

系统&服务管理进阶

NSD SERVICES

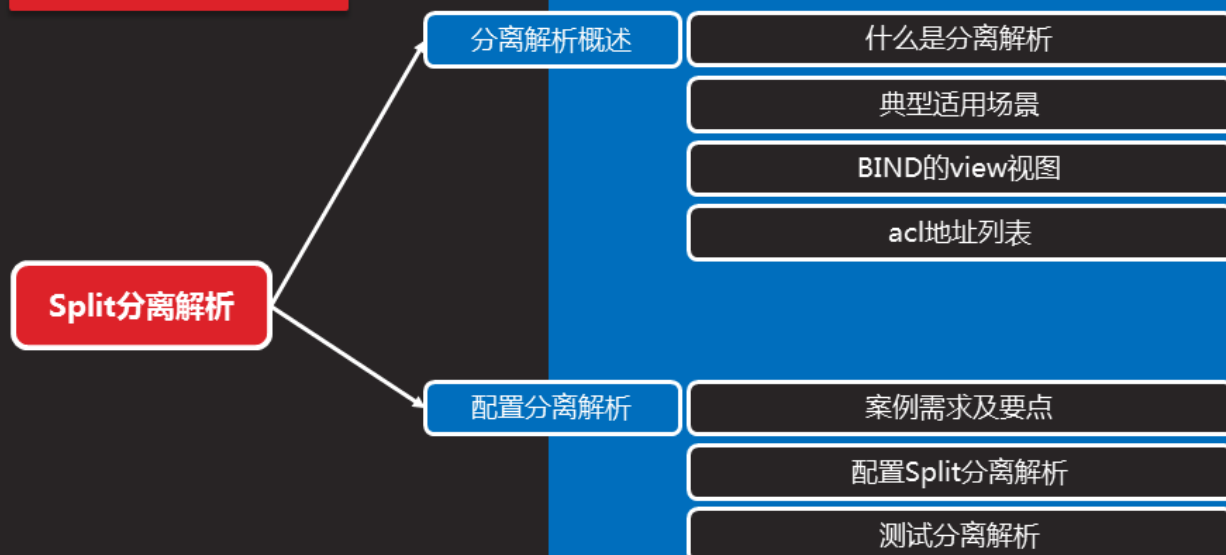
DAY03

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	Split分离解析
	10:30 ~ 11:20	
	11:30 ~ 12:00	RAID磁盘阵列
下午	14:00 ~ 14:50	进程管理
	15:00 ~ 15:50	
	16:10 ~ 17:00	日志管理
	17:10 ~ 18:00	总结和答疑



Split分离解析



分离解析概述

什么是分离解析

- 当收到客户机的DNS查询请求的时候
 - 能够区分客户机的来源地址
 - 为不同类别的客户机提供不同的解析结果（IP地址）

知识讲解

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\TsengYia>nslookup www.12306.cn
服务器:  gjjline.bta.net.cn
Address:  202.106.0.20

非权威应答:
名称:    12306.xdwscache.ourglb0.com
Addresses: 60.207.246.98
          43.255.177.55
Aliases:  www.12306.cn
          www.12306.cn.lxdns.com
```

从联通的客户机查询

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.10586]
(c) 2015 Microsoft Corporation. 保留所有权利。

C:\Users\TsengYia> nslookup www.12306.cn
服务器:  cache3-bj
Address:  211.98.2.4

非权威应答:
名称:    12306.xdwscache.ourglb0.com
Address:  122.70.142.160
Aliases:  www.12306.cn
          www.12306.cn.lxdns.com
```

从铁通的客户机查询



典型适用场景

- 访问压力大的网站，购买CDN提供的内容分发服务
 - 在全国各地/不同网络内部署大量镜像服务节点
 - 针对不同的客户机就近提供服务器



知识讲解



BIND的view视图

- 根据源地址集合将客户机分类
 - 不同客户机获得不同结果（待遇有差别）

```
view "联通" {
    match-clients { 来源地址1; ...; }
    zone "12306.cn" IN {
        ..... 地址库1;
    };
};
view "铁通" {
    match-clients { 来源地址2; ...; }
    zone "12306.cn" IN {
        ..... 地址库2;
    };
};
```

1. 同一个区域（12306.cn）在多个视图内分别定义，其地址库文件相互独立，从而实现解析结果的分离
2. 定义view视图后，不允许在view以外出现zone配置

知识讲解



acl地址列表

知识讲解

- 为大批量的客户机地址建立列表
 - 调用时指定列表名即可，列表名 any 可匹配任意地址
 - 根据view调用的顺序，“匹配即停止”

```
acl "liantong" {                                //联通网络
    IP地址1; IP地址2; ...
    网段1; 网段2; ...
    ...
};
acl "tietong" {                                //铁通网络
    IP地址3; IP地址4; ...
    网段3; 网段4; ...
    ...
};
```



配置分离解析

案例需求及要点

知识讲解

- 环境及需求
 - 权威DNS : `svr7.tedu.cn` 192.168.4.7
 - 负责区域 : `tedu.cn`
 - A记录分离解析 —— 以 `www.tedu.cn` 为例

客户机来自	解析结果
192.168.4.207、192.168.7.0/24	192.168.4.100
其他地址	1.2.3.4



案例需求及要点（续1）

知识讲解

- 基本配置步骤
 1. 建立2份地址库文件
 2. 针对来源地址定义acl列表
 3. 配置2个view，调用不同的地址库
 4. 重启named服务
 5. 测试分离解析结果



配置Split分离解析

- 1. 建立2份地址库文件
 - www的A记录指向不同的IP地址

```
[root@svr7 ~]# vim /var/named/tedu.cn.zone.lan
```

```
.. ..
```

```
www    IN    A      192.168.4.100      //对应解析结果1
```

```
[root@svr7 ~]# vim /var/named/tedu.cn.zone.other
```

```
.. ..
```

```
www    IN    A      1.2.3.4        //对应解析结果2
```

知识讲解



配置Split分离解析（续1）

- 2. 针对来源地址定义acl列表
 - 若地址比较少，也可以不建立列表

```
[root@svr7 ~]# vim /etc/named.conf
```

```
options {
```

```
    directory "/var/named";
```

```
};
```

```
acl "mylan" {
```

```
    192.168.4.207;
```

```
    192.168.7.0/24;
```

```
};
```

```
.. ..
```

```
//名为mylan的列表
```

知识讲解



配置Split分离解析（续2）

- 3. 配置2个view，调用不同的地址库

- 确认后重启 named 服务

```
view "mylan" {
    match-clients { mylan; };           //匹配列表 mylan
    zone "tedu.cn" IN {
        type master;
        file "tedu.cn.zone.lan"; }; };
view "other" {
    match-clients { any; };             //匹配任意地址
    zone "tedu.cn" IN {
        type master;
        file "tedu.cn.zone.other"; };};
```

```
[root@svr7 ~]# systemctl restart named
```

知识讲解



测试分离解析

- 分别从不同视图中的客户机测试
 - 从192.168.4.207查询，解析结果：192.168.4.100
 - 从其他主机查询，解析结果：1.2.3.4

```
[root@pc207 ~]# host www.tedu.cn 192.168.4.7
Using domain server:
Name: 192.168.4.7
```

```
.. ..
www.tedu.cn has address 192.168.4.110    //从客户机1查询
```

```
[root@svr7 ~]# host www.tedu.cn 192.168.4.7
Using domain server:
Name: 192.168.4.7
```

```
.. ..
www.tedu.cn has address 1.2.3.4          //从其他客户机查询
```

知识讲解



案例1：配置并验证Split分离解析

配置DNS服务，实现以下目标

- 1) 从主机192.168.4.207查询时，
www.tedu.cn ==> 192.168.4.100
- 2) 从其他客户端查询时，
www.tedu.cn ==> 1.2.3.4

课堂练习



RAID磁盘阵列

RAID磁盘阵列

RAID磁盘阵列

RAID阵列概述

RAID0/1/10

RAID5/6

RAID各级别特点对比

RAID阵列实现方式

RAID磁盘阵列

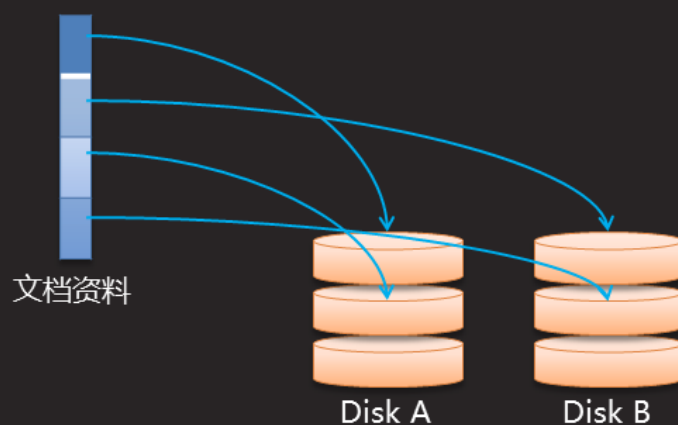
RAID阵列概述

- 廉价冗余磁盘阵列
 - Redundant Arrays of Inexpensive Disks
 - 通过硬件/软件技术，将多个较小/低速的磁盘整合成一个大磁盘
 - 阵列的价值：提升I/O效率、硬件级别的数据冗余
 - 不同RAID级别的功能、特性各不相同

RAID0/1/10

知识讲解

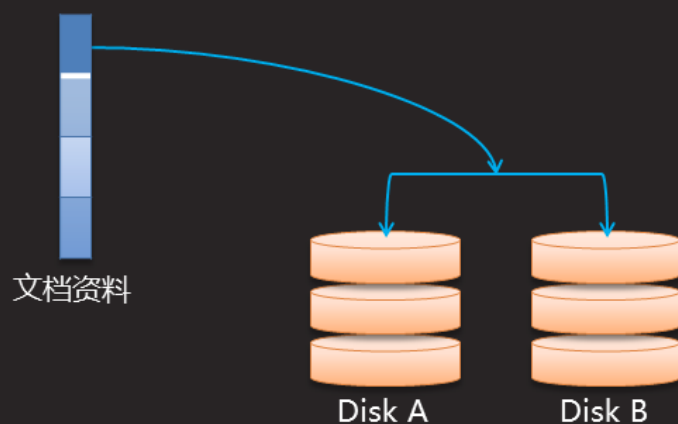
- RAID 0，条带模式
 - 同一个文档分散存放在不同磁盘
 - 并行写入以提高效率



RAID0/1/10 (续1)

知识讲解

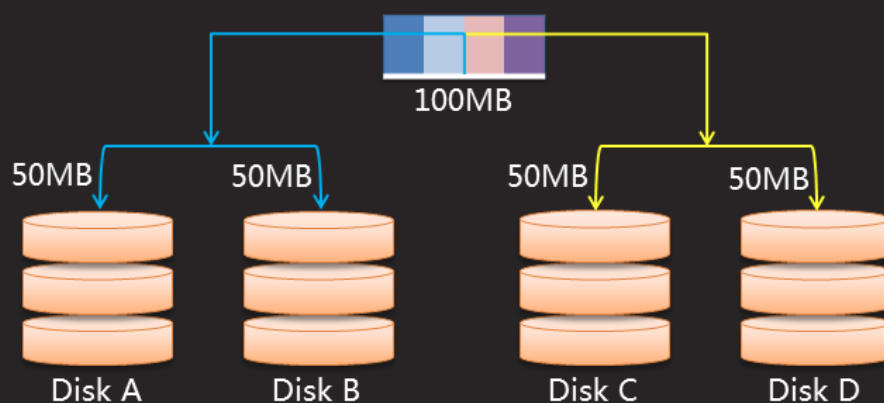
- RAID 1，镜像模式
 - 一个文档复制成多份，分别写入不同磁盘
 - 多份拷贝提高可靠性，效率无提升



RAID0/1/10 (续2)

知识讲解

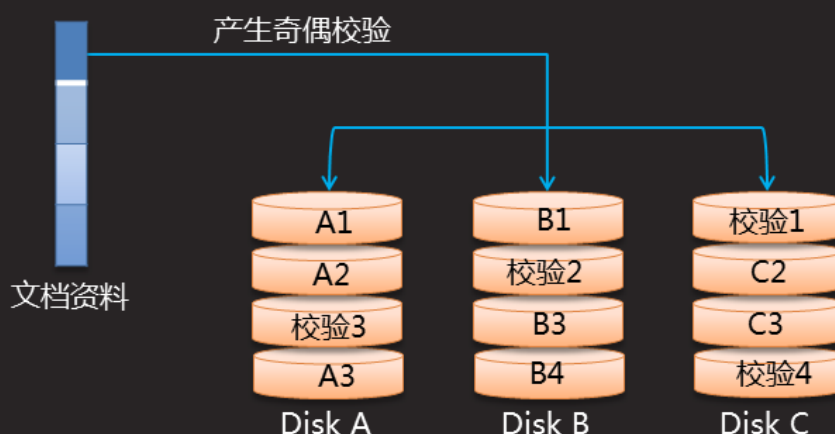
- RAID 0+1/RAID 1+0
 - 整合RAID 0、RAID 1的优势
 - 并行存取提高效率、镜像写入提高可靠性



RAID5/6

知识讲解

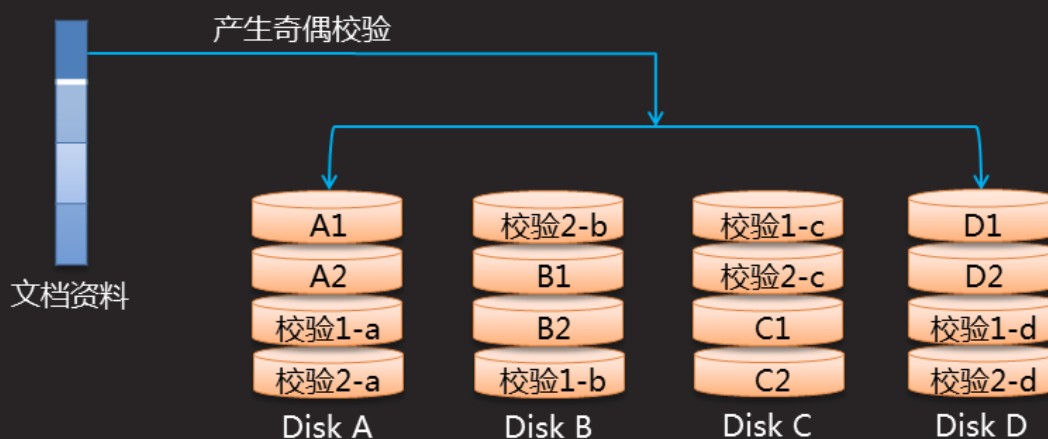
- RAID5, 高性价比模式
 - 相当于RAID0和RAID1的折中方案
 - 需要至少一块磁盘的容量来存放校验数据



RAID5/6 (续1)

- RAID6，高性价比/可靠模式
 - 相当于扩展的RAID5阵列，提供2份独立校验方案
 - 需要至少两块磁盘的容量来存放校验数据

知识讲解



RAID各级别特点对比

知识讲解

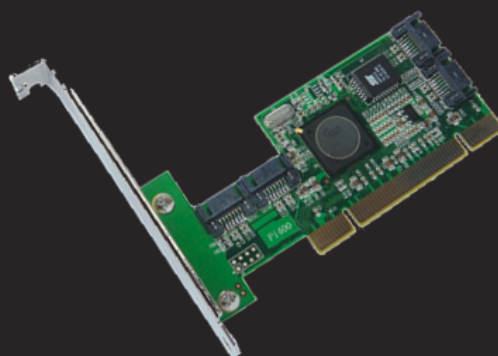
对比项	RAID 0	RAID 1	RAID 10	RAID 5	RAID 6
磁盘数	≥ 2	≥ 2	≥ 4	≥ 3	≥ 4
存储利用率	100%	$\leq 50\%$	$\leq 50\%$	$n-1/n$	$n-2/n$
校验盘	无	无	无	1	2
容错性	无	有	有	有	有
IO性能	高	低	中	较高	较高



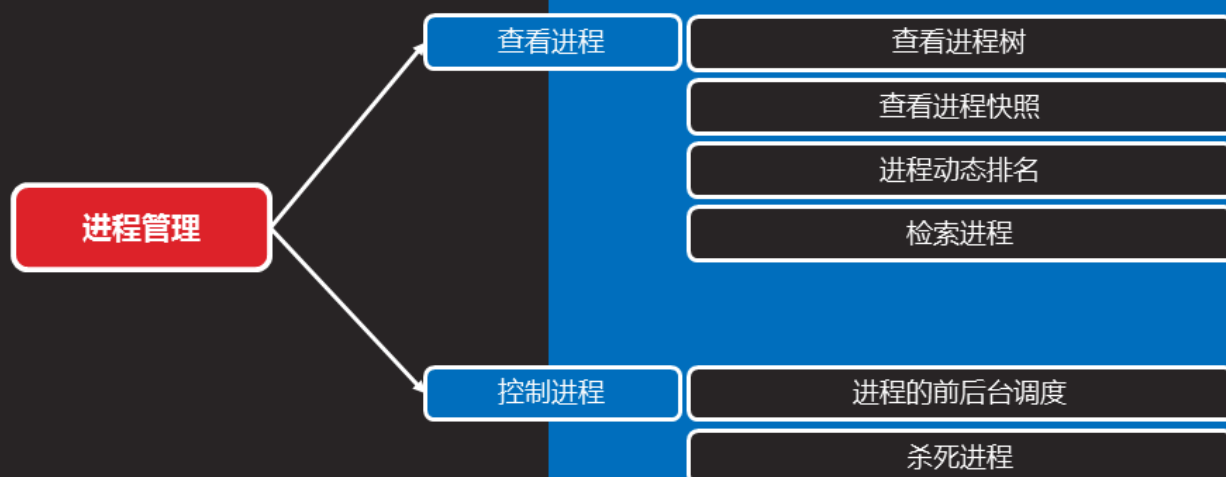
RAID阵列实现方式

知识讲解

- 硬RAID：由RAID控制卡管理阵列
 - 主板 → 阵列卡 → 磁盘 → 操作系统 → 数据
- 软RAID：由操作系统来管理阵列
 - 主板 → 磁盘 → 操作系统 → RAID软件 → 数据



进程管理



查看进程

查看进程树

知识讲解

- pstree — Processes Tree
 - 格式：`pstree [选项] [PID或用户名]`
- 常用命令选项
 - `-a`：显示完整的命令行
 - `-p`：列出对应PID编号

```
[root@svr7 ~]# pstree -p
systemd(1)─ModemManager(484)─{ModemManager}(504)
               │               └─{ModemManager}(513)
               └─NetworkManager(628)─{NetworkManager}(827)
                                   └─{NetworkManager}(830)
               ─abrt-watch-log(519)
               ─abrt-watch-log(520)
               ─abrt-d(518)
```



查看进程快照

- ps — Processes Snapshot
 - 格式：ps [选项]...
- 常用命令选项
 - aux：显示当前终端所有进程（a）、当前用户在所有终端下的进程（x），以用户格式输出（u）
 - -elf：显示系统内所有进程（-e）、以长格式输出（-l）信息、包括最完整的进程信息（-f）

知识讲解



查看进程快照（续1）

- ps aux 操作
 - 列出正在运行的所有进程

知识讲解

```
[root@svr7 ~]# ps aux
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.3	126904	7128	?	Ss	12月07	0:14	/usr/lib/syst
root	2	0.0	0.0	0	0	?	S	12月07	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S	12月07	0:00	[ksoftirqd/0]
root	7	0.0	0.0	0	0	?	S	12月07	0:00	[migration/0]
root	8	0.0	0.0	0	0	?	S	12月07	0:00	[rcu_bh]
root	9	0.0	0.0	0	0	?	S	12月07	0:00	[rcuob/0]
root	10	0.0	0.0	0	0	?	R	12月07	0:09	[rcu sched]

用户 进程ID %CPU %内存 虚拟内存 固定内存 终端 状态 起始时间 CPU时间 程序指令



查看进程快照 (续2)

- ps -elf 操作
 - 列出正在运行的所有进程

知识讲解

```
[root@svr7 ~]# ps -elf
F S UID          PID     PPID    C  PRI   NI     ADDR  SZ  WCHAN    STIME TTY          TIME CMD
4 S root           1         0    0   80    0     - 31726  ep_pol 12月07 ?        00:00:14 /usr/
1 S root           2         0    0   80    0     -      0 kthrea 12月07 ?        00:00:00 [kthr
1 S root           3         2    0   80    0     -      0 smpboo 12月07 ?        00:00:00 [ksof
1 S root           7         2    0  -40   - -      0 smpboo 12月07 ?        00:00:00 [migr
1 S root           8         2    0   80    0     -      0 rcu_gp 12月07 ?        00:00:00 [rcu_
1 S root           9         2    0   80    0     -      0 rcu_no 12月07 ?        00:00:00 [rcuo
1 R root          10         2    0   80    0     -      0 -      12月07 ?        00:00:09 [rcu
```

PPID : 父进程的PID号

PRI/NI : 进程优先级, 数值越小优先级越高



进程动态排名

- top 交互式工具
 - 格式: `top [-d 刷新秒数] [-U 用户名]`

知识讲解

```
[root@svr7 ~]# top -d5
top - 15:26:35 up 7 days, 4:13, 3 users, load average: 0.01, 0.02, 0.05
Tasks: 188 total, 2 running, 186 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni, 100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 1884232 total, 92336 free, 679404 used, 1112492 buff/cache
KiB Swap: 4194300 total, 4191084 free, 3216 used. 977560 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	126904	7128	2124	S	0.0	0.4	0:14.18	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.12	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.06	ksoftirqd/0
7	root	rt	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcu_bh
9	root	20	0	0	0	0	S	0.0	0.0	0:00.00	rcuob/0
10	root	20	0	0	0	0	S	0.0	0.0	0:09.53	rcu_sched
11	root	20	0	0	0	0	R	0.0	0.0	0:27.16	rcuos/0
12	root	rt	0	0	0	0	S	0.0	0.0	0:06.37	watchdog/0



进程动态排名（续1）

知识讲解

- top 交互操作指令
 - ? : 查看帮助（列出可用的按键指令）
 - P、M : 根据 %CPU、%MEM 降序排列
 - T : 根据进程消耗的 TIME 降序排列
 - k : 杀死指定的进程
 - q : 退出 top 程序



检索进程

知识讲解

- pgrep — Process Grep
 - 用途：pgrep [选项]... 查询条件
- 常用命令选项
 - -l : 输出进程名，而不仅仅是 PID
 - -U : 检索指定用户的进程
 - -t : 检索指定终端的进程
 - -x : 精确匹配完整的进程名



检索进程（续1）

知识讲解

- 列出名称包含|为 **gdm** 的进程信息

```
[root@svr7 ~]# pgrep -l 'gdm'
```

//包含关键词

```
1159 gdm
```

```
2777 gdm-session-wor
```

```
[root@svr7 ~]# pgrep -lx 'gdm'
```

//精确匹配

```
1159 gdm
```

- 列出属于用户 **zhsan** 的所有进程

```
[root@svr7 ~]# pgrep -l -U zhsan
```

```
2958 bash
```

```
2981 vim
```



案例2：查看进程信息

使用进程工具完成下列任务

课堂练习

- 1) 找出进程 **gdm** 的 PID 编号值
- 2) 列出由进程 **gdm** 开始的子进程树结构信息
- 3) 找出进程 **sshd** 的父进程的 PID 编号/进程名称
- 4) 查看当前系统的CPU负载/进程总量信息



控制进程



进程的前后台调度

- 前台启动
 - 输入正常命令行，运行期间占用当前终端
- 后台启动
 - 在命令行末尾添加 “&” 符号，不占用当前终端

```
[root@svr7 ~]# cp /dev/cdrom mycd.iso &  
[2] 22378           //后台制作ISO镜像文件
```



进程的前后台调度（续1）

知识讲解

- Ctrl + z 组合键
 - 挂起当前进程（暂停并转入后台）
- jobs 命令
 - 查看后台任务列表
- fg 命令
 - 将后台任务恢复到前台运行
- bg 命令
 - 激活后台被挂起的任务

缺省序号则为最近1个任务



进程的前后台调度（续2）

知识讲解

```
[root@svr7 ~]# jobs -l //查看后台任务列表
[1]+ 19078 停止          vim
[2]- 22756 Running      cp -i /dev/cdrom mycd.iso &
```

```
[root@svr7 ~]# fg
... //恢复已挂起的vim程序
```

```
[root@svr7 ~]# cp /dev/cdrom mycd2.iso
[3]+  Stopped          cp -i /dev/cdrom mycd2.iso
//按Ctrl+z键挂起任务
```

```
[root@svr7 ~]# bg 3 //后台运行第3个任务
[3]+ cp -i /dev/cdrom mycd2.iso &
```



杀死进程

知识讲解

- 干掉进程的不同方法

- Ctrl+c 组合键，中断当前命令程序
- kill [-9] PID...、kill [-9] %后台任务编号
- killall [-9] 进程名...
- pkill 查找条件

```
[root@svr7 ~]# killall -9 vim           //杀死同名的多个进程
[1]- 已杀死          vim file1.txt
[2]+ 已杀死          vim file2.txt
```

```
[root@svr5 ~]# pkill -9 -U hackli      //强制踢出用户
```



案例3：进程调度及终止

课堂练习

1. 运行“sleep 600”命令

- 另开一个终端，查出sleep程序的PID并杀死

2. 运行多个vim程序并都放入后台

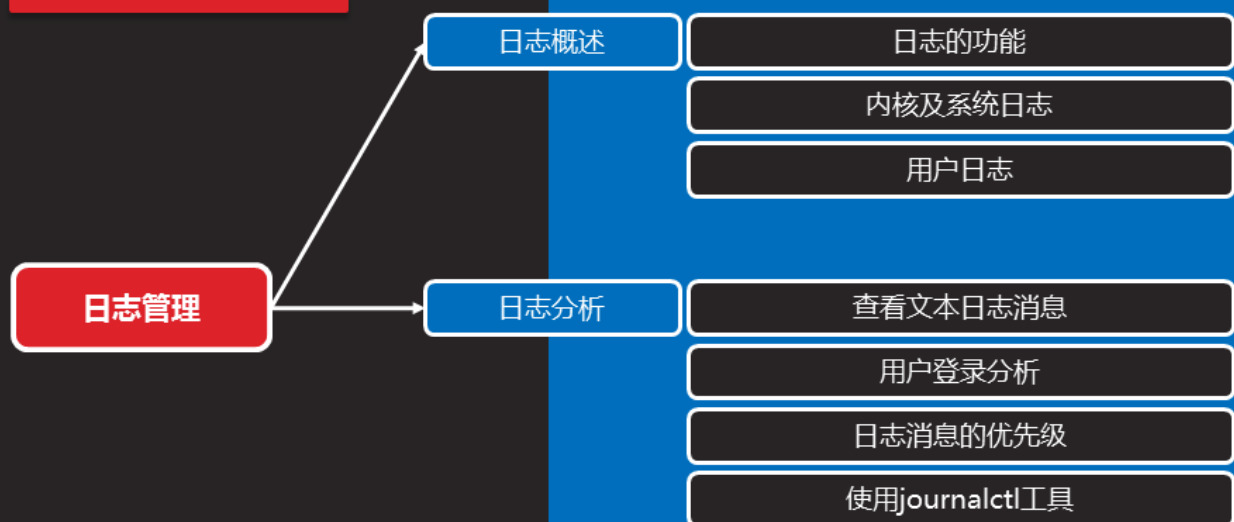
- 杀死所有vim进程

3. su切换为zhsan用户

- 另开一个终端，强制踢出zhsan用户



日志管理



日志概述

日志的功能

知识讲解

- 系统和程序的“日记本”
 - 记录系统、程序运行中发生的各种事件
 - 通过查看日志，了解及排除故障
 - 信息安全控制的“依据”



内核及系统日志

知识讲解

- 由系统服务rsyslog统一记录/管理
 - 日志消息采用文本格式
 - 主要记录事件发生的时间、主机、进程、内容

```
[root@svr7 ~]# tail /var/log/messages
```

```
.. ..
```

Aug 29 13:19:48	svr7	dhclient:	DHCPACK from 192.168.8.....
Aug 29 13:19:48	svr7	dhclient:	bound to 192.168.8.128 – re.....

时间、地点、人物，发生何事



内核及系统日志（续1）

- 常见的日志文件

知识讲解

日志文件	主要用途
<code>/var/log/messages</code>	记录内核消息、各种服务的公共消息
<code>/var/log/dmesg</code>	记录系统启动过程的各种消息
<code>/var/log/cron</code>	记录与cron计划任务相关的消息
<code>/var/log/maillog</code>	记录邮件收发相关的消息
<code>/var/log/secure</code>	记录与访问限制相关的安全消息



用户日志

- 由登录程序负责记录/管理
 - 日志消息采用二进制格式
 - 记录登录用户的时间、来源、执行的命令等信息

知识讲解

日志文件	主要用途
<code>/var/log/lastlog</code>	记录最近的用户登录事件
<code>/var/log/wtmp</code>	记录成功的用户登录/注销事件
<code>/var/log/btmp</code>	记录失败的用户登录事件
<code>/var/run/utmp</code>	记录当前登录的每个用户的相关信息



日志分析

查看文本日志消息

- 通用分析工具
 - tail、**tailf**、**less**、grep等文本浏览/检索命令
 - awk、sed等格式化过滤工具
- 专用分析工具
 - Webmin系统管理套件
 - Webalizer、AWStats等日志统计套件



用户登录分析

知识讲解

- users、who、w 命令
 - 查看已登录的用户信息，详细度不同
- last、lastb 命令
 - 查看最近登录成功/失败的用户信息

```
[root@svr7 ~]# last -2           //最近两条登入记录
root    pts/1    192.168.8.1  Thu Aug 29 10:56  still logged in
root    tty1     Thu Aug 29 10:56  still logged in
```

```
[root@svr7 ~]# lastb -2         //最近两条登录失败事件
root    ssh:notty 192.168.8.1  Thu Aug 29 10:56 - 10:56 (00:00)
zengye  tty1     Wed Aug 28 12:02 - 12:02
(00:00)
```



日志消息的优先级

知识讲解

- Linux内核定义的事件紧急程度
 - 分为 0~7 共8种优先级别
 - 其数值越小，表示对应事件越紧急/重要

```
[root@svr7 ~]# man 2 syslog
```

```
...
```

```
defined in <linux/kernel.h> as follows:
```

```
#define KERN_EMERG    "<0>" /* system is unusable */
#define KERN_ALERT    "<1>" /* action must be taken immediately */
#define KERN_CRIT     "<2>" /* critical conditions */
#define KERN_ERR       "<3>" /* error conditions */
#define KERN_WARNING   "<4>" /* warning conditions */
#define KERN_NOTICE    "<5>" /* normal but significant condition */
#define KERN_INFO      "<6>" /* informational */
#define KERN_DEBUG     "<7>" /* debug-level messages */
```



使用journalctl工具

知识讲解

- 提取由 systemd-journal 服务搜集的日志
 - 主要包括内核/系统日志、服务日志
- 常见用法
 - journalctl | grep 关键词
 - journalctl -u 服务名 [-p 优先级]
 - journalctl -n 消息条数
 - journalctl --since="yyyy-mm-dd HH:MM:SS" --until="yyyy-mm-dd HH:MM:SS"



案例4：系统日志分析

完成以下日志分析操作

课堂练习

- 1) 列出所有包含关键词 8909 的系统日志消息
- 2) 查看启动时识别的鼠标设备信息
- 3) 列出最近2条成功/不成功的用户登录消息
- 4) 列出最近10条重要程度在 ERR 及以上的日志消息
- 5) 列出所有与服务httpd相关的消息
- 6) 列出前4个小时内新记录的日志



总结和答疑



进程管理

问题现象

知识讲解

- 杀死指定的进程时失败
 - 问题1：执行 `kill %1` 杀后台进程提示：无此任务
 - 问题2：执行 `killall vim` 杀不死 vim 进程

```
[root@svr7 ~]# kill -9 %1
-bash: kill: %1: 无此任务
```

```
[root@svr7 ~]# killall vim
[root@svr7 ~]# jobs -l
[1]+  4668 停止 (tty 输出)    vim a.txt
```



故障分析及排除

知识讲解

- 原因分析
 - 报错1：%1 指当前用户的第1个后台进程（jobs -l），不是系统的后台进程
 - 报错2：个别交互进程会不能正常杀死
- 解决办法
 - 报错1：杀进程时指定正确的jobs后台编号
 - 报错2：发送 -9 信号强制杀死

```
[root@svr7 ~]# killall -9 vim
[1]+  已杀死          vim a.txt
```



