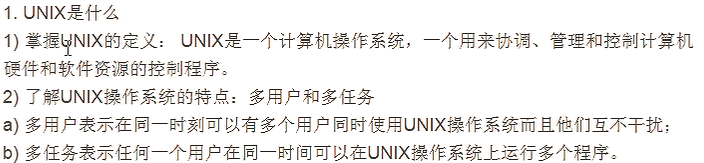


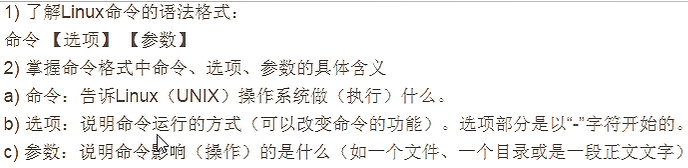


操作系统管理硬件和软件

操作系统由内核和系统调用接口组成，其中内核管理硬件，应用软件调用系统接口

、可移植、可扩展、交互式





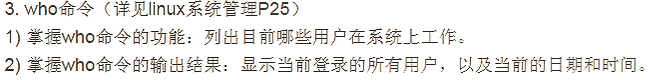
Init 0 关机

Ls（或者l） 显示当前文件所有

Ls -a 显示所有目录，包含隐藏目录

Ls -l（或者ll） 显示当前文件所有，文件的权限信息（列表方式）

Ls -l /root 显示指定文件所有（列表方式）



whoami 查看当前用户是谁

Date 显示系统当前日期和时间

Date ‘月日时分年’ 用于修改当前系统时间

Cal 看日历

eg: cal 8 1997 查看1997年8月

Hwclock -s 同步硬件时间 -w 同步系统时间

清屏 clear

新建用户

1. Useradd jack
2. Passwd jack 为jack添加密码

切换用户

Su jack

查看帮助

1. 命令 --help
2. Man 命令

显示当前目录

Pwd

Type可以补全命令

重要的目录：

Bin目录: 用来存放常用的（二进制）可执行文件

Sbin目录：用来存放系统的可执行文件

家目录：

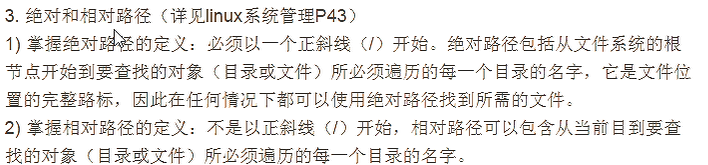
Root用户的家目录：/root/

普通用户的家目录都在：/home/

Dev 目录：设置文件目录

Etc目录：配置文件目录

Tmp目录：临时目录



Cd .. 返回上一级

Cd ~ 切换到家目录

Cd - 切换到根目录

Cd / 切换到根目录

Cd 切换到家目录

Cd - 回到上一次目录

Cp 源文件 目标文件

Cp -i 源文件 目标文件 复制时时若目标已经存在，则提示是否覆盖

Cd -r源文件 目标文件 递归式拷贝，多用于拷贝目录

Cd -p

Cd -f

Mv 源文件 目标文件

Mkdir 新建目录

Touch 新建文件

rm 文件名 删文件

rm -f 文件名 强制删文件（不提示）

Rm -r 文件名 删目录

Rm -rf 文件名 删目录（不提示）

多数情况下我们不删除目录，而是将要删除的文件 mv 源文件 /tmp/ 即保存到临时文件

查看文件：cat 文件名

Head 文件名 默认显示文件前10行，可以显示指定行数如head -2 文件名

显示前两行

tail文件名 默认显示文件后10行，可以显示指定行数如tail -2 文件名

显示最后两行

Tail -f文件名 动态显示文件内容，你增加一行他就显示一行

echo “你的内容” >> 文件名 可以快速把引号里面的内容写到你所指定的文件名里

more 文件名 文件内容查看命令

**用户权限和组：**

**新建用户：**

**新建了用户可通过**Less /etc/ passwd查看记录

所查看的每一行都是一个完整的记录：

root:x:0:0:root:/root:/bin/bash

用户名：密码：用户id:组id:用户描述信息：该用户所在家目录：/bin/bash（可以登录该操作系统）

下面需要注意：

root id为0，可以通过设置该用户的id值来设置root用户；且前1000以内为系统自用，新建用户时id值从1001开始

x表示该用户有密码但不可见具体内容，可以设置为空

/bin/bash/nologin 表示不可登录操作系统

查看用户密码文件 less /etc/shadow

查看组文件 less /etc/group 显示格式为 组名：组密码：组id

查看组密码文件 less /etc/gshadow

家目录

邮箱 /var/spool/mail/用户名

可通过 id 用户名 判断该用户存在不

比如id jack

uid=1001(jack) gid=1001(jack) groups=1001(jack)

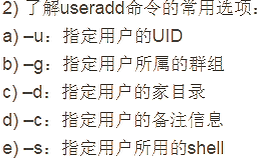
Groups 表示新建jack时创建的组，并把Jack添加进去

新建用户总结：

1. 在/etc/passwd
2. 在/etc/shadow 新增用户密码
3. 在/etc/group 新建组
4. 在/etc/gshadow 新增组密码
5. 在 /home/ 新建家目录
6. 在/var/spool/mail/用户名 新建邮箱（并配置）

因此新建用户可以通过useradd 用户名或者是完成上述步骤

**用户增删改查：**



新建用户

useradd -u 1011 -g jack -d /home/sb -c bigsb -s /bin/bash/ sb

root@iZuf6b37k32ul1j2uzz2dvZ:/# tail -1 /etc/passwd

sb:x:1011:1001:bigsb:/home/sb:/bin/bash/

删除用户

userdel -r 用户

更改用户家目录

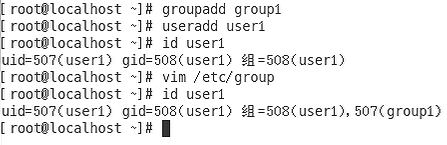
Usermod -d /home/genggai/ sb

更改家目录之后只是一个空文件，需要把原来的配置信息拷贝一下

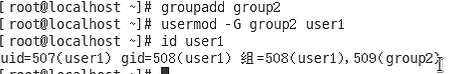
即：cp -r /home/sb/.[!.]\* /home/genggai

更改组：

通过vim /etc/goup 增加信息



通过usermod -G 组名 用户名 意思：通过覆盖的形式



通过usermod -aG 组名 用户名 意思:通过增加的形式（append）



**对文件的权限管理：**

Ls -l

**权限信息** 硬连接数 属主 属组 文件大小 文件创建日期 文件名

如：drwxr-xr-x

10位

第一位： - :文件类型

d :目录

l :软连接

b :(block):设备文件(硬盘)

p :管道文件

2-4：所属主的权限u

读r 4

写w 2

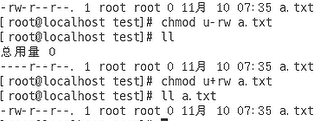
执行x 1

5-7：所属组的权限g

8-10：其他用户的权限o

更改用户的权限：

Chmod



或者 chmod u=xxx 文件 如 chmod u = rwx 其他组修改类似

注意：管理员拥有所有权限，尽管用chmod没赋予他权限，他仍然可以修改，其他用户不行

执行文件方法：  **./ 文件路径**

sh 文件名

. 文件路径

bash文件路径

查找文件路径的命令：whereis 文件名，find 文件名，locate文件名

**对目录的权限管理：**

显示信息：ll-d 路径 如：

root@iZuf6b37k32ul1j2uzz2dvZ:/# ll -d /

drwxr-xr-x 25 root root 4096 Jul 26 18:07 //

读r 即浏览目录的子文件/目录

写w 新建/删除/重命名文件

执行x 可以cd切换进去

注意：在目录里面新建文件时，需要进入该目录才行因此只有w是不够的，需要wx

如果需要新建 /temp/a.txt

那么只对/权限要求为x,只要能切换进来就行，但是想要在temp目录下新建文件，就需要求temp的权限为wx

需要注意:一个用户创建了文件，他把自己的可写权限去除了，但管理员和该用户仍可以修改

jack@iZuf6b37k32ul1j2uzz2dvZ:/tmp$ touch a.txt

jack@iZuf6b37k32ul1j2uzz2dvZ:/tmp$ ls -l

-rw-rw-r-- 1 jack jack 0 Jul 30 15:10 a.txt

jack@iZuf6b37k32ul1j2uzz2dvZ:/tmp$ chmod u-w a.txt

jack@iZuf6b37k32ul1j2uzz2dvZ:/tmp$ ls -l

-r--rw-r-- 1 jack jack 0 Jul 30 15:10 a.txt

jack@iZuf6b37k32ul1j2uzz2dvZ:/tmp$ vim a.txt

jack@iZuf6b37k32ul1j2uzz2dvZ:/tmp$ cat a.txt

11111111111111111111111111

**对属组属主的权限管理：**

chown(change owner)

chown 所属主名.所属组名 文件名如 chown root.root a.txt

只改属主：chown 属主 参数

只改属组：chown .属组 参数

递归修改：chown -R 属主.属组 参数 递归修改改目录下所有子目录和文件的权限

数字权限管理：

r:4 w:2 x:1

chmod 777 a.txt 即为-rwxrwxrwx

**文档合并与文档归档：**

>： 左边的内容覆盖给右面

结果 > 文件（把结果覆盖到文件）

cat /etc/passwd > new\_pass.txt

把/etc/passwd内容覆盖给new\_pass.txt,如果new\_pass.txt不存在，则新建一个文件

echo ： 回写输入内容

echo “hello” > new\_pass.txt

把hello内容覆盖给new\_pass.txt,如果new\_pass.txt不存在，则新建一个文件

>>： 左边的内容追加给右面

结果 >> 文件（把结果追加到文件）

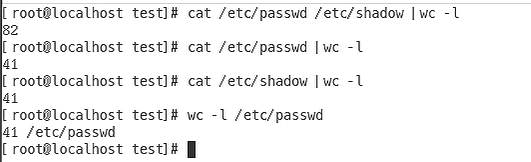
把/etc/passwd内容追加给new\_pass.txt,如果new\_pass.txt不存在，则新建一个文件

......

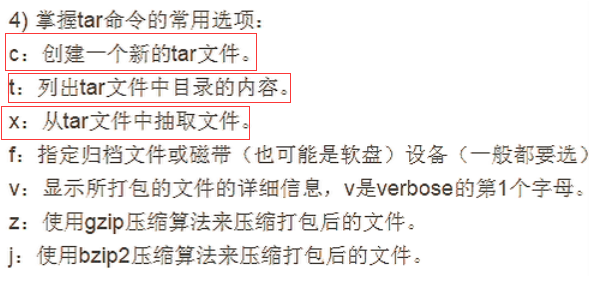
wc -l 文件名 统计文件有多少行

管道符 |

a|b 作用：a命令执行完毕后，把a当做条件执行b命令



归档(把一系列文件打包到一个.tar文件里)：



**打包：**



创建打包文件test.tar，并且把a.txt, b.txt, new\_file.txt,new\_pass.txt四个文件打包到test.tar（v显示详细信息）

**查看打包文件:**

tar -tf test.tar

**解包：**

tar xvf text.tar -C文件位置（可选）

默认把text.tar解压到当前文件，也可以指定解压文件的位置

**文档归档与两种解压方式：**

压缩文件

gzip 文件名 如gzip a.txt 会变成 a.txt.gz

解压文件

gunzip文件名 如 gunzip a.txt.gz 会变成a.txt

压缩文件

bzip2文件名 如gzip a.txt 会变成 a.txt.bz2

解压文件

bunzip2文件名 如 gunzip a.txt.bz2 会变成a.txt

打包并压缩文件(两种情况)

先打包在压缩

tar -cvf a.tar text.txt

gzip 文件名 如gzip a.tar 会变成 a.tar.gz

解压文件

gunzip文件名 如 gunzip a.tar.gz 会变成a.tar

打包压缩一起

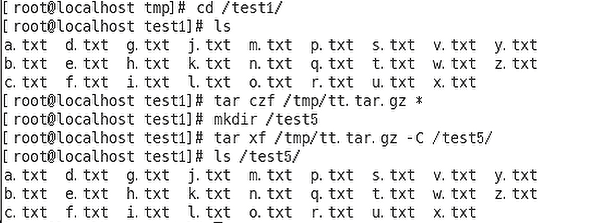
tar czvf a.tar.gz a.txt

解包

tar xvf text.tar -C文件位置（可选）

touch {a..z}.txt

表示创建a.txt到z.txt



**Vim编辑器：**

编辑模式下：

dd: 删除光标所在的一行

u: 撤销命令

ndd： 删除包含光标所在行及其以下共n行

yy： 复制当前行

nyy： 复制包含光标所在行及其以下共n行

p: 把复制的内容粘贴到光标所在的下一行

P: 把复制的内容粘贴到光标所在的上一行

npp: 把复制的内容粘贴n次

dd p: 剪切操作

shift+$: 光标跳到所在行的行尾

0： 光标跳到所在行的行首

gg:  把光标跳到整个内容的首部

ngg(nG): 把光标跳到第n行

G： 把光标跳到整个内容的尾部

H: 光标跳到屏幕的最上方

M: 光标跳到屏幕的最中间

L: 光标跳到屏幕的最下方

set nu: 显示行号(：扩展模式下)

/内容 查找内容

**系统启动流程：**

首先读BIOS，然后决定加载哪个操作系统（启动盘），然后读取磁盘内容（机械手臂从最外圈开始走），然后读取512bytes作为主引导记录（MBR）其中细分（前446：引导信息，接着64分区信息，最后2标志位）MBR告诉我们从该设备的某个分区来装载引导加载程序（可以理解为从我当前盘的哪个分区读操作系统代码即找到装有操作系统的分区）然后找到boot loader,boot loader（或者说是GRUB）负责装载系统内核,如windows，liunx,mac等操作系统（与操作系统硬件打交道），然后内核运行起来,kernel 启动一个init进程，进入操作系统，默为启动init 5 即图形界面

即kernel -> int process

“””

int 0: 关机

int 1: 单用户（以单用户模式启动操作系统，身份为root且不用输入密码，只启动 部分服务）

int 2: 多用户模式（没网络的3模式）

int 3: 多用户模式（命令界面）

int 4: 没有使用（留给开发测试用）

int 5: 图形界面

int 6: 重启

”””

单用户实列：

。。

1. 再读秒的时候按任意键（此时处于）
2. 按e(编辑)
3. 选kernel 然后按e 然后按空格进行编辑
4. 输入1回车，然后按b（启动引导），然后自动执行
5. 然后无需输入密码就进入，且为root用户（可用于root用户忘记密码）

**如何做GRUB加密，防止其他人随便进入单用户呢？**

**如下：**

使用 init 数字 切换系统级别（进入操作系统默为启动init 5 即图形界面）

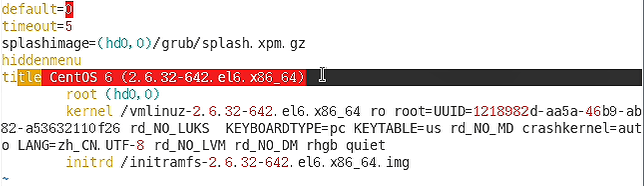
runlevel 显示操作系统处于哪个级别（第一位是上一次所处级别，第二位是当前级别。如： N 5）

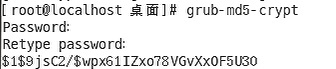
1个扇区的最小单位512bytes

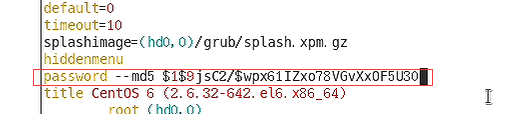
**grub加密**



1. default = 0表示第一个title，timeout为进入系统显示的时间

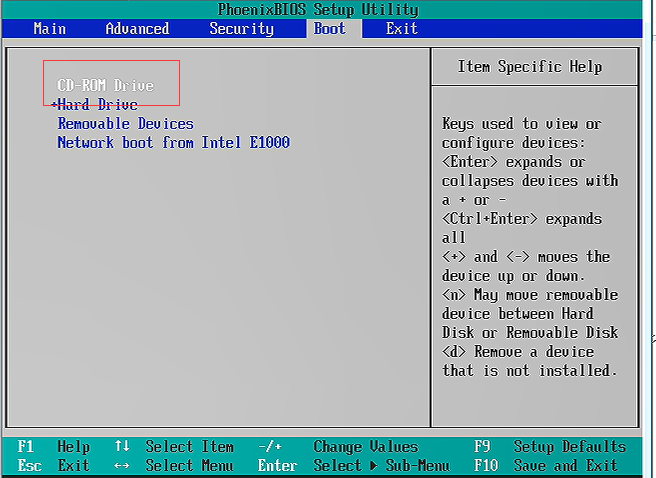


1. 输入password，通过加密后输出结果123456
2. 
3. 在title上添加password字段



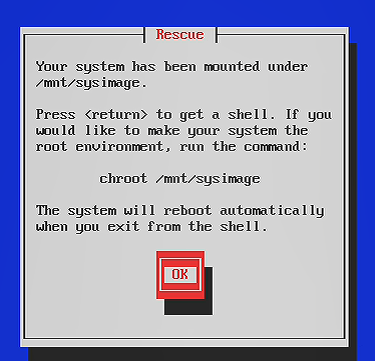
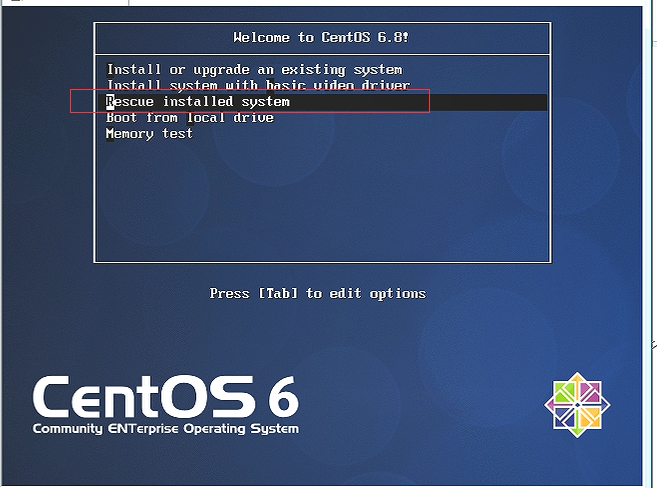
**bios加密**

**通过进入bios来****删除grub加密内容**

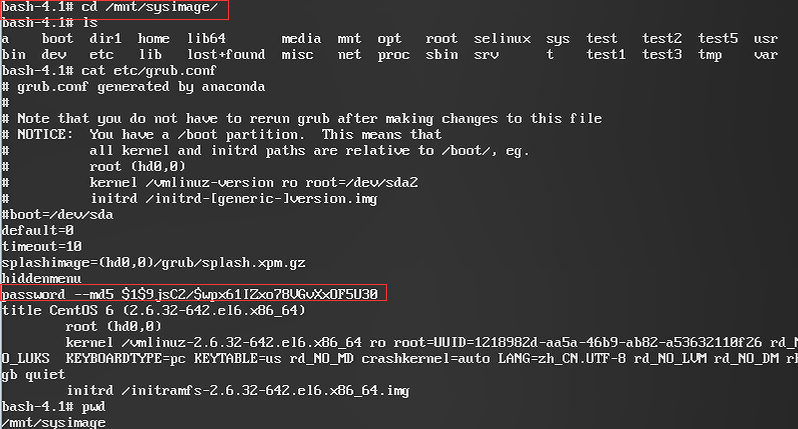


**然后f10保存退出**

**然后进入救援模式**

****

**删除grub加密内容**



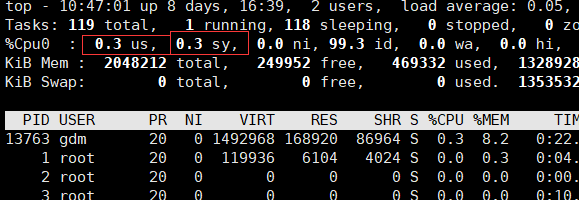
**把主板圆形电池扣下来放电一会儿重置bios**

**操作系统的进程管理：**

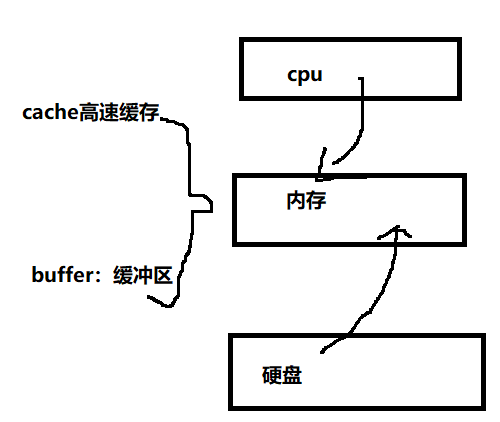
**Top命令**



1. 开一个终端就相当于调用了一个 /bin/bash命令
2. 主进程一死，由主进程生成的子进程会被回收
3. PID:进程号
4. USER:用户
5. SNI：越高，优先级越低
6. us:用户进程——应用程序
7. sy:系统进程——内核运行的程序



1. 缓冲区——把一堆零散的文件攒到一定数目之后一次性的提供给cpu运行，提高效率
2. cache（高速缓存）——为了缓解cpu和内存之间速度差，存放一些cpu经常调用的数据
3. buffer(缓冲区)——为了缓解硬盘和内存之间的速度差
4. cache，buffer——都是内存空间

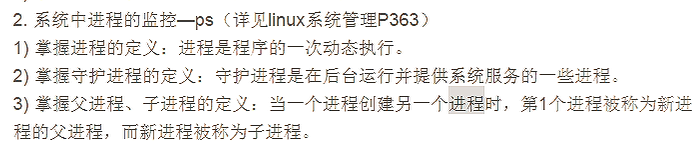


**Free命令**



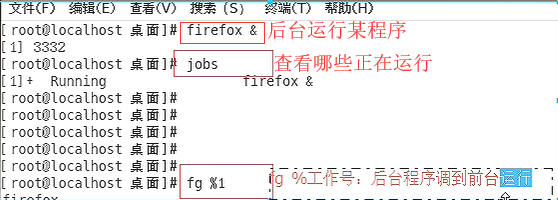
1. free -m:以m为单位显示内存
2. free -h:以G为单位显示内存
3. free :以字节为单位显示内存
4. swap:物理内存够用时不会使用swap

**进程管理（ps）：**



1. ps 用来显示当前进程状态
2. **ps -aus 显示所有的与用户相关的信息**
3. grep(过滤)： grep “txt” 文件名 **找出**文件包含txt的内容
4. 或者与管道符连用：ls /text | grep “txt”
5. grep -v “aaa”： **过滤**掉包含aaa的信息，即不包含aaa的内容
6. ps aux | grep 'init' | grep -v "grep"

**（kill命令）：通过向操作系统发送信号**



上图3523为pid(进程号)

1. kill -9  pid(进程号) 强制杀死某个进程(进程号)
2. kill -9 进程名 强制杀死包含进程名的所有进程
3. kill -9 %1(工作号 ) 强制杀死（jobs查看工作号）
4. fg %1(工作号) 后台调前台运行
5. bg %1(工作号) 后台运行
6. pstree 查看进程树

**磁盘管理：**

硬件级别读取数据为512bytes。

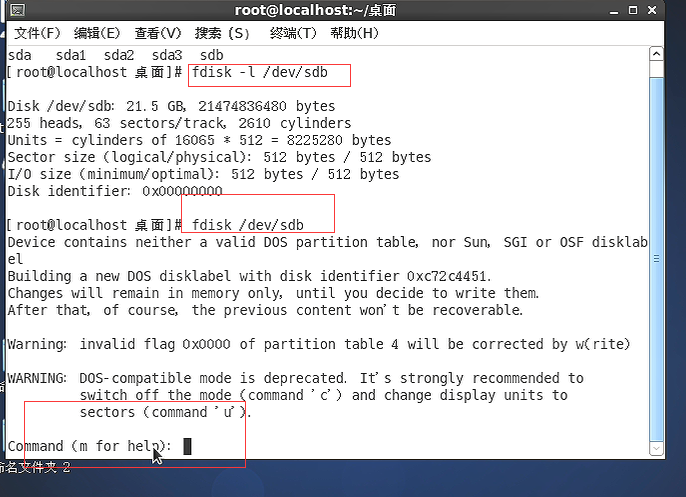
1个block = 8 \* 512bytes(8个扇区大小)

硬盘是硬件，软件控制硬件。操作系统操作文件读写，最后有操作系统通过一个块一个块写（为了提高效率）

块存储： 相当于一块大硬盘

文件存储： 相当于一个文件夹（文件的任何操作）

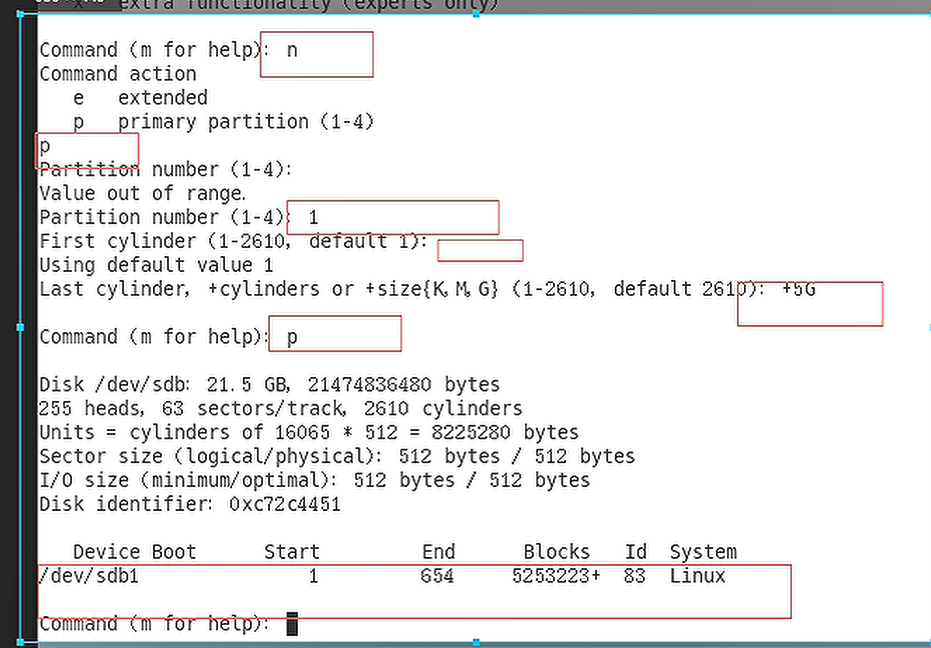
新加磁盘大小20G：



1. 首先分为四个主分区（最多只能用有四个主分区）每个分区5G

512bytes（前446：引导信息，接着64分区信息，最后2标志位）由64字节分区信息决定最多4个主分区

只有主分区可以装系统可以作为启动分区

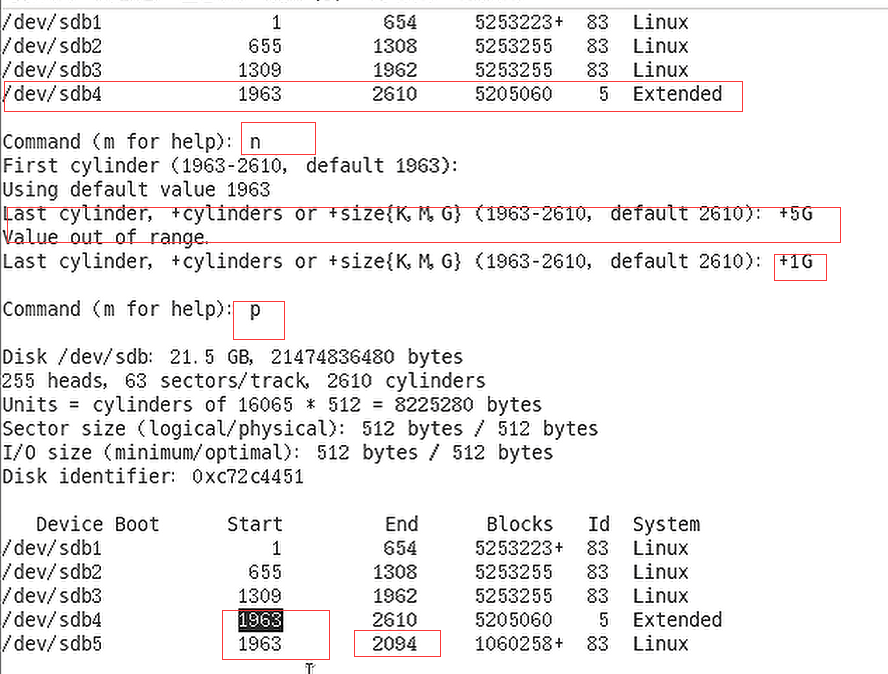


1. 删除第四个分区，并用来建扩展分区

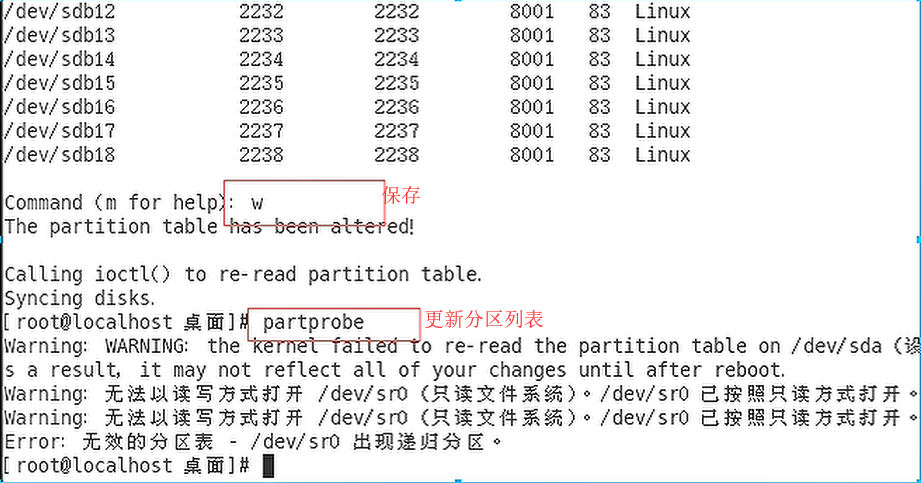
扩展分区：记录扩展分区信息，自己啥也不干



1. 在扩展分区上建一个1G大小的逻辑分区

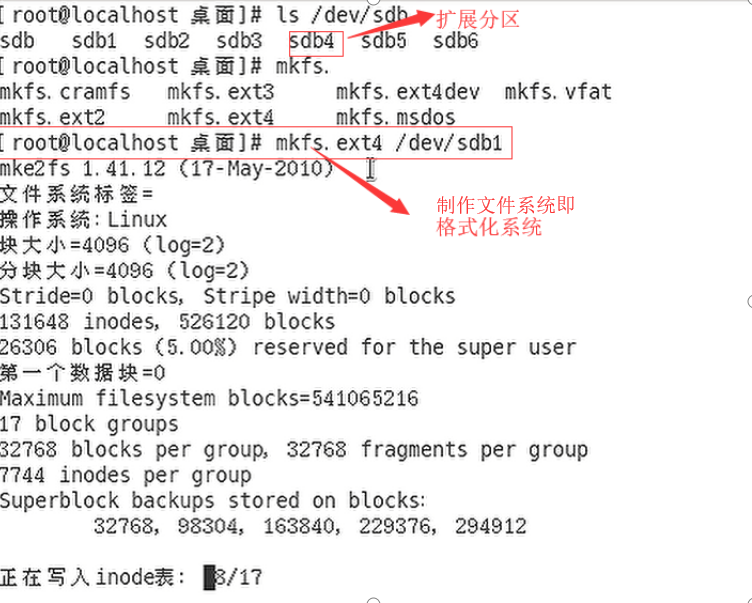


1. 一直新建，并保存



注意扩展分区只能建一个，扩展分区上最多只能建15个

制作系统，linux一般用mkfs.ext4



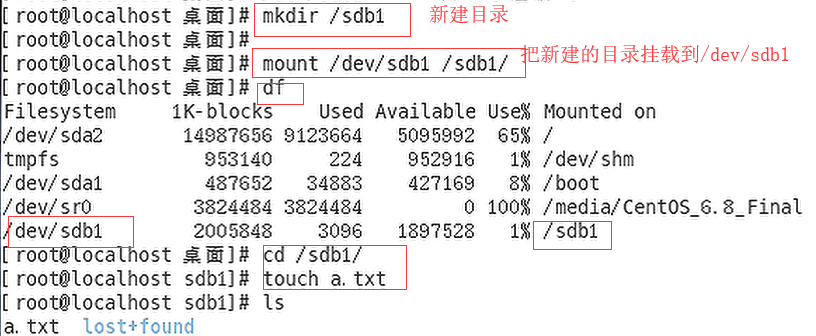


但是扩展分区下面的可以



1. 新建分区，然后格式化分区
2. 我们新建的是分区属于硬件，用户无法操作（我们操作C盘，就是操作里面的目录），我们必须做一个映射关系，由分区映射到具体的目录，以后在目录存内容，就会对应到原来的分区上。这个建立关系的过程叫挂载

如下例子：

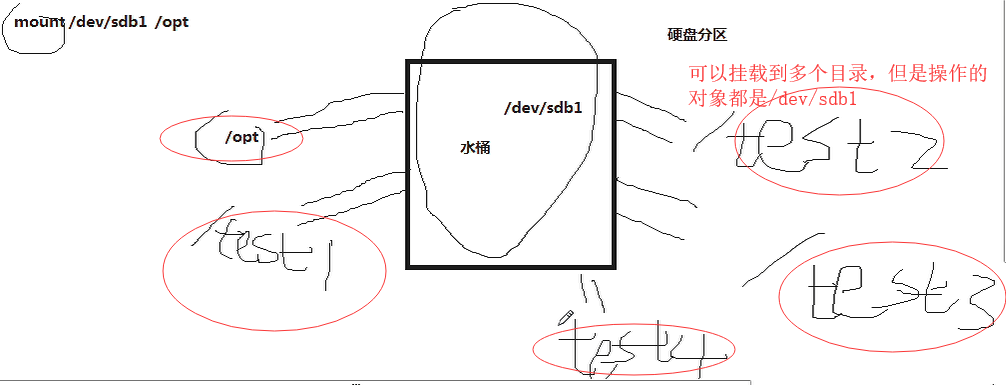


**文件系统与挂载：**

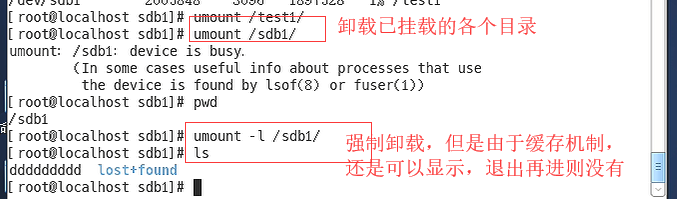
(mkfs.ext4)日志文件系统的特点：

先写到日志中，然后再从日志写到硬盘上。适用于少量数据，如果是大量数据建议用xfs

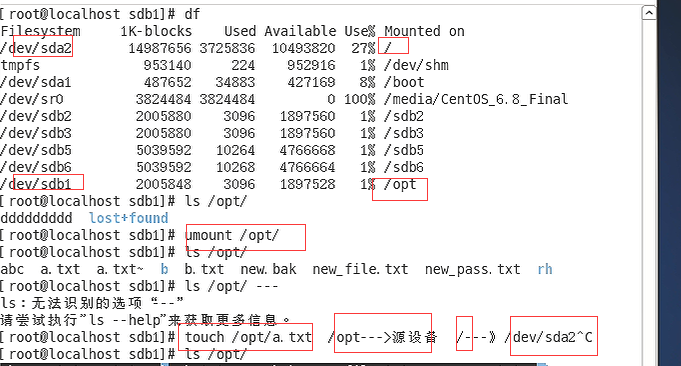
挂载多个目录

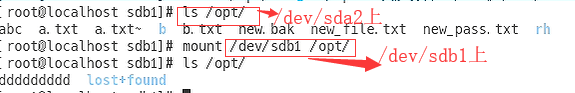


1. 卸载已经挂载的目录（umount 目录）



1. 对于新建目录和浏览目录更深层次理解。





1. 把/opt 目录 通过 umount /opt/ 卸载即/opt 不在指向 /dev/sdb1
2. ls /opt 浏览/opt内容，此时显示的是/dev/sdb2内容
3. 可以通过新建文件理解
4. touch /opt/a.txt 此时会先查找当前目录(/opt)有无源设备，因为通过 umount /opt/已经卸载，所以当前目录没有原设备，因此会查找当前目录的上一级目录即/(根目录)的原设备，发现/目录源设备为/dev/sda2。因此新建的a.txt的原设备就在 /dev/sda2上
5. 我们无法直接操作硬件(源设备)，目录相当于一个入口，我们通过在目录里面新建/删除文件，但实质都是将文件映射到对应的源设备上
6. ls /opt/ 即浏览源设备/dev/sda2/的内容

**总结：**

1. fdisk /dev/sdb 新建分区
2. partprobe 更新分区表信息
3. mkfs.ext4 /dev/sdb3 为/dev/sdb3 格式化制作文件系统
4. mount /dev/sdb3 /new/ 把/new/文件(目录)挂载到/dev/sdb3(设备上)
5. df(mount ) 查看挂载信息
6. cd /new/ 切换之后然后可以在此目录下操作，如新建文件等等

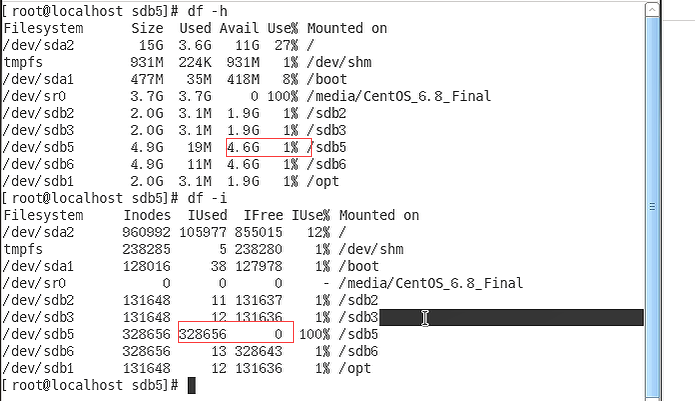
**磁盘用满的两种情况：**

**新建文件通常包含2部分信息：1是节点信息，2是文件内容**

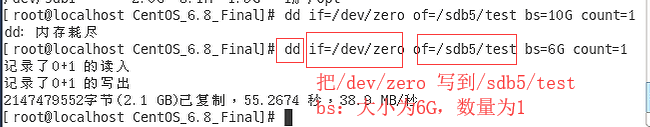
**Inode(i节点)**：存放数据元数据信息（如：通过ls -l 浏览的权限信息），只要一建立就存在，要想新建文件必须要有inode编号

1. **inode节点用完：**

**新建40万个txt空文件,但是在33万左右报设备上没有空间，原因是inode节点用完**



1. **硬盘空间满了（inode还有）：**



**ls -i 文件 查看文件inode号**

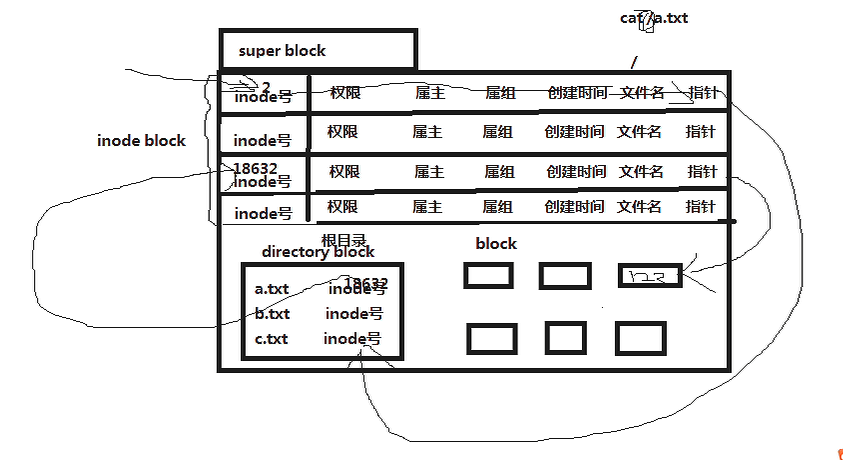
**文件系统**

**super block:**掌管全局，控制哪些编号可以使用以及哪些编号处于活跃状态，分配inode block

**directory block:**目录下有哪些文件名，以及文件对应的编号

**Inode block:该文件元数据信息等（1个block对应8个扇区）**

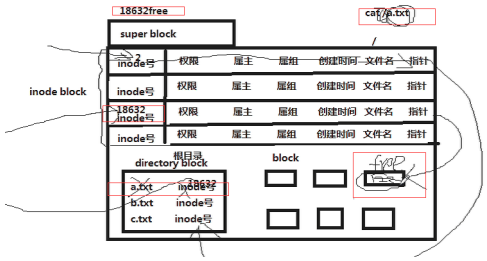
**如果：cat /a.txt,是如何寻址的呢？（操作系统是不认识文件名的，是通过文件的inode，找到对应的block的）**



1. ls -di / 查看根目录的inode(结果为2)
2. / （根）通过指针找到directory block 查看该文件是否存在
3. 如果存在,再通过该文件的inode找到改文件，在查看该文件的权限信息
4. 如果有权限, 则通过指针找到改block

**删除cat /a.txt,是怎样的呢？**

1. 找到文件位置18632，标记且设置为free,表示这个inode编号可以分配
2. 找到目录位置，去掉a.txt与inode的对应关系
3. 通过对应关系标记该文件block块，设置为free



**因此数据并没有真正的被删除（数据真实存在于block块中）：**

数据怎么恢复回来?

1. 把18632编号不能其他分配，且重新设置为忙碌的状态
2. 找到目录位置，添加a.txt与inode的对应关系
3. 通过对应关系标记该文件block块，去掉free状态
4. 数据恢复指的是重新建立关系的过程
5. 真正的数据删除往往是覆盖，即占用inode然后与block对应上关系,覆盖以前数据
6. 因此数据恢复往往是有概率的

**硬链接和软链接：**

**软链接（快捷方式）：**

ln -s 源路径 目标路径（快捷方式名）

修改链接文件，源文件也会发生改变。修改源文件，链接文件也会发生改变，但是删除快捷方式对源文件没影响，删除源文件快捷方式将失效。

**软链接源文件的inode号和目标文件的inode号不是一样的**

**软链接可以跨分区（软链接指向源文件名的，再由源文件指向inode）**

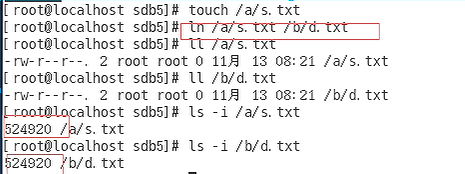
**硬链接：**

ln 源路径 目标路径（快捷方式名）

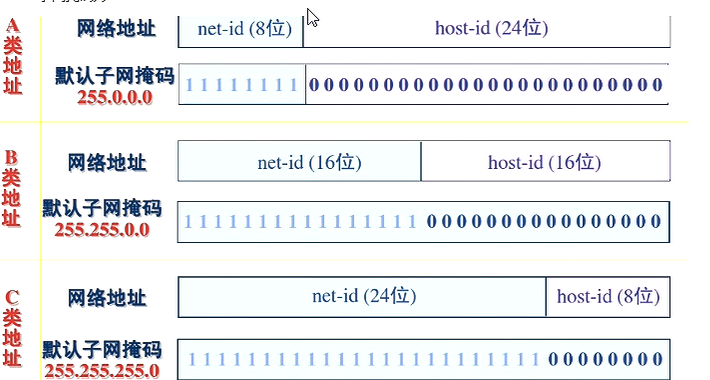
**硬连接源文件的inode号和目标文件的inode号是一样的**

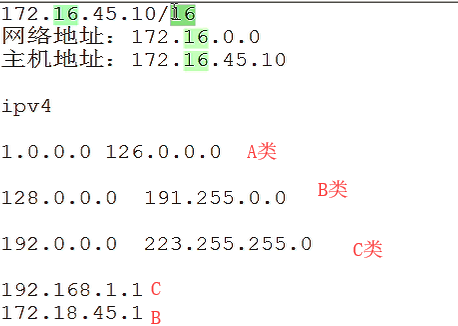
修改链接文件，源文件也会发生改变。修改源文件，链接文件也会发生改变，但是删除快捷方式对源文件没影响，删除源文件对链接文件也无影响。

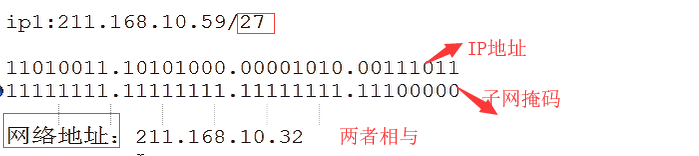
**硬链接不可以跨分区（硬链接直接指向inode，一个inode唯一定位一个资源,因此此时的inode虽然相同但属于不同的分区）**

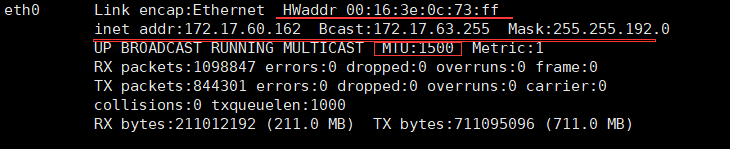


**ip地址与子网划分：**









MTU：网卡最大传输单元（字节）

HWaddr:MAC地址

inet addr: 网卡ip地址

Bcast：广播地址

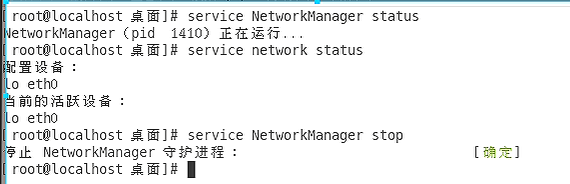
Mask：子网掩码

配置网络地址：ipconfig eth0 192.168.10.100 netmask 255.255.255.0

ipconfig eth0 192.168.10.100/24

1. 配置网卡vim/etc/sysconfig/network-scripts/ifcfg-eth0

BOOTPROTO:dhcp 动态获取ip地址，下面的就不用自己配置

1. 关掉守护进程
2. 重启网络服务： service network restart 或者 ifup eth0

chkconfig --list

左边服务名 右边是启动级别

chkconfig --level 2345 NetworlkManager off

把在2345下的状态设置为关闭（init 2345）

### 

### route-n 查看路由 cat /etc/resolv.conf 查看dns

**虚拟机网络模式：**

1. 桥连接：比如电脑里面的虚拟机，和本地电脑之间
2. NAT模式：私网ip通过路由器转化为公网ip
3. 单机模式

临时配ip: 

永久配置ip: vim/etc/sysconfig/network-scripts/ifcfg-eth0

**软件包：**

xxx.tar.gz: 编译安装（把你写的字符串编译成为计算机认识的二进制语言） 先编译在安装

RPM（红帽软件包管理公工具）

rpm包： 本身就是二进制 直接安装

一般来说奇数版本是测试版，偶数是稳定版

安装

rpm -ivh 软件包名 i表示install

rpm -ivh 软件包名 --force 强制安装，存在则覆盖

查看

rpm -qa 显示所有安装的 q表示query a为all

删除（不能使用--force）

rpm -e 软件包名 erase表示抹去

rpm -e 软件包名 --nodeps 忽略依赖的删除

关闭防火墙：

基于dns解析：

先找本地的/etc/hosts如果没有则找/etc/resolv.conf (dns)

修改主机名：

hostname jack

在/etc/hosts 在下添加主机记录即：静态ip jack

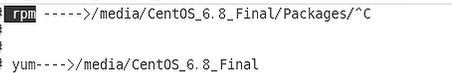
直接在网上安装软件包： rpm -ivh <http://...............>

在网上下载软件包： wget <http://...............>然后

rpm -ivh 软件包

查看文件安装位置： rpm -ql 文件名（l为location）

查看文件安装信息： rpm -qi 文件名（l为 information）



**rpm不能找到依赖关系，需要指定到/packages目录下**

yum erase 包名 卸载包

yum makecache 做好缓存，下次快速安装

重新安装包



**yum(python写的):**

yum—>安装源—>在repodata读取依赖关系—>找到所需要的软件包—>然后安装

1. **本地源安装：**

编辑 vim /etc/yum.repos.d/local.repo

[local] (id)

name=local yum (名字)

# 使用本地源就要使用文件协议file://+（yum源地址）

baseurl=file:///media/Centos\_6.8\_final（本地光盘） enabled=1 (开启)

gpgcheck=0

通过yum可以自动安装依赖包，它会先读取

/etc/yum.repos.d/local.repo里面的配置，然后安装

可以手动指定本地的安装源

yum clean all 清理缓存

需要安装createrepo来为test目录生成依赖关系

1. **编译安装（python）**

**解包—>运行configure脚本添加编译参数—>编译（依赖gcc glibc等库）make—>makeinstall**

**./configure --prefix=/usr/local/python3.6**

1. **下载源码：**

wget [https://www.python.org/ftp/python/3.6.4/](https://www.python.org/ftp/python/3.6.4/Python-3.6.4.tgz)[Python-3.6.4.tgz](https://www.python.org/ftp/python/3.6.4/Python-3.6.4.tgz)

1. **安装依赖**

yum install openssl-devel bzip2-devel expat-devel gdbm-devel readline-devel sqlite-devel

1. **解包：**

tar -xvf Python-3.6.4.tgz

1. **cd** Python-3.6.4
2. **./configure** --prefix=/usr/local/python3.6 (指定安装到/usr/local/python3.6下)
3. **编译make**
4. **make install**

配置PATH：

**在linux中,一切皆文件**借助which ls 显示/usr/bin/ls 可知要想通过python3.6直接运行python则需要像ls命令一样做些配置

即对应关系为ls --------------------PSTH------------------------/usr/bin/ls

1. （短暂修改）**本地的PYTH变量**

读取变量的值：echo $PYTH /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin

1. 添加值在pyth后:

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/usr/local/python3.6/bin

1. **（永久修改）修改到配置文件里面： vim /etc/profile** 末尾添加如下

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin:/usr/local/python3.6/bin

export=PATH （expor表示为了让PATH生效）

1. **SSH服务：**

服务器：下载openssh-server

客户端：下载openssh-client（如ssh）

1. 连接ssh: ssh ip的地址 -p 22
2. 查看ssh配置 vi /etc/ssh/sshd\_config
3. 重启服务 restart sshd service

搭建服务：

修改配置之后，想要生效必须重启

查看防火墙状态 service iptables status

1. **简单的apache服务：**
2. **准备环境**

centos6关闭防火墙：service iptables stop(临时修改)

chfconfig iptables off(开机就关闭防火墙)

centos7关闭防火墙：systemctl restart httpd（一步到位）

systemctl命令是将service和chkconfig命令结合在了一起

service iptables status 查看防火墙状态

关闭selinux: setenforce 0(临时修改)

vim etc/sysconfig/selinux(永久设为disabled）

1. **安装软件包**

yum install httpd -y(apache)

1. **修改配置文件**
2. vim etc/httpd/conf/httpd.conf

端口：80

默认项目根目录：DocumentRoot "/var/www/html"访问的时候直接到这里，因此可以不写这个前缀

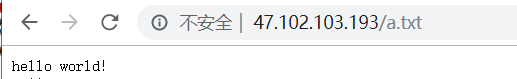
1. **重启服务**
2. **测试**

自己写内容：

1. 在/var/www/html文件下新建a.txt

比如写入helloworld

1. 在浏览器访问服务的IP地址，结果如下：



系统默认显示

1. 系统默认显示/var/www/html里面的indx.html，如果你没有指定显示内容，系统则默认显示index.html内容

比如在/var/www/html建一个index.html文件，如果你没有像上图一样指定（a.txt），则浏览器默认显示index.html内容

1. **Centos 6 和centos 7的区别**

**(2)文件系统**  
[CentOS6] ext4  
[CentOS7] xfs

**(3)内核版本**  
[CentOS6] 2.6.x-x  
[CentOS7] 3.10.x-x

**(5)防火墙**  
[CentOS6] iptables  
[CentOS7] firewalld

**(6)默认数据库**  
[CentOS6] MySQL  
[CentOS7] MariaDB  
**(7)文件结构**  
[CentOS6] /bin, /sbin, /lib, and /lib64在/下  
[CentOS7] /bin, /sbin, /lib, and /lib64移到/usr下  
**(8)主机名**  
[CentOS6] /etc/sysconfig/network  
[CentOS7] /etc/hostname

**(12)服务相关**

[CentOS6]  
 $ service service\_name start  
 $ service service\_name stop  
 $ service sshd restart/status/reload  
  
[CentOS7]  
 $ systemctl start service\_name  
 $ systemctl stop service\_name  
 $ systemctl restart/status/reload sshd

**自启动**  
[CentOS6]  
 $ chkconfig service\_name on/off  
  
[CentOS7]  
 $ systemctl enable service\_name  
 $ systemctl disable service\_name  
**服务一览**  
[CentOS6]  
 $ chkconfig --list  
  
[CentOS7]  
 $ systemctl list-unit-files  
 $ systemctl --type service  
**强制停止**  
[CentOS6]  
 $ kill -9 <PID>  
  
[CentOS7]  
 $ systemctl kill --signal=9 sshd

**1）网络信息**  
[CentOS6]  
 $ netstat  
 $ netstat -I  
 $ netstat -n  
  
[CentOS7]  
 $ ip n  
 $ ip -s l  
 $ ss  
  
**2）IP地址MAC地址**  
[CentOS6]  
 $ ifconfig -a  
  
[CentOS7]  
 $ ip address show  
  
**3）路由**  
[CentOS6]  
 $ route -n  
 $ route -A inet6 -n  
  
[CentOS7]  
 $ ip route show  
 $ ip -6 route show  
  
**(14)重启关闭  
  
1）关闭**  
[CentOS6]  
 $ shutdown -h now   
  
[CentOS7]  
 $ poweroff  
 $ systemctl poweroff  
  
**2）重启**  
[CentOS6]  
 $ reboot  
 $ shutdown -r now  
  
[CentOS7]  
 $ reboot  
 $ systemctl reboot  
  
**3）单用户模式**  
[CentOS6]  
 $ init S  
[CentOS7]  
 $ systemctl rescue  
  
**4）启动模式**  
[CentOS6]  
[GUICUI]  
$ vim /etc/inittab  
  id:3:initdefault:  
[CUIGUI]  
$ startx  
  
[CentOS7]  
[GUICUI]  
$ systemctl isolate multi-user.target  
[CUIGUI]  
$systemctl isolate graphical.target  
**默认**  
$ systemctl set-default graphical.target  
$ systemctl set-default multi-user.target  
**当前**$ systemctl get-default