHub 交保护费，那就只能自己搭建一台 Git 服务器作为私有仓库使用。

搭建 Git 服务器需要准备一台运行 Linux 的机器，强烈推荐用 Ubuntu 或 Debian，这样，通过几条简单的 `apt` 命令就可以完成安装。

假设你已经有 `sudo` 权限的用户账号，下面，正式开始安装。

第一步，安装 `git`：

```
$ sudo apt-get install git
```

第二步，创建一个 `git` 用户，用来运行 `git` 服务：

```
$ sudo adduser git
```

第三步，创建证书登录：

收集所有需要登录的用户的公钥，就是他们自己的 `id_rsa.pub` 文件，把所有公钥导入到 `/home/git/.ssh/authorized_keys` 文件里，一行一个。

第四步，初始化 Git 仓库：

先选定一个目录作为 Git 仓库，假定是 `/srv/sample.git`，在 `/srv` 目录下输入命令：

```
$ sudo git init --bare sample.git
```

Git 就会创建一个裸仓库，裸仓库没有工作区，因为服务器上的 Git 仓库纯粹是为了共享，所以不让用户直接登录到服务器上去改工作区，并且服务器上的 Git 仓库通常都以 `.git` 结尾。然后，把 owner 改为 `git`：

```
$ sudo chown -R git:git sample.git
```

第五步，禁用 shell 登录：

出于安全考虑，第二步创建的 git 用户不允许登录 shell，这可以通过编辑 `/etc/passwd` 文件完成。找到类似下面的一行：

```
git:x:1001:1001:,,,:/home/git:/bin/bash
```

改为：

```
git:x:1001:1001:,,,:/home/git:/usr/bin/git-shell
```

这样，`git`用户可以正常通过 ssh 使用 git，但无法登录 shell，因为我们为`git`用户指定的`git-shell`每次一登录就自动退出。

第六步，克隆远程仓库：

现在，可以通过`git clone`命令克隆远程仓库了，在各自的电脑上运行：

```
$ git clone git@server:/srv/sample.git

Cloning into 'sample'...

warning: You appear to have cloned an empty repository.
```

剩下的推送就简单了。

管理公钥

如果团队很小，把每个人的公钥收集起来放到服务器的`/home/git/.ssh/authorized_keys`文件里就是可行的。如果团队有几百号人，就没法这么玩了，这时，可以用 Gitosis 来管理公钥。

这里我们不介绍怎么玩 Gitosis 了，几百号人的团队基本都在 500 强了，相信找个高水平的 Linux 管理员问题不大。

管理权限

有很多不但视源代码如生命，而且视员工为窃贼的公司，会在版本控制系统里设置一套完善的权限控制，每个人是否有读写权限会精确到每个分支甚至每个目录下。因为 Git 是为 Linux 源代码托管而开发的，所以 Git 也继承了开源社区的精神，不支持权限控制。不过，因为 Git 支持钩子（hook），所以，可以在服务器端编写一系列脚本来控制提交等操作，达到权限控制的目的。Gitolite 就是这个工具。

这里我们也不介绍 Gitolite 了，不要把有限的生命浪费到权限斗争中。

小结

- 搭建 Git 服务器非常简单，通常 10 分钟即可完成；
- 要方便管理公钥，用 Gitosis；
- 要像 SVN 那样变态地控制权限，用 Gitolite。