



# 进程，系统性能和计划任务

讲师：王晓春

# 本章内容



- ◆ 进程相关概念
- ◆ 进程工具
- ◆ 系统性能相关工具
- ◆ 计划任务

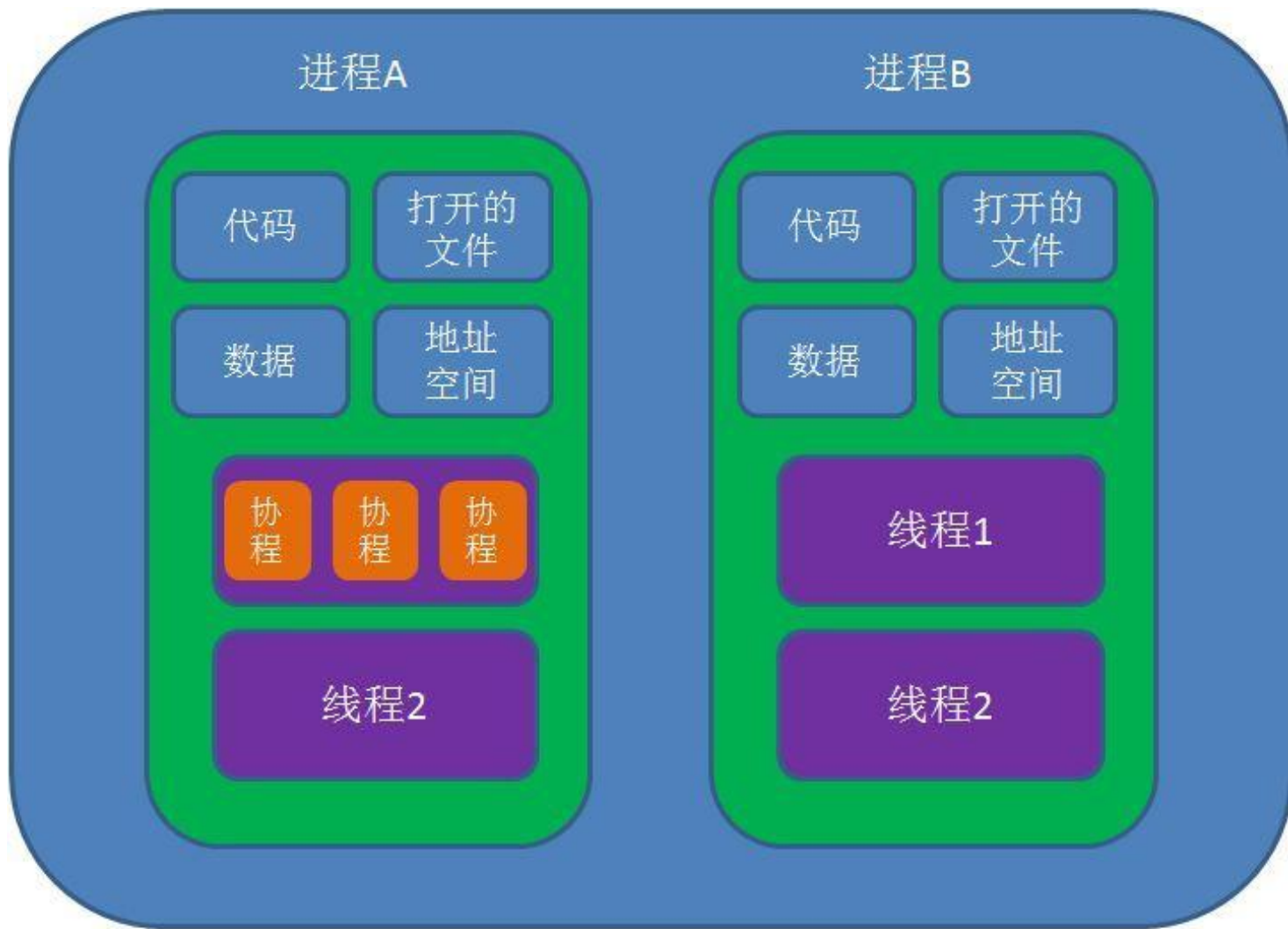
- ◆ 内核功用：进程管理、文件系统、网络功能、内存管理、驱动程序、安全功能等
- ◆ Process: 运行中的程序的一个副本，是被载入内存的一个指令集合
  - 进程ID ( Process ID , PID ) 号码被用来标记各个进程
  - UID、GID、和SELinux语境决定对文件系统的存取和访问权限
  - 通常从执行进程的用户来继承
  - 存在生命周期
- ◆ task struct : Linux内核存储进程信息的数据结构格式
- ◆ task list : 多个任务的task struct组成的链表
- ◆ 进程创建：
  - init : 第一个进程
  - 进程：都由其父进程创建，fork()，父子关系，CoW

# 进程，线程和协程



马哥教育

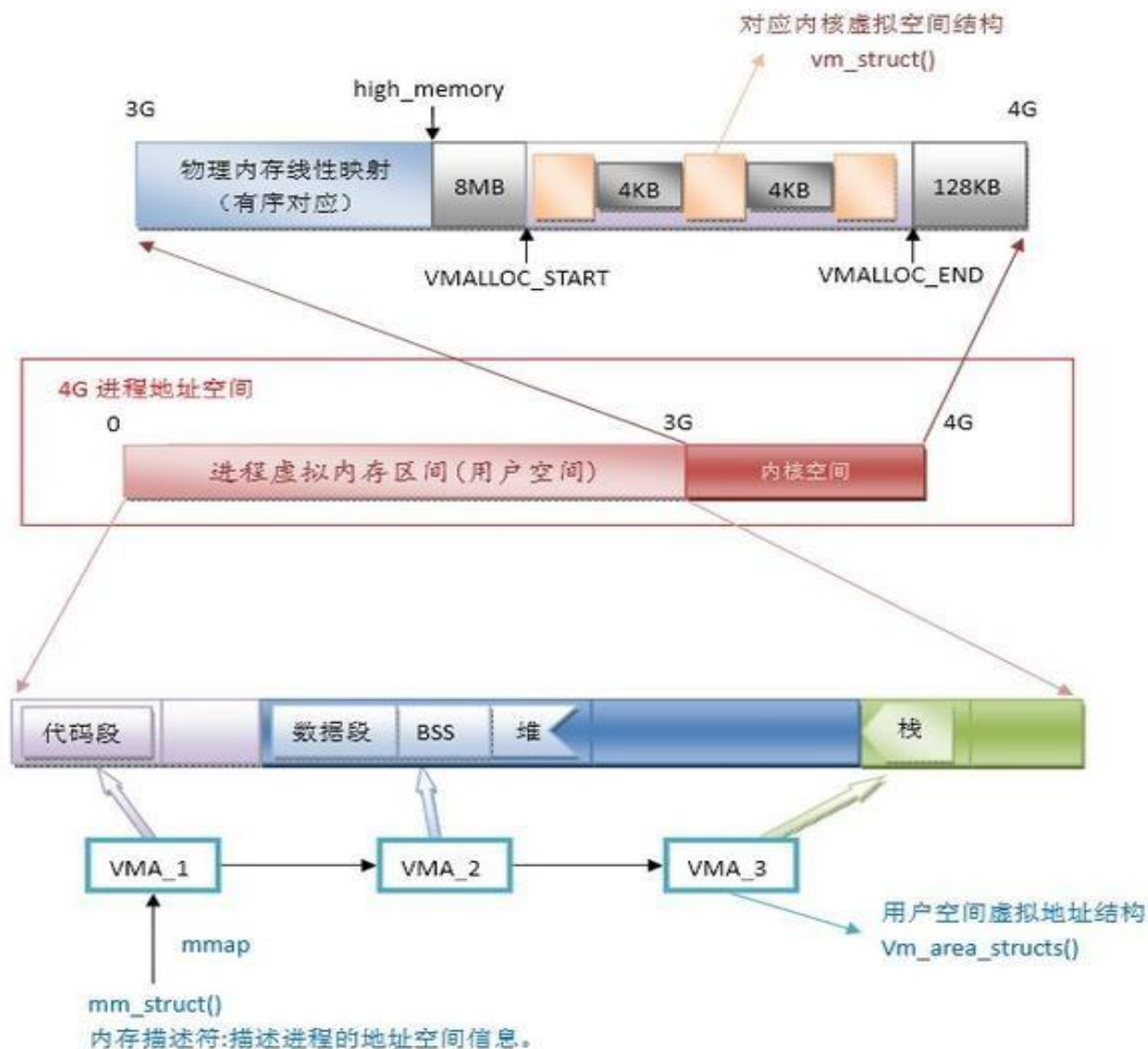
IT 人的高薪职业学院



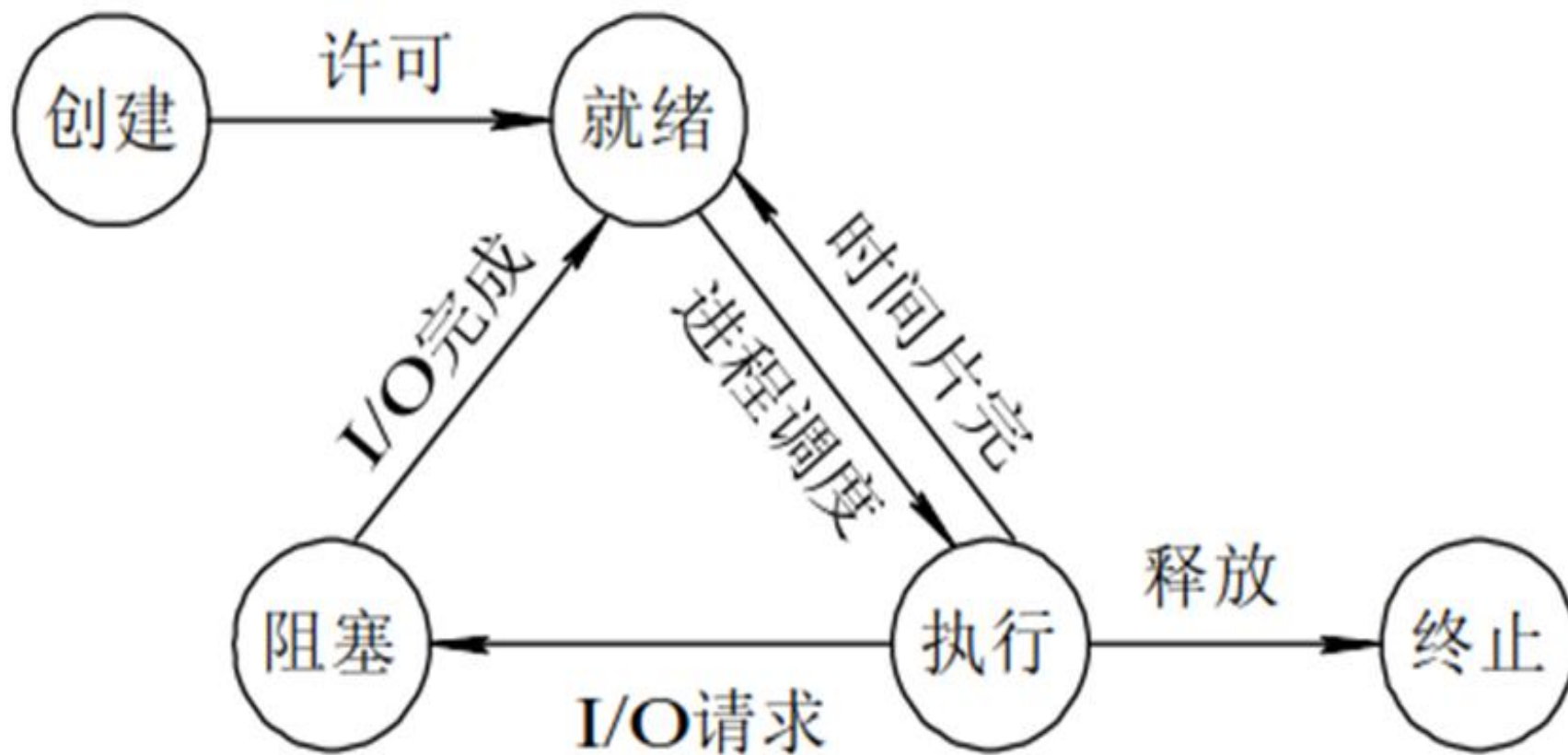
操作系统

- ◆ 查看线程：cat /proc/PID/status |grep -i threads
- ◆ Page Frame: 页框，用存储页面数据，存储Page 4k
- ◆ 物理地址空间和线性地址空间
- ◆ MMU：Memory Management Unit 负责转换线性和物理地址
- ◆ TLB：Translation Lookaside Buffer 翻译后备缓冲器  
用于保存虚拟地址和物理地址映射关系的缓存
- ◆ LRU：Least Recently Used 近期最少使用算法,释放内存

# 用户和内核空间



# 进程的基本状态和转换



- ◆ 创建状态：进程在创建时需要申请一个空白PCB(process control block进程控制块)，向其中填写控制和管理进程的信息，完成资源分配。如果创建工作无法完成，比如资源无法满足，就无法被调度运行，把此时进程所处状态称为创建状态
- ◆ 就绪状态：进程已准备好，已分配到所需资源，只要分配到CPU就能够立即运行
- ◆ 执行状态：进程处于就绪状态被调度后，进程进入执行状态
- ◆ 阻塞状态：正在执行的进程由于某些事件（I/O请求，申请缓存区失败）而暂时无法运行，进程受到阻塞。在满足请求时进入就绪状态等待系统调用
- ◆ 终止状态：进程结束，或出现错误，或被系统终止，进入终止状态。无法再执行



# 状态之间转换六种情况

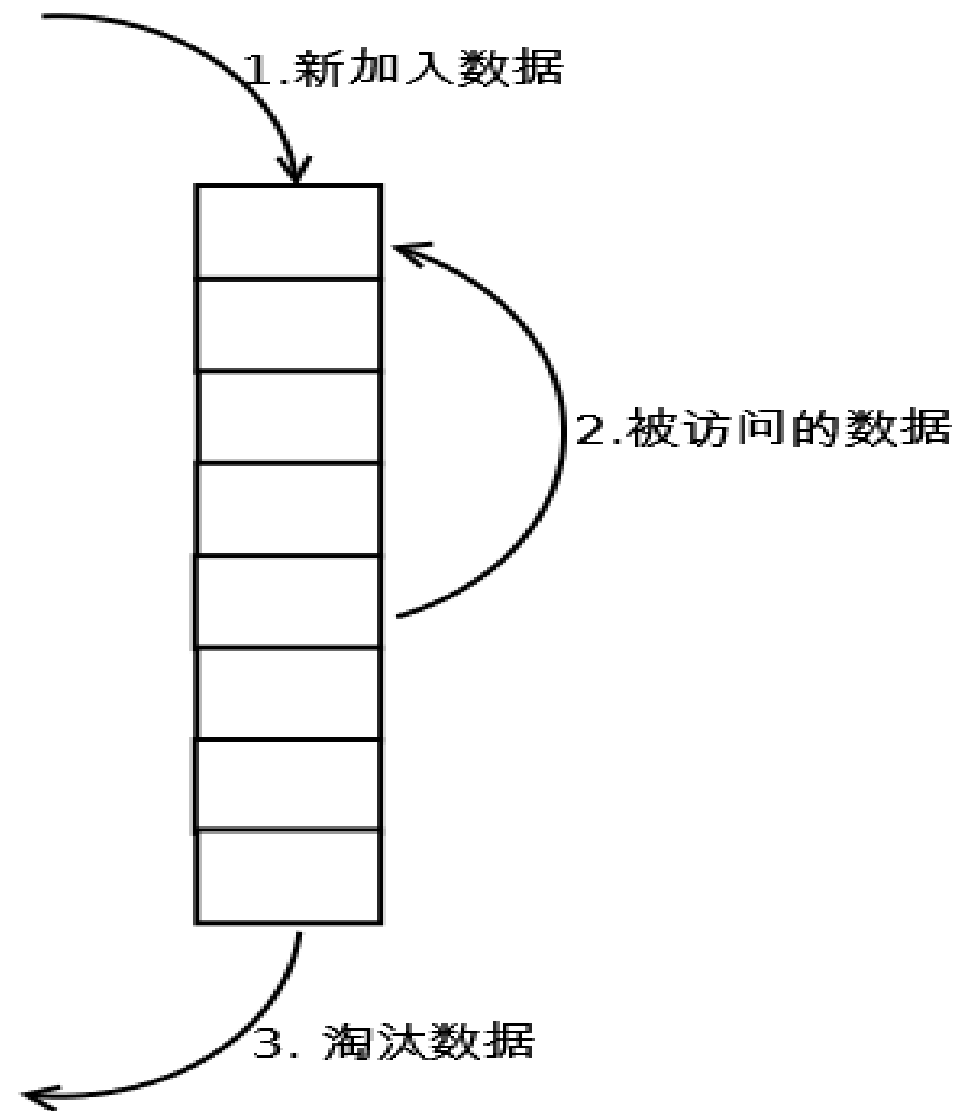
- ◆ 运行——>就绪：1，主要是进程占用CPU的时间过长，而系统分配给该进程占用CPU的时间是有限的；2，在采用抢先式优先级调度算法的系统中,当有更高优先级的进程要运行时，该进程就被迫让出CPU，该进程便由执行状态转变为就绪状态
- ◆ 就绪——>运行：运行的进程的时间片用完，调度就转到就绪队列中选择合适的进程分配CPU
- ◆ 运行——>阻塞：正在执行的进程因发生某等待事件而无法执行，则进程由执行状态变为阻塞状态，如发生了I/O请求
- ◆ 阻塞——>就绪:进程所等待的事件已经发生，就进入就绪队列
- ◆ 以下两种状态是不可能发生的：
- ◆ 阻塞——>运行：即使给阻塞进程分配CPU，也无法执行，操作系统在进行调度时不会从阻塞队列进行挑选，而是从就绪队列中选取
- ◆ 就绪——>阻塞：就绪态根本就没有执行，谈不上进入阻塞态

# LRU算法



◆ 假设序列为 4 3 4 2 3 1 4 2  
物理块有3个，则

- ◆ 第1轮 4调入内存 4
- ◆ 第2轮 3调入内存 3 4
- ◆ 第3轮 4调入内存 4 3
- ◆ 第4轮 2调入内存 2 4 3
- ◆ 第5轮 3调入内存 3 2 4
- ◆ 第6轮 1调入内存 1 3 2
- ◆ 第7轮 4调入内存 4 1 3
- ◆ 第8轮 2调入内存 2 4 1



## ◆ IPC: Inter Process Communication

同一主机：pipe      管道

socket    套接字文件

signal    信号

shm      shared memory

semaphore    信号量，一种计数器

不同主机：socket    IP和端口号

RPC      remote procedure call

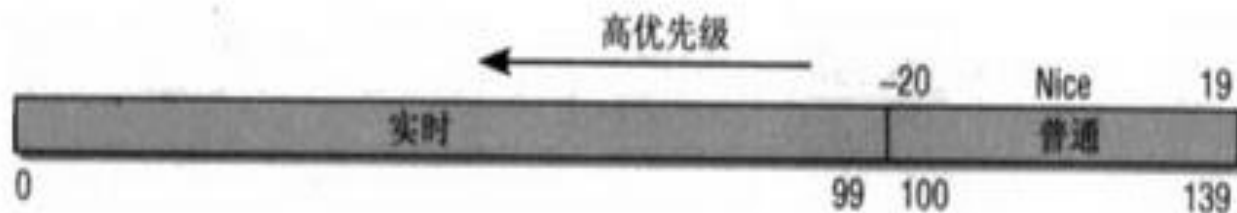
MQ      消息队列，如：Kafka，RabbitMQ，ActiveMQ

# 进程优先级

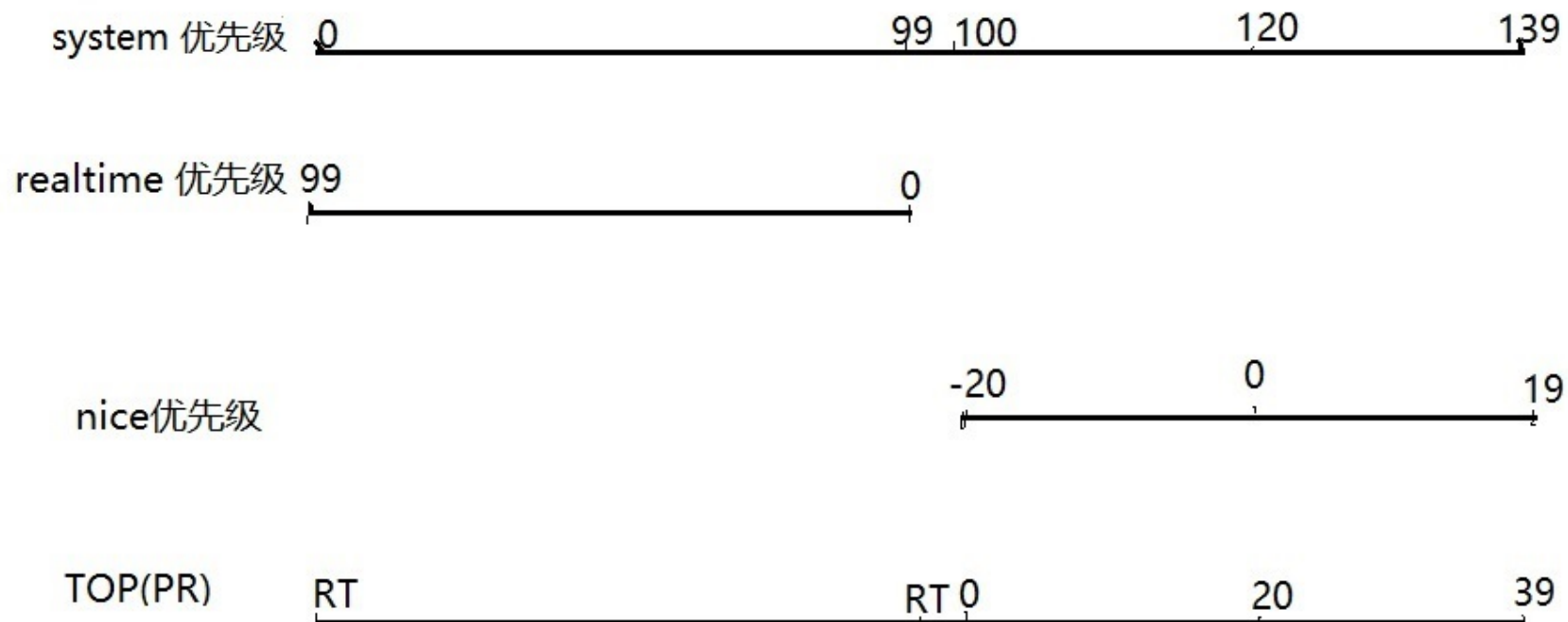


linux2.6内核将任务优先级进行了一个划分, 实时优先级范围是0到MAX\_RT\_PRIO-1 (即99), 而普通进程的静态优先级范围是从MAX\_RT\_PRIO到MAX\_PRIO-1 (即100到139),

优先级范围	描述
0——99	实时进程
100——139	非实时进程



## Centos 优先级



## ◆ 进程优先级：

系统优先级：数字越小，优先级越高

0-139：各有140个运行队列和过期队列

实时优先级：99-0 值最大优先级最高

nice值：-20到19，对应系统优先级100-139

## ◆ Big O：时间复杂度，用时和规模的关系

$O(1)$ ,  $O(\log n)$ ,  $O(n)$ 线性,  $O(n^2)$ 抛物线,  $O(2^n)$

- ◆ Linux内核：抢占式多任务

- ◆ 进程类型：

  - 守护进程: daemon, 在系统引导过程中启动的进程，和终端无关进程

  - 前台进程：跟终端相关，通过终端启动的进程

  - 注意：两者可相互转化

- ◆ 进程状态：

  - 运行态：running

  - 就绪态：ready

  - 睡眠态：

    - 可中断：interruptable

    - 不可中断：uninterruptable

  - 停止态：stopped, 暂停于内存，但不会被调度，除非手动启动

  - 僵死态：zombie，结束进程，父进程结束前，子进程不关闭

## ◆ 进程的分类：

CPU-Bound：CPU密集型，非交互

IO-Bound：IO密集型，交互

## ◆ Linux系统状态的查看及管理工具：pstree, ps, pidof, pgrep, top, htop, glance, pmap, vmstat, dstat, kill, pkill, job, bg, fg, nohup

## ◆ pstree命令：

pstree            display a tree of processes

## ◆ ps: process state

ps                report a snapshot of the current processes

Linux系统各进程的相关信息均保存在/proc/PID目录下的各文件中



# 查看进程进程ps



- ◆ ps [OPTION]...
- ◆ 支持三种选项：
  - UNIX选项 如-A -e
  - BSD选项 如a
  - GNU选项 如--help
- 选项：默认显示当前终端中的进程
  - a 选项包括所有终端中的进程
  - x 选项包括不链接终端的进程
  - u 选项显示进程所有者的信息
  - f 选项显示进程树,相当于 --forest
  - k|--sort 属性 对属性排序,属性前加- 表示倒序
  - o 属性... 选项显示定制的信息 pid、cmd、%cpu、%mem
  - L 显示支持的属性列表

- ◆ -C cmdlist 指定命令，多个命令用，分隔
- ◆ -L 显示线程
- ◆ -e 显示所有进程，相当于-A
- ◆ -f 显示完整格式程序信息
- ◆ -F 显示更完整格式的进程信息
- ◆ -H 以进程层级格式显示进程相关信息
- ◆ -u userlist 指定有效的用户ID或名称
- ◆ -U userlist 指定真正的用户ID或名称
- ◆ -g gid或groupname 指定有效的gid或组名称
- ◆ -G gid或groupname 指定真正的gid或组名称
- ◆ -p pid 显示指pid的进程
- ◆ --ppid pid 显示属于pid的子进程
- ◆ -t ttylist 指定tty,相当于 t
- ◆ -M 显示SELinux信息，相当于Z

# ps 输出属性



- ◆ VSZ: Virtual memory SiZe , 虚拟内存集 , 线性内存
- ◆ RSS: ReSident Size, 常驻内存集
- ◆ STAT : 进程状态
  - R : running
  - S: interruptable sleeping
  - D: uninterruptable sleeping
  - T: stopped
  - Z: zombie
  - +: 前台进程
  - l: 多线程进程
  - L : 内存分页并带锁
  - N : 低优先级进程
  - <: 高优先级进程
  - s: session leader , 会话 ( 子进程 ) 发起者

# ps



- ◆ ni: nice值
- ◆ pri: priority 优先级
- ◆ rtprio: 实时优先级
- ◆ psr: processor CPU编号
- ◆ 示例：

```
ps axo pid,cmd,psr,ni,pri,rtprio
```

- ◆ 常用组合：

```
aux
```

```
-ef
```

```
-eFH
```

```
-eo pid,tid,class,rtprio,ni,pri,psr,pcpu,stat,comm
```

```
axo stat,euid,ruid,tty,tpgid,sess,pgrp,ppid,pid,pcpu,comm
```

- ◆ 查询你拥有的所有进程

```
ps -x
```

- ◆ 显示指定用户名(RUID)或用户ID的进程

```
ps -fU apache
```

```
ps -fU 48
```

- ◆ 显示指定用户名(EUID)或用户ID的进程

```
ps -fu wang
```

```
ps -fu 1000
```

- ◆ 查看以root用户权限（实际和有效ID）运行的每个进程

```
ps -U root -u root
```

- ◆ 列出某个组拥有的所有进程（实际组ID：RGID或名称）

```
ps -fG nginx
```

- ◆ 列出有效组名称（或会话）所拥有的所有进程

```
ps -fg mysql
```

```
ps -fg 27
```

- ◆ 显示指定的进程ID对应的进程

```
ps -fp 1234
```

- ◆ 以父进程ID来显示其下所有的进程，如显示父进程为1234的所有进程

```
ps -f --ppid 1234
```

- ◆ 显示指定PID的多个进程

```
ps -fp 1204,1239,1263
```

- ◆ 要按tty显示所属进程

```
ps -ft pts/0
```

- ◆ 以进程树显示系统中的进程如何相互链接

```
ps -e --forest
```

- ◆ 以进程树显示指定的进程

```
ps -f --forest -C sshd
```

```
ps -ef --forest | grep -v grep | grep sshd
```

- ◆ 要显示一个进程的所有线程,将显示LWP ( 轻量级进程 ) 以及NLWP ( 轻量级进程数 ) 列

```
ps -fL -C nginx
```

- ◆ 要列出所有格式说明符

```
ps L
```

- ◆ 查看进程的PID , PPID , 用户名和命令

```
ps -eo pid,ppid,user,cmd
```

- ◆ 自定义格式显示文件系统组,ni值开始时间和进程的时间

```
ps -p 1234 -o pid,ppid,fgroup,ni,lstart,etime
```

- ◆ 使用其PID查找进程名称：

```
ps -p 1244 -o comm=
```

- ◆ 要以其名称选择特定进程，显示其所有子进程

```
ps -C sshd,bash
```

- ◆ 查找指定进程名所有的所属PID，在编写需要从std输出或文件读取PID的脚本时这个参数很有用

```
ps -C httpd,sshd -o pid=
```

- ◆ 检查一个进程的执行时间

```
ps -eo comm,etime,user | grep nginx
```



## ◆ 查找占用最多内存和CPU的进程

```
ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head
```

```
ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%cpu | head
```

## ◆ 显示安全信息

```
ps -eM
```

```
ps --context
```

## ◆ 使用以下命令以用户定义的格式显示安全信息

```
ps -eo euser,ruser,suser,fuser,f,comm,label
```

## ◆ 使用watch实用程序执行重复的输出以实现对进程进行实时的监视，如下面的命令显示每秒钟的监视

```
watch -n 1 'ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head'
```

## ◆ 进程优先级调整

静态优先级：100-139

进程默认启动时的nice值为0，优先级为120

只有根用户才能降低nice值（提高优先性）

## ◆ nice命令

`nice [OPTION] [COMMAND [ARG]...]`

## ◆ renice命令

`renice [-n] priority pid...`

## ◆ 查看

`ps axo pid,comm,ni`

◆ 最灵活：ps 选项 | 其它命令

◆ 按预定义的模式：pgrep

pgrep [options] pattern

-u uid: effective user , 生效者

-U uid: real user , 真正发起运行命令者

-t terminal: 与指定终端相关的进程

-l: 显示进程名

-a: 显示完整格式的进程名

-P pid: 显示指定进程的子进程

◆ 按确切的程序名称：/sbin/pidof

pidof bash

◆ /proc/uptime 包括两个值，单位 s

系统启动时长，空闲进程的总时长（按总的CPU核数计算）

◆ uptime 和 w

显示当前时间，系统已启动的时间、当前上线人数，系统平均负载（1、5、15分钟的平均负载，一般不会超过1，超过5时建议警报）

□ 系统平均负载:

指在特定时间间隔内运行队列中的平均进程数

□ 通常每个CPU内核的当前活动进程数不大于3，那么系统的性能良好。如果每个CPU内核的任务数大于5，那么此主机的性能有严重问题

□ 如果linux主机是1个双核CPU，当Load Average 为6的时候说明机器已经被充分使用

## ◆ top : 有许多内置命令

排序 :

P : 以占据的CPU百分比,%CPU

M : 占据内存百分比,%MEM

T : 累积占据CPU时长,TIME+

首部信息显示 :

uptime信息 : l命令

tasks及cpu信息 : t命令

cpu分别显示 : 1 (数字)

memory信息 : m命令

退出命令 : q

修改刷新时间间隔 : s

终止指定进程 : k

保存文件 : W

# top命令



## ◆ 栏位信息简介

us : 用户空间

sy : 内核空间

ni : 调整nice时间

id : 空闲

wa : 等待IO时间

hi : 硬中断

si : 软中断 ( 模式切换 )

st : 虚拟机偷走的时间

## ◆ 选项：

- d # 指定刷新时间间隔，默认为3秒
- b 全部显示所有进程
- n # 刷新多少次后退出
- H 线程模式，示例：top -H -p `pidof mysqld`

## ◆ htop命令：EPEL源

### 选项：

- d #: 指定延迟时间；
- u UserName: 仅显示指定用户的进程
- s COLUMN: 以指定字段进行排序

### 子命令：

- s：跟踪选定进程的系统调用
- l：显示选定进程打开的文件列表
- a：将选定的进程绑定至某指定CPU核心
- t：显示进程树

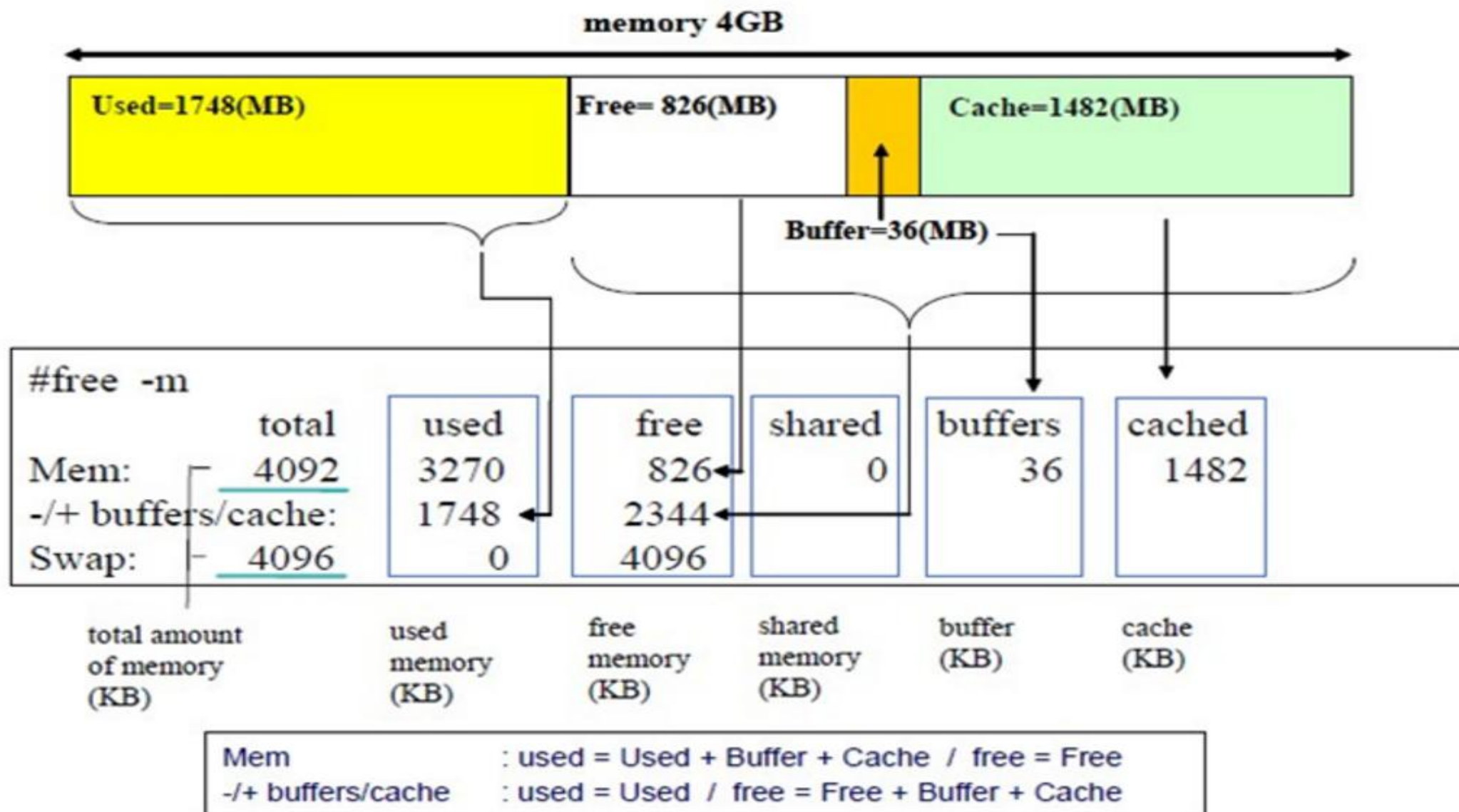
## ◆ 内存空间使用状态：

free [OPTION]

- b 以字节为单位
- m 以MB为单位
- g 以GB为单位
- h 易读格式
- o 不显示-/+buffers/cache行
- t 显示RAM + swap的总和
- s n 刷新闻隔为n秒
- c n 刷新n次后即退出



# Free命令



- ◆ vmstat命令：虚拟内存信息

vmstat [options] [delay [count]]

vmstat 2 5

- ◆ procs:

r：可运行（正运行或等待运行）进程的个数，和核心数有关

b：处于不可中断睡眠态的进程个数(被阻塞的队列的长度)

- ◆ memory：

swpd: 交换内存的使用总量

free：空闲物理内存总量

buffer：用于buffer的内存总量

cache：用于cache的内存总量

- ◆ swap:

si：从磁盘交换进内存的数据速率(kb/s)

so：从内存交换至磁盘的数据速率(kb/s)

## ◆ io :

bi : 从块设备读入数据到系统的速率(kb/s)

bo: 保存数据至块设备的速率

## ◆ system :

in: interrupts 中断速率, 包括时钟

cs: context switch 进程切换速率

## ◆ cpu :

us: Time spent running non-kernel code

sy: Time spent running kernel code

id: Time spent idle. Linux 2.5.41前,包括IO-wait time.

wa: Time spent waiting for IO. 2.5.41前, 包括in idle.

st: Time stolen from a virtual machine. 2.6.11前, unknown.

## ◆ 选项 :

-s: 显示内存的统计数据

- ◆ iostat : 统计CPU和设备IO信息  
示例 : `iostat 1 10`
- ◆ iftop : 显示带宽使用情况 , EPEL源  
示例 : `iftop -n -i eth1`
- ◆ pmap命令 : 进程对应的内存映射
- ◆ `pmap [options] pid [...]`
  - x: 显示详细格式的信息示例 : `pmap 1`
- ◆ 另外一种实现  
`cat /proc/PID/maps`

- ◆ glances命令：EPEL源
- ◆ glances [-bdehmnrsvyz1] [-B bind] [-c server] [-C conffile] [-p port] [-P password] [--password] [-t refresh] [-f file] [-o output]
- ◆ 内建命令：
  - a Sort processes automatically
  - c Sort processes by CPU%
  - m Sort processes by MEM%
  - p Sort processes by name
    - i Sort processes by I/O rate
  - d Show/hide disk I/O stats
  - f Show/hide file system stats
  - n Show/hide network stats
  - s Show/hide sensors stats
  - y Show/hide hddtemp stats
  - l Show/hide logs
  - b Bytes or bits for network I/O
  - w Delete warning logs
  - x Delete warning and critical logs
  - 1 Global CPU or per-CPU stats
  - h Show/hide this help screen
  - t View network I/O as combination
  - u View cumulative network I/O
  - q Quit (Esc and Ctrl-C also work)

## ◆ 常用选项：

- b: 以Byte为单位显示网卡数据速率
- d: 关闭磁盘I/O模块
- f /path/to/somefile: 设定输入文件位置
- o {HTML|CSV}：输出格式
- m: 禁用mount模块
- n: 禁用网络模块
- t #: 延迟时间间隔
- 1：每个CPU的相关数据单独显示

## ◆ C/S模式下运行glances命令

### ◆ 服务器模式：

`glances -s -B IPADDR`

IPADDR: 指明监听的本机哪个地址

### ◆ 客户端模式：

`glances -c IPADDR`

IPADDR：要连入的服务器端地址

◆ dstat命令：系统资源统计,代替vmstat,iostat

◆ dstat [-afv] [options..] [delay [count]]

-c 显示cpu相关信息

-C #,#,...,total

-d 显示disk相关信息

-D total,sda,sdb,...

-g 显示page相关统计数据

-m 显示memory相关统计数据

-n 显示network相关统计数据

-p 显示process相关统计数据

-r 显示io请求相关的统计数据

-s 显示swapped相关的统计数据



- ◆ --tcp
- ◆ --udp
- ◆ --unix
- ◆ --raw
- ◆ --socket
- ◆ --ipc
- ◆ --top-cpu : 显示最占用CPU的进程
- ◆ --top-io: 显示最占用io的进程
- ◆ --top-mem: 显示最占用内存的进程
- ◆ --top-latency: 显示延迟最大的进程

- ◆ iostat命令是一个用来监视磁盘I/O使用状况的top类工具iostat具有与top相似的UI，其中包括PID、用户、I/O、进程等相关信息，可查看每个进程是如何使用IO
- ◆ iostat输出
  - 第一行：Read和Write速率总计
  - 第二行：实际的Read和Write速率
  - 第三行：参数如下：
    - 线程ID（按p切换为进程ID）
    - 优先级
    - 用户
    - 磁盘读速率
    - 磁盘写速率
    - swap交换百分比
    - IO等待所占的百分比
    - 线程/进程命令

# iotop常用参数

- ◆ -o, --only 只显示正在产生I/O的进程或线程，除了传参，可以在运行过程中按o生效
- ◆ -b, --batch 非交互模式，一般用来记录日志
- ◆ -n NUM, --iter=NUM 设置监测的次数，默认无限。在非交互模式下很有用
- ◆ -d SEC, --delay=SEC 设置每次监测的间隔，默认1秒，接受非整形数据例如1.1
- ◆ -p PID, --pid=PID 指定监测的进程/线程
- ◆ -u USER, --user=USER 指定监测某个用户产生的I/O
- ◆ -P, --processes 仅显示进程，默认iotop显示所有线程
- ◆ -a, --accumulated 显示累积的I/O，而不是带宽
- ◆ -k, --kilobytes 使用kB单位，而不是对人友好的单位。在非交互模式下，脚本编程有用

# iostat常用参数和快捷键



- ◆ -t, --time 加上时间戳，非交互非模式
- ◆ -q, --quiet 禁止头几行，非交互模式，有三种指定方式
  - -q 只在第一次监测时显示列名
  - -qq 永远不显示列名
  - -qqq 永远不显示I/O汇总
- ◆ 交互按键
  - left和right方向键：改变排序
  - r：反向排序
  - o：切换至选项--only
  - p：切换至--processes选项
  - a：切换至--accumulated选项
  - q：退出
  - i：改变线程的优先级

# nload 查看网络实时吞吐量

- ◆ nload是一个实时监控网络流量和带宽使用情况，以数值和动态图展示进出的流量情况
- ◆ 安装：yum install nload (EPEL源)
- ◆ 界面操作
  - 上下方向键、左右方向键、enter键或者tab键都就可以切换查看多个网卡的流量情况
  - 按 F2 显示选项窗口
  - 按 q 或者 Ctrl+C 退出 nload
- ◆ 示例：
  - nload：默认只查看第一个网络的流量进出情况
  - nload eth0 eth1：在nload后面指定网卡，可以指定多个
- ◆ 设置刷新闻隔：默认刷新闻隔是100毫秒，可通过 -t命令设置刷新时间（单位是毫秒）  
nload -t 500 eth0
- ◆ 设置单位：显示两种单位一种是显示Bit/s、一种是显示Byte/s，默认是以Bit/s，也可不显示/s
  - u h|b|k|m|g|H|B|K|M|G 表示的含义：h: auto, b: Bit/s, k: kBit/s, m: MBit/s, H: auto, B: Byte/s, K: kByte/s, M: MByte/s
  - nload -u M eth0

- ◆ lsuf : list open files查看当前系统文件的工具。在linux环境下，一切皆文件，用户通过文件不仅可以访问常规数据，还可以访问网络连接和硬件如传输控制协议 (TCP) 和用户数据报协议 (UDP)套接字等，系统在后台都为该应用程序分配了一个文件描述符
- ◆ 命令参数
  - -a : 列出打开文件存在的进程
  - -c<进程名> : 列出指定进程所打开的文件
  - -g : 列出GID号进程详情
  - -d<文件号> : 列出占用该文件号的进程
  - +d<目录> : 列出目录下被打开的文件
  - +D<目录> : 递归列出目录下被打开的文件

## ◆ 命令参数

- -n<目录>：列出使用NFS的文件
- -i<条件>：列出符合条件的进程(4、6、协议、:端口、 @ip )
- -p<进程号>：列出指定进程号所打开的文件
- -u：列出UID号进程详情
- -h：显示帮助信息
- -v：显示版本信息。
- -n: 不反向解析网络名字

- ◆ 进程管理
- ◆ 查看由登陆用户启动而非系统启动的进程  
`lsof /dev/pts/1`
- ◆ 指定进程号，可以查看该进程打开的文件  
`lsof -p 9527`
- ◆ 文件管理
- ◆ 查看指定程序打开的文件  
`lsof -c httpd`
- ◆ 查看指定用户打开的文件  
`lsof -u root | more`
- ◆ 查看指定目录下被打开的文件  
`lsof +D /var/log/`  
`lsof +d /var/log/`  
参数+D为递归列出目录下被打开的文件，参数+d为列出目录下被打开的文件



## ◆ 查看所有网络连接

```
lsof -i -n
```

```
lsof -i@127.0.0.1
```

通过参数-i查看网络连接的情况，包括连接的ip、端口等以及一些服务的连接情况，例如：sshd等。也可以通过指定ip查看该ip的网络连接情况

## ◆ 查看端口连接情况

```
lsof -i :80 -n
```

通过参数-i:端口可以查看端口的占用情况，-i参数还有查看协议，ip的连接情况等

## ◆ 查看指定进程打开的网络连接

```
lsof -i -n -a -p 9527
```

参数-i、-a、-p等，-i查看网络连接情况，-a查看存在的进程，-p指定进程

## ◆ 查看指定状态的网络连接

```
lsof -n -P -i TCP -s TCP:ESTABLISHED
```

-n:no host names, -P:no port names,-i TCP指定协议，-s指定协议状态通过多个参数可以清晰的查看网络连接情况、协议连接情况等

## ◆ 恢复删除文件

```
lsof |grep /var/log/messages
```

```
rm -f /var/log/messages
```

```
lsof |grep /var/log/messages
```

```
cat /proc/653/fd/6
```

```
cat /proc/653/fd/6 > /var/log/messages
```

- ◆ kill命令：向进程发送控制信号，以实现对进程管理,每个信号对应一个数字，信号名称以SIG开头（可省略），不区分大小写

显示当前系统可用信号：kill -l 或者 trap -l

常用信号：man 7 signal

- 1) SIGHUP 无须关闭进程而让其重读配置文件
- 2) SIGINT 中止正在运行的进程；相当于Ctrl+c
- 3) SIGQUIT 相当于ctrl+\
- 9) SIGKILL 强制杀死正在运行的进程
- 15) SIGTERM 终止正在运行的进程
- 18) SIGCONT 继续运行
- 19) SIGSTOP 后台休眠

指定信号的方法：(1) 信号的数字标识：1, 2, 9

(2) 信号完整名称：SIGHUP

(3) 信号的简写名称：HUP

- ◆ 按PID : `kill [-SIGNAL] pid ...`
  - `kill -n SIGNAL pid`
  - `kill -s SIGNAL pid`
- ◆ 按名称 : `killall [-SIGNAL] comm...`
- ◆ 按模式 : `pkill [options] pattern`
  - `-SIGNAL`
  - `-u uid: effective user , 生效者`
  - `-U uid: real user , 真正发起运行命令者`
  - `-t terminal: 与指定终端相关的进程`
  - `-l: 显示进程名 ( pgrep可用 )`
  - `-a: 显示完整格式的进程名 ( pgrep可用 )`
  - `-P pid: 显示指定进程的子进程`

## ◆ Linux的作业控制

前台作业：通过终端启动，且启动后一直占据终端

后台作业：可通过终端启动，但启动后即转入后台运行（释放终端）

## ◆ 让作业运行于后台

(1) 运行中的作业： Ctrl+z

(2) 尚未启动的作业： COMMAND &

## ◆ 后台作业虽然被送往后台运行，但其依然与终端相关；退出终端，将关闭后台作业。如果希望送往后台后，剥离与终端的关系

nohup COMMAND &>/dev/null &

screen ; COMMAND

## ◆ 查看当前终端所有作业：jobs

## ◆ 作业控制：

fg [[%]JOB\_NUM]：把指定的后台作业调回前台

bg [[%]JOB\_NUM]：让送往后台的作业在后台继续运行

kill [%JOB\_NUM]：终止指定的作业

# 并行运行



◆ 同时运行多个进程，提高效率

◆ 方法1

```
vi all.sh
```

```
f1.sh&
```

```
f2.sh&
```

```
f3.sh&
```

◆ 方法2

```
(f1.sh&);(f2.sh&);(f3.sh&)
```

◆ 方法3

```
{ f1.sh& f2.sh& f3.sh& }
```

## ◆ Linux任务计划、周期性任务执行

- 未来的某时间点执行一次任务

at                      指定时间点，执行一次性任务

batch                  系统自行选择空闲时间去执行此处指定的任务

- 周期性运行某任务

cron

◆ 包：at

◆ at 命令：at [option] TIME

◆ 常用选项：

-V 显示版本信息

-t time 时间格式 [[CC]YY]MMDDhhmm[.ss]

-l 列出指定队列中等待运行的作业；相当于atq

-d 删除指定的作业；相当于atrm

-c 查看具体作业任务

-f /path/file 指定的文件中读取任务

-m 当任务被完成之后，将给用户发送邮件，即使没有标准输出

◆ 注意：作业执行命令的结果中的标准输出和错误以邮件通知给相关用户



- ◆ TIME : 定义出什么时候进行 at 这项任务的时间
  - HH:MM [YYYY-mm-dd]
  - noon, midnight, teatime ( 4pm )
  - tomorrow
  - now+#{minutes,hours,days, OR weeks}

- ◆ HH:MM 02:00

在今日的 HH:MM 进行，若该时刻已过，则明天此时执行任务

- ◆ HH:MM YYYY-MM-DD 02:00 2016-09-20

规定在某年某月的某一天的特殊时刻进行该项任务

- ◆ HH:MM[am|pm] [Month] [Date]

04pm March 17

17:20 tomorrow

- ◆ HH:MM[am|pm] + number [minutes|hours|days|weeks]

在某个时间点再加几个时间后才进行该项任务

now + 5 min

02pm + 3 days

## ◆ 执行方式：

- 1 ) 交互式
- 2 ) 输入重定向
- 3 ) at -f 文件

## ◆ 依赖与atd服务,需要启动才能实现at任务

## ◆ at队列存放在/var/spool/at目录中

## ◆ /etc/at.{allow,deny}控制用户是否能执行at任务

白名单：/etc/at.allow 默认不存在，只有该文件中的用户才能执行at命令

黑名单：/etc/at.deny 默认存在，拒绝该文件中用户执行at命令，而没有在at.deny 文件中的使用者则可执行

如果两个文件都不存在，只有 root 可以执行 at 命令

# 周期性任务计划cron

- ◆ 周期性任务计划：cron

- ◆ 相关的程序包：

  - cronie：主程序包，提供crond守护进程及相关辅助工具

  - crontabs：包含CentOS提供系统维护任务

  - cronie-anacron：cronie的补充程序，用于监控cronie任务执行状况，如cronie中的任务在过去该运行的时间点未能正常运行，则anacron会随后启动一次此任务

- ◆ 确保crond守护处于运行状态：

CentOS 7:

`systemctl status crond`

CentOS 6:

`service crond status`

- ◆ 计划周期性执行的任务提交给crond，到指定时间会自动运行

系统cron任务：系统维护作业

`/etc/crontab`

用户cron任务：

`crontab`命令

- ◆ 日志：`/var/log/cron`

◆ 系统cron任务:/etc/crontab

◆ 注释行以 # 开头

◆ 详情参见 man 5 crontab

```
# Example of job definition:
```

```
# .----- minute (0 - 59)
```

```
# | .----- hour (0 - 23)
```

```
# | | .----- day of month (1 - 31)
```

```
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
```

```
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
```

```
# | | | | |
```

```
# * * * * * user-name command to be executed
```

◆ 例如：晚上9点10分运行echo命令

```
10 21 * * * wang /bin/echo "Howdy!"
```

## ◆ 时间表示法：

- (1) 特定值  
给定时间点有效取值范围内的值
- (2) \*  
给定时间点上有效取值范围内的所有值  
表示 “每...”
- (3) 离散取值  
#, #, #
- (4) 连续取值  
#-#
- (5) 在指定时间范围上，定义步长  
/#: #即为步长

# 时间格式

- ◆ @yearly            0 0 1 1 \*
- ◆ @annually        0 0 1 1 \*
- ◆ @monthly        0 0 1 \* \*
- ◆ @weekly        0 0 \* \* 0
- ◆ @daily        0 0 \* \* \*
- ◆ @hourly        0 \* \* \* \*
- ◆ @reboot        Run once after reboot

- ◆ 示例：每3小时echo和wall命令

0 \*/3 \* \* \* wang /bin/echo "howdy" ; wall "welcome to Magedu!"



# 计划任务



- 系统的计划任务:

/etc/crontab

配置文件

/etc/cron.d/

配置文件

/etc/cron.hourly/

脚本

/etc/cron.daily/

脚本

/etc/cron.weekly/

脚本

/etc/cron.monthly/

脚本

- ◆ 运行计算机关机时cron不运行的任务，CentOS6以后版本取消anacron服务，由crond服务管理
- ◆ 对笔记本电脑、台式机、工作站、偶尔要关机的服务器及其它不一直开机的系统很重要对很有用
- ◆ 配置文件：/etc/anacrontab，负责执行/etc/cron.daily /etc/cron.weekly /etc/cron.monthly中系统任务
  - 字段1：如果在这些日子里没有运行这些任务.....
  - 字段2：在重新引导后等待这么多分钟后运行它
  - 字段3：任务识别器，在日志文件中标识
  - 字段4：要执行的任务
- ◆ 由/etc/cron.hourly/0anacron执行
- ◆ 当执行任务时，更新/var/spool/anacron/cron.daily 文件的时间戳

- ◆ CentOS7使用systemd-tmpfiles-setup服务实现
- ◆ CentOS6使用/etc/cron.daily/tmpwatch定时清除临时文件
- ◆ 配置文件：
  - /etc/tmpfiles.d/\*.conf
  - /run/tmpfiles.d/\*.conf
  - /usr/lib/tmpfiles.d/\*.conf
- ◆ /usr/lib/tmpfiles.d/tmp.conf
  - d /tmp 1777 root root 10d
  - d /var/tmp 1777 root root 30d
- ◆ 命令：
  - systemd-tmpfiles --clean|remove|create configfile

## ◆ crontab命令定义

每个用户都有专用的cron任务文件：`/var/spool/cron/USERNAME`

## ◆ crontab命令：

`crontab [-u user] [-l | -r | -e] [-i]`

`-l`                列出所有任务

`-e`                编辑任务

`-r`                移除所有任务

`-i`                同`-r`一同使用，以交互式模式移除指定任务

`-u user`          仅root可运行，指定用户管理cron任务

## ◆ 控制用户执行计划任务：

`/etc/cron.{allow,deny}`

# at和crontab



- ◆ 一次性作业使用 at

- ◆ 重复性作业使用 crontab

Create	at <i>time</i>	crontab -e
List	at -l	crontab -l
Details	at -c <i>jobnum</i>	crontab -l
Remove	at -d <i>jobnum</i>	crontab -r
Edit	N/A	crontab -e

- ◆ 没有被重定向的输出会被邮寄给用户

- ◆ root 能够修改其它用户的作业

- ◆ 注意：运行结果的标准输出和错误以邮件通知给相关用户
  - (1) `COMMAND > /dev/null`
  - (2) `COMMAND &> /dev/null`
  
- ◆ 对于cron任务来讲，%有特殊用途；如果在命令中要使用%，则需要转义，将%放置于单引号中，则可不用转义

## ◆ 思考：

### ◆ (1) 如何在秒级别运行任务？

```
* * * * * for min in 0 1 2; do echo "hi"; sleep 20; done
```

### ◆ (2) 如何实现每7分钟运行一次任务？

## ◆ sleep命令：

```
sleep NUMBER[SUFFIX]...
```

SUFFIX:

s: 秒, 默认

m: 分

h: 小时

d: 天

- ◆ 1、每周的工作日1：30，将/etc备份至/backup目录中，保存的文件名称格式为“etcbak-yyyy-mm-dd-HH.tar.xz”，其中日期是前一天的时间
- ◆ 2、每两小时取出当前系统/proc/meminfo文件中以S或M开头的信息追加至/tmp/meminfo.txt文件中
- ◆ 3、工作日时间，每10分钟执行一次磁盘空间检查，一旦发现任何分区利用率高于80%，就执行wall警报



- ◆ 博客 : <http://mageedu.blog.51cto.com>
- ◆ 主页 : <http://www.magedu.com>
- ◆ QQ : 1661815153, 113228115
- ◆ QQ群 : 203585050, 279599283

# 祝大家学业有成

# 谢 谢

咨询热线 400-080-6560