代码管理工具GitLab 可视化界面在windows 安装教程

--作者：Able

1.Windows下TortoiseGit over Putty or Openssh提交项目到GitLab（可视化工具）：（安装教程按照1进行）

**具体安装教程：**

**Windows下TortoiseGit over Putty or Openssh提交项目到GitLab**

Gitlab默认的配置推荐使用shell命令行与server端进行交互,作为搞SCM的表示无压力,但是作为开发人员还是比较适应windows的GUI图形管理,方便他们将注意力集中到代码编写上,所以今天刚好接着昨天给大家介绍一下使用windows下的git client工具实现将本地代码提交到远程服务器.

解决方案：

环境部署

操作系统                 Windows7 x64 Sp1

TortoiseGit             TortoiseGit-1.8.11.0-64bit.msi

Git                          Git-1.9.4-preview20140815.exe

Putty                      PuTTY 0.63

OpenSSH                owned by Git

下载地址:  <http://yunpan.cn/QaaLGsd23BsWL> (访问密码 96d7)

注: 此处下载地址中的安装包都为64位版本,32位请另行下载.

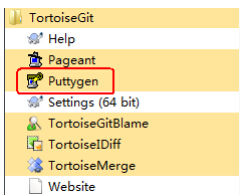
**一.部署前的准备工作**

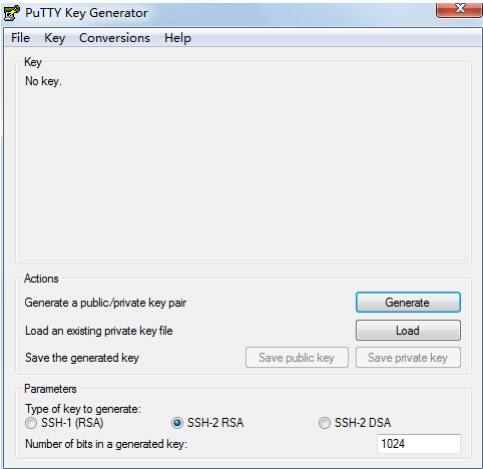
安装TortoiseGit,Git,Putty,默认下一步安装即可,其中Git为Git client环境安装包,确保windows包含git环境.

**二.配置PuTTY authentication agent**

在安装TortoiseGit后,可以选择使用OpenSSH客户端还是使用Putty客户端,如果使用的是默认的Putty客户端,由于TortoiseGit GUI连接不支持server端自定义端口配置,若GitLab使用的SSH端口为自定义端口,则需要使用PuTTY的authentication agent去做一个本地的端口转发.

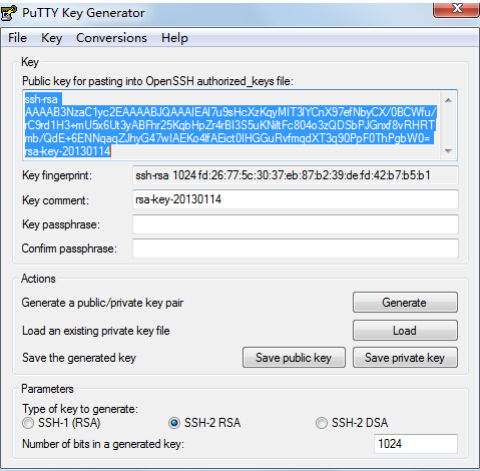
1.首先使用TortoiseGit自带的Puttygen创建本地的公/私钥对

[](http://www.showerlee.com/wp-content/uploads/2014/08/11.jpg)

[](http://www.showerlee.com/wp-content/uploads/2014/08/2.jpg)

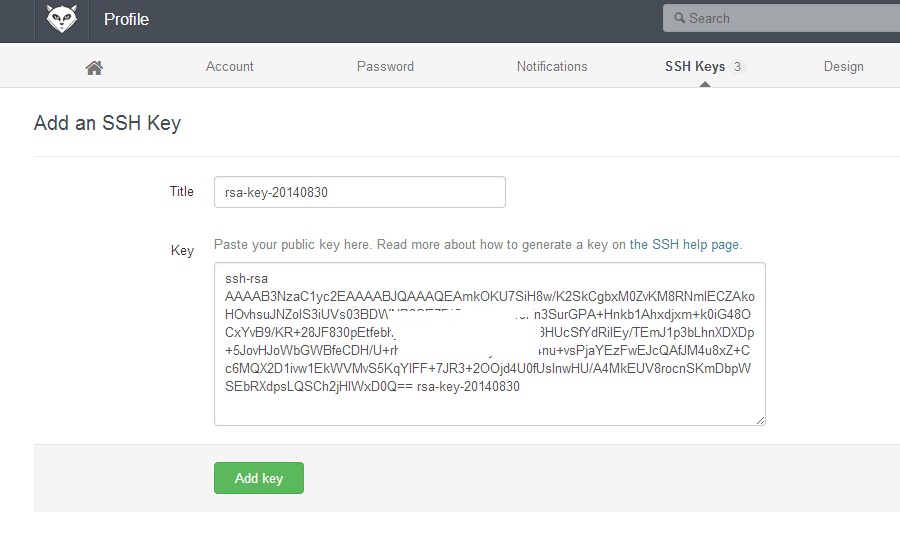
2.点击Generate按钮,在窗口空白处按照提示晃动鼠标,生成公/私钥对,并保存到本地

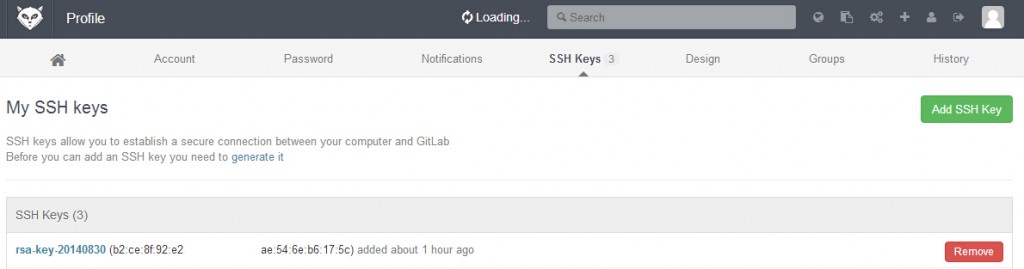
其中testkey为公钥,testkey.ppk为私钥

[](http://www.showerlee.com/wp-content/uploads/2014/08/3.jpg)

[](http://www.showerlee.com/wp-content/uploads/2014/08/4.jpg)

3.将窗口生成的Public key粘贴到GitLab站点具体使用账号的SSh Keys内即完成公钥上传.

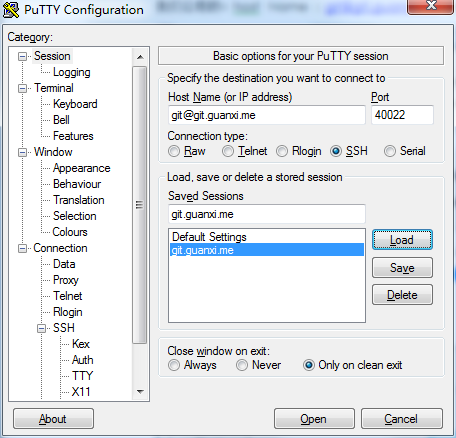
[](http://www.showerlee.com/wp-content/uploads/2014/08/23.jpg)

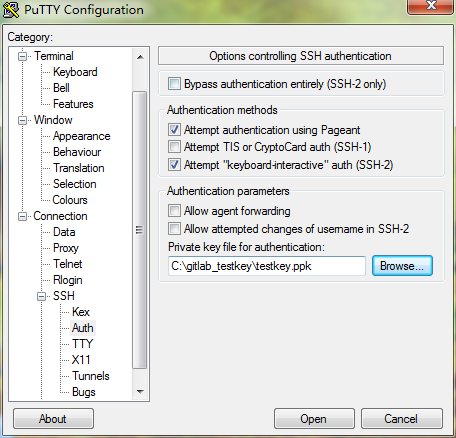
[](http://www.showerlee.com/wp-content/uploads/2014/08/5.jpg)

4.使用PuTTY连接gitlab服务器,open之前需选择之前本地生成的私钥

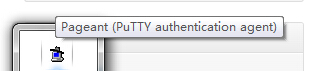
这里服务端domain为git.example.com,

我们公司的：Host Name : [git@git.guanxi.me](mailto:git@git.guanxi.me) port:42202





5.连接成功会在右下角任务栏出现任务图标 (一般情况下回出现)

[](http://www.showerlee.com/wp-content/uploads/2014/08/9.jpg)

**三.配置TortoiseGit**

1.从windows开始菜单选择TortoiseGit-Setting工具,在对话框里选择Git项,填写相关info

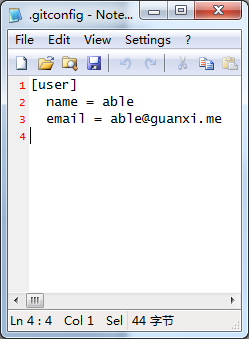
注: 这里的User Info中Name建议与你的Gitlab的账户名保持一致,且该账户必须有该project的相关git操作权限,如果没有可能会造成GIT相关操作报permisson deny错误.当然如果你Gitlab本身刚搭建好,只有系统初始的admin账户,未添加其他账户,则这个权限问题会被系统忽略.

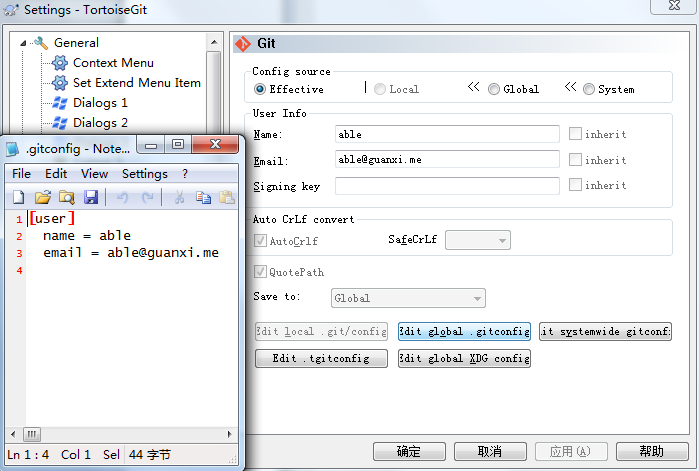
当然另外一种可能是:

大多数国内文档这里未提及这点可能在于国人的windows系统99.9%都是Ghost版本,初始都为admin最高权限,所以这边权限问题直接忽略,但如果你的windows系统是家庭版,或者未使用系统最高权限登陆,则会出现上述报错情况,这个问题也是在我一个朋友那边发现的,他的机子是Surface的Win8家庭版,一般都是普通用户登录,So让我倒持了大半天,终于发现了问题所在...

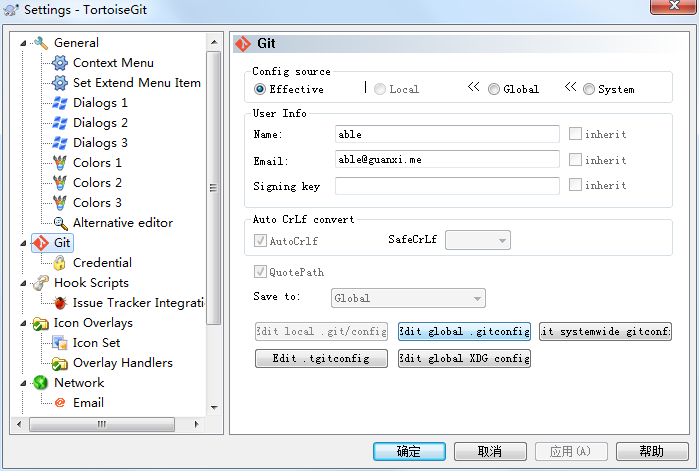
如果有git相关管理经验的朋友,这里可以解释一下.

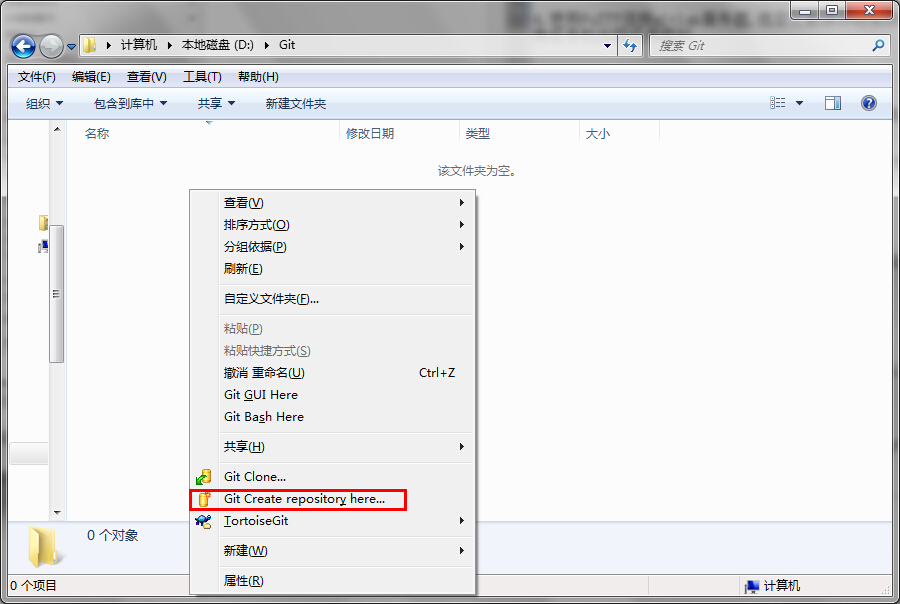
这个如果不可以编辑，可以通过点击 图中 来手动输入，账号名和邮箱

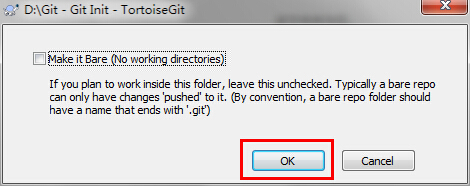
按图示 格式输入 ： 输入后点击确定即可



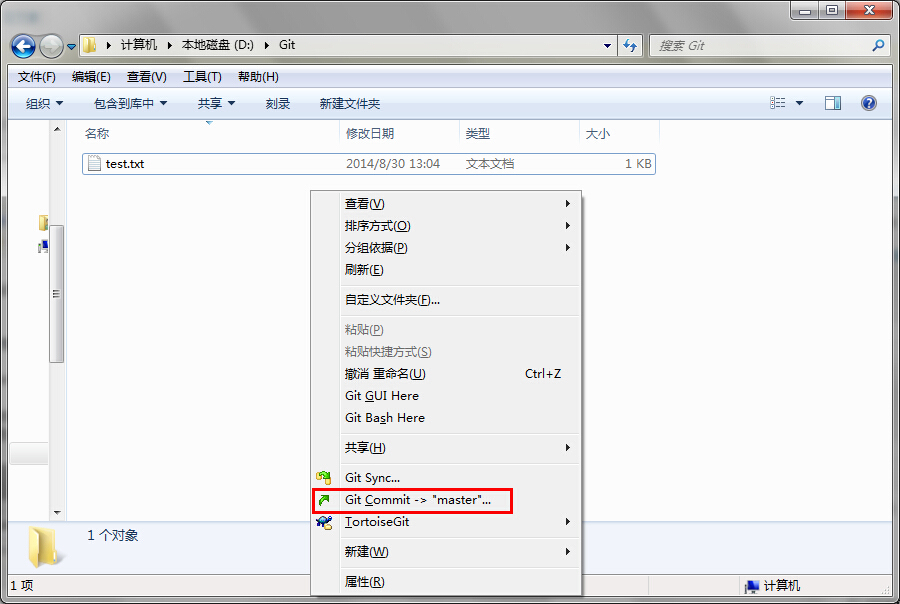
再次进入 就可以看到自己的账号名和邮箱有配置效果：（至此，gitlab客户端本地配置已经完毕。）

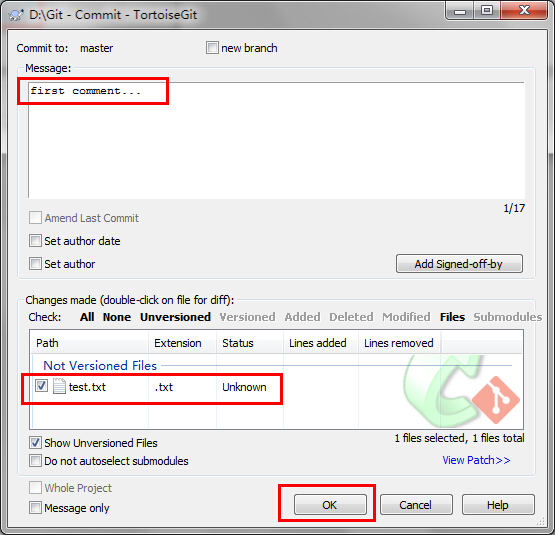


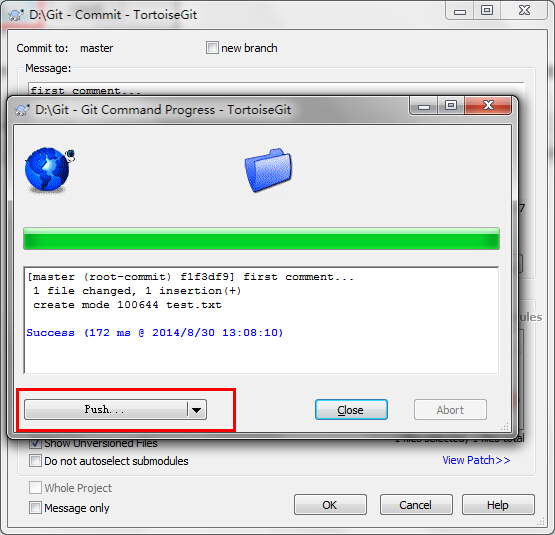
2.在D盘新建一个目录,例如"D:\Git",并进入目录右键目录空白处选择"Git Create repository here...",弹出对话框点确认,这样即建立了一个本地Git仓库. [](http://www.showerlee.com/wp-content/uploads/2014/08/111.jpg)

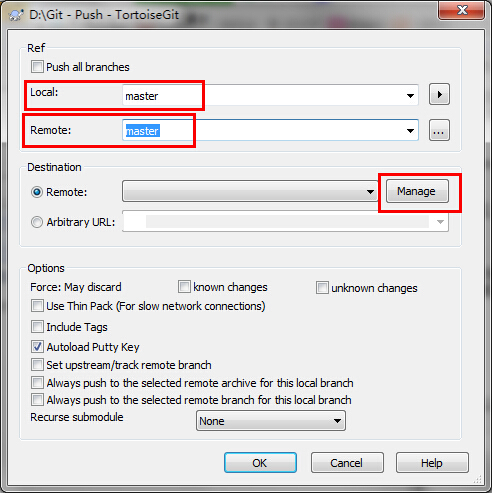
[](http://www.showerlee.com/wp-content/uploads/2014/08/12.jpg)

3.在该仓库目录下创建一个测试项目文本文件,右键目录空白处,选择Git Commit -> "master" ...,在弹出对话框里输入提示注释,选择要加入的版本控制文件,确定即可提交.

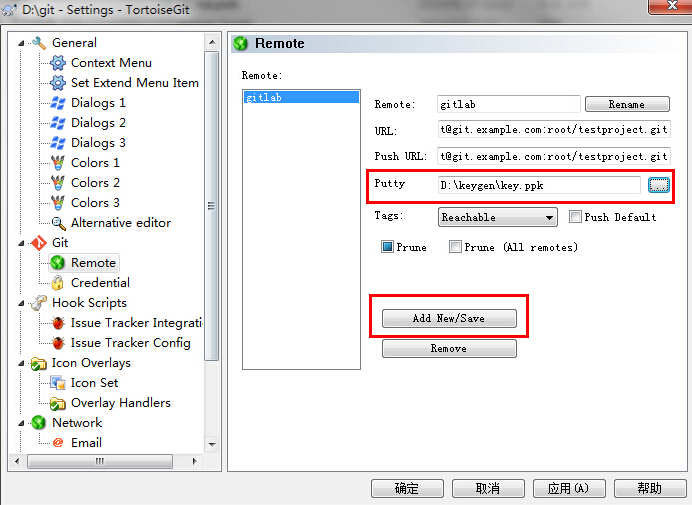
[](http://www.showerlee.com/wp-content/uploads/2014/08/18.jpg)

[](http://www.showerlee.com/wp-content/uploads/2014/08/13.jpg)

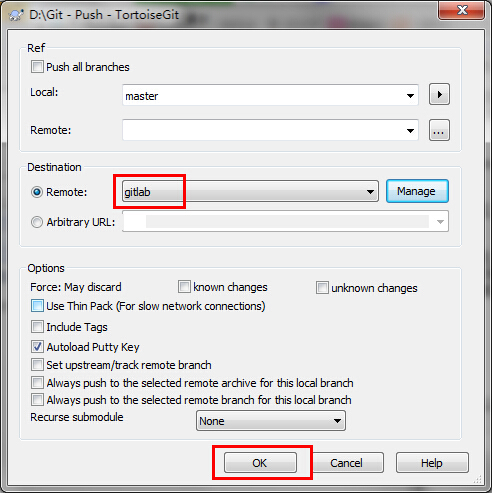
4.现在只是在本地实现使用Git管理项目,在此界面若显示Success则本地提交成功,接下来点击Push..,把我们的改动递交到Git服务器上.  
[](http://www.showerlee.com/wp-content/uploads/2014/08/14.jpg)

5.此时会弹出Push对话框,在Ref - Remote栏里设定当前分支名为master,然后点击Destination - Remote栏的Manage按钮.  
[](http://www.showerlee.com/wp-content/uploads/2014/08/15.jpg)

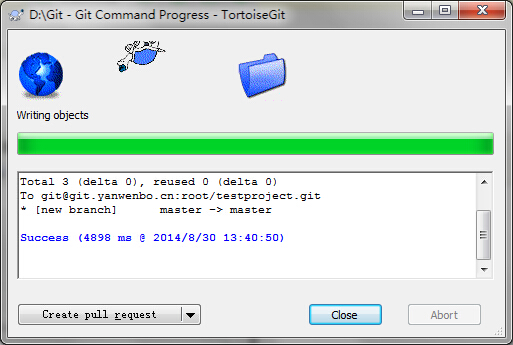
6.在弹出的设置对话框中按照如图所示,填写服务器Remote名称,URL(git@git.yanwenbo.cn:root/tesproject.git),之前保存的本地私钥testkey.ppk,点击Add New/Save按钮保存这一设置,然后点击确定退出返回之前的对话框.

[](http://www.showerlee.com/wp-content/uploads/2014/08/101.jpg)

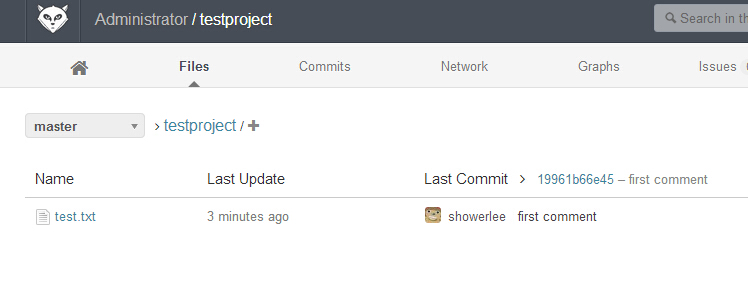
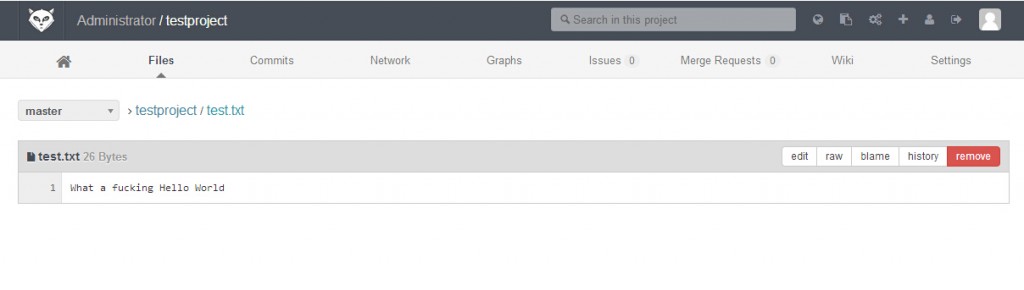
7.最终确认提交

[](http://www.showerlee.com/wp-content/uploads/2014/08/17.jpg)

8.push成功

[](http://www.showerlee.com/wp-content/uploads/2014/08/20.jpg)

9. Giblab后台查看提交结果.

[](http://www.showerlee.com/wp-content/uploads/2014/08/21.jpg)[](http://www.showerlee.com/wp-content/uploads/2014/08/22.jpg)

大功告成...

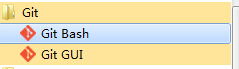
后记...

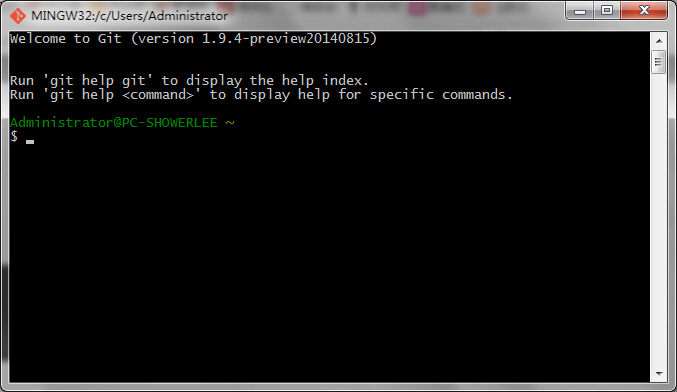
之前介绍的是利用TortoiseGit配合putty实现SSH自定义端口的连接GitLab

但是在给一个朋友配的过程中发现他的SURFACE WIN8系统死活无法PUSH成功,但我的win7同样方法就没问题,索性就用OPENSSH的命令行方法实现.

具体过程这里较之前的稍作更改

1. 利用git bash创建秘钥,方法和linux命令行创建秘钥命令行格式基本一致

[](http://www.showerlee.com/wp-content/uploads/2014/08/110.jpg)

[](http://www.showerlee.com/wp-content/uploads/2014/08/24.jpg)

a.客户端生成秘钥

# cd ~

# ssh-keygen -t rsa

一路回车后生成公钥和秘钥对

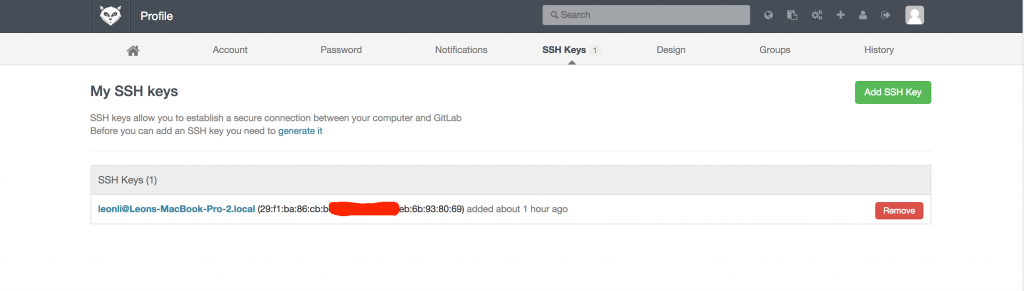
# cat ~/.ssh/id\_rsa.pub

—————————————————————————————————————————————————

将这里生成的秘钥全部复制并粘贴到gitlab web SSH Keys后台保存即可

—————————————————————————————————————————————————

上传后如图:

[](http://www.showerlee.com/wp-content/uploads/2014/08/QQ20140829-9@2x.png)

b.测试SSH连接

# ssh -p22 [git@git.example.com](mailto:git@git.example.com)

若报如下错误:

——————————————————————————————————————

PTY allocation request failed on channel 1

/usr/bin/env: ruby: No such file or directory

Connection to git.example.com closed.

——————————————————————————————————————

说明服务端ruby环境变量未在此目录/usr/bin/ruby

在服务器端加此软链即可:

# ln -s /usr/local/ruby/bin/ruby /usr/bin/ruby

注:若服务器端SSH自定义端口,则需要在客户端~/.ssh/config下添加端口配置

假定自定义SSH端口为2222

# echo “Port 2222” >> ~/.ssh/config

c.重新连接

# ssh -p22 [git@git.example.com](mailto:git@git.example.com)

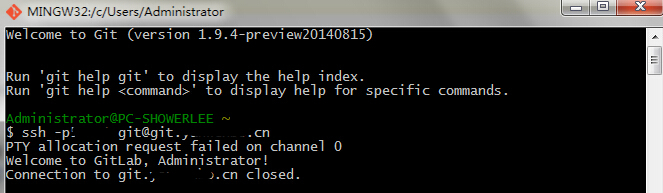
——————————————————————————————————————

PTY allocation request failed on channel 1

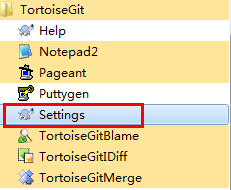
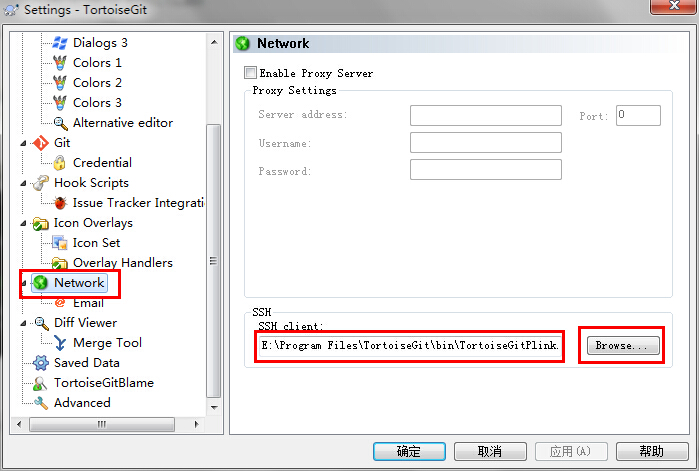
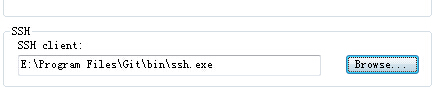
Welcome to GitLab, Anonymous!

Connection to git.example.com closed.

——————————————————————————————————————

[](http://www.showerlee.com/wp-content/uploads/2014/08/32.jpg)

2.将TortoiseGit默认的SSH客户端TortoiseGitPlink(Putty)更改为openssh,具体路径与你的实际安装路径为准.

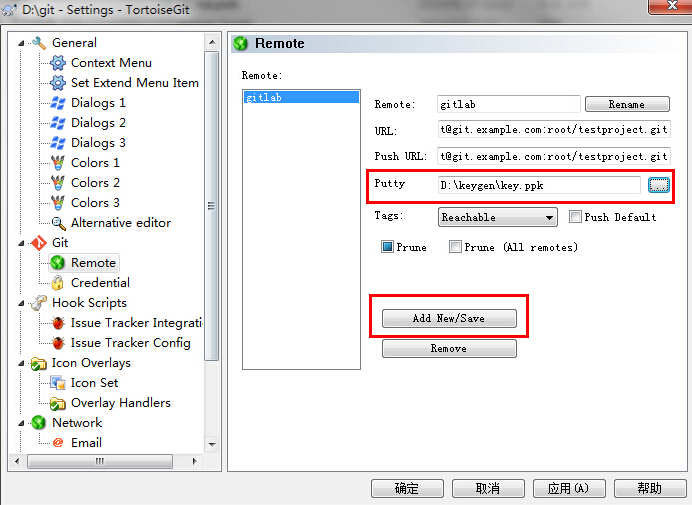
[](http://www.showerlee.com/wp-content/uploads/2014/08/41.jpg)[](http://www.showerlee.com/wp-content/uploads/2014/08/51.jpg)[](http://www.showerlee.com/wp-content/uploads/2014/08/61.jpg)

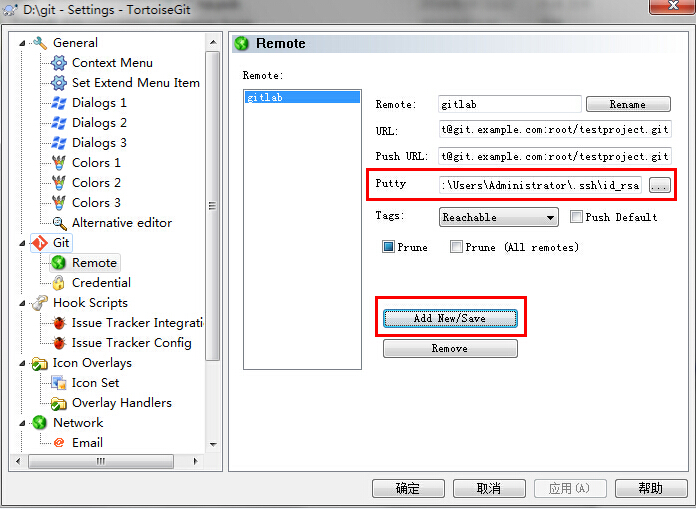
3.更改TortoiseGit的秘钥import路径为Git bash命令行生成的秘钥路径

将之前到此处的配置中如下PuttyGen生成的秘钥路径更改成你的openssh生成的秘钥

笔者是windows管理员账户登录

所以路径为 C:\Users\Administrator\.ssh\id\_rsa

[](http://www.showerlee.com/wp-content/uploads/2014/08/101.jpg)

[](http://www.showerlee.com/wp-content/uploads/2014/08/91.jpg)

4.其他配置保持不变,按照上文操作即可完成openssh认证远程Git over SSH for Gitlab.

<完>