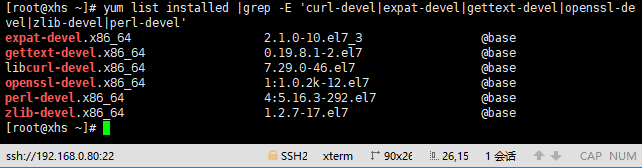
Git安装教程(centos 7)

1. 安装git服务端
2. 检查是否有一些需要的库

yum list installed |grep -E 'curl-devel|expat-devel|gettext-devel|openssl-devel|zlib-devel|perl-devel'



比如上图显示已经安装，没有的话用以下命令安装

yum install curl-devel expat-devel gettext-devel openssl-devel zlib-devel perl-devel

1. 安装git

yum install git

1. 创建一个git用户组和用户，用来运行git服务

groupadd git

useradd git –g git

1. 创建证书登录

cd /home/git

mkdir .ssh

chmod 755 .ssh

touch .ssh/authorized\_keys

chmod 644 .ssh/authorized\_keys

1. 初始化git仓库

cd /home

mkdir gitrepo

chown git:git gitrepo/

cd gitrepo

git init --bare simple.git（创建一个裸仓库）

chown –R git:git simple.git

1. 禁用shell登录

出于安全考虑，第二步创建的git用户不允许登录shell，这可以通过编辑/etc/passwd文件完成。找到类似下面的一行：

git:x:1001:1001:,,,;/homt/git:/bin/bash 改为：

git:x:1001:1001:,,,:/home/git:/usr/bin/git-shell

1. 本地客户端使用服务端克隆这个仓库

git clone [git@192.168.0.80:/home/gitrepo/simple.git](mailto:git@192.168.0.80:/home/gitrepo/simple.git)

出现Cloning into 'simple'...

warning: You appear to have cloned an empty repository.

Checking connectivity... done.

说明复制成功，也说明git服务搭建成功。

（搭建结束）

1. Git公匙和权限管理
2. 利用git工具克隆gitolite仓库

此时当前目录是/root

git clone <https://github.com/sitaramc/gitolite.git>

1. gitolite安装

查看/home/git/下面有没有bin文件夹，没有创建，有的话继续

./gitolite/install –to /home/git/bin/

1. 使用ssh-keygen 生成git管理员公私匙/root/.ssh/id\_rsa.pub

切换用户su git

/home/git/bin/gitolite setup –pk /root/.ssh/id\_rsa.pub

exit 退出git用户

1. 使用命令克隆管理仓库的仓库gitolite-admin

git clone [git@192.168.0.80:gitolite-admin.git](mailto:git@192.168.0.80:gitolite-admin.git)

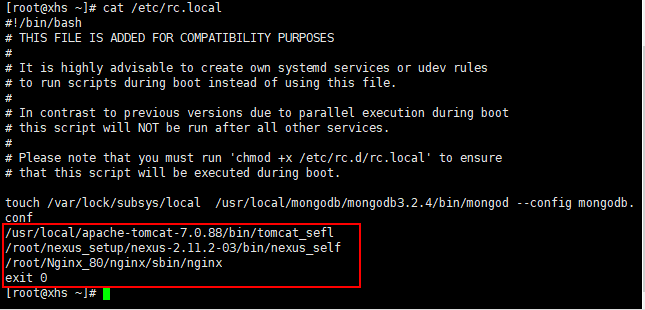
如果一切顺利，能克隆下来，然后就是配置gitolite-admin仓库中的conf/gitolite.conf文件（权限），keydir目录下放的是公匙

1. 配置完成后，记得commit and push,不然配置的东西不生效。需要注意的是，虽然是同一个机器，但也有本地和远程之分，刚才的本地仓库就是克隆下来到/root/gitolite-admin这个文件，远程仓库则是指/home/git/repositories/gitolite-admin这个文件。

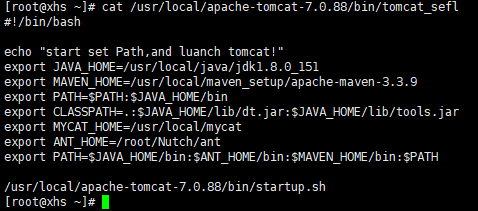
Linux开机自启设置方式

主要有两种方式

1. 一种是写入脚本在/etc/rc.local文件中

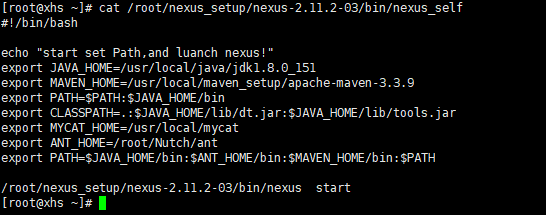


例如上面第一个tomcat\_sefl可执行脚本。如下图所示，需要注意的是需要导入JAVA\_HOME等环境变量，不然启不动。

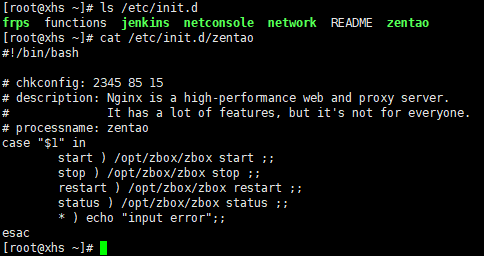


nexus\_self和tomcat启动一样，都需要导入JAVA\_HOME，

最后执行exit 0（表示正常退出）。



1. 制作启动服务，这种可以接受start,stop,restart ,status等命令
2. 在/etc/init.d/下面编写启动脚本



1. 将启动脚本权限改为可执行

chmod a+x /etc/init.d/zentao

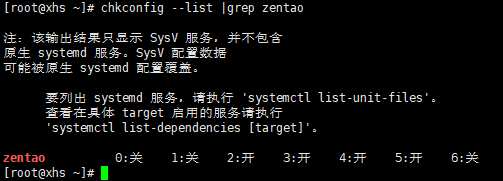
1. 添加启动项

chkconfig --add zentao

chkconfig --zentao on

1. 检查是否设置成功

chkconfig –list |grep zentao



1. 需要注意的是，在这个脚本的编写过程中，其中开头有几个虽然是以注释形式出现的属性，但是要注册服务的话必须拥有，其中chkconfig 2345 85 15 这三个数字表示。2345:表示0-6这七个等级开关（上图最后一行），85表示开启时候的优先级，15表示关闭的时候的优先级，所以开启的时候尽量数字写大点，优先级低点，关闭的时候优先级高点，最先关闭。

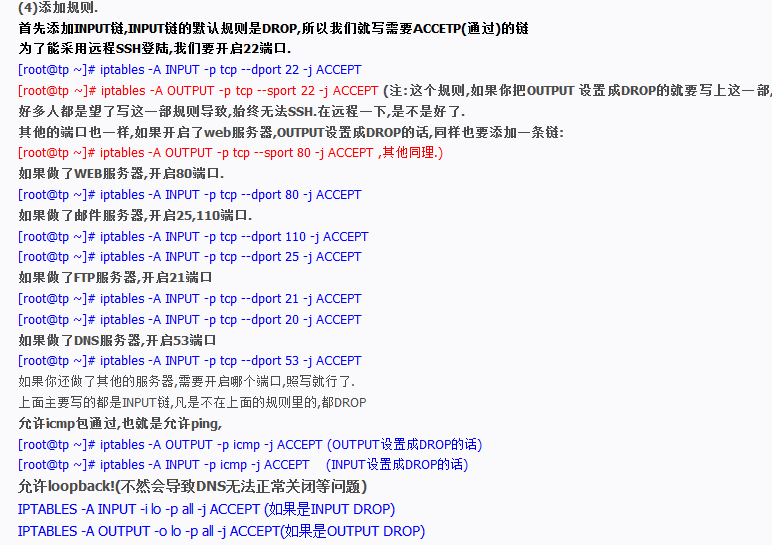
关闭防火墙（开放端口）

1. centos6.5

查看防火墙状态 service iptable status

临时关闭：service iptables stop

永久关闭：chkconfig iptables off



1. centos 7

查看防火墙状态 firewall-cmd –state

或者是 systemctl status firewalld

启动：systemctl start firewalld

停止：systemctl stop firewalld





# Jenkins安装(CentOS6.5)

## 前置条件

1. 安装jdk1.7或以上版本
2. 安装git
3. 安装maven

## 安装

添加Jenkins库到yum库，Jenkins将从这里下载安装。

sudo wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat/jenkins.repo

sudo rpm --import https://jenkins-ci.org/redhat/jenkins-ci.org.key

sudo yum install jenkins

* 安装成功后Jenkins将作为一个守护进程随系统启动
* 系统会创建一个“jenkins”用户来允许这个服务，如果改变服务所有者，同时需要修改/var/log/jenkins, /var/lib/jenkins, 和/var/cache/jenkins的所有者
* 启动的时候将从/etc/sysconfig/jenkins获取配置参数
* 默认情况下，Jenkins运行在8080端口，在浏览器中直接访问该端进行服务配置
* Jenkins的RPM仓库配置被加到/etc/yum.repos.d/jenkins.repo

## 停止/启动

sudo service jenkins start/stop/restart

sudo chkconfig jenkins on

启动报如下错误   
Starting Jenkins bash: /usr/bin/java: No such file or directory   
修改Jenkins启动配置文件，指定java安装路径。

vim /etc/init.d/jenkins

在candidates中第一行添加java路径，如下：

candidates="

/opt/soft/jdk1.8.0\_60/bin/java

/etc/alternatives/java

/usr/lib/jvm/java-1.6.0/bin/java

/usr/lib/jvm/jre-1.6.0/bin/java

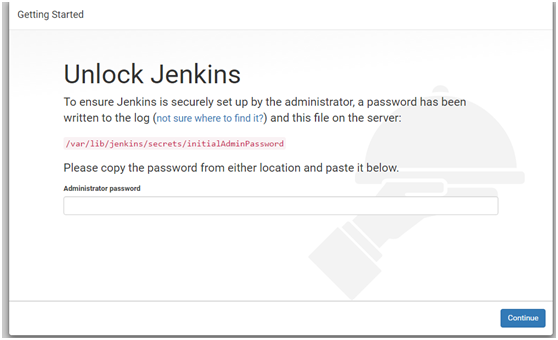
/usr/lib/jvm/java-1.7.0/bin/java

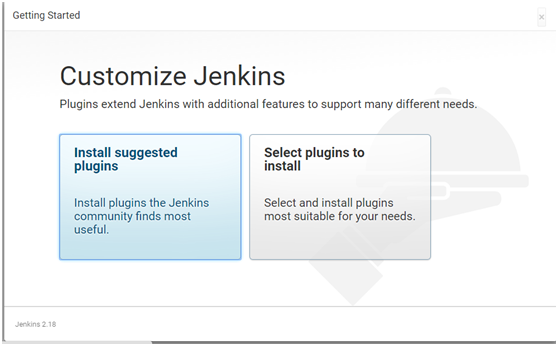
/usr/lib/jvm/jre-1.7.0/bin/java

/usr/lib/jvm/java-1.8.0/bin/java

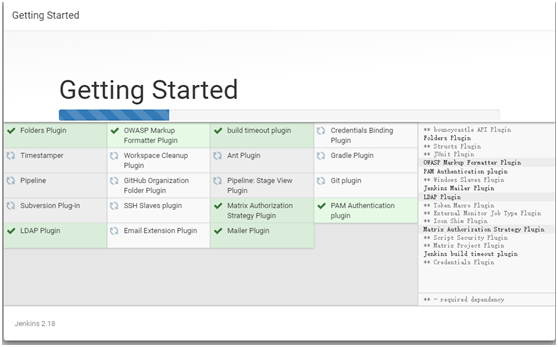
/usr/lib/jvm/jre-1.8.0/bin/java

/usr/bin/java

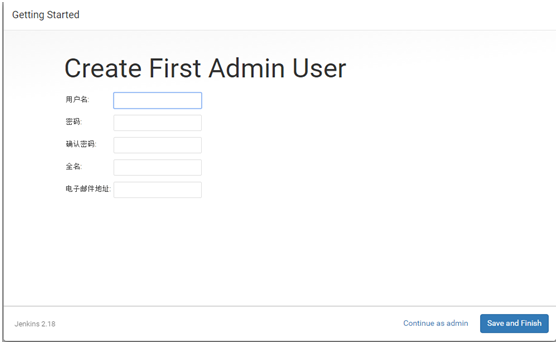
在浏览器中访问   
首次进入会要求输入初始密码如下图，   
初始密码在：/var/lib/jenkins/secrets/initialAdminPassword   


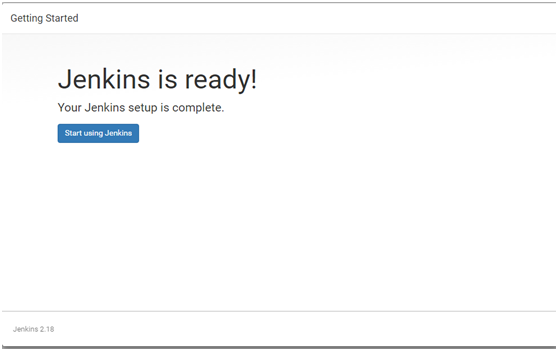


选择Install suggested plugins 安装默认的插件，下面Jenkins就会自己去下载相应的插件安装了



创建超级管理员账号

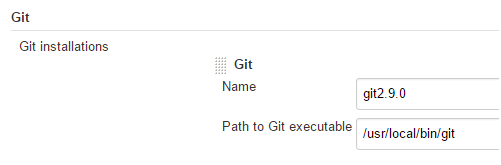




#### 配置

在Jenkins->Global Tool Configuration下配置git。

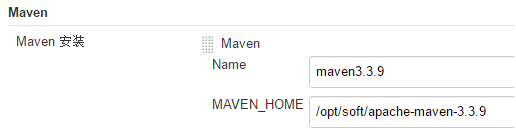
* Path to Git executable：填写git的安装路径



### java配置

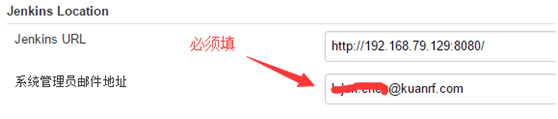
在服务器上执行echo $JAVA\_HOME便可看到java home。   

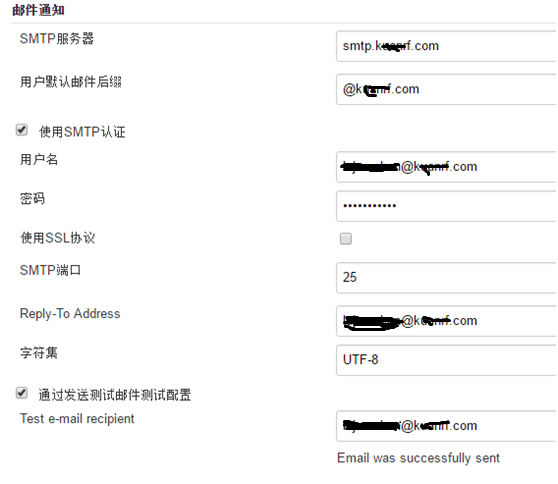

#### 配置



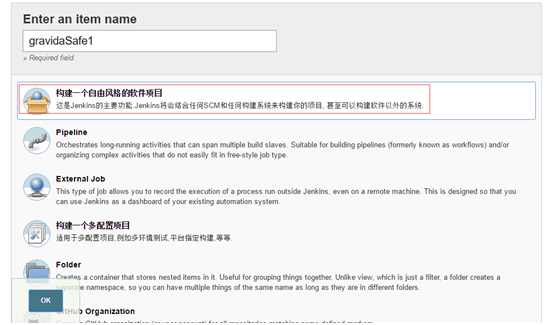
### email配置

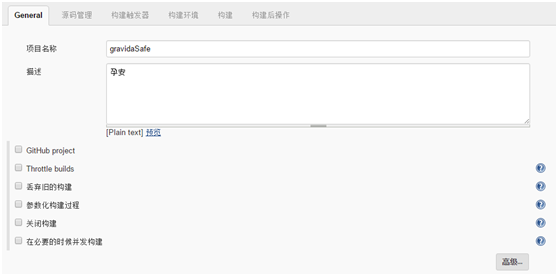
配置邮箱，在构建失败的时候会向指定邮箱发送告知邮件。

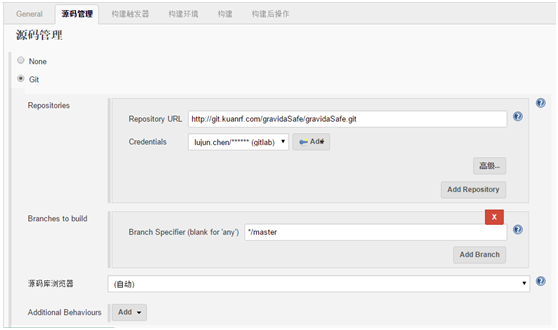


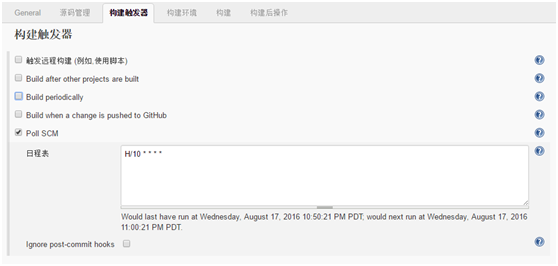


## 构建配置









Poll SCM：定时检查源码变更（根据SCM软件的版本号），如果有更新就checkout最新code下来，然后执行构建动作。我的配置如下：

\*/5 \* \* \* \* （每5分钟检查一次源码变化）

Build periodically：周期进行项目构建（它不管care源码是否发生变化），配置如下：

0 2 \* \* \* （每天2:00 必须build一次源码）





到这里就可以进行构建了，左边菜单有个立刻构建按钮，点击便可以构建。   
第一次构建会有点慢，maven会去下很多插件和jar包。

## 发布项目

### tomcat配置

tomcat-users.xml添加tomcat的manager访问用户，用于发布war包。

<role rolename="manager-script"/>

<user username="clj" password="66YYtre" roles="manager-script"/>

webapps下只保留manager，删除其他文件夹。   
conf\Catalina\localhost下添加manager.xml，配置如下。

<Context privileged="true" antiResourceLocking="false"

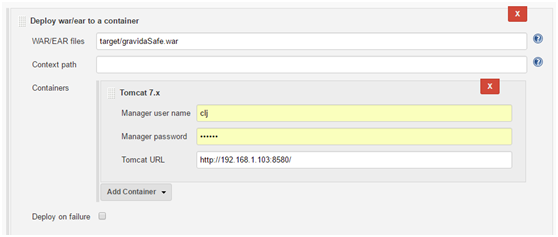
docBase="${catalina.home}/webapps/manager">

<Valve className="org.apache.catalina.valves.RemoteAddrValve" allow="^.\*$" />

</Context>

### 发布到Tomcat

Jenkins安装Deploy to container Plugin插件   
Jenkins配置tomcat服务



在构建的时候，构建成功同时会把构建的war部署到对应tomcat下，同时可以指定多个tomcat，集群化部署的时候很方便。

* [Jenkins官网下载地址](http://pkg.jenkins-ci.org/redhat/)
* [官方教程](https://wiki.jenkins-ci.org/display/JENKINS/Installing+Jenkins+on+Red+Hat+distributions)

# Mysql(5.7)安装(CentOS6.5)

# 压缩版（tar.gz）

安装环境：系统是 centos6.5

1、下载

　　　　下载地址：http://dev.mysql.com/downloads/mysql/5.6.html#downloads

　　　　下载版本：我这里选择的5.6.33，通用版，linux下64位

　　　　也可以直接复制64位的下载地址，通过命令下载：wget <http://dev.mysql.com/get/Downloads/MySQL-5.6/mysql-5.6.33-linux-glibc2.5-x86_64.tar.gz>

//最新找到的5.7版本

wget https://cdn.mysql.com//Downloads/MySQL-5.7/mysql-5.7.22-linux-glibc2.12-x86\_64.tar.gz

2、解压

|  |  |
| --- | --- |
| 1  2  3  4 | #解压  tar -zxvf mysql-5.6.33-linux-glibc2.5-x86\_64.tar.gz  #复制解压后的mysql目录  cp -r mysql-5.6.33-linux-glibc2.5-x86\_64 /usr/local/mysql |

3、添加用户组和用户

|  |  |
| --- | --- |
| 1  2  3  4 | #添加用户组  groupadd mysql  #添加用户mysql 到用户组mysql  useradd -g mysql mysql |

4、安装

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | cd /usr/local/mysql/<br>mkdir ./data/mysql  chown -R mysql:mysql ./  ./scripts/mysql\_install\_db --user=mysql --datadir=/usr/local/mysql/data/mysql  cp support-files/mysql.server /etc/init.d/mysqld  chmod 755 /etc/init.d/mysqld  cp support-files/my-default.cnf /etc/my.cnf    #修改启动脚本  vi /etc/init.d/mysqld    #修改项：  basedir=/usr/local/mysql/  datadir=/usr/local/mysql/data/mysql    #启动服务  service mysqld start    #测试连接  ./mysql/bin/mysql -uroot    #加入环境变量，编辑 /etc/profile，这样可以在任何地方用mysql命令了  export PATH=$PATH:/usr/local/mysql//bin<br>source /etc/profile      #启动mysql  service mysqld start  #关闭mysql  service mysqld stop  #查看运行状态  service mysqld status |

# Linux系统信息

1. **查看内存**

一、free

　　对于free命令，可以加上-m参数，使其显示单位为Mb。如下：



　　这里出现了一些名词，有些是我不知道的，我专门去查了一下它们所代表的含义。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| total | used | free | shared | buffers | cached |
| Mem | 表示物理 内存总量 | 表示总计分配给缓存（包含buffers 与cache ）使用的数量，但其中可能部分缓存并未实际使用 | 未被分配的内存 | 共享内存 | 系统分配但未被使用的buffers 数量 | 系统分配但未被使用的cache 数量 |
| 实际使用的buffers 与cache 总量，也是实际使用的内存总量 | 未被使用的buffers 与cache 和未被分配的内存之和，这就是系统当前实际可用内存 |  |  |  |  |  |
| Swap |  |  |  |  |  |  |

同样是做为缓存，buffers和cache又有啥区别呢？于是又查了些资料，发现buffers实际应该是叫“缓冲”，其英文解释是：A buffer is something that has yet to be "written" to disk.就是说，buffers是将要写入硬盘中的数据缓存。而cache的英文解释：A cache is something that has been "read" from the disk and stored for later use.

就是说，cache是从硬盘或者存储设备读进计算机后缓存在内存中的数据。

　　想起看《计算机组成原理》的时候，有看到因特尔的cpu与内存之间有cache，这之间好像有什么区别的样子。

继续查资料，发现了page cache 和 buffer cache。简单说来，page cache用来缓存文件数据，buffer cache用来缓存磁盘数据。在有文件系统的情况下，对文件操作，那么数据会缓存到page cache，如果直接采用dd等工具对磁盘进行读写，那么数据会缓存到buffer cache.

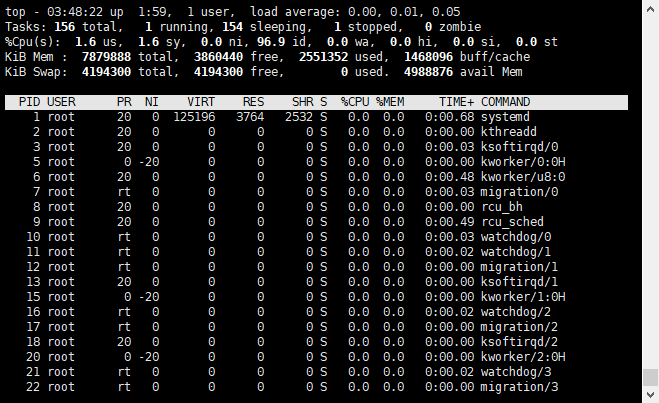
　　对于cache和buffer，cached是cpu与内存间的，buffer是内存与磁盘间的，都是为了解决速度不对等的问题。

　　在free中，buffer 是buffer cache的内存，是块设备的读写缓冲区；cache是page cache的内存， 文件系统的cache。

这里还有一个名词，Swap，即虚拟内存。当物理内存不够用的时候，就要释放掉一部分空间，以供当前运行程序使用。这些被释放掉的内存的数据就被临时保存在虚拟内存中。但是，并非所有的数据都全被保存，只有那些使用malloc或new生成的对象，他们在文件中并没有相应的“储备”文件，被称为匿名内存数据，需要临时地存进Swap。故Swap又称匿名数据交换空间。

至于shared，暂时没查到与之相关的资料，只是在某链接里提了一下其不常用。

二、top命令



其中，第一行的内容与uptime命令的结果是一样的

[linuxidc@ubuntu ~]$>uptime

10:05:01 up 34 min, 1 user, load average: 0.00, 0.11, 0.16

　　第二行的名词也比较好理解，就是进程总数与处于R、S、T、Z状态下的进程数：

　　第三行就是CPU的使用情况了，如下：

|  |  |
| --- | --- |
| %Cpu(s) |  |
| us | 用户空间占用CPU百分比 |
| sy | 内核空间占用CPU百分比 |
| ni | 用户进程空间内改变过优先级的进程占用CPU百分比 |
| id | 空闲CPU百分比 |
| wa | 等待输入输出的CPU时间百分比 |
| hi | CPU服务于硬件中断所耗费的时间总额 |
| si | CPU服务软中断所耗费的时间总额 |
| st | Steal Time |

　　第四、五行就是内存的使用情况了，和free命令里相似。

　　然后是列表的含义：

|  |  |
| --- | --- |
| PID | 进程ID |
| USER | 进程所有者 |
| PR | 优先级 |
| NI | nice值，负值表示高优先级，正值表示低优先级 |
| VIRT | 进程使用的虚拟内存总量 |
| RES | 进程使用的、未被换出的物理内存大小 |
| SHR | 共享内存大小 |
| S | 进程状态 |
| %CPU | 上次更新到现在的CPU时间占用百分比 |
| %MEM | 进程使用的物理内存百分比 |
| TIME+ | 进程使用CPU总时间 |
| COMMAND | 命令名、命令行 |

1. **查看cpu信息**

**1.1 查看CPU个数**

# cat /proc/cpuinfo | grep "physical id" | uniq | wc -l

2 \*\*uniq命令：删除重复行;wc –l命令：统计行数\*\*

**1.2 查看CPU核数**

# cat /proc/cpuinfo | grep "cpu cores" | uniq  
cpu cores : 4

**1.3 查看CPU型号**

# cat /proc/cpuinfo | grep 'model name' |uniq  
model name : Intel(R) Xeon(R) CPU E5630 @ 2.53GHz

**总结：**该服务器有2个4核CPU，型号Intel(R) Xeon(R) CPU E5630 @ 2.53GHz

1. **查看硬盘容量**

**df –hl**

-a或--all：包含全部的文件系统；

--block-size=<区块大小>：以指定的区块大小来显示区块数目；

-h或--human-readable：以可读性较高的方式来显示信息；

-H或--si：与-h参数相同，但在计算时是以1000 Bytes为换算单位而非1024 Bytes；

-i或--inodes：显示inode的信息；

-k或--kilobytes：指定区块大小为1024字节；

-l或--local：仅显示本地端的文件系统；

-m或--megabytes：指定区块大小为1048576字节；

--no-[sync](http://man.linuxde.net/sync)：在取得磁盘使用信息前，不要执行sync指令，此为预设值；

-P或--portability：使用POSIX的输出格式；

--sync：在取得磁盘使用信息前，先执行sync指令；

-t<文件系统类型>或--[type](http://man.linuxde.net/type)=<文件系统类型>：仅显示指定文件系统类型的磁盘信息；

-T或--print-type：显示文件系统的类型；

-x<文件系统类型>或--exclude-type=<文件系统类型>：不要显示指定文件系统类型的磁盘信息；

--[help](http://man.linuxde.net/help)：显示帮助；

--version：显示版本信息。

1. **查看文件大小**

**du 命令**

-a或-all 显示目录中个别文件的大小。

-b或-bytes 显示目录或文件大小时，以byte为单位。

-c或--total 除了显示个别目录或文件的大小外，同时也显示所有目录或文件的总和。

-k或--kilobytes 以KB(1024bytes)为单位输出。

-m或--megabytes 以MB为单位输出。

-s或--summarize 仅显示总计，只列出最后加总的值。

-h或--human-readable 以K，M，G为单位，提高信息的可读性。

-x或--one-[file](http://man.linuxde.net/file)-xystem 以一开始处理时的文件系统为准，若遇上其它不同的文件系统目录则略过。

-L<符号链接>或--dereference<符号链接> 显示选项中所指定符号链接的源文件大小。

-S或--separate-[dirs](http://man.linuxde.net/dirs" \o "dirs命令" \t "_blank) 显示个别目录的大小时，并不含其子目录的大小。

-X<文件>或--exclude-from=<文件> 在<文件>指定目录或文件。

--exclude=<目录或文件> 略过指定的目录或文件。

-D或--dereference-args 显示指定符号链接的源文件大小。

-H或--si 与-h参数相同，但是K，M，G是以1000为换算单位。

-l或--count-links 重复计算硬件链接的文件。