# linux指令

|  |
| --- |
| RMP 是 LINUX 下的一种bai软件的可执行程序，就是Redhat的包管理器，后缀是.rpm。  i表示安装，v表示显示安装过程，h表示显示进度  rpm -ivh jenkins-2.235.5-1.2.noarch.rpm 安装软件  rpm -Uvh jenkins-2.235.5-1.2.noarch.rpm 升级软件  rpm -e \*.rpm 反安装  rpm -qpi \*.rpm 查询软件包的详细信息  rpm -qf \*.rpm 包名 查询某个文件是属于那个rpm包  rpm -qpl \*.rpm 查该软件包会向系统里面写入哪些文件  rpm -qa | grep XXXX(moudle name) 查看某个包是否被安装  rpm –help  RPM常用命令：  -i --import 安装软件包 是导bai入密钥，这密钥是红帽公du司提供的  --nodeps 不验证软件包的依赖  -v 可视化，提供更多的详细信息的输出  -h 显示安装进度  另外的常用的附带参数为：  --force 强制安装，即使覆盖其他包的文件也要安装  -a 查询所有已经安装的软件包  -f 查询 文件所属于的软件包  -q 查询软件包（通常用来看下还未安装的软件包）  -l 显示软件包的文件列表  -d 显示被标注为文档的文件列表  -c 显示被标注为配置文件的文件列表 最后这两个用的很少了rpm 包名字结构：  glibc-2.17-196.el7\_4.2.x86\_64  glibc 软件名  -2 主版本号  .17 次版本号  -el7 修订号  x86 CPU架构平台  64 支持系统位数  -el我是哪个平台的，  rpm包有个特点，我是centOS7的rpm包一般只能装到CentOS7的系统里，  一般是CentOS6的软件包只能装到CentOS6的系统里。  ================================================  <https://www.cnblogs.com/sui776265233/p/10175878.html>    yum也是一个安装命令，用以解决安装包时的依赖性问题, 一个在线的软件安装命令  例如我们需要安装一个软件A，而软件A-->B-->C-->D.通过yum我们只需要安装A，其他依赖包会自动安装。  查看本地仓库文件  cd /etc/yum.repos.d/  ls  更换centos的软件源 原因：很多http被墙了，在国内无法访问。 1.下载源仓库文件,xxx.repo  curl -o /etc/yum.repos.d/ali.repo http://mirrors.aliyun.com/repo/Centos-7.repo  将repo文件保存到/etc/yum.repos.d/目录中。  让当前库文件生效  yum clean cache  yum make cache  使用yum进行软件包安装卸载  $>yum list //列出所有软件包  $>yum list installed //列出已经安装的软件包  $>yum list installed | grep nano //列出已经安装的软件包  $>yum search nano //在yum的软件源中搜索软件  $>yum remove nano //卸载软件  $>yum -y install nano //直接安装，不需要yes确认.  $>yum list installed | grep nano //查看是否安装了Nano  只想下载不安装  $>mkdir /home/centos/rpms  $echo 以下命令只下载软件，不安装软件  $>sudo yum install --downloadonly //只下载  --downloaddir=/home/centos/rpms //指定下载目录  Wget  $>sudo systemctl disable firewalld.service //"开机自启"禁用  $>sudo systemctl stop firewalld.service //停止防火墙  $>sudo systemctl status firewalld.service //查看防火墙状态  >su root  $>yum search ifconfig  $>yum -y install net-tools //安装网络工具  =================================  -h帮助  -y当安装过程提示选择全部为"yes"  -q不显示安装的过程  yum的一切信息都存储在一个叫yum.reops.d目录下的配置文件中，通常位于/etc/yum.reops.d目录下。  在这个目录下面有很多文件，都是.repo结尾的，repo文件是yum源（也就是软件仓库）的配置文件，通常一个repo文件定义了一个或者多个软件仓库的细节内容，例如我们将从哪里下载需要安装或者升级的软件包，repo文件中的设置内容将被yum读取和应用！  进入/etc/yum.repo.d目录下，  vi CentOS-Base.repo  https://img-blog.csdn.net/20180410090206108  [extras]这个表示的是名称，--->yum的ID，必须唯一，本地有多个yum源的时候，这里必须是唯一的  name=CentOS-$releasever - Base ----->具体的yum源名字，其实相当于对它的描述描述信息 $r releasever你可以使用这个变量参考红帽企业Linux发行版，也就是说表示当前发行版的大版本号。  baseurl是镜像服务器地址，只能写具体的确定地址。下面的例子中，只能有一个baseurl，但里面可以包含多个url  mirrorlist是镜像服务器的地址列表，里面有很多的服务器地址。这里有一个变量$arch，cpu体系，还有一个变量：$basearch，cpu的基本体系组  baseurl和mirrorlist都是指向yum源的地址，不同点是包含地址的多少。你若自己写的话，我们一般只写一个地址，直接用baseurl就行  gpgcheck=0 ---->要不要验证呢，取消验证。1，使用公钥检验rpm的正确性。  gpgcheck若是1将对下载的rpm将进行gpg的校验，校验密钥就是gpgkey，一般自己的yum源是不需要检测的。gpgcheck=0，那么gpgkey就可以不填写  在来看一下yum.conf文件  https://img-blog.csdn.net/20180410090329830  cachedir=/var/cache/yum #yum下载的RPM包的缓存目录  keepcache=0 #缓存是否保存，1保存，0不保存。  debuglevel=2 #调试级别(0-10)，默认为2  logfile=/var/log/yum.log #yum的日志文件所在的位置  1 yum清空缓存列表  yum clean packages 清除缓存目录下的软件包，清空的是(/var/cache/yum)下的缓存  yum clean headers 清除缓存目录下的 headers  yum clean oldheaders 清除缓存目录下旧的 headers  yum clean, yum clean all (= yum clean packages; yum clean oldheaders) 清除缓存目录下的软件包及旧的headers  2 yum显示信息  yum list # yum list显示所有已经安装和可以安装的程序包  这些列表里面的包的来源就是/etc/yum.repo.d。 base docker-ce-stable epel/x86\_64/metalink epel extras rpmforge updates  yum list rpm 显示安装包信息rpm  显示installed ，这里是包名，版本和仓库名  yum list httpd  这里是可获得包，说明还没有安装。  yum info rpm 显示安装包rpm的详细信息  3 yum安装  yum -y install httpd  如果你不加-y他会问你要不要安装。如果你想自己来控制有些包是否安装，这里不要加-y，如果你想自动安装，不进行交互，这里加入-y。这个就是yum 的安装了，非常简单。  安装完之后查询一下  yum list httpd，Installed 说明已经安装好了  4 yum删除  yum remove httpd 删除程序包httpd ，也就是卸载。  yum deplist rpm 查看程序rpm依赖情况    5 yum包的升级  yum check-update 检查可更新的程序  yum update 全部更新，升级所有包，以及升级软件和系统内核，这就是一键升级。他可以更新CentOS的内核到最新版本。  yum update package1 更新指定程序包package1，  yum upgrade package1 升级指定程序包package1  6 yum 解决依赖的原理  YUM 解决依赖关系问题，自动下载软件包。yum是基于C/S架构。C指的是客户端， S指的是服务器，想ftp，http,file、关于yum为什么能解决依赖关系：所有的Yum 源里面都有repodata，它里面是有XML格式文件，里面有说明需要什么包。例如上堂课实验的：mysql-server rpm 需要：perl-DBI ，python，或者是php等等包。  7 yum 组的管理  yum进行安装的时候可以一组一组的进行安装，先来看看有那些组yumgrouplist。可用的组有这么多。  Available Environment Groups:首先是可用的环境分组。  Compute Node 计算节点  Infrastructure Server 基础设施服务器  File and Print Server 文件和打印服务  Cinnamon Desktop Cinnamon桌面  Installed Groups:已安装的组，开发工具  Development Tools  Available Groups:可用的组。  如果我们想以组的方式来安装，加入我想安装安全工具：  输入：yumgroupinstall "Security Tools"  看到这里询问是否要安装，当然是N不安装了。如果想直接安装，不询问的话，这里也在后面加上选项 –y。  yum groupremove group1 删除程序组group1  ==================================  参考  1 安装  yum install 全部安装  yum install package1 安装指定的安装包package1  yum groupinsall group1 安装程序组group1  2 更新和升级  yum update 全部更新  yum update package1 更新指定程序包package1  yum check-update 检查可更新的程序  yum upgrade package1 升级指定程序包package1  yum groupupdate group1 升级程序组group1  3 查找和显示  yum info package1 显示安装包信息package1  yum list 显示所有已经安装和可以安装的程序包  yum list package1 显示指定程序包安装情况package1  yum groupinfo group1 显示程序组group1信息yum search string 根据关键字string查找安装包  4 删除程序  yum remove &#124; erase package1 删除程序包package1  yum groupremove group1 删除程序组group1  yum deplist package1 查看程序package1依赖情况  5 清除缓存  yum clean packages 清除缓存目录下的软件包  yum clean headers 清除缓存目录下的 headers  yum clean oldheaders 清除缓存目录下旧的 headers  yum clean, yum clean all (= yum clean packages; yum clean oldheaders) 清除缓存目录下的软件包及旧的header |

# yum install和yum localinstall的区别

|  |
| --- |
| yum install会去yum仓库查找相应的软件并安装，仓库中的软件都是解决了依赖关系的，而yum localinstall是用来安装本地rpm包的命令，首先rpm包要先下载到本地，然后在本地目录执行yum localinstall \*.rpm，yum会自动搜寻依赖关系并安装  而rmp -i \*.rpm不会自行解决依赖关系，缺少依赖就会报错 |

# Wget 命令详解

|  |
| --- |
| <https://www.jianshu.com/p/2e2ba8ecc22a>  <https://www.cnblogs.com/lxz88/p/6278268.html>  <https://www.cnblogs.com/sx66/p/11887022.html> |

# Contos7 查看ip

|  |
| --- |
| ifconfig |

# centos7账号密码

|  |
| --- |
| root/ww++123123  shipeng/qq++123123 |

# gitlab安装

|  |
| --- |
| **mvbqjaefaaeibbcc**  free -m 查看内存  gitlab  root/ww123456  wget https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el7/gitlab-ce-12.4.2-ce.0.el7.x86\_64.rpm  wget https://mirrors.tuna.tsinghua.edu.cn/gitlab-ce/yum/el8/gitlab-ce-13.2.4-ce.0.el8.x86\_64.rpm  rpm -ivh gitlab-ce-12.4.2-ce.0.el7.x86\_64.rpm  rpm -ivh gitlab-ce-13.2.4-ce.0.el8.x86\_64.rpm  把虚拟机防火墙关 systemctl stop firewalld.service  安装依赖包  yum install -y curl postfix policycoreutils-python openssh-server wget  yum localinstall gitlab-ce-12.3.5-ce.0.el7.x86\_64.rpm –y  vim /etc/gitlab/gitlab.rb  搜索 / external\_url (n下一个)  external\_url = ’<http://gitlab.oldux.com/>’  修改端口  nginx['listen\_port'] = 8888                gitlab-ctl reconfigure  sudo gitlab-ctl start # 启动所有 gitlab 组件； sudo gitlab-ctl stop # 停止所有 gitlab 组件； sudo gitlab-ctl restart # 重启所有 gitlab 组件； sudo gitlab-ctl status # 查看服务状态； sudo gitlab-ctl reconfigure # 启动服务； sudo vim /etc/gitlab/gitlab.rb # 修改默认的配置文件； gitlab-rake gitlab:check SANITIZE=true --trace # 检查gitlab； sudo gitlab-ctl tail # 查看日志；    #这里也是多方总结  git clone https://gitlab.com/xhang/gitlab.git  cd gitlab/  git fetch  git diff origin/10-2-stable origin/10-2-stable-zh > /tmp/10.2.diff  cd /opt/gitlab/embedded/service/gitlab-rails  git apply /tmp/10.2.diff  patch -d/opt/gitlab/embedded/service/gitlab-rails -p1 < 10.2.diff |

# 汉化

|  |
| --- |
| 汉化的版本一定要和gitlab对应，否则汉化后会502  查看服务器上gitlab的版本号  cat /opt/gitlab/embedded/service/gitlab-rails/VERSION  查看汉化包对应的gitlab的版本号，要先解压汉化包  tar xf gitlab-12-3-stable-zh.tar.gz  cat gitlab-12-3-stable-zh/VERSION  \cp -r gitlab-12-3-stable-zh/\* /opt/gitlab/embedded/service/gitlab-rails/  gitlab-ctl start    不用管错误，重启gitlab    还需要设置 |

# gitlab-用户-用户组

|  |
| --- |
|  |

# gitlab提交

|  |
| --- |
| 线创建用户      创建项目    邀请成员    首次拉取或提交代码，必须填写已存在的用户和密码      然后就会自动添加凭证了    ===================================  提交现有代码到该仓库，必须现在现在凭证里设置  =========================================  您可以使用以下选项之一直接在 GitLab 中创建文件。 命令行指引 您还可以按照以下说明从计算机中上传现有文件。 Git 全局设置 git config --global user.name "Administrator"  git config --global user.email "admin@example.com" 创建一个新仓库 git clone http://localhost/devops/demo.git  cd demo  touch README.md  git add README.md  git commit -m "add README"  git push -u origin master 推送现有文件夹 cd existing\_folder  git init  git remote add origin http://localhost/devops/demo.git  git add .  git commit -m "Initial commit"  git push -u origin master 推送现有的 Git 仓库 cd existing\_repo  git remote rename origin old-origin  git remote add origin http://localhost/devops/demo.git  git push -u origin --all  git push -u origin --tags |

# 启用或禁用 GitLab CI/CD

|  |
| --- |
| 在这里插入图片描述 |

# 报错

|  |
| --- |
|  |

# master分支默认是保护分支

|  |
| --- |
| 直接推送到master会报错  remote: GitLab: You are not allowed to push code to protected branches on this project.    新装的gitlab。GitLab一些分支默认被保护，仅仅拥有master级别的用户才能提交到保护分支，而且master分支默认是保护分支，其他用户需要通过合并issue请求来提交上去  解决如下： |

# SSH公钥以及解决使用命令git pull/push 输入密码问题

|  |
| --- |
| 添加ssh是为了解决使用ssh  拉取或推送代码    <https://blog.csdn.net/weixin_45620266/article/details/103571622>  <https://blog.csdn.net/hjtcn/article/details/78995073>  ssh-keygen -t rsa -C "lisi@163.com" |

# gitlab备份和恢复

|  |
| --- |
| 默认存储位置    修改后    ===    =========================    恢复        ==========================  升级 |

# Jenkins

|  |
| --- |
| itcast/123456  sed -i 's/http:\/\/updates.jenkins-ci.org\/download/https:\/\/mirrors.tuna.tsinghua.edu.cn\/jenkins/g' default.json && sed -i 's/http:\/\/www.google.com/https:\/\/www.baidu.com/g' default.json  cd /root  cat /var/lib/jenkins/secrets/initialAdminPassword  systemctl start jenkins  service jenkins restart  wget <https://pkg.jenkins.io/redhat/jenkins-2.235.4-1.1.noarch.rpm>  卸载  rpm -e jenkins #rpm卸载  rpm -ql jenkins #检查是否卸载成功  find / -iname jenkins | xargs -n 1000 rm -rf 彻底删除残留文件  yum -y install policycoreutils openssh-server openssh-clients postfix  systemctl enable sshd && systemctl start sshd  systemctl status sshd  systemctl enable postfix && systemctl start postfix  systemctl start firewalld  systemctl status firewalld  firewall-cmd --add-service=ssh --permanent  firewall-cmd --add-service=http --permanent  firewall-cmd --reload  firewall-cmd --zone=public --add-port=82/tcp --permanent  firewall-cmd --reload |

# 介绍

|  |
| --- |
|  |

# 安装

|  |
| --- |
| yum localinstall jenkins-2.235.5-1.2.noarch.rpm  启动 systemctl start Jenkins    systemctl status jenkins  没安装java |