系统运维分享

一、系统运维职责

运维的职责覆盖了产品从设计到发布、运行维护、变更升级及至下线的生命周期

产品发布前

这个阶段[运维工程师](https://baike.baidu.com/item/%E8%BF%90%E7%BB%B4%E5%B7%A5%E7%A8%8B%E5%B8%88" \t "https://baike.baidu.com/item/%E8%BF%90%E7%BB%B4/_blank)的职责是参与设计并把有关运维准入，主要包括：

（1） 产品的业务熟悉；

（2） 产品架构设计的合理性评估，包括是否存在单点，是否可容错，是否有强耦合等，同时需要提供产品设计的合理性建议以使产品能够满足上线发布并稳定运行的基本要求；

（3） 资源评估，包括所需的服务器资源、网络资源以及资源的分布等，同时把相关产品对资源预算申请的合理性，控制服务成本；

（4） 资源就位，将申请的服务器及基础环境/域名准备就位。

产品发布

这个阶段运维工程师负责发布的具体工作，将具体的软件和系统/硬件资源整合形成产品并对外提供服务。

对于已在线服务的更新也属于发布范畴，这个时候的产品发布一般要保障在线发布，在不中断对外服务的情况下完成产品的升级。对于大型复杂的变更也存在中止服务部署完成后再重新提供服务的情况，但这种情况需要运维工程师通过尽可能的技术手段来避免。

产品运行维护

这个阶段的主要工作包括：

（1） 监控：对服务运行的状态进行实时的监控，随时发现服务的运行异常和资源消耗情况；输出重要的日常服务运行报表以评估服务/业务整体运行状况，发现服务隐患；

（2） 故障处理：对服务出现的任何异常进行及时处理，尽可能避免问题的扩大化甚至中止服务。这之前运维工程师需要针对各类服务异常，如机房/网络故障、程序bug等问题制定处理的预案，问题出现时可以自动或手动执行预案达到止损的目的。除了日常小故障外，运维工程师还需要考虑产品不同程度受损情况下的灾难恢复，包括诸如地震等不可抗力导致大规模机房故障、在线产品被删除等对产品造成致命伤害的情况。

容量管理：包括服务规模扩张后的资源评估、扩容、机房迁移、流量调度等规划和具体实施。

产品性能/成本优化

产品对外提供服务最重要的一点是用户体验，用户体验中非常重要的是产品的可用性和响应速度。而如何用最合理的资源（如机器、[带宽](https://baike.baidu.com/item/%E5%B8%A6%E5%AE%BD" \t "https://baike.baidu.com/item/%E8%BF%90%E7%BB%B4/_blank)等）支持产品提供高可用和高速度的用户体验，这也是运维工程师的重要职责。

产品下线

服务器操作系统，又名网络操作系统。相比个人版操作系统，在一个具体的网络中，服务器操作系统要承担额外的管理、配置、稳定、安全等功能，处于每个网络中的心脏部位，其网络操作系统的别称也由此而来。

[服务器操作系统](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/6639898" \t "https://baike.baidu.com/item/%E4%B8%BB%E6%B5%81%E6%9C%8D%E5%8A%A1%E5%99%A8%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/_blank)主要分为四大流派：WINDOWS、[NETWARE](https://baike.baidu.com/item/NETWARE/602929" \t "https://baike.baidu.com/item/%E4%B8%BB%E6%B5%81%E6%9C%8D%E5%8A%A1%E5%99%A8%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/_blank)、[UNIX](https://baike.baidu.com/item/UNIX/219943" \t "https://baike.baidu.com/item/%E4%B8%BB%E6%B5%81%E6%9C%8D%E5%8A%A1%E5%99%A8%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/_blank)、[LINUX](https://baike.baidu.com/item/LINUX/27050" \t "https://baike.baidu.com/item/%E4%B8%BB%E6%B5%81%E6%9C%8D%E5%8A%A1%E5%99%A8%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/_blank)。

一、了解服务器--系统基础

服务器的市场竞争非常激烈，国外有IBM、HP（惠普）、DELL（戴尔）、SUN等著名厂商，国内有华为、联想、浪潮、曙光等一线厂商都提供不同级别的服务器产品

服务器也是由主板、内存、CPU、磁盘、网卡、显卡、电源、主机箱、RAID卡、IDRAC卡等硬件设备组成

服务器分为塔式服务器、机架式服务器、刀片服务器。



PowerEdge T640服务器（塔式服务器）



PowerEdge R330机架式服务器

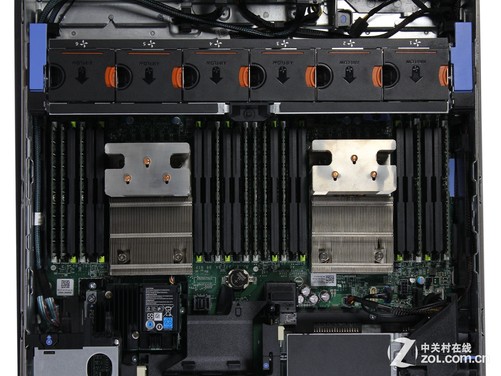
机架式服务器（1u=4.45cm）



PowerEdge R740服务器（2u=88.9mm=8.89cm）



服务器内部架构

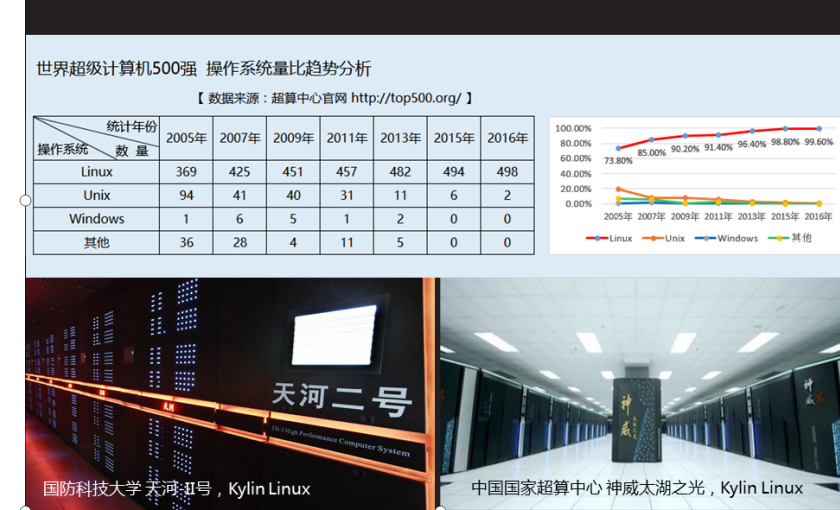






1. 走进linux

（1）、linux简介



互联网数据中心(Internet Data Center)简称IDC

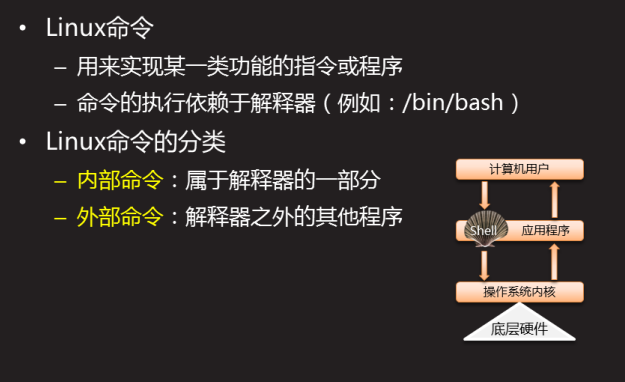
Linux在企业中的应用：

企业服务器

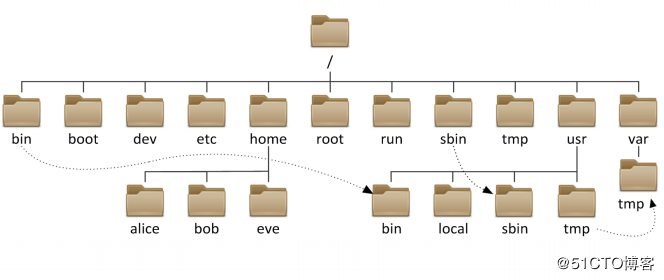
嵌入式系统

高性能大型运算





Linux一切皆文件！

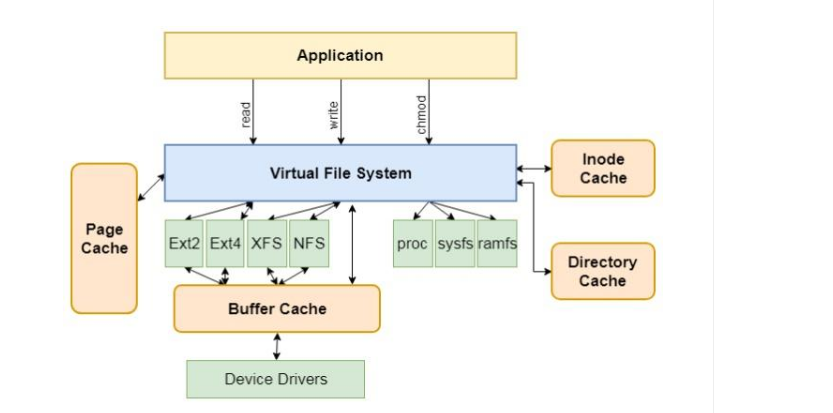




/bin 二进制可执行命令  
/dev 设备特殊文件  
/etc 系统管理和配置文件  
/etc/rc.d 启动的配置文件和脚本  
/home 用户主目录的基点，比如用户user的主目录就是/home/user，可以用~user表示  
/lib 标准程序设计库，又叫动态链接共享库，作用类似windows里的.dll文件  
/sbin 系统管理命令，这里存放的是系统管理员使用的管理程序  
/tmp 公用的临时文件存储点  
/root 系统管理员的主目录  
/mnt 系统提供这个目录是让用户临时挂载其他的文件系统。  
/lost+found 这个目录平时是空的，系统非正常关机而留下“无家可归”的文件（windows下叫什么.chk）就在这里  
/proc 虚拟的目录，是系统内存的映射。可直接访问这个目录来获取系统信息。  
/var 某些大文件的溢出区，比方说各种服务的日志文件  
/usr 最庞大的目录，要用到的应用程序和文件几乎都在这个目录。其中包含：  
/usr/x11r6 存放x window的目录  
/usr/bin 众多的应用程序  
/usr/sbin 超级用户的一些管理程序  
/usr/doc linux文档  
/usr/include linux下开发和编译应用程序所需要的头文件  
/usr/lib 常用的动态链接库和软件包的配置文件  
/usr/man 帮助文档  
/usr/src 源代码，linux内核的源代码就放在/usr/src/linux里  
/usr/local/bin 本地增加的命令  
/usr/local/lib 本地增加的库根文件系统

**（2）、Linux的文件系统**

目前在Linux操作系统中支持很多种文件系统，包括Ext2、Ext4、Btrfs和XFS等，多达几十个文件系统。虽然支持的文件系统种类很多，但从用户层面使用方式无任何差别，用户并不感知其中的差异



索引节点inode

硬盘分区，格式化、创建文件系统

被格式化的磁盘分为两部分：第一部分是Inode 第二部分是block

block是用来存储实际数据用的，例如：照片、视频等普通文件数据

inode是用来存储这些数据的属性的（也就是ls-l的结果）

inode包含的属性信息有文件大小、属主、归属的用户组、读写权限、问价类型、修改时间，还有指向文件实体指针的功能（inode节点—-block的对应关系），但是唯独不包含文件名

访问一个文件【通过文件名找到inode—->block 】

查看硬盘分区情况：lsblk

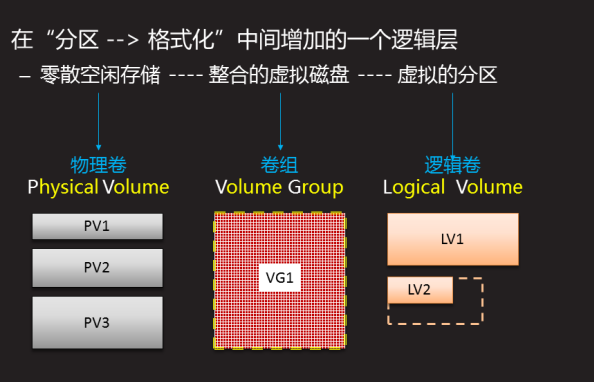
查看硬盘使用情况：df -hT

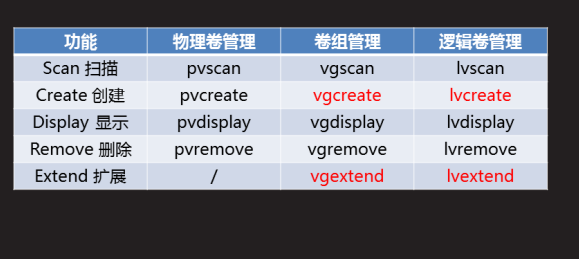
对硬盘进行分区（交互式）：fdisk

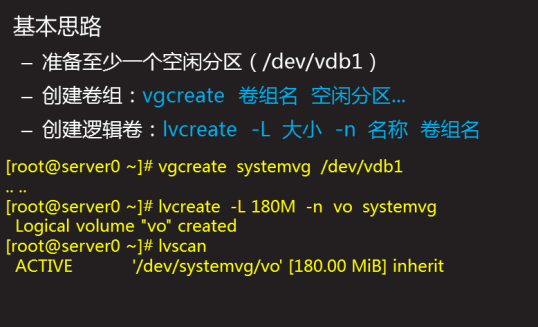
查看分区详细信息：fidsk -l /dev/vda

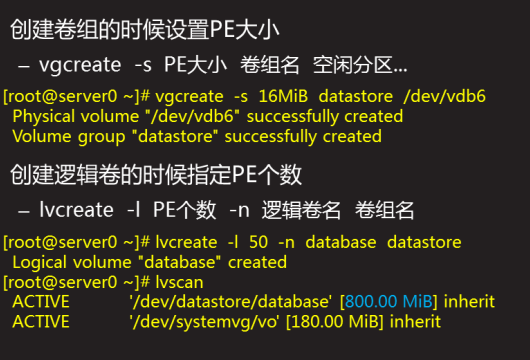
LVM逻辑卷

物理卷（PV），卷组（VG），逻辑卷（LV）

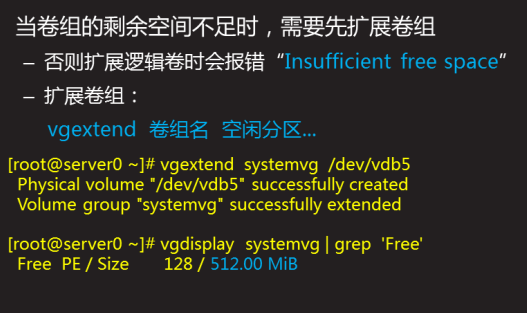




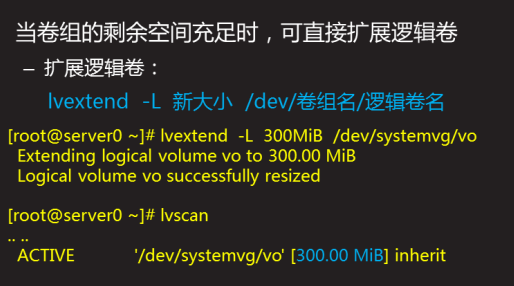




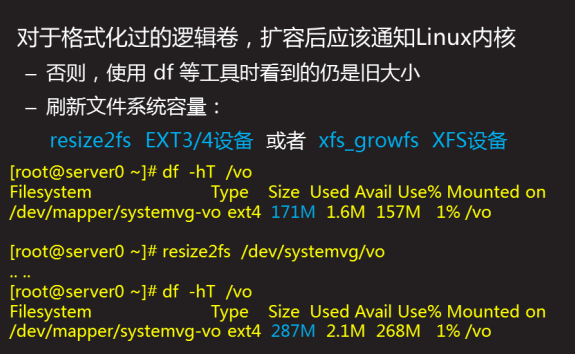
扩展卷组



扩展逻辑卷



格式化扩展的逻辑卷



SYSSTAT是一个软件包，包含**监测系统性能及效率的一组工具**，这些工具对于我们收集系统性能数据，比如CPU使用率、硬盘和网络吞吐数据，这些数据的收集和分析，有利于我们判断系统是否正常运行，是提高系统运行效率、安全运行服务器的得力助手”。

安装sysstat：yum -y install sysstat

sysstat工具包包含工具：vmstat,netstat,ifstat

iostat 工具提供CPU使用率及硬盘吞吐效率的数据；  #比较核心的工具

mpstat 工具提供单个处理器或多个处理器相关数据；

**pidstat**: 关于运行中的进程/任务、CPU、内存等的统计信息

sar 工具负责收集、报告并存储系统活跃的信息，可以查看包括负载平均值，CPU负载，RAM，磁盘I/O网络I/O和其他一些统计信息；     #统计数据的核心工具

例子1：iostat

%user：CPU处在用户模式下的时间百分比。

%nice：CPU处在带NICE值的用户模式下的时间百分比。

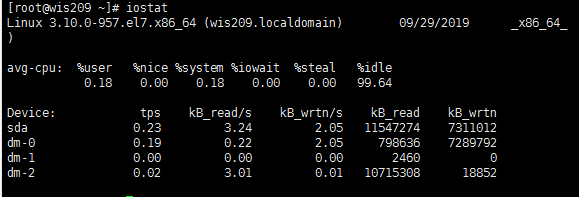
%system：CPU处在系统模式下的时间百分比。

%iowait：CPU等待输入输出完成时间的百分比，值过高，表示硬盘存在I/O瓶颈

%steal：管理程序维护另一个虚拟处理器时，虚拟CPU的无意识等待时间百分比。

%idle：CPU空闲时间百分比。值高，表示CPU较空闲，值高但系统响应慢时，可能是CPU等待分配内存，应加大内存容量。值持续低于10，表明CPU处理能力相对较低，系统中最需要解决的资源是CPU。小于70% IO压力就较大了，一般读取速度有较多的wait。

每隔两秒刷新一次，总共刷新三次：iostat 2 3

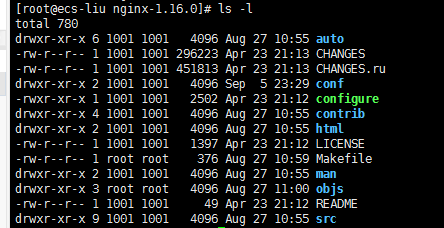


例子2：sar -u 1 5 输出CPU使用情况的统计信息，每秒输出一次，一共五次

文件类型



查看文件类型命令：ls -l



常用选项：-a 显示目录下所有文件，包括隐藏文件（以‘.’开头的文件）

-h 一人类易读的单位换算文件大小，K，M，G

-d 显示目录本身详细信息

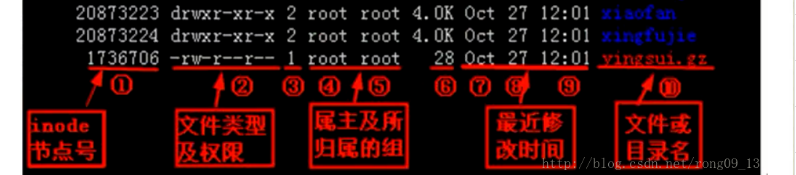
-i 显示inode

-t 按时间排序，最近的排在第一，以此类推

-r 按时间前后倒序排序

常用组合 ： -lh -la -ld -ltr -lhi

ls -lhi



删除文件：rm 只是删除

**Linux上进程有5种状态:**1. 运行(正在运行或在运行队列中等待)  
2. 中断(休眠中, 受阻, 在等待某个条件的形成或接受到信号)  
3. 不可中断(收到信号不唤醒和不可运行, 进程必须等待直到有中断发生)  
4. 僵死(进程已终止, 但进程描述符存在, 直到父进程调用wait4()系统调用后释放)  
5. 停止(进程收到SIGSTOP, SIGSTP, SIGTIN, SIGTOU信号后停止运行运行)  
**ps工具标识进程的5种状态码:**D 不可中断 uninterruptible sleep (usually IO)  
R 运行 runnable (on run queue)  
S 中断 sleeping  
T 停止 traced or stopped  
Z 僵死 a defunct (”zombie”) process

查看进程情况：ps -aux (-elf)

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND  
USER: 行程拥有者  
PID: pid  
%CPU: 占用的 CPU 使用率  
%MEM: 占用的记忆体使用率  
VSZ: 占用的虚拟记忆体大小  
RSS: 占用的记忆体大小  
TTY: 终端的次要装置号码 (minor device number of tty)  
STAT: 该行程的状态:  
D: 不可中断的静止  
R: 正在执行中  
S: 静止状态  
T: 暂停执行  
Z: 不存在但暂时无法消除  
W: 没有足够的记忆体分页可分配  
<: 高优先序的行程  
N: 低优先序的行程  
L: 有记忆体分页分配并锁在记忆体内 (即时系统或捱A I/O)  
START: 行程开始时间  
TIME: 执行的时间  
COMMAND:所执行的指令

查看进程的树形结构：pstree (软件包tree，yum -y install pstree)

查看当前终端的进程：ps

结束进程： kill （-9） PID，

kill命令可以带信号号码选项，也可以不带。如果没有信号号码，kill命令就会发出终止信号(15)，这个信号可以被进程捕获，使得进程在退出之前可以清理并释放资源。也可以用kill向进程发送特定的信号。kill -l 查看所有信号,重用 信号如下：

HUP    1    终端断线  
INT     2    中断（同 Ctrl + C）  
QUIT    3    退出（同 Ctrl + \）  
TERM   15    终止  
KILL    9    强制终止  
CONT   18    继续（与STOP相反， fg/bg命令）  
STOP    19    暂停（同 Ctrl + Z）

pkill  正在运行的程式名

杀死指定名字的进程：killall 参数 进程名

查看内核版本：uname -a

查看系统当前用户连接情况：who 或者w

查看服务器cpu信息：lscpu 或者 cat /proc/cpu/info

查看系统负载：top

#### **第一行显示服务器概况 top命令**

如下所示，第一行列出了服务器运行了多长时间，当前有多少个用户登录，服务器的负荷情况等，使用uptime命令能获得同样的结果。

top - 21:48:39 up 8:57, 2 users, load average: 0.36, 0.24, 0.14

/ / / \

当前时间 运行时长 当前登录用户数 平均负载（1分钟、5分钟、15分钟）

平均负载的值越小代表系统压力越小，越大则代表系统压力越大。通常，我们会以最后一个数值，也就是15分钟内的平均负载作为参考来评估系统的负载情况。

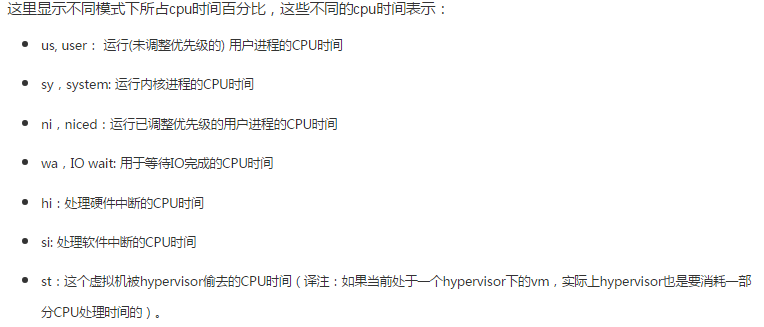
对于只有单核cpu的系统，1.0是该系统所能承受负荷的边界值，大于1.0则有处理需要等待，处于超负荷运作。

一个单核cpu的系统，平均负载的合适值是0.7以下。如果负载长期徘徊在1.0，则需要考虑马上处理了。超过1.0的负载，可能会带来非常严重的后果。

当然，多核cpu的系统是在前述值的基础上乘以cpu内核的个数。如对于多核cpu的系统，有N个核则所能承受的边界值为N.0。

第二行是进程情况，running是正在运行的进程，sleeping是休眠的进程，stopped是已经停止的进程，zombie是僵尸进程。

第三行是cpu使用情况。



用户进程 us 占比高，I/O 操作 wa 低：说明系统缓慢的原因在于进程占用大量 CPU，通常还会伴有教低的空闲比率 id，说明 CPU 空转时间很少。

I/O 操作 wa 低，空闲比率 id 高：可以排除 CPU 资源瓶颈的可能。

I/O 操作 wa 高：说明 I/O 占用了大量的 CPU 时间，需要检查交换空间的使用，交换空间位于磁盘上，性能远低于内存，当内存耗尽开始使用交换空间时，将会给性能带来严重影响，所以对于性能要求较高的服务器，一般建议关闭交换空间。另一方面，如果内存充足，但 wa 很高，说明需要检查哪个进程占用了大量的 I/O 资源。

-us:用户进程消耗的CPU时间百分  
us的值比较高时，说明用户进程消耗的CPU时间多，但是如果长期超50%的使用，那么我们就该考虑优化程序算法或者进行加速（比如PHP/PERL）  
-sy:内核进程消耗的CPU时间百分比（sy的值高时，说明系统内核消耗的CPU资源多，这并不是良性表现，我们应该检查原因）

-wa:IO等待消耗的CPU时间百分比  
wa的值高时，说明IO等待比较严重，这可能由于磁盘大量作随机访问造成，也有可能磁盘出现瓶颈（块操作）。  
-id:CPU处于空闲状态时间百分比,如果空闲时间(cpu id)持续为0并且系统时间(cpu sy)是用户时间的两倍(cpu us) 系统则面临着CPU资源的短缺.  
解决办法:  
当发生以上问题的时候请先调整应用程序对CPU的占用情况.使得应用程序能够更有效的使用CPU.同时可以考虑增加更多的CPU. 关于CPU的使用情况还可以结合mpstat, ps aux top prstat –a等等一些相应的命令来综合考虑关于具体的CPU的使用情况,和那些进程在占用大量的CPU时间.一般情况下，应用程序的问题会比较大一些.比如一些SQL语句不合理等等都会造成这样的现象.

用sar进行CPU利用率的分析

#sar -u 2 10  
Linux 2.6.18-53.el5PAE (localhost.localdomain) 03/28/2009

07:40:17 PM CPU %user %nice %system %iowait %steal %idle  
07:40:19 PM all 12.44 0.00 6.97 1.74 0.00 78.86  
07:40:21 PM all 26.75 0.00 12.50 16.00 0.00 44.75  
07:40:23 PM all 16.96 0.00 7.98 0.00 0.00 75.06  
07:40:25 PM all 22.50 0.00 7.00 3.25 0.00 67.25  
07:40:27 PM all 7.25 0.00 2.75 2.50 0.00 87.50  
07:40:29 PM all 20.05 0.00 8.56 2.93 0.00 68.46  
07:40:31 PM all 13.97 0.00 6.23 3.49 0.00 76.31  
07:40:33 PM all 8.25 0.00 0.75 3.50 0.00 87.50  
07:40:35 PM all 13.25 0.00 5.75 4.00 0.00 77.00  
07:40:37 PM all 10.03 0.00 0.50 2.51 0.00 86.97  
Average: all 15.15 0.00 5.91 3.99 0.00 74.95

在显示内容包括：  
%user：CPU处在用户模式下的时间百分比。

%nice：CPU处在带NICE值的用户模式下的时间百分比。  
%system：CPU处在系统模式下的时间百分比。  
%iowait：CPU等待输入输出完成时间的百分比。

%steal：管理程序维护另一个虚拟处理器时，虚拟CPU的无意识等待时间百分比。  
%idle：CPU空闲时间百分比。

在所有的显示中，我们应主要注意%iowait和%idle，%iowait的值过高，表示硬盘存在I/O瓶颈，%idle值高，表示CPU较空闲，如果%idle值高但系统响应慢时，有可能是CPU等待分配内存，此时应加大内存容量。%idle值如果持续低于10，那么系统的CPU处理能力相对较低，表明系统中最需要解决的资源是CPU。

用sar进行运行进程队列长度分析：

#sar -q 2 10  
Linux 2.6.18-53.el5PAE (localhost.localdomain) 03/28/2009

07:58:14 PM runq-sz plist-sz ldavg-1 ldavg-5 ldavg-15  
07:58:16 PM 0 493 0.64 0.56 0.49  
07:58:18 PM 1 491 0.64 0.56 0.49  
07:58:20 PM 1 488 0.59 0.55 0.49  
07:58:22 PM 0 487 0.59 0.55 0.49  
07:58:24 PM 0 485 0.59 0.55 0.49  
07:58:26 PM 1 483 0.78 0.59 0.50  
07:58:28 PM 0 481 0.78 0.59 0.50  
07:58:30 PM 1 480 0.72 0.58 0.50  
07:58:32 PM 0 477 0.72 0.58 0.50  
07:58:34 PM 0 474 0.72 0.58 0.50  
Average: 0 484 0.68 0.57 0.49

runq-sz 准备运行的进程运行队列。  
plist-sz 进程队列里的进程和线程的数量  
ldavg-1 前一分钟的系统平均负载(load average)  
ldavg-5 前五分钟的系统平均负载(load average)  
ldavg-15 前15分钟的系统平均负载(load average)

顺便说一下load avarage的含义

load average可以理解为每秒钟CPU等待运行的进程个数.

在Linux系统中，sar -q、uptime、w、top等命令都会有系统平均负载load average的输出，那么什么是系统平均负载呢？  
系统平均负载被定义为在特定时间间隔内运行队列中的平均任务数。如果一个进程满足以下条件则其就会位于运行队列中：  
- 它没有在等待I/O操作的结果  
- 它没有主动进入等待状态(也就是没有调用'wait')  
- 没有被停止(例如：等待终止

# uptime  
20:55:40 up 24 days, 3:06, 1 user, load average: 8.13, 5.90, 4.94  
命令输出的最后内容表示在过去的1、5、15分钟内运行队列中的平均进程数量。  
一般来说只要每个CPU的当前活动进程数不大于3那么系统的性能就是良好的，如果每个CPU的任务数大于5，那么就表示这台机器的性能有严重问题。对 于上面的例子来说，假设系统有两个CPU，那么其每个CPU的当前任务数为：8.13/2=4.065。这表示该系统的性能是可以接受的。

 查看CPU利用率。建议CPU指标如下

　　a) User Time：65%～70%

　　b) System Time：30%～35%

　　c) Idle：0%～5%

　　如果us,sy高于这个指标可以判断CPU有瓶颈

　　使用top查看

　　查看运行队列

　　每个CPU都会维持一个运行队列，理想情况下，调度器会不断让队列中的进程运行。进程不是处在sleep状态就是run able状态。如果CPU过载，就会出现调度器跟不上系统的要求，导致可运行的进程会填满队列。队列愈大，程序执行时间就愈长。“load”用来表示运行队列，用top 命令我们可以看到CPU的1分钟，5分钟和15分钟内的运行队列的大小。这个值越大表明系统负荷越大。超过 1.00（1核CPU来说），那么说明CPU已经超出负荷，交通严重的拥堵。

　　使用top或者uptime查看

　　查看上下文切换

　　每个CPU（或多核CPU中每个核心）在同一时间只能执行一个线程，Linux采用抢占式调度。即为每个线程分配一定的执行时间，当到达执行时间，线程中有IO阻塞或高优先级线程要执行时，Linux将切换执行的线程，在切换时要存储目前线程的执行状态，并恢复要执行的线程状态，这个过程称之为上下文切换。对于java应用，典型的是在进行文件IO操作，网络IO操作，锁等待或线程sleep时，当前线程会进入阻塞或者休眠状态，从而触发上下文切换，上下文切换过多会造成内核占用过多的CPU使用，使得应用的响应速度下降。

　　使用vmstat查看cs

结论

检查system的运行队列,以及确定不要超出每个处理器3个可运行状态线程的限制.

　　确定CPU 利用率中user/system比例维持在70/30

　　当CPU 开销更多的时间在system mode,那就说明已经超负荷并且应该尝试重新调度优先级

　　当I/O 处理得到增长,CPU 范畴的应用处理将受到影响

ps:对于JAVA应用，CPU瓶颈可以通过jprofiler监控分析

磁盘瓶颈

　　iostat查看IO信息。如果 %util 接近 100%，说明产生的I/O请求太多，I/O系统已经满负荷，该磁盘可能存在瓶颈。

　　另外还需要注意iowait这个值，iowait 值高就意味着磁盘缓慢或负载过大。还有不要信任svctm这个字段。

　　监控swap 和系统分区,要确保virtual memory不是文件系统I/O 的瓶颈.

ps:磁盘瓶颈可以通过pidstat -d 定位程序

#iostat -x 1 每隔一秒刷新一次

Linux 2.6.18-53.el5PAE (localhost.localdomain)  03/27/2009

avg-cpu:  %user   %nice %system %iowait  %steal   %idle

          30.72    0.00    5.00    5.72    0.00   58.56

Device:   rrqm/s   wrqm/s    r/s    w/s     rsec/s   wsec/s   avgrq-sz avgqu-sz   await    svctm    %util

sda         0.79       21.81     9.15   8.08   237.99   239.29    27.69       1.32         76.31    4.07      7.02

sdb         0.69      19.13      3.26   2.99   153.08   176.92    52.85       0.43         68.80    5.96      3.72

sdc         3.47       89.30     10.95  7.30   213.30   772.94    54.04       1.32         72.43    4.18      7.63

每项数据的含义如下，

rrqm/s:     每秒进行 merge 的读操作数目。即 rmerge/s

wrqm/s:     每秒进行 merge 的写操作数目。即 wmerge/s

r/s:       每秒完成的读 I/O 设备次数。即 rio/s

w/s:       每秒完成的写 I/O 设备次数。即 wio/s

rsec/s:     每秒读扇区数。即 rsect/s

wsec/s:     每秒写扇区数。即 wsect/s

rkB/s:     每秒读K字节数。是 rsect/s 的一半，因为每扇区大小为512字节。

wkB/s:     每秒写K字节数。是 wsect/s 的一半。

avgrq-sz:   平均每次设备I/O操作的数据大小 (扇区)。即 (rsect+wsect)/(rio+wio)

avgqu-sz:   平均I/O队列长度。即 aveq/1000 (因为aveq的单位为毫秒)。

await:     平均每次设备I/O操作的等待时间 (毫秒)。即 (ruse+wuse)/(rio+wio)

svctm:     平均每次设备I/O操作的服务时间 (毫秒)。即 use/(rio+wio)

%util:     一秒中有百分之多少的时间用于 I/O 操作，或者说一秒中有多少时间

I/O队列是非空的,即use/1000 (因为use的单位为毫秒),

如果 %util 接近 100%，说明产生的I/O请求太多，I/O系统已经满负荷，该磁盘可能存在瓶颈。

svctm 一般要小于 await (因为同时等待的请求的等待时间被重复计算了)，

svctm 的大小一般和磁盘性能有关，CPU/内存的负荷也会对其有影响，请求过多也会间接导致 svctm 的增加。

await 的大小一般取决于服务时间(svctm) 以及 I/O 队列的长度和 I/O 请求的发出模式。如果 svctm 比较接近 await，说明 I/O 几乎没有等待时间；如果 await 远大于 svctm，说明 I/O 队列太长，应用得到的响应时间变慢，如果响应时间超过了用户可以容许的范围，这时可以考虑更换更快的磁盘，调整内核 elevator 算法，优化应用，或者升级 CPU。

   队列长度(avgqu-sz)也可作为衡量系统 I/O 负荷的指标，但由于 avgqu-sz 是按照单位时间的平均值，所以不能反映瞬间的 I/O 洪水。

io/s = r/s +w/s

await=(ruse+wuse)/io（每个请求的等待时间）

await\*io/s=每秒内的I/O请求总共需要等待的ms

avgqu-sz=await\*(r/s+w/s)/1000(队列长度)

以下数据其实与/proc/diskstats中除设备号与设备名外的其它数据是一一对应关系，只是统计的方法略有差别而已。

#cat /sys/block/sda/stat

17949157  1547772 466744707 174070520 15855905 42781288 469298468 2406092114        2 137680700 2581025934

1. 第五行内存和交换分区使用情况，也可用free -h查看。

p：指定监控进程PID来仅仅监控某个进程的状态

i：不显示闲置或者僵尸进程，再按一次i，恢复显示所有进程

k：输入想要kill掉的进程pid号，默认使用15信号终止，可用9强制终止

s：改变刷新时间间隔，默认是5s刷新一次，若输入0秒，持续刷新

M：切换内存信息

M：根据驻留内存大小进行排序

P：根据CPU使用百分比大小进行排序

T：根据时间/累计时间进行排序

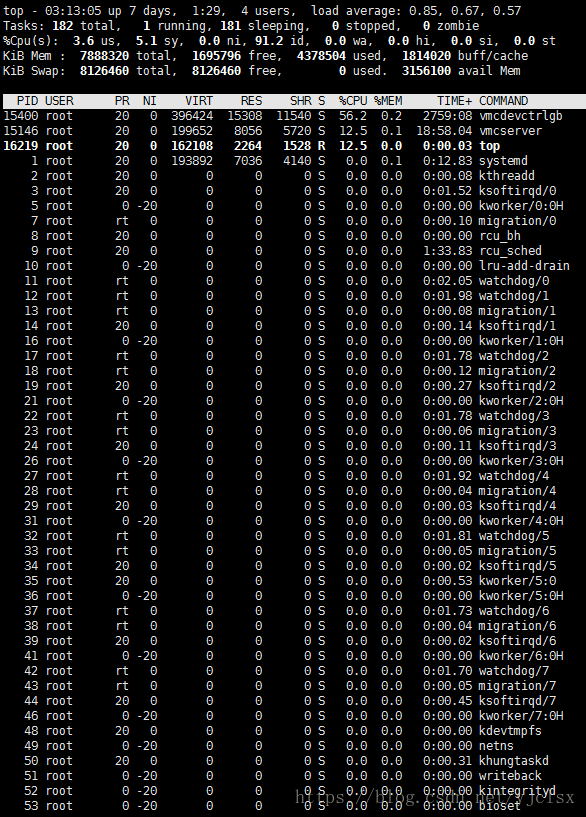
n : 更新的次数，完成后将会退出 top，如，-n1(-n2...)：刷新一次（刷新两次....）后自动退出

q：退出top命令

c : 切换显示模式，共有两种模式，一是只显示执行档的名称，另一种是显示完整的路径与名称S : 累积模式，会将己完成或消失的子行程 ( dead child process ) 的 CPU time 累积起来，显示完整命令：top -c

b : 批次档模式，搭配 "n" 参数一起使用，可以用来将 top 的结果输出到档案内

d : 改变显示的更新速度，或是在交谈式指令列( interactive command)按 s。设置信息更新时间，top -d 3 //表示更新周期为3秒



**查看日志**文件内容命令有

* cat 由第一行开始显示文件内容；**cat -n filename**（打印行号），缺点：大型文件非常慢
* tac 从最后一行开始显示，可以看出 tac 是 cat 的倒着写
* nl 显示的时候，顺道输出行号！
* more 一页一页的显示文件内容
* less 与 more 类似，但是比 more 更好的是，他可以往前翻页！
* head 只看头几行；head -100 filename 输出文件头100行，不加-100，默认输出前10行
* tail 只看尾巴几行；tail -100 filename 输出文件尾部100行，不加-100，默认输出后10行

Linux中常用的工作模式：

0：关机，相当于halt  
1：单用户模式  
2：无网络支持的多用户模式  
3：有网络支持的多用户模式  
4：保留，未使用  
5：有网络支持有[X-Window](https://www.baidu.com/s?wd=X-Window&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)支持的多用户模式  
6：重新引导系统，即重启，相当于reboot

查看内存使用情况：free -h（-g）或者cat /proc/meminfo

服务管理：

Centos6：

service [服务名] start/restart/stop/status

列出所有的服务及每个级别的开启状态：Chkconfig --list

Chkconfig --level 345 network on/off

另外还可以省略级别，默认是针对级别2、3、4和5操作的。

chkconfig network on

把某个服务加入系统服务或者删除:

chkconfig --add network

chkconfig --del network

Centos7:systemd管理服务

systemctl enable crond.service #让某个服务开机启动（.service可以省略）

systemctl disable crond.service #不让开机启动(相当于centos6中的chkconfig crond.service on/off)

systemctl status crond.service #查看服务状态

systemctl start crond.service #启动某个服务

systemctl stop crond.service #停止某个服务

systemctl restart crond.service #重启某个服务

systemctl is-enabled crond #查看某个服务是否开机启动

systemctl list-unit-files #列出 systemd 服务

systemctl list-dependencies [target] #查看对特定 target 启用的服务请执行

网络配置：

Centos6及以前：network.service服务

Centos7开始：network.service和NetworkManager

network.service是secondary，NetworkManager.service才是primary

网卡配置文件路径：/etc/sysconf/network-scripts/ifcfg-\*

查看各网卡信息：ifconfig（需安装net-tools软件）

查看特定网卡信息：ifconfig eth0

临时设置ip:

ifconfig eth0 192.168.0.100/24(回车即生效，重启电脑或者网络服务，ip消失)

开机、禁用网卡：

 ifconfig [NIC\_NAME] down/up

 例子：ifconfig enp0s3 down，禁用网卡，此时用ifconfig看不到该网卡

ifdown 命令用于禁用网卡，ifup 命令用于启用网卡。

注意：这两个命令不支持以 enpXXX 命名的新的网络设备。

ifdown/ifup 的常用语法：

# ifdown [NIC\_NAME]

# ifup [NIC\_NAME]

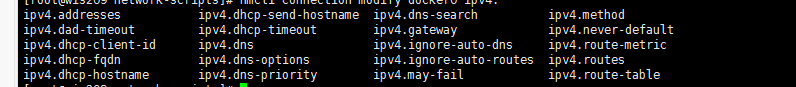
例子：ifdown eth1

ip link set 网卡名 down/up

Ip link show 或者 ip address show

Nmcli命令，需要 NetworkManager服务开启

更改ip：nmcli connection modify docker0 ipv4.addresses 192.168.0.101/24 ipv4.method manual connection.autoconnect yes



启动、禁止网卡：nmcli connection up/down 网卡名

在 Linux 系统中，可以通过查看 /proc/sys/vm/swappiness 内容的值来确定系统对 SWAP 分区的使用原则。当swappiness 内容的值为 0 时，表示最大限度地使用物理内存，物理内存使用完毕后，才会使用 SWAP 分区。当swappiness 内容的值为 100 时，表示积极地使用 SWAP 分区，并且把内存中的数据及时地置换到 SWAP 分区。 默认值为 0，表示需要在物理内存使用完毕后才会使用 SWAP 分区

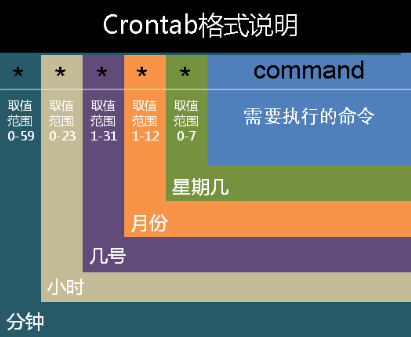
1. 自动化运维
2. 编写shell脚本，添加crontab周期性计划

查看自己账户的周期性计划：crontab -l

编辑周期性计划：crontab -e

删除周期性计划：crontab -r

用户任务调度：用户定期要执行的工作，比如用户数据备份、定时邮件提醒等。用户可以使用crontab工具来定制自己的计划任务。所有用户定义的crontab文件都被保存在/var/spool/cron目录中。其文件名与用户名一致。crontab使用者权限文件如下：



在以上各个字段中，还可以使用以下特殊字符：

星号（\*）：当 minute 为 \* 时表示每分钟都要执行 command，hour 为 \* 时表示每小时都要执行程序，其余类推。

中杠（-）：当 minute 为 a-b 时表示从第 a 分钟到第 b 分钟这段时间内要执行，hour 为 a-b 时表示从第 a 到第 b 小时都要执行，例如“2-6”表示“2,3,4,5,6”，其余类推。

斜杠（/）：当 minute 为 \*/n 时表示每 n 分钟个时间间隔执行一次，例如\*/10，表示每十分钟执行一次。hour 为 \*/n 表示每 n 小时个时间间隔执行一次，例如\*/2，表示每两小时执行一次，其余类推。

逗号（,）：当 minute 为 a, b, c,... 时表示第 a, b, c,... 分钟要执行，hour 为 a, b, c,... 时表示第 a, b, c...个小时要执行，例如，“1,2,5,7,8,9”，其余类推。

例子：

每1分钟执行一次command ：\* \* \* \* \* command

每小时的第3和第15分钟执行：3,15 \* \* \* \* command

在上午8点到11点的第3和第15分钟执行：3,15 8-11 \* \* \* command

每隔两天的上午8点到11点的第3和第15分钟执行：3,15 8-11 \*/2 \* \* command

每个星期一的上午8点到11点的第3和第15分钟执行：3,15 8-11 \* \* 1 command

每晚的21:30重启smb ：30 21 \* \* \* /etc/init.d/smb restart

1. 自动化运维工具

常用自动化运维工具：Ansible、Puppet、SaltStack

自动化运维工具：ansible（通过管理服务器来操作）

Ansible —基于 Pythonparamiko 开发，分布式，无需客户端，轻量级，配置语法使用 YMAL 及 Jinja2 模板语言，

    更强的远程 命令执行操作 ，Ansible 是一个简单的自动化运维管理工具，可以用来自动化部署应用、配置、编排 task(持续 交 付、无宕机更新等)，采用paramiko 协议库（fabric 也使用这个），通过 SSH 或者 ZeroMQ 等连接主机

（1）安装ansible

Yum -y install ansible

1. 配置host文件，添加主机组

[root@m01 ~]# cd /etc/ansible/

[root@m01 ansible]# ls

ansible.cfg  hosts  roles

[root@m01 ansible]# vim hosts

[root@m01 ansible]# tail hosts

## 10.25.1.57

# Here's another example of host ranges, this time there are no

# leading 0s:

## db-[99:101]-node.example.com

[test]                       #  定义一个组名称 ，到时候批量的时候可以使用模块名字，也可以是IP地址   
****172.16.1.41  
172.16.1.31****

1. ****在批量管理服务器中，配置秘钥对实现免密****

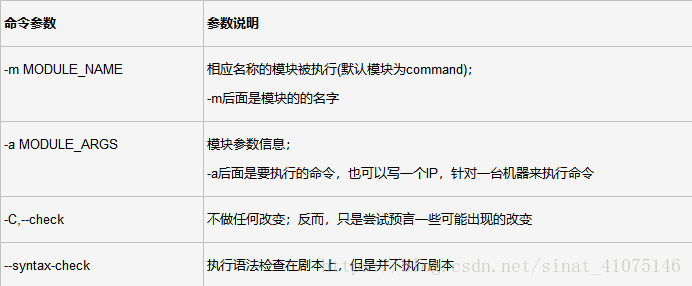
****ssh-keygen -f /root/.ssh/id\_rsa -N ‘’****

****ssh-copy-id -i ~/.ssh/id\_rsa root@ip地址****

输入密码即可，可通过“ssh root@ip”来验证是否需要密码

1. 操作

格式 :ansible 主机组（ip） -m 模块名 -a ‘参数’ #全部主机就用all



ansible test -m command -a "hostname"         ----->> ****command  是ansible的一个默认模块，test  是配置文件里的模块名称 ，这里也可以加IP地址 。****

Ansible查看模块：ansible-doc -l

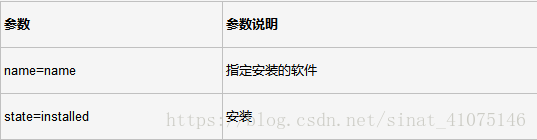
查看模块数量：ansible-doc -l | wc -l



ping   模块

[root@m01 ~]# ansible test -m ping

yum模块



ansible test -m yum -a "name=vim state=installed"

service模块



ansible test -m service -a "name=crond state=restarted enabled=yes"

Script模块：在远程主机上执行主控端的脚本，相当于scp+shell组合。

ansible all -m script -a "/home/test.sh 12 34"

Shell模块：功能：执行远程主机的shell脚本文件，或者执行命令

ansible all -m shell -a "/home/test.sh"

Copy模块：实现主控端向目标主机copy文件。

[root@localhost ~]# ansible all -m copy -a "src=/home/test.sh dest=/tmp/ owner=root group=root mode=0755"   
#src 主控端文件位置  
#dest 被控端目标位置  
#owner 文件复制过去后的所有者  
#group 文件复制过去后的所属组  
#mode 文件的权限设定，执行a+x这种方式

Ansible-playbook剧本

（5）、Ansible    剧本功能 ；

[root@m01 scripts]# vim playbook.yml   ---->>这里必须是 yml  后缀

[root@m01scripts]#cat playbook.yml   ---->>> 创建一个定时任务的剧本  ；

#command play-book

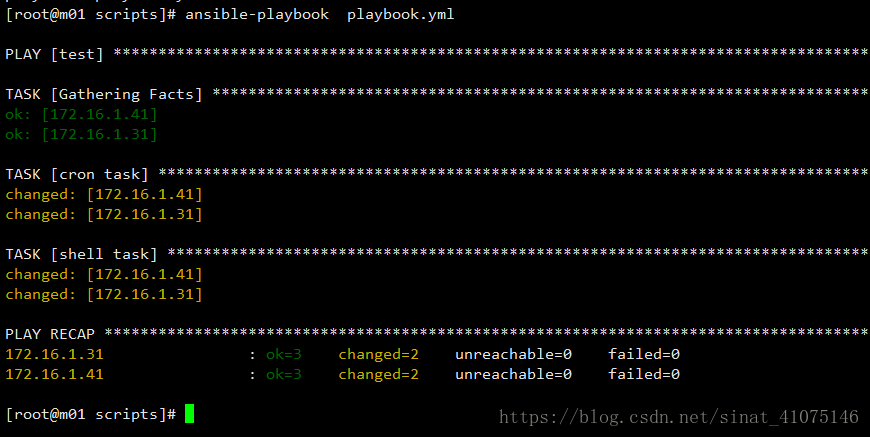
- hosts: test

  tasks:

    - name: cron taskzh      cron: name=test01 minute=0 hour=1 job='/bin/sh /server/scripts/yum.sh  /dev/null'

    - name: shell task

      shell: echo "$HOSTNAME" >>/tmp/hostname.txt



执行剧本的几个命令 ；

ansible-playbook --syntax-check cron.yml   ---->> 检查语法

ansible-playbook -C cron.yml           ---->> 模拟执行

ansible-playbook  cron.yml            ---->>执行这个剧本



Rsync

一、rsync的概述

rsync是类unix系统下的数据镜像备份工具，从软件的命名上就可以看出来了——remote sync。rsync是Linux系统下的文件同步和数据传输工具，它采用“rsync”算法，可以将一个客户机和远程文件服务器之间的文件同步，也可以在本地系统中将数据从一个分区备份到另一个分区上。如果rsync在备份过程中出现了数据传输中断，恢复后可以继续传输不一致的部分。rsync可以执行完整备份或增量备份。它的主要特点有：

1.可以镜像保存整个目录树和文件系统；

2.可以很容易做到保持原来文件的权限、时间、软硬链接；无须特殊权限即可安装；

3.可以增量同步数据，文件传输效率高，因而同步时间短；

4.可以使用rcp、ssh等方式来传输文件，当然也可以通过直接的socket连接；

5.支持匿名传输，以方便进行网站镜象等；

6.加密传输数据，保证了数据的安全性；

-v是“--verbose”,即详细模式输出； -z表示“--compress”，即传输时对数据进行压缩处理；

-r表示“--recursive”，即对子目录以递归的模式处理；-t是“--time”，即保持文件时间信息；

-o表示“owner”，用来保持文件属主信息；-p是“perms”，用来保持文件权限；

-g是“group”，用来保持文件的属组信息；

--progress用于显示数据镜像同步的过程；

--delete选项指定以rsync服务器端为基础进行数据镜像同步，也就是要保持rsync服务器端目录与客户端目录的完全一致；

--exclude选项用于排除不需要传输的文件类型；

在对rsync服务器配置结束以后，下一步就需要在客户端发出rsync命令来实现将服务器端的文件备份到客户端来。rsync是一个功能非常强大的工具，其命令也有很多功能特色选项，我们下面就对它的选项一一进行分析说明。  
Rsync的命令格式可以为以下六种：  
　　rsync [OPTION]... SRC DEST  
　　rsync [OPTION]... SRC [USER@]HOST:DEST  
　　rsync [OPTION]... [USER@]HOST:SRC DEST  
　　rsync [OPTION]... [USER@]HOST::SRC DEST  
　　rsync [OPTION]... SRC [USER@]HOST::DEST  
　　rsync [OPTION]... rsync://[USER@]HOST[:PORT]/SRC [DEST]  
　　对应于以上六种命令格式，rsync有六种不同的工作模式：  
　　1)拷贝本地文件。当SRC和DES路径信息都不包含有单个冒号":"分隔符时就启动这种工作模式。如：rsync -a /data /backup  
　　2)使用一个远程shell程序(如rsh、ssh)来实现将本地机器的内容拷贝到远程机器。当DST路径地址包含单个冒号":"分隔符时启动该模式。如：rsync -avz \*.c foo:src  
　　3)使用一个远程shell程序(如rsh、ssh)来实现将远程机器的内容拷贝到本地机器。当SRC地址路径包含单个冒号":"分隔符时启动该模式。如：rsync -avz foo:src/bar /data  
　　4)从远程rsync服务器中拷贝文件到本地机。当SRC路径信息包含"::"分隔符时启动该模式。如：rsync -av root@172.16.78.192::www /databack  
　　5)从本地机器拷贝文件到远程rsync服务器中。当DST路径信息包含"::"分隔符时启动该模式。如：rsync -av /databack root@172.16.78.192::www  
　　6)列远程机的文件列表。这类似于rsync传输，不过只要在命令中省略掉本地机信息即可。如：rsync -v rsync://172.16.78.192/www

 使用例子，把192.168.1.2的/home/下的文件同步到本地的/home/下面：rsync -aSvH --delete /home/[root@192.168.1.2:/home/](mailto:root@192.168.1.2:/home/)

    主要SCR目录的写法、比如 rsync src/  和 src 是有区别的。 src/是src文件夹下的所有文件作为传送对象。没有/的src的话是，src这个文件夹整体拷贝传送。

rsync执行中需要ssh认证等，可以实现配置，然后在cron中定时执行同步就好了。

-a, --archive 归档模式，表示以递归方式传输文件，并保持所有文件属性，等于-rlptgoD

-v, --verbose 详细模式输出

-q, --quiet 精简输出模式

-l, --links 保留软链结

-H, --hard-links 保留硬链结

-p, --perms 保持文件权限  
-o, --owner 保持文件属主信息  
-g, --group 保持文件属组信息

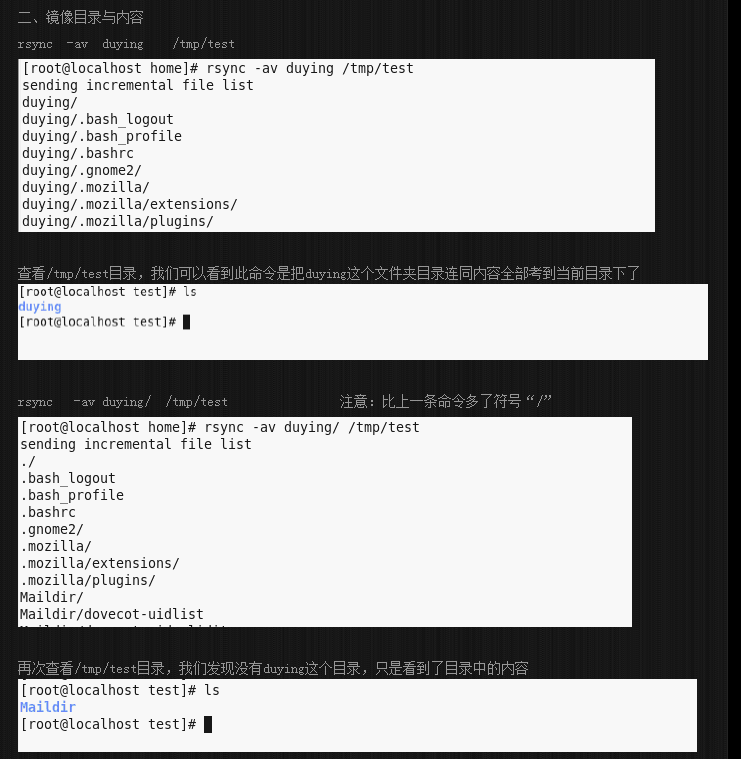
-t, --times 保持文件时间信息  
-S, --sparse 对稀疏文件进行特殊处理以节省DST的空间

-W, --whole-file 拷贝文件，不进行增量检测  
-x, --one-file-system 不要跨越文件系统边界  
-B, --block-size=SIZE 检验算法使用的块尺寸，默认是700字节  
-e, --rsh=COMMAND 指定使用rsh、ssh方式进行数据同步  
--rsync-path=PATH 指定远程服务器上的rsync命令所在路径信息  
-C, --cvs-exclude 使用和CVS一样的方法自动忽略文件，用来排除那些不希望传输的文件  
--existing 仅仅更新那些已经存在于DST的文件，而不备份那些新创建的文件  
--delete 删除那些DST中SRC没有的文件  
--delete-excluded 同样删除接收端那些被该选项指定排除的文件  
--delete-after 传输结束以后再删除  
--ignore-errors 及时出现IO错误也进行删除

-z, --compress 对备份的文件在传输时进行压缩处理  
--exclude=PATTERN 指定排除不需要传输的文件模式  
--include=PATTERN 指定不排除而需要传输的文件模式  
--exclude-from=FILE 排除FILE中指定模式的文件  
--include-from=FILE 不排除FILE指定模式匹配的文件

--config=FILE 指定其他的配置文件，不使用默认的rsyncd.conf文件  
--port=PORT 指定其他的rsync服务端口

--progress 在传输时现实传输过程  
--log-format=formAT 指定日志文件格式  
--password-file=FILE 从FILE中得到密码



rsync的优点与不足

与传统的cp、tar备份方式对比，rsync具有安全性高、备份迅速、支持增量备份等优点，通过rsync可以解决对实时性要求不高的数据备份需求，例如，定期地备份文件服务器数据到远端服务器，对本地磁盘定期进行数据镜像等。

但是随着系统规模的不断扩大，rsync的缺点逐渐被暴露了出来。首先，rsync做数据同步时，需要扫描所有文件后进行对比，然后进行差量传输。如果文件很大，扫面文件是非常耗时的，而且发生变化的文件往往是很少一部分，因此rsync是非常低效的方式。其次，rsync不能实时监测、同步数据，虽然它可以通过Linux守护进程的方式触发同步，但是两次触发动作一定会有时间差，可能导致服务器端和客户端数据出现不一致。

对调文件两列，分隔符为“，”

sed -r 's/^(.\*),(.\*),(.\*)$/\1,\3,\2/' 1.txt

awk -F, '{OFS=","; print $1,$2,$3,$4,$NF,$(NF-1)}' 1.txt

-F定义文件分隔符“，”，OFS定义输出内容分隔符“，”，若没定义，输出列默认以“ ”空格作为分隔符

echo a,b,c,d,e |awk -F "," -v O[FS](https://www.baidu.com/s?wd=FS&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)="," '{tem=$NF; $NF=$(NF-1);$(NF-1)=tem;}'1  
a,b,c,e,d

Linux的账号管理包括用户与用户组，它们两者是多对多的关系，即一个用户可以属于多个用户组，且一个用户组可以包含多个用户。一个用户组中的用户具有相同的权限。 在Linux中，用户的账户信息和用户组信息都记录在指定的文件中，这些文件构成了Linux用户管理的基础。下面我们就详细介绍Linux用户管理相关的文件。

# **Linux的账号文件**

## **1./etc/passwd**

这个文件虽然叫passwd，但实际存放Linux所有的账户信息，其中每一行表示一个账户信息，每条账户信息由7个字段构成，每个字段用:分隔。 这个文件中除了有root账户、我们自定义的用户账户外，还有很多系统账户，如下所示：

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

bin:x:1:1:bin:/bin:/sbin/nologin

daemon:x:2:2:daemon:/sbin:/sbin/nologin

adm:x:3:4:adm:/var/adm:/sbin/nologin

lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin

sync:x:5:0:sync:/sbin:/bin/sync

shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown

### **下面详细介绍每个参数的含义：**

#### **1.1. 账号名称**

#### **1.2. 密码**

在passwd文件中，密码以x表示，实际的密码存放在专门的密码文件/etc/shadow中。为何要这么玩呢？ 早起的Unix系统就是将密码存放在passwd文件中的这个位置上。但由于很多程序都需要读这个文件获取用户信息，那么密码就很容易被获取到，虽然密码采用了MD5消息摘要后再存储在这里，但仍然有被破解的风险，因此为了安全，现在的Linux系统都将密码单独存放在/etc/shadow中。

#### **1.3. UID**

UID=User Identifier，它是用户的标识。Linux通过UID来区分一个个账户。 UID是一个0-65535范围内的数字，不同的取值有不同的含义：

* UID=0 表示该账户为系统管理员。root的UID即为0.当然，你可把其他账户的UID设为0，此时该账户就拥有与root一样的权限。
* UID=1~99 Linux自行创建的账户，权限和普通账户没啥区别，只是为了规范。
* UID=100~499 系统操作相关的账户。这些账户是由用户创建的，权限和普通账号的权限没啥区别，也只是为了规范。
* UID=500~65535 普通用户的UID。

#### **1.4. GID**

GID＝Group Identifier，他是用户组的ID。它的取值范围和UID遵循一样的规范。

#### **1.5. 账号说明信息**

这个字段是账户的说明信息，并没什么卵用，只是为了看看。

#### **1.6. 用户主目录**

每个账户都有一个主目录，用户可以在各自的主目录下为所欲为。 这个字段就是设置账户的主目录的路径。

#### **1.7. 该账户登录后的shell**

当一个用户登录后，Linux会给该用户分配一个shell来和Linux内核通信，这样该用户就能操控这台计算机了。那么Linux的shell有好多种，究竟给登录用户分配哪个shell呢？这就是在这个参数中设定。 当然，如果不想让用户操控这台计算机的话，可以将这个参数设为/sbin/nologin，这样在用户登录后Linux不会给他分配shell，从而该用户无法操控这台机器。

## **2./etc/shadow**

这个文件存放所有账号的密码，它的权限为-r——–，也就是只有root才能查看这个文件，它具体的内容如下：

root:$1$cfvc4Vwh$KuVS9yJeWOcLkEXKbTryf.:16477:0:99999:7:::

bin:\*:16300:0:99999:7:::

daemon:\*:16300:0:99999:7:::

adm:\*:16300:0:99999:7:::

lp:\*:16300:0:99999:7:::

和passwd一样，也是一行代表一个账号，参数之间用:分隔，每个参数的含义如下：

#### **2.1 账号名称**

#### **2.2 密码**

这里才是账号真正的密码存放的地方，它采用了MD5消息摘要。

#### **2.3 最近修改密码的日期**

这个字段记录了上次修改密码的时间。 这个字段表示从1970-1-1到上次修改密码的天数。 如上述root密码的修改时间为16477，也就是1970-1-1往后16477天，也就是2015年2月11日。

#### **2.4 密码不可以被修改的天数**

这个值是从上次修改时间算起，密码不可以被修改的天数。

#### **2.5 密码需要被重新设置的天数**

从第三个字段开始，经过多少天之后就需要重新设置密码。

#### **2.6 密码更改期限前的发出警告的天数**

#### **2.7 密码过期后的宽限天数**

在宽限期间内账号仍然可以登录，只不过会给出改密码的提示。

#### **2.8 账号失效日期**

这个值也是从1970-1-1往后的天数，到了那天，这个账号就无法登录。 收费服务可以使用这个功能。

#### **2.9 保留字段**

保留字段目前还没有任何作用。

## **3./etc/group**

这个文件中记录的是GID和用户名名的对应关系。具体内容如下：

root:x:0:root

bin:x:1:root,bin,daemon

daemon:x:2:root,bin,daemon

sys:x:3:root,bin,adm

#### **3.1 用户组名**

#### **3.2 用户组密码**

用户组密码很少使用，这个密码由用户组管理员设置。 当然和用户密码一样，用户组的密码也存储在一个独立文件中：/etc/gshadow

#### **3.3 GID**

用户组的ID

#### **3.4 该组的用户的用户名**

本组成员的用户名。多个成员间用,隔开，注意：不能有任何空格！

## **4./etc/gshadow**

这个文件用于存储用户组的密码，这个文件主要作用是：如果root管理员非常忙，没空管每个用户组的成员，那么可以指定几个用户组管理员，专门管理每组成员的添加与删除。 本文件具体内容如下：

root:::root

bin:::root,bin,daemon

daemon:::root,bin,daemon

#### **4.1 用户组名**

#### **4.2 密码**

密码为!表示本用户组无管理员。

#### **4.3 本用户组的管理员账号**

#### **4.4 该用户组的成员账号**

# **账号管理**

了解了这些文件的构成后，接下来我们使用目录来操作这些文件，从而实现用户账号的管理。

## **1. 新增用户：useradd**

useradd [-参数 参数值] 用户名

参数详解：

### **1.1 -u**

为账户指定一个UID。

### **1.2 -g**

为账户指定一个初始化的GID。

### **1.3 -G**

为账户指定还需要加入的用户组。

### **1.4 -M**

### **1.5 -m**

### **1.6 -c**

设置账户的描述信息。

### **1.7 -d**

设置账户的主目录。

### **1.8 -r**

创建一个系统管理员账号。

### **1.9 -s**

指定这个账户的shell。

### **1.10 -e**

设置账号失效日期，格式为：YYYY-MM-DD。

### **1.11 -f**

设置密码失效日。-1表示永不失效。

### **1.12 -D**

查看useradd默认的参数值。 当不用任何参数创建账号时，会使用useradd默认的参数创建，我们可以使用useradd -D查看默认的参数，如下：

GROUP=100#默认的用户组GID为100

HOME=/home#默认在/home下创建用户主目录

INACTIVE=-1#密码默认用户失效

EXPIRE=#账号也用户失效

SHELL=/bin/bash#默认的shell为bash

SKEL=/etc/skel#将这个路径下的目录复制一份作为用户主目录

CREATE\_MAIL\_SPOOL=yes#是否用户的mailbox

## **2. 设置密码：passwd、chage**

这两个命令其实都是对/etc/shadow文件中的内容进行修改。passwd能设置/etc/passwd文件中的一部分参数，而chage能设置另一部分参数。

### **2.1 passwd**

passwd [-参数 参数值] 用户名

PS：不加用户名表示修改当前账户的密码。

#### **2.1.1 -l**

将账号锁住，实则将shadow文件中的密码改为！

#### **2.1.2 -u**

将账号解锁。

#### **2.1.3 -S**

列出该用户密码的详细信息。

#### **2.1.4 -n**

设置不可以修改密码的天数。

#### **2.1.5 -x**

设置必须要修改密码的天数。

#### **2.1.6 -w**

设置过期前警告的天数。

#### **2.1.7 -i**

设置密码失效日期。

### **2.2 chage**

chage [-参数 参数值] 用户名

#### **2.2.1 -l**

列出该账号的详细密码信息。

#### **2.2.2 -d**

查看该账号的最近一次改密时间。

#### **2.2.3 -E**

设置账号失效日。

#### **2.2.4 -I**

设置密码失效日。

#### **2.2.5 -m**

设置密码不可以被修改的天数。

#### **2.2.6 -M**

设置密码有效期。

#### **2.2.7 -W**

设置密码过期警告的天数。

## **3. 修改账户信息：usermod**

该命令用于修改/etc/passwd和/etc/shadow中的信息。

usermod [-参数 参数值] 用户名

#### **3.1 -L**

冻结该账号。其实将shadow的密码改为！

#### **3.2 -U**

解冻该账号。

#### **3.3 其他参数请自行查阅。**

## **4. 删除账户：userdel**

userdel [-r] username-r ：将用户相关的数据均删除，包括如下数据：/etc/passwd和/etc/shadow中的账户数据/etc/group和/etc/gshadow中的分组信息/home/username和/var/spool/mail/username中的个人数据

find /backup/\* -type f -mtime +5 -name "\*.tar.gz" -exec rm {} \; #删除该文件夹下超过5天的文件

su命令

语法：

su(选项)(参数)

选项：

-c<指令>或--command=<指令>：执行完指定的指令后，即恢复原来的身份；

-f或——fast：适用于csh与tsch，使shell不用去读取启动文件；

-l或——login：改变身份时，也同时变更工作目录，以及HOME,SHELL,USER,logname。此外，也会变更PATH变量；

-m,-p或--preserve-environment：变更身份时，不要变更环境变量；

-s<shell>或--shell=<shell>：指定要执行的shell；

--help：显示帮助；

--version；显示版本信息。

实例：

变更帐号为root并在执行ls指令后退出变回原使用者：

su -lc ls root

变更帐号为root并传入-f选项给新执行的shell：

su root -f

变更帐号为test并改变工作目录至test的家目录：

su - test

了解su命令后，自然对su－应该也掌握了吧。

su命令和su -命令最大的本质区别就是：前者只是切换了root身份，但Shell环境仍然是普通用户的Shell；而后者连用户和Shell环境一起切换成root身份了。只有切换了Shell环境才不会出现PATH环境变量错误。su切换成root用户以后，pwd一下，发现工作目录仍然是普通用户的工作目录；而用su -命令切换以后，工作目录变成root的工作目录了。

# **[Linux下实现 OpenSSL 简单加密与解密字符串](https://www.cnblogs.com/linuxprobe-sarah/p/10129520.html)**

**场景**

shell脚本中存在明文密码  
客户要求禁止使用明文密码,密码做加密处理.

**方案**

在网上了解到了[Linux](https://www.linuxprobe.com/" \o "linux" \t "https://www.cnblogs.com/linuxprobe-sarah/p/_blank) OpenSSL加密解密工具  
可以指定各种加密算法为字符,文件做加密处理.  
加密的案例比较多,解密的寥寥无几.  
有兴趣的可以去查下中文教程  
案例中使用加密算法 : AES

**实例**

##所以命令都在Linux中执行##

##查看是否安装了该工具,我的环境自带了,文章不做安装描述##

openssl

##加密(执行后获得加密码)##

echo abc | openssl aes-128-cbc（加密算法） -k 123（密码自己设置） -base64

得到一串字符串，解密需要用到该字符串

##解密(加密码太长,用\*表示了)##

echo \*\*\*\*\* | openssl aes-128-cbc -d（解密） -k 123 -base64

#### cp 命令

* -a：此选项通常在复制目录时使用，它保留链接、文件属性，并复制目录下的所有内容。其作用等于dpR参数组合。
* -d：复制时保留链接。这里所说的链接相当于Windows系统中的快捷方式。
* -f：覆盖已经存在的目标文件而不给出提示。
* -i：与-f选项相反，在覆盖目标文件之前给出提示，要求用户确认是否覆盖，回答"y"时目标文件将被覆盖。
* -p：除复制文件的内容外，还把修改时间和访问权限也复制到新文件中。
* -r：若给出的源文件是一个目录文件，此时将复制该目录下所有的子目录和文件。
* -l：不复制文件，只是生成链接文件。

alias cp 如果返回有cp=’cp -i’，就说明cp有别名，需要取消别名，不然cp覆盖已有文件时，都会有提示信息

unalias cp #取消别名，此时cp不会有提示信息，或者用cp -rf ，也不会有提示信息

alias cp=’cp-i’ #再恢复别名



\cp a b 或者 /bin/cp a b

Yes | cp a b

**grep** 同时满足多个关键字和满足任意关键字

① grep -E "word1|word2|word3"   file.txt

   满足任意条件（word1、word2和word3之一）将匹配。

② grep word1 file.txt | grep word2 |grep word3

   必须同时满足三个条件（word1、word2和word3）才匹配

cat ./$(ls -tr) >> nps.log #把一个文件夹下所有文件的内容按时间先后顺序，合并到另一个文件中

Sed

Awk