Gerrit使用指南

目录

[1 Gerrit初始设置 3](#_Toc309049306)

[2 Gerrit流程 5](#_Toc309049307)

[3 RD本地操作 6](#_Toc309049308)

[3.1提交到本地的git库中 6](#_Toc309049309)

[3.2提交到gerrit中 6](#_Toc309049310)

[4 登陆Gerrit操作 8](#_Toc309049311)

[4.1手动添加评审人员【可选】 8](#_Toc309049312)

[4.2代码评审 8](#_Toc309049313)

[4.3将代码submit到git服务器 10](#_Toc309049314)

[5 Hudson编译 12](#_Toc309049315)

# 1 Gerrit初始设置

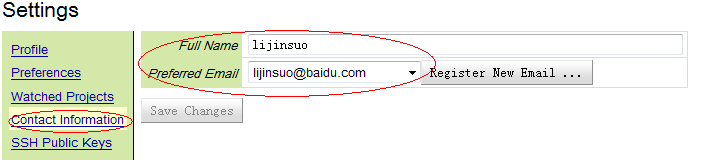
1. 首先登陆gerrit系统，输入git用户名和密码，进入系统





1. 设置邮件地址

点击setting-Contact Information，选择fullname，注册Email地址



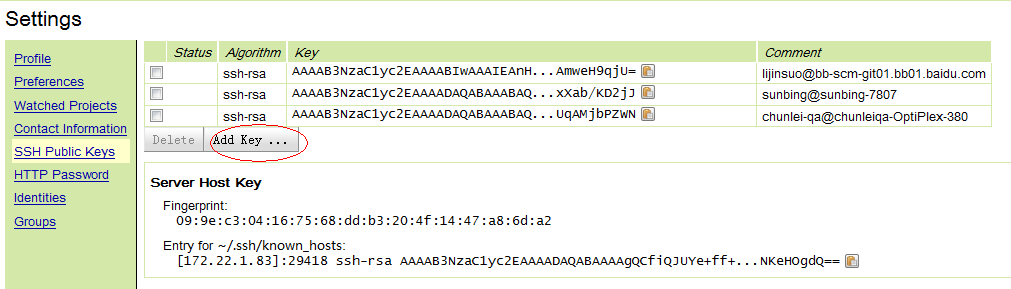
**注意：设置完成后，请hi或者邮件通知SCM，添加提交代码的权限。**

1. 将ssh公钥上传到自己账户设置中的SSH Public Keys中

$ ssh-keygen –t rsa

$cat ~/.ssh/id\_rsa.pub

选择settings-SSH Public Keys，点击Add key,将本机ssh公钥添加。



1. 测试SSH连接

$ ssh –p 8235 账户@git.scm.baidu.com，,

返回以下以下类似信息

Connection to hostname closed

# 2 Gerrit流程



# 3 RD本地操作

## 3.1提交到本地的git库中

1. 创建workspace

例如：

$ repo init --repo-url=http://git.scm.baidu.com:8088/git/repo.git/ -u [http://git.scm.baidu.com:8088/git/manifest.git/](http://172.22.1.83:8088/git/manifest.git/) -b master

1. 创建branch

$ repo start testbranch --all

或者使用git命令

$ git checkout -b --track origin/master

1. 修改文件文件，提交到本地的git库中

$ vi 文件

$ git commit -am “comment”

## 3.2提交到gerrit中

1. 修改jira状态，提交代码
2. 开发人员登陆到jira系统中，手动将本次修改对应的jita记录由inprogress修改为resolved
3. 提交到git服务器，自动生成gerrit记录。

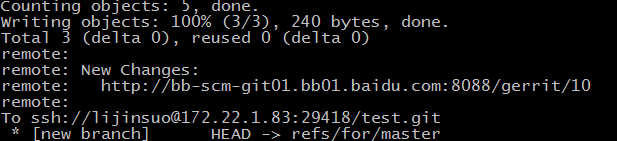
* **修改单个git库**

$ repo upload

或者使用 --re=reviewer的邮件地址，自动将对应人员添加为评审人员，如果要添加多个reviewer，可用“,”分隔。

$ repo upload . [--re=lijinsuo@baidu.com](mailto:--re=lijinsuo@baidu.com),donghe@baidu.com

返回以下类似信息

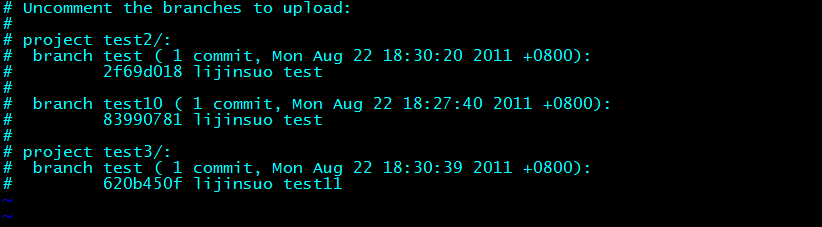


或者使用git命令

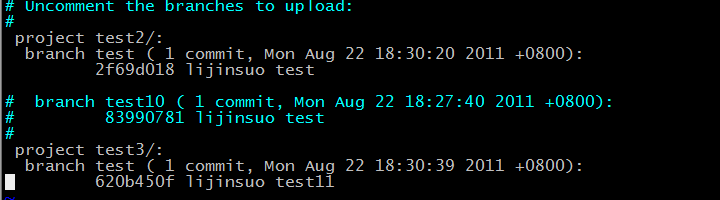
$ git push ssh://gerrit账号@git.scm.baidu.com:8235/projectname HEAD:refs/for/master

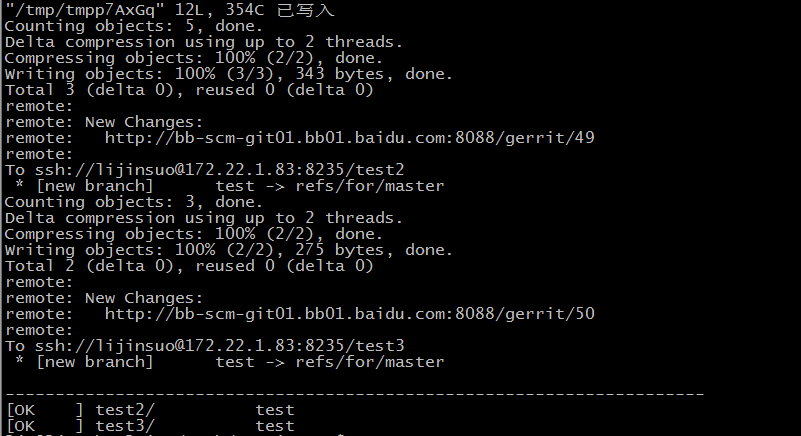
* **修改多个git库**

执行repo upload，弹出交互界面，选择要push的提交。



将要push的提交，将前面的#去掉，然后保存。





* **下载了master的代码，需要提交到millennium-baidu分支上**

首先建立一个本地分支

$ git checkout -b millennium-baidu --track origin/millennium-baidu

修改代码

$ git add

$ git commit –m “xxxxx”

提交到gerrit上

$ repo upload . [--re=donghe@baidu.com](mailto:--re=donghe@baidu.com)

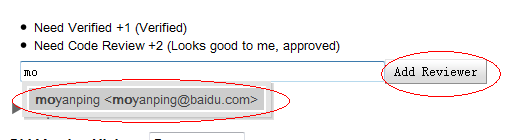
或者使用git命令

$ git push ssh://lijinsuo@git.scm.baidu.com:8235/build HEAD:refs/for/millennium-baidu

# 4 登陆Gerrit操作

## 4.1手动添加评审人员【可选】

1. 登陆gerrit系统，选择ALL或MY，查找自己提交的评审记录，gerrit系统自动将comment信息设置为评审记录的subject
2. 点击记录，进入到记录界面，输入评审人员账户或email信息，点击Add Reviewr，添加评审人员和验证人员。



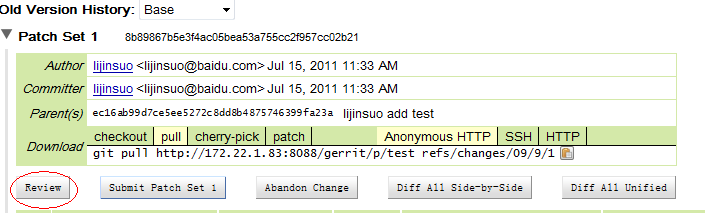
## 4.2代码评审

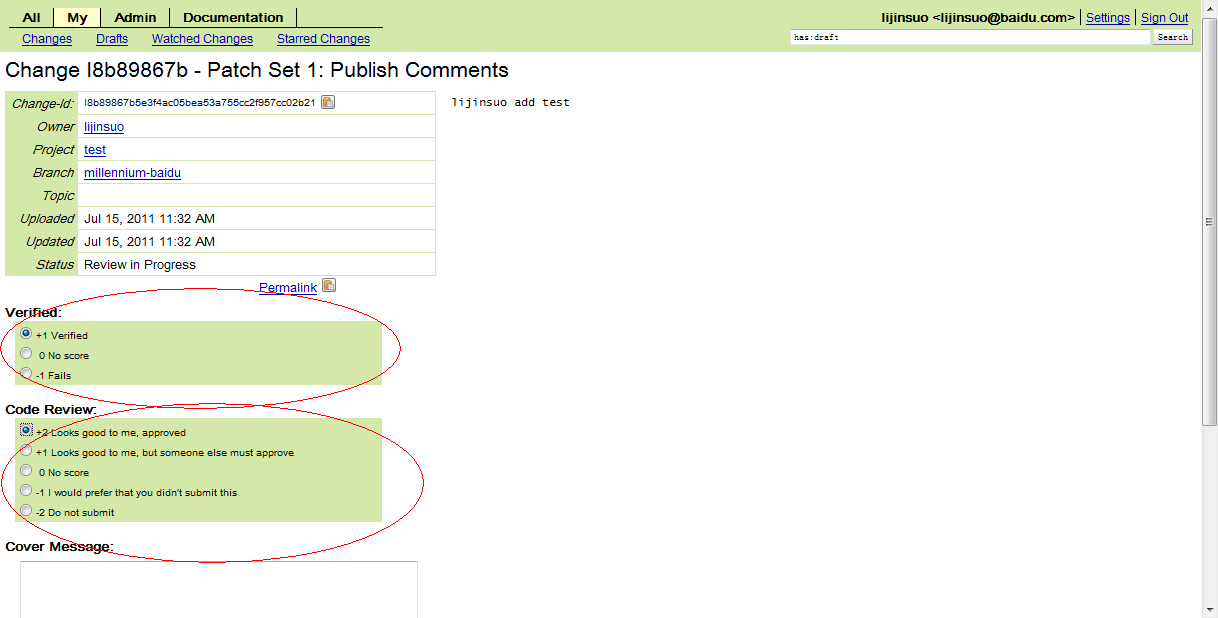
1. 评审人员对记录进行评审，验证人员对记录进行验证工作。

可点击Diff All Side-by-Side或Diff ALL Unified查看修改前后的差别。



1. 点击review，添加意见。





1. 当评审未通过，评审人员设置Verified为-1和CodeReview为-2，gerrit自动发邮件通知开发人员评审失败，并自动将对应的jira记录由Resolved修改为Inprogress
2. 评审失败，重新提交的办法：

推荐仍然使用原来的评审记录ID，即在原评审记录中增加新的patchset。

1. 修改代码
2. $ git add 修改文件
3. $ git commit --amend
4. $ repo upload .

如果要使用新的评审记录ID，可使用如下方法。

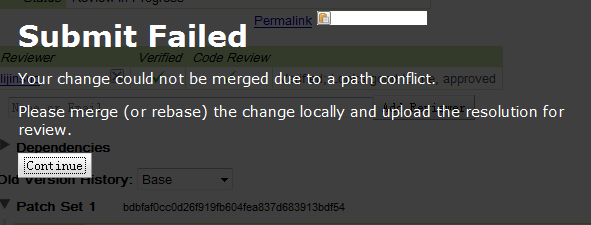
1. 修改代码
2. $ git add 修改文件
3. $ git commit -m “comment”
4. $ repo upload .
5. 当评审和验证通过后，评审人员设置Verified为+1和CodeReview为+2，gerrit自动发邮件通知开发评审通过。

## 4.3将代码submit到git服务器

1. 当代码评审通过后，具有submit权限的人员将rd修改合并到git服务器的正式分支上，RD提交生效。



1. 当submit时有文件冲突时，gerrit会自动发邮件通知开发人员，并自动将对应的jira记录由Resolved修改为Inprogress



1. Submit失败的解决办法：
2. RD进入到有冲突的git库中，执行以下命令解决。

$ git fetch --all

$ git rebase origin/master

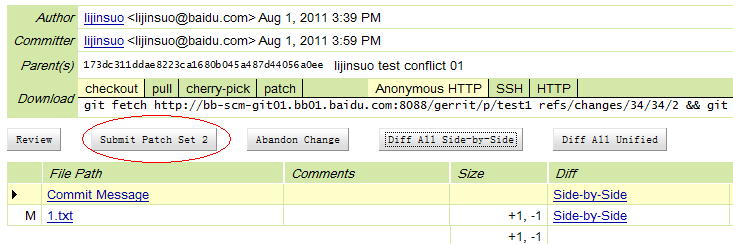
1. 修改文件，解决冲突，解决完毕，执行以下操作

$ git add 冲突文件

$ git rebase --continue

1. RD将对应的jira记录由Resolved修改为Inprogress
2. 重新提交到gerrit平台上，本次修改自动使用原gerrit记录ID，生成新的patchset

$ repo upload



1. 评审人员重新review，并将新的patchset提交到git服务器

# 5 Hudson编译

1. hudson会每分钟轮询检查git服务器的代码是否有change，当监控到git服务器有新的代码提交时，自动触发构建。
2. 若构建成功，hudson会自动将本次所有提交对应的jira记录由resolved修改为integrated,并自动将本次编译的BuildNumber和ImagePath回填到对应的jira记录中
3. 当构建失败，hudson不会去修改jira状态，直到下一次构建成功，hudson会将本次构建成功与上次构建成功之间所有提交对应的jira状态由resolved修改为integrated