# 安装VMwareTools：

VMwareTools是VMware虚拟机中很重要的一个工具包，有些时候在虚拟机中安装完操作系统会缺少网卡驱动，不能上网，这时只要安装VMwareTools就可以解决问题，下面以CentOS为例，来说明VMwareTools的安装方法

1、打开VMware Workstation虚拟机，开启CentOS系统

虚拟机-安装VMware Tools

登录CentOS终端命令行

2、mkdir /media/mnt    #新建挂载目录

mount /dev/cdrom    /media/mnt/      #挂载VMware Tools安装盘到/media/mnt/目录

cd /media/mnt/    #进入安装目录

ll   #查看

cp    VMwareTools-8.8.1-528969.tar.gz    /home    #复制文件到/home目录

系统运维  www.osyunwei.com  温馨提醒：qihang01原创内容©版权所有,转载请注明出处及原文链接

3、tar zxvf VMwareTools-9.6.2-1688356.tar.gz #解压(VMwareTools-9.6.2-1688356.tar.gz这个名称不同的版本是不同的，这里是以VMware 10.03的版本为例)

cd vmware-tools-distrib   #进入文件目录

./vmware-install.pl  #安装

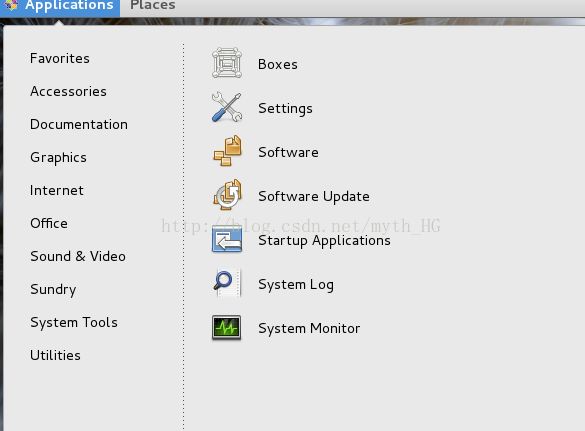
一直按enter即可

最后，重启服务器，VMwareTools安装成功。

为了在/mnt/hgfs/生产共享文件夹，需要执行/usr/bin/下的./vmware-config-tools.pl，一路enter回车，即可

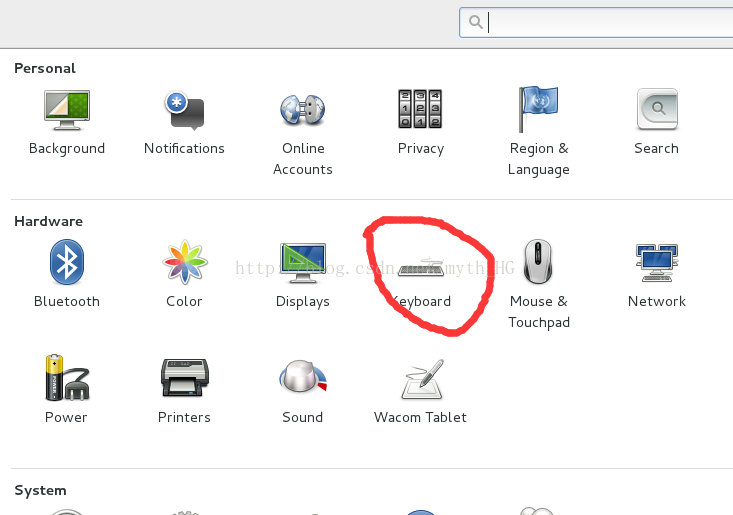
# CentOS7设置终端快捷键：

**首先打开Applications,选择System Tools**

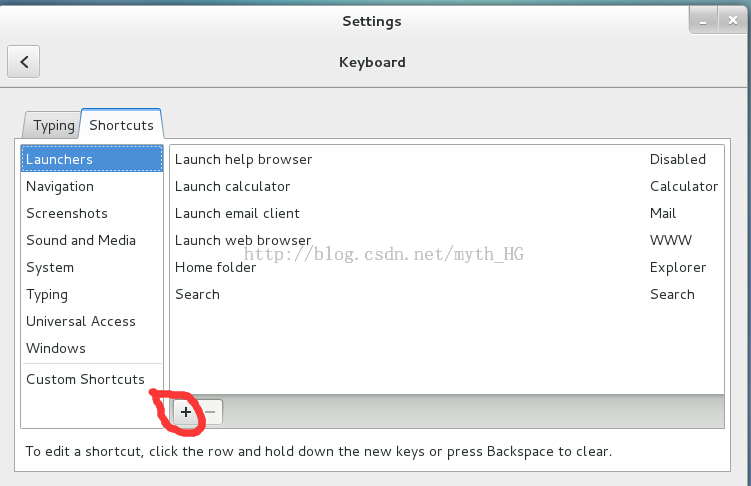
**IMG_256**

IMG_258

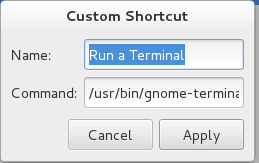
**选择Setting进入下面界面**



**选择Keyboard，进入下面界面，点击Shortcuts**

****

**再点击+号，弹出对话框**

****

**Name：****Run a Terminal**

**Command:  /usr/bin/gnome-terminal (Terminal 程序的路径）**

**点击apply**

**一开始是禁用（disabled）的**

**点击disabled出现下面情形**

IMG_262

**此时按下你喜欢的快捷键比如Ctrl+Alt+T**

**完成。**

# Xshell工具上传下载文件到linux：

**Xshell本地window和linux互传文件：**

在linux中rz 和 sz 命令允许开发板与主机通过串口进行传递文件了，下面我们就来简单的介绍一下rz 和 sz 命令的例子。

rz，sz是Linux/Unix同Windows进行ZModem文件传输的命令行工具。  
优点就是不用再开一个sftp工具登录上去上传下载文件。

sz：将选定的文件发送（send）到本地机器  
rz：运行该命令会弹出一个文件选择窗口，从本地选择[文件上传](http://www.111cn.net/tags.php/%CE%C4%BC%FE%C9%CF%B4%AB/" \t "http://blog.csdn.net/is_zhoufeng/article/details/_blank)到Linux服务器

从服务端发送文件到客户端：

sz filename

从客户端上传文件到服务端：

rz

在弹出的框中选择文件，上传文件的用户和组是当前登录的用户

# 配置防火墙，开启80端口、3306端口：

**CentOS 7.0默认使用的是firewall作为防火墙，这里改为iptables防火墙。**

**1、关闭firewall：**

**#停止firewall服务**

**systemctl stop firewalld.service**

**#禁止firewall开机启动**

**systemctl disable firewalld.service**

**2、安装iptables防火墙**

**#安装**

**yum install iptables-services**

**#编辑防火墙配置文件**

**vi /etc/sysconfig/iptables**

**# Firewall configuration written by system-config-firewall**

**# Manual customization of this file is not recommended.**

**\*filter**

**:INPUT ACCEPT [0:0]**

**:FORWARD ACCEPT [0:0]**

**:OUTPUT ACCEPT [0:0]**

**-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT**

**-A INPUT -p icmp -j ACCEPT**

**-A INPUT -i lo -j ACCEPT**

**-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT**

**-A INPUT -m state --state NEW -m tcp -p tcp --dport 80 -j ACCEPT**

**-A INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT**

**-A INPUT -j REJECT --reject-with icmp-host-prohibited**

**-A FORWARD -j REJECT --reject-with icmp-host-prohibited**

**COMMIT**

**:wq! #保存退出**

**#最后重启防火墙使配置生效**

**systemctl restart iptables.service**

**#设置防火墙开机启动**

**systemctl enable iptables.service**

**二、关闭SELINUX**  
**#修改配置文件**

**vi /etc/selinux/config**

**#SELINUX=enforcing #注释掉**

**#SELINUXTYPE=targeted #注释掉**

**SELINUX=disabled #增加**

**:wq! #保存退出**

**#使配置立即生效**

**setenforce 0**

# 安装apache：

**yum install httpd**

**可能会用到的：**

**systemctl start httpd.service #启动apache**

**systemctl stop httpd.service #停止apache**

**systemctl restart httpd.service #重启apache**

**systemctl enable httpd.service #设置apache开机启动**

**restart一下，然后：**  
**输入localhost**

# 安装JDK:

## **[CentOS 卸载自带OpenJdk](http://www.cnblogs.com/zsxfbj/archive/2010/09/22/CentOS_Unstall_OpenJDK.html)**

安装好的CentOS会自带OpenJdk,用命令 java -version ，会有下面的信息：

java version "1.6.0"  
OpenJDK  Runtime Environment (build 1.6.0-b09)  
OpenJDK 64-Bit Server VM (build 1.6.0-b09, mixed mode)

最好还是先卸载掉openjdk,在安装sun公司的jdk.

先查看 rpm -qa | grep java

显示如下信息：

java-1.4.2-gcj-compat-1.4.2.0-40jpp.115  
java-1.6.0-openjdk-1.6.0.0-1.7.b09.el5

卸载：

rpm -e --nodeps java-1.4.2-gcj-compat-1.4.2.0-40jpp.115  
rpm -e --nodeps java-1.6.0-openjdk-1.6.0.0-1.7.b09.el5

还有一些其他的命令

rpm -qa | grep gcj

rpm -qa | grep jdk

如果出现找不到openjdk source的话，那么还可以这样卸载

 yum -y remove java java-1.4.2-gcj-compat-1.4.2.0-40jpp.115  
 yum -y remove java java-1.6.0-openjdk-1.6.0.0-1.7.b09.el5

**安装自己的JDK：**

rpm -ivh jdk-xxxx.rpm

vi /etc/profile

export JAVA\_HOME=/usr/java/jdk1.6.0\_10   
export CLASSPATH=.:$JAVA\_HOME/jre/lib/rt.jar:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar   
export PATH=$PATH:$JAVA\_HOME/bin

执行source /etc/profile.

# 安装rpmforge：

**rpmforge是包含4000多种CentOS软件的软件库，被CentOS社区认为是安全和稳定的软件库，安装rpmforge-release-0.5.3-1.el7.rf.x86\_64.rpm如下：**

**从CSDN下载 ：**

**<http://download.csdn.net/detail/mimi00x/8081263>**

**从官网http://repoforge.org链接到http://wiki.centos.org/AdditionalResources/Repositories/RPMForge，没有CentOS 7的链接，观察CentOS 4、5、6的链接的规律，猜测CentOS 7的包名为rpmforge-release-0.5.3-1.el7.rf.x86\_64.rpm。**

rpm --import http://apt.sw.be/RPM-GPG-KEY.dag.txt  
rpm -K rpmforge-release-0.5.3-1.el7.rf.x86\_64.rpm  
rpm -i rpmforge-release-0.5.3-1.el7.rf.x86\_64.rpm

rpmforge的repo文件已经在目录/etc/yum.repos.d/下，禁掉rmpforge的repo：

enabled = 0  
yum install \* --enablerepo=rpmforge

done! ^\_^

二、安装rpmforege　

  1. 在http://pkgs.repoforge.org/rpmforge-release/下载相应版本的rpm文件。（我的版本是rpmforge-release-0.5.3-1.el6.rf.x86\_64.rpm）

  2. 在shell中执行

rpm -ivh rpmforge-release-0.5.3-1.el6.rf.x86\_64.rpm

### **启用rpmforge-extras库**

用文本编辑器(vi、vim、gedit等都可以)打开文件/etc/yum.repos.d/rpmforge.repo，找到[rpmforge-extras]，把enabled=0改成enabled=1，如下所示：

[IMG_256](http://www.cnblogs.com/blackpuppy/p/javascript:void(0);)

### Name: RPMforge RPM Repository for RHEL 6 - dag

### URL: http://rpmforge.net/**[**rpmforge**]**

# ...

**[**rpmforge-extras**]**

name = RHEL $releasever - RPMforge.net - extras

baseurl = http://apt.sw.be/redhat/el6/en/$basearch/extras

mirrorlist = http://mirrorlist.repoforge.org/el6/mirrors-rpmforge-extras

#mirrorlist = file:///etc/yum.repos.d/mirrors-rpmforge-extras

enabled = 1

protect = 0

gpgkey = file:///etc/pki/rpm-gpg/RPM-GPG-KEY-rpmforge-dag

gpgcheck = 1

**[**rpmforge-testing**]**

# ...

### **关闭rpmforge-extras库、清理**

升级完成后，关闭rpmforge-extras库。与步骤2.2类似，用文本编辑器打开/etc/yum.repos.d/rpmforge.repo，找到[rpmforge-extras]，把enabled=1改成enabled=0，如下所示：

[IMG_258](http://www.cnblogs.com/blackpuppy/p/javascript:void(0);)

### Name: RPMforge RPM Repository for RHEL 6 - dag### URL: http://rpmforge.net/

[rpmforge]# ...

[rpmforge-extras]# ...

enabled = 0

# ...

[rpmforge-testing]# ...

[IMG_259](http://www.cnblogs.com/blackpuppy/p/javascript:void(0);)

最后，清理yum的缓存：

# yum clean all

# 安装升级git：

**Window安装git：**

**利用的软件：**

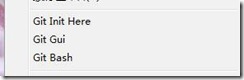
|  |  |
| --- | --- |
| **软件** | **下载地址** |
| Git（版本Git-1.9.2） | <http://git-scm.com/download/> |
| TortoiseGit（版本TortoiseGit-1.8.8.0） | <http://code.google.com/p/tortoisegit/> |
| TortoiseGit语言汉化包 | <http://code.google.com/p/tortoisegit/wiki/Download?tm=2> |

（其中TortoiseGit为Git图形客户端）

1、 安装Git。按照默认安装，到这一步的时候，界面如下：

[](http://images.cnitblog.com/blog/98693/201405/041429255649300.jpg)

选择第二项，这样，安装完毕后，在Windowns系统的右键菜单中，将出现

[](http://images.cnitblog.com/blog/98693/201405/041429308616718.jpg)

在开发机上,生产密钥并上传到服务器上  
ssh-keygen -t rsa   #一路回车，不需要设置密码

ssh-keygen -t rsa -C "your\_email@youremail.com" (不过最好不要带：.com, .cn后缀)

#上传公钥到服务器(默认SSH端口22)  
scp ~/.ssh/id\_rsa.pub  192.168.100.202:/tmp

或编辑`/etc/hosts`文件，在`/etc/hosts`文件里添加如下文本：  
  
# local git server  
192.168.100.202 zgit  
  
然后再上传自己的公钥到服务器  
scp ~/.ssh/id\_rsa.pub  zgit:/tmp/

2、 安装TortoiseGit。默认安装就行。

3、 安装TortoiseGit语言汉化包。默认安装。安装完毕后，设置TortoiseGit为中文（这个不难，在此就不讲了）。

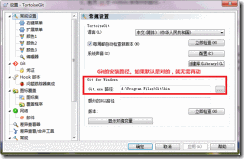
**二、** **配置TortoiseGit。**

1、 在windows系统中，单击右键，如图所示：

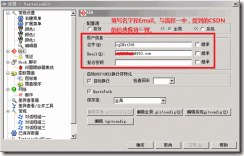
[](http://images.cnitblog.com/blog/98693/201405/041429491114361.jpg)

选择“TortoiseGit-->设置”。

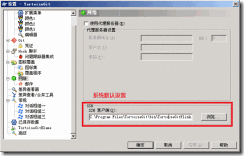
2、 配置Git在windows系统中的路径。在设置界面中，选择“常规设置”

[](http://images.cnitblog.com/blog/98693/201405/041430101271748.gif)

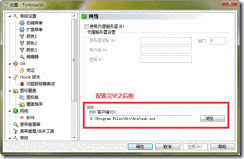
3、 配置用户名和邮箱。在设置界面，选择“Git”。界面右端，选中“全局”。在名字和Email中，填写注册CSDN时（流程一）提到的名字和邮箱。

[](http://images.cnitblog.com/blog/98693/201405/041430296586348.gif)

4、 配置SSH。在设置界面，选择“网络”。默认SSH设置为：

[](http://images.cnitblog.com/blog/98693/201405/041430474868161.gif)

点击“浏览”，设置路径为Git的SSH.exe。一般在Git的安装目录Bin下边。有一个ssh.exe的文件，选中。如图所示：

[](http://images.cnitblog.com/blog/98693/201405/041431010482271.gif)

**linux安装GIT （开发机和服务机安装步骤相同） ：**

**安装ssh：**

yum install openssh-server openssh-client

**仅能安装也能更新升级：**

**yum install git**

创建git服务器管理用户：

#useradd -m git

#passwd git //注意这里要设置一个密码，建议不要不设密码，网上很多资料都不设密码

# mkdir /home/git/repositories

创建一个git仓库，并验证git是否可用,在使用git之前，要设置git的两个全局变量，设置方式为：

1. # git config--global user.name "linxl"

2. # git config--global user.email ["350825854@qq.com"](mailto:\"350825854@qq.com\")

设置git仓库目录的权限:

1. # chown git:git /home/git/repositories
2. # chmod 755 /home/git/repositories

# su - git //切换到git用户

在开发机上,生产密钥

ssh-keygen -t rsa   #一路回车，不需要设置密码

ssh-keygen -t rsa -C "your\_email@youremail.com" (不过最好不要带：.com, .cn后缀)

创建并初始化一个验证测试repository：

$ mkdir /home/git/repo //假设要创建的repository放在/home/git/repo目录下

$ cd /home/git/repo

$ mkdir test\_repo

$ cd test\_repo

$ git init --bare

测试新创建的repository是否可用：

 su - test (test为另外一个user或者exit，使用root用户)

 $ cd /home/test

$ mkdir repos

$ cd repos

$ git clone /home/git/repo/test\_repo

如果显示" Initialized empty Gitrepository in /home/git/repo/test\_repo/.git/",则说明成功的clone了一个local repository。之后就可以在此repository下进行开发了，比如，添加一个readme.txt文件。

1. $ cd test\_repo

2. $ vim readme.txt (创建一个文件)

3. $ git add . (注意后面有一个[.])

4. $ git commit -a

5. $ git push origin master

注意：如果此时报错，最大的原因是test\_repo的refs和objects目录的权限不够，执行如下操作即可（git server端的repository下的这两个目录）。

1. $ chmod -R 777 refs/

2. $ chmod -R 777objects/

# 服务器端 gitosis的安装：

**（目录的权限控制对公钥免密码很重要）**

参考： <http://blog.csdn.net/dengsilinming/article/details/7490335>

安装python的setup tool：

python-setuptools的安装（因为gitosis的安装脚本是用python写的）

#yum install python-setuptools

获取并安装gitosis：

git clone git://github.com/res0nat0r/gitosis.git//下载gitosis到/home/git/目录下

cd gitosis

python setup.pyinstall //如果没有报错，则说明安装成功

配置gitosis：

在开发机上,生产密钥并上传到服务器上  
ssh-keygen -t rsa   #一路回车，不需要设置密码

上传公钥到服务器(默认SSH端口22):

scp ~/.ssh/id\_rsa.pub 192.168.100.202:/tmp

为了git操作时不用输入密码,还要在服务器端做以下配置:

# su 到git用户  
su - git

$ cat /tmp/id\_dsa.pub >> /home/git/.ssh/authorized\_keys //将root的公匙导入到git目录的 authorized\_keys文件中

$ chmod 600 /home/git/.ssh/authorized\_keys

初始化git服务器:

# su 到git用户  
su - git

git账户根目录下（/home/git/）

 $ chmod a+r /tmp/

gitosis-init < /tmp/id\_rsa.pub

初始化之后，会在/home/git/repositories创建gitosis-admin.git项目，可以通过维护这个项目来对gitosis进行配置。

除此以外，还需要对gitosis-admin.git/hooks/post-update目录赋上特殊权限：

chmod 755 /home/git/repositories/gitosis-admin.git/hooks/post-update

#显示以下信息即表示成功

#Initialized empty Git repository in /home/git/repositories/gitosis-admin.git/

#Reinitialized existing Git repository in /home/git/repositories/gitosis-admin.git/

4)#删除密钥

rm -rf /tmp/id\_rsa.pub

5)在个人开发机上导出项目管理  
```# su 到git用户  
su - git

mkdir -p /home/git/repo  
cd /home/git/repo  
git clone [git@zgit:](mailto:git@zgit:gitosis-admin.git)[gitosis-admin.git](mailto:git@zgit:gitosis-admin.git) //注意路径直接上版本库的目录

1. **在个人开发机增加及设置管理项目**  
   ```  
   cd /home/git/repo/gitosis-admin  
     
   # 查看git服务器已经上传密钥  
   ls keydir    
     
   cat keydir/ltl@jackliu-ThinkPad.pub    
     
   #ltl@jackliu-ThinkPad.pub为已经上传的开发机生成的公密  
     
   #显示密钥 最后的字符串为 密钥用户名 这里为 ltl@jackliu-ThinkPad  
   vim gitosis.conf  
     
   #在文件尾增加以下内容  
     
   [group test-git]            # 具有写权限的组名称  
   writable = test-git         # 该组可写的项目名称  
   members = ltl@jackliu-ThinkPad  guangyun.ni@yeepay.com     #该组的成员(密钥用户名) 多个用户协同开发时，以空格分隔  
     
   # 如果要增加只读的组 参考如下  
   # [group test-git-readnoly]          # 具有都权限的组名称  
   # readonly = test-git                # 该组只读的项目名称  
   # members = ltl@jackliu-ThinkPad     # 该组的成员  
     
     
   #提交修改  
   git add .  
   git commit -a -m "add test-git repo"  
   git push  
     
   ```  
     
   **5.在个人开发机上初始，增加及使用项目test-git**  
     
   ```  
   cd ~/repo    
     
   mkdir test-git     
     
   cd test-git    
     
   git init    
     
   touch readme    
     
   git add .     
     
   git commit -a -m "init test-git"    
     
   git remote add origin git@zgit:test-git.git    
     
   git push origin master    
     
   ```  
     
   **6.增加协同开发者的公钥key到git服务器**  
      
    - 执行`cd repo/gitosis-admin/keydir`切换目录  
      
    - 把协同开发者的id\_rsa.pub 文件里的数据 拷贝到 对应的开发者的`密钥用户名.pub`文件。如把密钥用户名 guangyun.ni@yeepay.com 的 id\_rsa.pub 文件中文本 粘贴到 guangyun.ni@yeepay.com.pub 文件里，并保存  
     
    - 然后将添加数据后的目录更新到git服务器  
      
    ```  
     
   需要说明的是，必须由个人开发机上的keydir目录和gitosis.conf更新到远程服务器，才可以让其他协同开发者一起更新代码。

# 安装配置gitolite:

# 安装配置gitweb：

**安装gitweb：**

# yum install gitweb

**修改gitweb.conf:**

# vi /etc/gitweb.conf

$projectroot = "/home/git/repositories/"

将$projectroot设置为git代码库所在的文件夹

完整的giweb.conf配置：

01 # path to git projects (<project>.git)

02 #$projectroot = "/var/cache/git";

03 $projectroot = "/home/git/repositories";

04

05 # directory to use for temp files

06 $git\_temp = "/tmp";

07

08 # target of the home link on top of all pages

09 $home\_link = $my\_uri || "/";

10

11 # html text to include at home page

12 $home\_text = "indextext.html";

13

14 # file with project list; by default, simply scan the projectroot dir.

15 $projects\_list = $projectroot;

16

17 # stylesheet to use

18 @stylesheets = ("/gitweb/static/gitweb.css");

19

20 # javascript code for gitweb

21 $javascript = "/gitweb/static/gitweb.js";

22

23 # logo to use

24 $logo = "/gitweb/static/git-logo.png";

25

26 # the 'favicon'

27 $favicon = "/gitweb/static/git-favicon.png";

28

29 # git-diff-tree(1) options to use for generated patches

30 #@diff\_opts = ("-M");

31 @diff\_opts = ();

**修改git.conf:**

# vi /etc/httpd/conf.d/git.conf

Alias /git /var/www/git

<Directory /var/www/git>

options +ExecCGI

AddHandler cgi-script .cgi

DirectoryIndex gitweb.cgi

</Directory>

将 "Alias /git /var/www/git" 改为 "Alias /gitweb /var/www/git"

然后启动Apache： # systemctl start httpd.service

在浏览器中访问：<http://192.168.100.92/gitweb/>



# 安装配置gitlab:

## 简述安装gitlab

git服务器比较有名的是gitosis和gitolite，这两个管理和使用起来稍微有些复杂，没有web页面，而gitlab则是类似于github的一个工具，github无法免费建立私有仓库，并且为了代码安全，于是在内网安装了一个自己实验室的一个git服务器，多方比较，选择了gitlab，gitlab有很多依赖，而bitnami制作了一键安装的包，下载链接：https://bitnami.com/redirect/to/37478/bitnami-gitlab-7.0.0-0-linux-installer.run ，使得安装如此简洁。 安装的过程中，会设置一个管理员账户，这个可以用来登陆，并且，gitlab的成员不能自己随便申请，需要管理员进行邀请；

下载bitnami的一键安装包：

https://bitnami.com/redirect/to/39039/bitnami-gitlab-7.1.1-0-linux-x64-installer.run

下载完成后，这个东西就是傻瓜安装。

安装需要root权限：

chmod 777 bitnami-gitlab-7.10.1-0-linux-x64-installer.run

$ sudo ./bitnami-gitlab-7.10.1-0-linux-x64-installer.run

[sudo] password for wuzhang:

一路 Yes ，更改端口号，防止和web的80冲突，我改为1234了，域名：自己电脑的IP地址！然后自动安装！

用火狐访问一下：http://192.168.1.118:1234

登陆界面，确实比gitweb好看多了！输

入用户名：wuzhang

密码：\*\*\*\*\*\*\*\*（至少8位哦）

安装完成，在浏览器地址栏:http://192.168.1.118:1234

**以下所有操作基于root用户，安装路径为默认路径**

## 安装gitlab服务器

下载地址***[https://bitnami.com/redirect/to/39039/bitnami-gitlab-7.1.1-0-linux-x64-installer.run](https://bitnami.com/redirect/to/39039/bitnami-gitlab-7.1.1-0-linux-x64-installer.run" \t "_blank)***

添加执行权限

chmod +x bitnami-gitlab-7.4.3-0-linux-installer.run

运行安装文件

./bitnami-gitlab-7.4.3-0-linux-installer.run

安装所带的软件

- GitLab 7.4.3

- Apache 2.4.10

- ImageMagick 6.7.5

- MySQL 5.5.40

- Git 1.9.0

- Ruby 1.9.3-p550

- Rails 4.1.6

- RubyGems 1.8.12

安装过程

The installer detects that exists a 'git' user in the system. This installer will change the config uration for this user. **Do** you want **tocontinue** the installation? [y/N]: y

-----------------------------------------------------------------------

Welcome **to** the Bitnami Gitlab Stack Setup Wizard.

-----------------------------------------------------------------------

**Select** the components you want **to** install; clear the components you **donot** want

**to** install. Click **Nextwhen** you **are** ready **tocontinue**.

GitLab : Y (Cannot be edited)

GitLab CI [Y/n] :Y

**Is** the selection above correct? [Y/n]: Y

-----------------------------------------------------------------------

Installation folder

Please, choose a folder **to** install Bitnami Gitlab Stack

**Select** a folder [/opt/gitlab-7.4.3-0]:注意此处可以直接按回车,否则在当前目录创建一个你自己命名的文件夹

-----------------------------------------------------------------------

**Create** Admin account 设置管理员

Bitnami Gitlab Stack admin **user** creation

Email Address [**user**@example.com]: xxxxxxx@qq.com

Login [**user**]: master

Password :\*\*\*\*\*\*\*\*

Please confirm your password :\*\*\*\*\*\*\*\*

-----------------------------------------------------------------------

Hostname that will be used **tocreate** internal URLs. If this **valueis** incorrect,

you may be unable **to** access your Gitlab installation **from** other computers. It **is**

advisable **to** use a **Domain** instead **of** an IP address **for** compatibility **with**

different browsers.

**Domain** [192.168.82.181]:这里写域名

**Do** you want **to** configure mail support? [y/N]: y

-----------------------------------------------------------------------

Configure SMTP Settings

This **is** required so your application can send notifications via email.

**Default** email provider:

[1] GMail

[2] Custom

Please choose an **option** [1] : 2选择2自定义，因为Gmail已经屏蔽了

-----------------------------------------------------------------------

Configure SMTP Settings

This data **is** stored **in** the application configuration files **and** may be visible **to**

others. **For** this reason, it **is** recommended that you **donot** use your personal

account credentials.

Username []: xxxxx@qq.com

Password :

Re-enter :

SMTP Host []: smtp.qq.com

SMTP Port []: 25

Secure **connection**

[1] None

[2] SSL

[3] TLS

Please choose an **option** [3] :3

-----------------------------------------------------------------------

Setup **is** now ready **tobegin** installing Bitnami Gitlab Stack **on** your computer.

**Do** you want **tocontinue**? [Y/n]: y

-----------------------------------------------------------------------

Please wait while Setup installs Bitnami Gitlab Stack **on** your computer.

Installing

0% \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 50% \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 100%

#########################################

-----------------------------------------------------------------------

Setup has finished installing Bitnami Gitlab Stack **on** your computer.

Info: **To** access the Bitnami Gitlab Stack, **goto**

http://192.168.82.181:80**from** your browser.访问路径

Press [Enter] **tocontinue**:

## 配置邮件服务

Gitlab不能注册用户的，需要由经理创建用户并以邮件的方式告知开发人员，开发人员在通过邮件的链接设置密码才能登陆到gitlab。

1. **更改config/environments/production.rb**

vi /opt/gitlab-7.4.3-0/apps/gitlab/

htdocs/config/environments/production.rb

在文件最后设置

config.action\_mailer.perform\_deliveries = true

config.action\_mailer.raise\_delivery\_errors = true

config.action\_mailer.delivery\_method = :smtp

config.action\_mailer.smtp\_settings = {

:address => "smtp.qq.com",

:port => "25",

:domain => "smtp.qq.com",

:authentication => :plain,

:user\_name => "xxxxxxxx@qq.com",

:password => "xxxxxxxx",

:enable\_starttls\_auto => true

}

config.eager\_load = true

config.assets.js\_compressor = :uglifier

config.allow\_concurrency = false

**2,更改config/environments/production.rb**

进入目录

cd /opt/gitlab-7.1.1-0/apps/gitlab/htdocs/config/initializers/

拷贝文件

cp smtp\_settings.rb.sample smtp\_settings.rb

编辑文件

vi smtp\_settings.rb

if Rails.env.production?

Gitlab::Application.config.action\_mailer.delivery\_method = :smtp

ActionMailer::Base.smtp\_settings = {

address: "smtp.qq.com",

port: 25,

user\_name: "xxxxxxxx@qq.com",

password: "xxxxxxxx",

domain: "qq.com",

authentication: :login,

enable\_starttls\_auto: true

}

End

3,  **config/gitlab.yml**

备份gitlab.yml

修改gitlab.yml

Vi /opt/gitlab-7.4.3-0/apps/gitlab/htdocs/config/gitlab.yml

在第38行修改

email\_from: [xxxxxxxx@qq.com](mailto:xxxxxxxx@qq.com)

## 启动与关闭服务器

进入文件夹

Cd /opt/gitlab-7.4.3-0

执行文件

./ctlscript.shstart启动所有服务

参数说明

./ctlscript.sh help

./ctlscript.sh (start|stop|restart|status)

./ctlscript.sh (start|stop|restart|status) mysql

./ctlscript.sh (start|stop|restart|status) apache

./ctlscript.sh (start|stop|restart|status) redis

./ctlscript.sh (start|stop|restart|status) gitlab\_sidekiq

./ctlscript.sh (start|stop|restart|status) gitlabci\_sidekiq

./ctlscript.sh (start|stop|restart|status) gitlabci\_runner

help - this screen

start - start the service(s)

stop - stop the service(s)

restart - restart or start the service(s)

status - show the status of the service(s)

## 目录结构及日志

apache2/: Apache Web server.

ruby/: Ruby language.

redis/: Redis server.

mysql/: MySQL Database.

apps/

gitlab/: GitLab application folder

conf/: GitLab Apache configuration files

htdocs/: GitLab application files

apache错误日志

/opt/gitlab-7.4.3-0/apache2/logs/error\_log

apache访问日志

/opt/gitlab-7.4.3-0/apache2/logs/access\_log

## mysql数据库

mysql数据库默认用户名和密码：

username: bitnami

password: "48312c6b4d"

由于是一键安装，没有给出mysql的用户名密码，在database.yml文件里可以找到。路径：

/opt/gitlab-7.4.3-0/apps/gitlab/htdocs/config/database.yml

## 备份和恢复

使用**Gitlab**一键安装包安装**Gitlab**非常简单, 同样的**备份**恢复与迁移也非常简单. 使用一条命令即可创建完整的**Gitlab备份**:

**gitlab**-rake **gitlab**:backup:create

使用以上命令会在/var/opt/**gitlab**/backups目录下创建一个名称类似为1393513186\_**gitlab**\_backup.tar的压缩包, 这个压缩包就是**Gitlab**整个的完整部分, 其中开头的1393513186是**备份**创建的日期.

### Gitlab 修改备份文件默认目录

你也可以通过修改/etc/**gitlab**/**gitlab**.rb来修改默认存放**备份**文件的目录:

**gitlab**\_rails['backup\_path'] = '/mnt/backups'

/mnt/backups修改为你想存放**备份**的目录即可, 修改完成之后使用**gitlab**-ctl reconfigure命令重载配置文件即可.

### Gitlab 自动备份

也可以通过crontab使用**备份**命令实现自动**备份**:

sudo su -

crontab -e

加入以下, 实现每天凌晨2点进行一次自动**备份**:

0 2 \* \* \* /opt/**gitlab**/bin/**gitlab**-rake **gitlab**:backup:create

### Gitlab 恢复

同样, **Gitlab**的从**备份**恢复也非常简单:

# 停止相关数据连接服务

**gitlab**-ctl stop unicorn

**gitlab**-ctl stop sidekiq

# 从1393513186编号**备份**中恢复

**gitlab**-rake **gitlab**:backup:restore BACKUP=1393513186

# 启动**Gitlab**

sudo **gitlab**-ctl start

### Gitlab迁移

迁移如同**备份**与恢复的步骤一样, 只需要将老服务器/var/opt/**gitlab**/backups目录下的**备份**文件拷贝到新服务器上的/var/opt/**gitlab**/backups即可(如果你没修改过默认**备份**目录的话). 但是需要注意的是新服务器上的**Gitlab**的版本必须与创建**备份**时的**Gitlab**版本号相同. 比如新服务器安装的是最新的7.60版本的**Gitlab**, 那么迁移之前, 最好将老服务器的**Gitlab** 升级为7.60在进行**备份**.

# 安装配置gitblit：