**Gitolite**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**重命名、删除仓库**

rename repository（重命名仓库）：

1、在gitolite配置文件中(conf/gitolite.conf)，修改你仓库的名字。

2、登录服务器（当然前提是你需要管理员权限），修改仓库文件夹的名字。

3、commit，push你的gitolite配置文件。

注意：第二步和第三步的顺序不能对调，因为这样会创建一个空仓库。仓库重命名之后，请在本地修改仓库的地址。

delete repository（删除仓库）：

1、在gitolite配置文件中删除你的仓库。 commit，push你的配置文件。

2、登录服务器，删除仓库所在的文件夹 (例如： /home/git/repositories/projectname.git)。

3、删除那些不再需要的用户和公钥。

**参考：**[**http://www.2eggs.org/archives/466**](http://www.2eggs.org/archives/466)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* demo \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* 实例5 \*/

repo project

...

// 除了根目录的Makefile文件外，new对其他文件都具有写权限

- NAME/Makefile = new

RW Name/ = new

/\* 推荐用法 \*/

repo project

// 有效防止误操作：

// 指定admin对branch的读写权限，防止误操作将本地的临时性branch推送到服务器端

// 如果确定需要新增一个branch，则在下面新增一行，例如新增dev分支

RW+ master$ = admin // admin 有读，写，强制写master分支的权限

RW+ dev$ = admin // admin 有读，写，强制写dev分支的权限

RW dev$ = dev1 dev2 // 普通developer仅能读写dev分支，且不能强制写

- refs/tags/v[0-9] = dev2 dev2 // 限制普通用户不能创建以v加上数字开头的release tag

// 如果不同的用户(组)分别负责完全独立的两个子系统，则可通过类似以下这种方式排除互相干扰

- NAME/net/ = dev1 // dev1 不能修改net子系统下的文件

- NAME/arch/ = dev2 // dev2 不能修改arch子系统下的文件

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Gitlab 安装配置 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

. gitolite 向 gitlab 迁移

第一步,将gitolite复制到gitlab仓库目录下

*# cp -r /gitroot/gitolite/repositories/\* /var/opt/gitlab/git-data/repositories/*

执行导入处理程序

*# gitlab-rake gitlab:import:repos*

上面程序会处理一下目录结构，

进入gitlab web界面，创建仓库与导入的仓库同名，这样就完成了导入工作。

提示

转换最好在git用户下面操作，否则你需要运行

# chown git:git -R /var/opt/gitlab/git-data/repositories