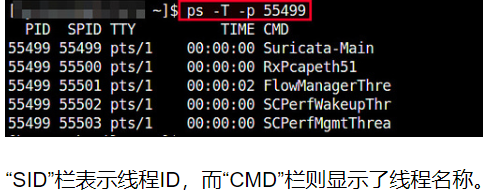
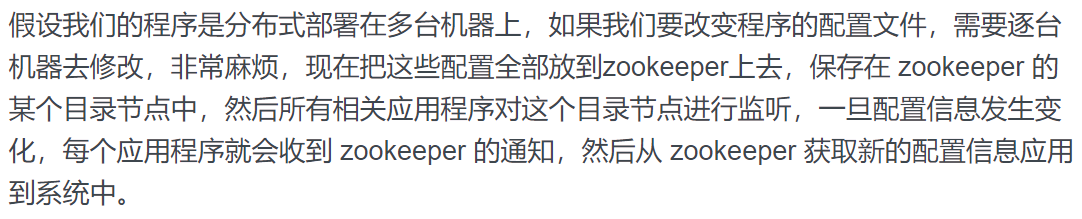
Ps -t -p pid

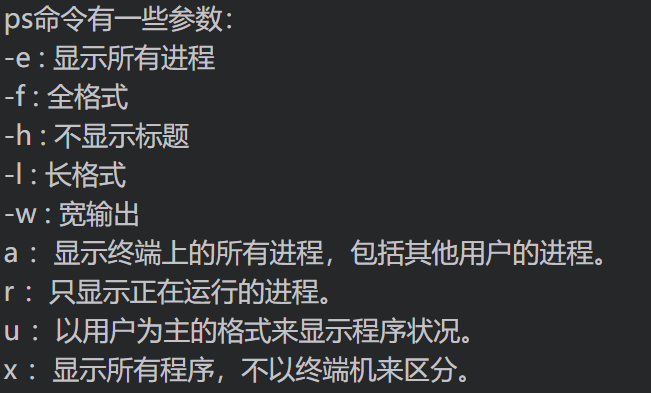






ps -ef表示查看全格式的全部进程

PS parament



Readlink -f 追溯符号链接所指向的位置

A=`ls -l`

把ls -l的结果赋给A ls -l 作为命令来执行。

$0

Shell本身的文件名

当无法确定当前所在目录时，可以使用以下方式：

#!/bin/sh

DIR=$(cd $(dirname $0) && pwd)

表示的意思是将当前目录赋给DIR，并显示出当前路径

绿色文件---------- 可执行文件，可执行的程序

红色文件-----------压缩文件或者包文件

蓝色文件----------目录

白色文件----------普通，如文本文件，配置文件，源码文件等

浅蓝色文件----------链接文件，主要是使用ln命令建立的文件

红色闪烁----------表示链接的文件有问题

黄色文件----------表示设备文件

灰色文件----------表示其它文件

系统文件目录：

**/bin 存放经常使用的命令**

/boot 存放启动时的核心文件，连接文件和镜像文件

/dev 访问设备

**/etc 系统管理所需要的配置文件和子目录**

/home 用户主目录

/lib 动态链接共享库

/lost+found 当非法关机后，存放的文件

/proc 系统内存的映射，访问这个目录来获取系统信息

**/var 程序日志存放处**

/usr 用户应用程序和文件存放

/usr/bin 系统用户使用的用户程序 /sbin 比较高级的管理程序和系统守护程序

/usr/src 内核源代码默认的放置目录

YAML（Yet Another Markup Language）（发音 /ˈjæməl/ ）

一种基于Unicode容易阅读，容易和脚本语言交互的，用来表达资料序列的编程语言。

curl -X POST --header"Content-Type:application/json" --data ‘{}’ url (使用post模拟json格式请求接口)

**curl -H "Content-Type:application/json" -X POST --data** "{\"token\": \"IhcjQdkKgbVsV6jRnh\", \"timestamp\": \"20180828104900\", \"bd\": \"xx\", \"acc\": \"1.0\", \"extra\": \"xxxx\"}" **<http://127.0.0.1:8080/ci>**

Curl -X POST -H ‘content-type: application/xml’ -d **@/apps/myxmlfile.txt <http://172.19.219.xx:8081/csp/faq/actDiaUserInfo.action>**

**Curl -X POST -H ‘content-type: application/json’ -d @/apps/myjsonfile.txt <http://192.168.129.xx/AntiRushServer/api/ActivityAntiRush>**

**curl -H 'content-type: application/json' -X POST -d** '{"accountType":"4","channel":"1","channelId":"YW\_MMY","uid":"13154897541","phoneNumber":"13154897541","loginSource":"3","loginType":"1","userIp":"192.168.2.3","postTime":"14633fffffffffff81286","userAgent":"Windows NT","imei":"352600051025733","macAddress":"40:92:d4:cb:46:43","serialNumber":"123"}' http://192.168.129.xx/AntiRushServer/api/ActivityAntiRush

**Tee -a -i file1 file2.... 输入字符串，将添加到file12中**

**<< EOF**

**...**

**EOF**

**其中的"..."表示用户输入内容。**

**创建一个新的组，并添加组 ID。**

**groupadd －g 344 runoob**

创建gz1用户，该用户组为group1 ，并在/home目录下自动创建gz1目录（-m）

useradd -m -g group1 gz1

Chown user:usergroup mynotes修改文件拥有者

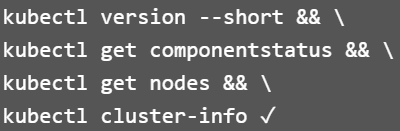
将当前前目录下的所有文件与子目录的拥有者皆设为 runoob，群体的使用者 runoobgroup:

chown -R runoob:runoobgroup \*

Docker容器将帮助你隔离和打包你的软件及其所有依赖项。

Kubernetes将帮助你部署和协调你的容器。

&& \ 前一个命令正常执行才会执行后一个命令



apt-get属于ubuntu、Debian的包管理工具

1 常见的安装包格式 deb包,安装deb包的命令是“dpkg -参数”

2 包管理工具 apt-get

sudo apt-get install –y 省略确认

**yum则属于Redhat、Centos包管理工具**

**1 常见的安装包格式 rpm包,安装rpm包的命令是“rpm -参数”**

**2 包管理工具 yum**

安装：rpm -ivh \*.rpm 卸载：rpm -e packgename

mount是Linux下的一个命令，它可以将分区挂接到Linux的一个文件夹下，从而将分区和该目录联系起来，因此我们只要访问这个文件夹，就相当于访问该分区了。

mount [参数] /dev/sdb1(需要挂载的分区) /sdb1(挂载目录)

用AUFS，把 base 和 top 一起 mount 到 ./mnt 目录：

mount -t aufs -o br=./top:./base none ./mnt

-t aufs：mount 的文件类型，使用的是 aufs

-o：传递个 aufs 的选项，每个文件类型的选项不同

br：表示 branch，也就是 aufs 需要的的各个目录

none：这个本来是设备的名字，但是我们并没有用到任何设备，只会用到文件夹，因此这里为 none

./mnt：挂载点，也就是内容最终出现的目录

AUFS 的全称是 Advanced Multi-layered unification filesytem，它的主要功能是：把多个目录结合成一个目录，对外使用。  
把多个目录 mount 成一个，读写操作步骤如下：

**默认情况下，最上层的目录为读写层，只能有一个,下面可以有一个或者多个只读层读文件**

读文件，从最上面一个开始往下逐层去找，打开第一个找到的文件，读取其中的内容

写文件，如果在最上层找到了该文件，直接打开。否则，从上往下开始查找，找到文件后，把文件复制到最上层，然后再打开这个copy

ps -ef|grep -v “grep”(注：排除名字叫grep的匹配条件)|wc -l(统计行数)