linux 常用命令--------雪松整理

Q 群: 198173206　欢迎 linux 系统运维朋友加入！
博客:　http://hi.baidu.com/quanzhou722/blog

错误在所难免，还望指正！

==========
1 文件管理2 软件管理3 系统管理
4 服务管理5 网络管理6 磁盘管理
7 用户管理8 脚本相关9 服务配置
==========

---
1 文件管理

创建空白文件
touch

不提示删除非空目录
rm -rf 目录名
(-r:递归删除 -f 强制)

##########

恢复 rm 删除的文件(ext3)

查看磁盘分区格式
df -T

卸载挂载
umount /data/

ext3grep /dev/sdb1 --ls --inode 2
记录信息继续查找目录下文件 inode 信息
ext3grep /dev/sdb1 --ls --inode 131081　#　此处是 inode
记录下 inode 信息开始恢复目录
ext3grep /dev/sdb1 --restore-inode 49153

##########

windows 文本转 linux 文本
dos2unix

linux 文本转 windows 文本
unix2dos

转换编码

iconv -f gbk -t utf8 原.txt > 新.txt

查看 md5 值
md5sum

硬链接
ln

符号连接
ln -s

查看上下翻页且显示行号
cat ? | nl |less
q 退出

结束状态
ctrl+z

查看文件开头内容
head

查看文件结尾内容
tail -f   #   监视日志文件

检查文件类型
file

重命名
rename
rename source_pattern target_pattern source_files

更改默认权限
umask

按数字排序
sort -n

按数字倒叙
sort -nr

过滤重复行
sort -u

删除重复的行
uniq

重复的行出现次数
uniq -c

只显示不重复行
uniq -u

将两个文件合并用 tab 键分隔开
paste a b

将两个文件合并指定'+'符号隔开
paste -d'+' a b

将多行数据合并到一行用 tab 键隔开
paste -s a

设置隐藏属性
chattr [+-=] [ASacdistu] 文件或目录

向下分面器
more

搜索
locate 字符串

查看行数
wc -l

实时某个目录下查看最新改动过的文件
watch -d -n 1 'df; ls -FlAt /path'

快速备份一个文件
cp filename{,.bak}

##################################

配置编辑器
gconf-editor

vi 配置文件
vi /etc/vimrc

编辑文件
vim

打开行号
:set nu

取消行号
:set nonu

跳转到 200
200G

取消高亮
:nohl

设置自动缩进
:set autoindent

查看文本格式
set ff

改为 unix 格式
set binary

向前翻页
ctrl+ U

向后翻页
ctrl+ D

全部替换
g/字符 1/s//字符 2/g
%s/字符 1/字符 2/g
#　普通用户打开文档不能保存
#　tee 是一个把 stdin 保存到文件的小工具,而%,是 vim 当中一个只读寄存器的名字,总保存着当前编辑文件的文件路径
:w !sudo tee %

##################################

按文件名查找
find /etc -name http

查找某一类型文件
find . -type f

按照文件权限查找
find / -perm

按照文件属主查找
find / -user

按照文件所属的组来查找文件
find / -group

文件使用时间在 N 天以内
find / -atime -n

文件使用时间在 N 天以前
find / -atime +n

文件内容改变时间在 N 天以内
find / -mtime -n

文件内容改变时间在 N 天以前
find / -mtime +n

文件状态改变时间在 N 天前
find / -ctime +n

文件状态改变时间在 N 天内
find / -ctime -n

linux 文件无创建时间
Access 使用时间
Modify 内容修改时间
Change 状态改变时间(权限、属主)

查找文件长度大于 1M 字节的文件
find / -size +1000000c -print

按名字查找文件传递给-exec 后命令
find /etc -name "passwd*" -exec grep "xuesong" {} \;

查找文件名，不取路径
find . -name 't*' -exec basename {} \;

###################################

==========

---
2 软件管理

解包 tar
tar xvf 1.tar -C 目录

打包 tar
tar -cvf 1.tar *

查看 tar
tar tvf 1.tar

给 tar 追加文件
tar -rvf 1.tar 文件名

解包 gz
tar zxvpf gz.tar.gz

打包 gz
tar zcvpf gz.tar.gz

查看 gz
tar ztvpf gz.tar.gz

解压 bzip2
bzip2   -dv 1.tar.bz2

解压 gzip 到 tar
gzip -dv 1.tar.gz

压缩 tar 到 gz
gzip -v 1.tar

bzip2 压缩
bzip2 -v 1.tar

查看 bzip2
bzcat

rpm 安装
rpm -ivh

卸载
rpm -e lynx

强制卸载
rpm -e lynx --nodeps

测试
rpm --test lynx

查看所有安装的 rpm 包
rpm -qa

查找包是否安装
rpm -qa | grep http

解压 zip
unzip zip.zip

压缩 zip
zip zip.zip *

rar3.6 下载地址
http://www.rarsoft.com/rar/rarlinux-3.6.0.tar.gz

压缩文件为 rar 包
rar a rar.rar *.jpg

解压 rar 包
unrar x rar.rar

7z 压缩
7z a 7z.7z *

7z 解压
7z e 7z.7z

查看字符行
zgrep 字符 1.gz

打包/home, /etc ， 但排除 /home/dmtsai
tar --exclude /home/dmtsai -zcvf myfile.tar.gz /home/* /etc

在 /home 当中，比 2005/06/01 新的文件才备份
tar -N "2005/06/01" -zcvf home.tar.gz /home

下载
wgetrc

文本上网
lynx

制作镜像
cp -v /dev/dvd /software/rhel4.6.iso9660

清除编译结果
make clean

源码安装
./configure --help |less
./configure --prefix=/usr/local/
make
make install

perl 程序编译
perl Makefile.PL
make
make test
make install

perl 程序编译
python file.py

编译 c 程序
gcc -g hello.c -o hello

==========

---
3 系统管理

终端锁屏 Ctrl+S

解锁屏 Ctrl+Q

PS1 环境变量控制提示显示
PS1='[\u@ \H \w \A \@#]\$'

开机启动模式
vi /etc/inittab

查找命令的目录
whereis

查看当前要执行的命令所在的路径
which

让部命令后台运行
命令 &

将前台放入后台(暂停)
ctrl+z

查看后台运行程序
jobs

启动后台暂停进程
bg 2

调回后台进程
fg 2

后台运行不受 shell 退出影响
nohup cmd &

清空整个屏幕
clear

重新初始化屏幕
reset

查询用过的命令默认 1000 条
history

清楚记录命令
history -c
cat /root/.bash_history

查看登陆过的用户信息
last
last -n user
who /var/log/wtmp

列出登录系统失败的用户相关信息
lastb -a

/var/log/btmp

防火墙日志
tail -f /var/log/messages

ssh 日志
tail -f /var/log/secure

随机生成密码
mkpasswd
-l 位数 -C 大小 -c 小写 -d 数字 -s 特殊字符
mkpasswd -l 8   -C 2 -c 2 -d 4 -s 0

当前在线用户
who
w

查看当前用户名
whoami

查看登陆用户名
logname

显示月历
cal

查看服务器启动时间
uptime

设日期
date -s 20091112

设时间
date -s 18:30:50

同步时间
/usr/sbin/ntpdate stdtime.gov.hk

7 天前日期
`date -d "7 days ago" +%Y%m%d`

日期格式转换
`date +%Y-%m-%d -d '20110902'`

日期和时间
date +%Y-%m-%d_%X

查看时间
hwclock

账户影子文件

/etc/shadow

列出所有语系
locale -a

修改语言
LANG=en

修改编码
vi /etc/sysconfig/i18n
utf-8 <---> GBK

强制字符集
export LC_ALL=C

查询静态主机名
vi /etc/hosts

最大连接
/etc/security/limits.conf

grub 开机启动项添加
/etc/grub.conf
title ms-dos
rootnoverify (hd0,0)
chainloader +1

别名
alias

监测命令
watch

查看 Linux 版本信息
uname -a
cat /proc/version
cat /etc/issue
lsb_release -a

查看 cpu 信息
more /proc/cpuinfo

查看 cpu 型号和逻辑核心数
cat /proc/cpuinfo | grep name | cut -f2 -d: | uniq -c

cpu 运行的位数
getconf LONG_BIT

物理 cpu 个数
cat /proc/cpuinfo | grep physical | uniq -c

结果大于 0 支持 64 位
cat /proc/cpuinfo | grep flags | grep ' lm ' | wc -l

查看 cpu 是否支持虚拟化
cat /proc/cpuinfo|grep flags
pae 支持半虚拟化　Intel VT 支持全虚拟化

查看内存信息
more /proc/meminfo

查看全面硬件信息
dmidecode

查看服务器型号
dmidecode | grep "Product Name"

查看软 raid 信息
cat /proc/mdstat

查看硬 raid 信息
cat /proc/scsi/scsi

查看硬件信息
lspci

查看是否支持 raid
lspci|grep RAID

显示开机信息
dmesg

进程树
pstree

每隔一秒报告虚拟内存统计信息 9 次
vmstat 1 9

把 180 号进程的优先级加 1
renice +1 180

终止某个 PID 进程
kill -9 PID

指定三天后下午 5:00 执行/bin/ls
at 5pm + 3 days /bin/ls

编辑周期任务
crontab -e
分钟　小时　　天　月　星期　　命令或脚本
01　　1-3/2　　*　　*　　*　　命令或脚本

直接将命令写入周期任务
echo "40 7 * * 2 /root/sh">>/var/spool/cron/root

查看自动周期性任务
crontab -l

删除自动周期性任务
crontab -r

禁止或允许用户使用周期任务
cron.deny 和 cron.allow

启动自动周期性服务
service crond 启动|停止|重启|状态>

是否打开了某个端口
netstat -anlp | grep 端口号

查看监听
netstat -anlp

sudo 命令权限添加
visudo
用户 别名(可用 all)=NOPASSWD:命令 1，命令 2
wangming linuxfan=NOPASSWD:/sbin/apache start,/sbin/apache restart
UserName ALL=(ALL) ALL

制作补丁
diff suzu.c suzu2.c > sz.patch

安装补丁
patch suzu.c < sz.patch

显示打开指定文件的所有进程
lsof 文件

查看端口的进程
lsof -i :32768

显示消耗内存最多的 10 个运行中的进程，以内存使用量排序.cpu +3
ps aux |grep -v USER | sort -nk +4 | tail

查看内核模块
lsmod

yum 扩展源
http://download.fedoraproject.org/pub/epel
wget http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm
rpm -Uvh epel-release-5-4.noarch.rpm

升级所有包版本,依赖关系，系统版本内核都升级
yum -y update

升级指定的软件包
yum -y update 软件包名

不改变软件设置更新软件，系统版本升级，内核不改变
yum -y upgrade

yum 搜索相关包
yum search mail

会改变配置文件,改变旧的依赖关系，改变系统版本
dist-upgrade

编辑启动项
/boot/grub/grub.conf

开机启动脚本顺序
/etc/profile
/etc/profile.d/*.sb
~/bash_profile
~/.bashrc
/etc/bashrc

---

top

前五行是系统整体的统计信息。
第一行是任务队列信息，同 uptime 命令的执行结果。其内容如下：
01:06:48 当前时间
up 1:22 系统运行时间，格式为时:分
1 user 当前登录用户数
load average: 0.06, 0.60, 0.48 系统负载，即任务队列的平均长度。
三个数值分别为 1 分钟、5 分钟、15 分钟前到现在的平均值。

第二、三行为进程和 CPU 的信息。当有多个 CPU 时，这些内容可能会超过两行。内容如下：
Tasks: 29 total 进程总数
1 running 正在运行的进程数
28 sleeping 睡眠的进程数
0 stopped 停止的进程数
0 zombie 僵尸进程数
Cpu(s): 0.3% us 用户空间占用 CPU 百分比
1.0% sy 内核空间占用 CPU 百分比
0.0% ni 用户进程空间内改变过优先级的进程占用 CPU 百分比
98.7% id 空闲 CPU 百分比
0.0% wa 等待输入输出的 CPU 时间百分比
0.0% hi
0.0% si

最后两行为内存信息。内容如下：

Mem: 191272k total 物理内存总量
173656k used 使用的物理内存总量
17616k free 空闲内存总量
22052k buffers 用作内核缓存的内存量
Swap: 192772k total 交换区总量
0k used 使用的交换区总量
192772k free 空闲交换区总量
123988k cached 缓冲的交换区总量。
内存中的内容被换出到交换区，而后又被换入到内存，但使用过的交换区尚未被覆盖，
该数值即为这些内容已存在于内存中的交换区的大小。
相应的内存再次被换出时可不必再对交换区写入。

进程信息区
统计信息区域的下方显示了各个进程的详细信息。首先来认识一下各列的含义。

序号 列名 含义
a PID 进程 id
b PPID 父进程 id
c RUSER Real user name
d UID 进程所有者的用户 id
e USER 进程所有者的用户名
f GROUP 进程所有者的组名
g TTY 启动进程的终端名。不是从终端启动的进程则显示为 ？
h PR 优先级
i NI nice 值。负值表示高优先级，正值表示低优先级
j P 最后使用的 CPU，仅在多 CPU 环境下有意义
k %CPU 上次更新到现在的 CPU 时间占用百分比
l TIME 进程使用的 CPU 时间总计，单位秒
m TIME+ 进程使用的 CPU 时间总计，单位 1/100 秒
n %MEM 进程使用的物理内存百分比
o VIRT 进程使用的虚拟内存总量，单位 kb。VIRT=SWAP+RES
p SWAP 进程使用的虚拟内存中，被换出的大小，单位 kb。
q RES 进程使用的、未被换出的物理内存大小，单位 kb。RES=CODE+DATA
r CODE 可执行代码占用的物理内存大小，单位 kb
s DATA 可执行代码以外的部分(数据段+栈)占用的物理内存大小，单位 kb
t SHR 共享内存大小，单位 kb
u nFLT 页面错误次数
v nDRT 最后一次写入到现在，被修改过的页面数。
w S 进程状态。
D=不可中断的睡眠状态
R=运行
S=睡眠
T=跟踪/停止
Z=僵尸进程
x COMMAND 命令名/命令行

y WCHAN 若该进程在睡眠，则显示睡眠中的系统函数名
z Flags 任务标志，参考 sched.h
##################################

iptables

内建三个表：nat mangle 和 filter
filter 预设规则表，有 INPUT、FORWARD 和 OUTPUT 三个规则链

INPUT 进入
FORWARD 转发
OUTPUT 出去
ACCEPT 将封包放行
REJECT 拦阻该封包
DROP 丢弃封包不予处理

-A 在所选择的链(INPUT 等)末添加一条或更多规则
-D 删除一条
-E 修改
-p tcp、udp、icmp 0 相当于所有 all!取反
-P 设置缺省策略(与所有链都不匹配强制使用此策略)
-s IP/掩码 (IP/24) 主机名、网络名和清楚的 IP 地址 !取反
-j 目标跳转，立即决定包的命运的专用内建目标
-i 进入的（网络）接口 [名称] eth0
-o 输出接口[名称]
-m 模块
--sport 源端口
--dport 目标端口

#配置文件
vi /etc/sysconfig/iptables
#将防火墙中的规则条目清除掉
iptables -F
#注意:iptables -P INPUT ACCEPT
#导入防火墙规则
iptables-restore <规则文件
#保存防火墙设置
/etc/init.d/iptables save
#重启防火墙服务
/etc/init.d/iptables restart
#查看规则
iptables -L -n
iptables -L -n --line-numbers
#从某个规则链中删除一条规则
iptables -D INPUT --dport 80 -j DROP
iptables -D INPUT 8
#取代现行规则
iptables -R INPUT 8 -s 192.168.0.1 -j DROP
#插入一条规则

```
iptables -I INPUT 8 --dport 80 -j ACCEPT
#查看转发
iptables -t nat -nL
#在内核里打开 ip 转发功能
echo 1 > /proc/sys/net/ipv4/ip_forward

####################################

#允许本地回环
iptables -A INPUT -s 127.0.0.1 -p tcp -j ACCEPT
#允许已建立的或相关连的通行
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
#开放对外访问
iptables -P OUTPUT ACCEPT
#指定某端口针对 IP 开放
iptables -A INPUT -s 192.168.10.37 -p tcp --dport 22 -j ACCEPT
#允许的 IP 或 IP 段访问
iptables -A INPUT -s 192.168.10.37 -p tcp -j ACCEPT
#开放对外开放端口
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
#关闭入口
iptables -P INPUT DROP
#关闭转发
iptables -P FORWARD DROP

####################################

        iptables 规则文件

# Generated by iptables-save v1.2.11 on Fri Feb    9 12:10:37 2007
*filter
:INPUT DROP [637:58967]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [5091:1301533]
#允许的 IP 或 IP 段访问
-A INPUT -s 127.0.0.1 -p tcp -j ACCEPT
-A INPUT -s 192.168.0.0/255.255.0.0 -p tcp -j ACCEPT
#开放对外开放端口
-A INPUT -p tcp --dport 80 -j ACCEPT
#指定某端口针对 IP 开放
-A INPUT -s 192.168.10.37 -p tcp --dport 22 -j ACCEPT
#拒绝所有协议(INPUT 允许的情况)
#-A INPUT -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,URG RST -j DROP
#  允许已建立的或相关连的通行
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
#拒绝 ping
-A INPUT -p tcp -m tcp -j REJECT --reject-with icmp-port-unreachable
COMMIT
# Completed on Fri Feb    9 12:10:37 2007

####################################
```

常用实例

```
#允许在 IP 访问指定端口
iptables -A INPUT -s 192.168.62.1 -p tcp --dport 22 -j ACCEPT
#禁止使用某端口
iptables -A INPUT -p tcp --dport 22 -j DROP
iptables -A INPUT -p tcp --dport 31 -j REJECT
#禁止 QQ 端口
iptables -D FORWARD -p udp --dport 8000 -j REJECT
#禁止 icmp 端口
#除 192.168.62.1 外，禁止其它人 ping 我的主机
iptables -A INPUT -i eth0 -s 192.168.62.1/32 -p icmp -m icmp --icmp-type 8 -j ACCEPT
iptables -A INPUT -i eth0 -p icmp -m icmp --icmp-type 8 -j DROP
#其它情况不允许
iptables -A INPUT -i eth0 -j DROP
#禁止指定 IP 访问
iptables -A INPUT -p tcp -s IP -j DROP
#禁止指定 IP 访问端口
iptables -A INPUT -p tcp -s IP --dport port -j DROP
#阻止所有没有经过你系统授权的 TCP 连接
iptables -t filter -A INPUT -i eth0 -p tcp --syn -j DROP
#添加网段转发
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -j MASQUERADE
#IP 包流量限制
iptables -A INPUT -f -m limit --limit 100/s --limit-burst 100 -j ACCEPT
iptables -A INPUT -i eth0 -p icmp -j DROP
#端口映射
iptables -t nat -A PREROUTING -i eth0 -p tcp -s 192.168.62.0/24 --dport 80 -j REDIRECT --to-ports
3128
#列出某规则链中的所有规则
iptables -L INPUT
#删除某个规则链 ,不加规则链，清除所有非内建的
iptables -X allowed
#将封包计数器归零
iptables -Z INPUT
#定义新的规则链
iptables -N allowed
#定义过滤政策
iptables -P INPUT DROP
#修改某自订规则链的名称
iptables -E allowed disallowed
#比对通讯协议类型是否相符
#-p ! tcp  排除 tcp 以外的 udp、icmp。-p all 所有类型
iptables -A INPUT -p tcp
#比对封包的来源  IP
#192.168.0.0/24  ！反向对比
iptables -A INPUT -s 192.168.1.1
#比对封包的目的地  IP
iptables -A INPUT -d 192.168.1.1
```

#比对封包是从哪片网卡进入
#eth+表示所有的网卡
iptables -A INPUT -i eth0
#比对封包要从哪片网卡送出
iptables -A FORWARD -o eth0
#比对某段时间内封包的平均流量
#例子是用来比对：每小时平均流量是否超过一次 3 个封包。 除了每小时平均次外，也可以每秒钟、每分钟或每天平均一次，默认值为每小时平均一次，参数如后： /second、 /minute、/day。除了进行封数量的比对外，设定这个参数也会在条件达成时，暂停封包的比对动作，以避免因骇客使用洪水攻击法，导致服务被阻断。
iptables -A INPUT -m limit --limit 3/hour
#比对瞬间大量封包的数量
#例子是用来比对一次同时涌入的封包是否超过 5 个(这是默认值)，超过此上限的封包将被直接丢弃。使用效果同上。
iptables -A INPUT -m limit --limit-burst 5
#比对来自本机的封包
#是否为某特定使用者所产生的，这样可以避免服务器使用 root 或其它身分将敏感数据传送出，可以降低系统被骇的损失。可惜这个功能无法比对出来自其它主机的封包。
iptables -A OUTPUT -m owner --uid-owner 500
#比对来自本机的封包
iptables -A OUTPUT -m owner --gid-owner 0
iptables -A OUTPUT -m owner --pid-owner 78
iptables -A OUTPUT -m owner --sid-owner 100
#用来比对联机状态
iptables -A INPUT -m state --state RELATED,ESTABLISHED
联机状态共有四种：INVALID、ESTABLISHED、NEW 和 RELATED。
#-j 参数指定进行的处理动作，处理动作包括： ACCEPT、REJECT、DROP、REDIRECT、MASQUERADE、LOG、DNAT、SNAT、MIRROR、QUEUE、RETURN、MARK,说明：
iptables -A FORWARD -p TCP --dport 22 -j REJECT --reject-with tcp-reset
#可以传送的封包有几个选择：ICMP port-unreachable、ICMP echo-reply 或是 tcp-reset
REDIRECT 将封包重新导向到另一个端口(PNAT)
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080
MASQUERADE 改写封包来源 IP 为防火墙 NIC IP
iptables -t nat -A POSTROUTING -p TCP -j MASQUERADE --to-ports 1024-31000
LOG 将封包相关讯息纪录在 /var/log 中
iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT packets"
SNAT 改写封包来源 IP 为某特定 IP 或 IP 范围
iptables -t nat -A POSTROUTING -p tcp-o eth0 -j SNAT --to-source 194.236.50.155-194.236.50.160:1024-32000
DNAT 改写封包目的地 IP 为某特定 IP 或 IP 范围
iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 --dport 80 -j DNAT --to-destination 192.168.1.1-192.168.1.10:80-100
MIRROR 镜射封包
QUEUE 中断过滤程序
RETURN 结束在目前规则炼中的过滤程序
MARK 将封包标上某个代号
##################################
iptables 配置实例
允许某段 IP 访问任何端口

iptables -A INPUT -s 192.168.0.3/24 -p tcp -j ACCEPT
设定预设规则 (拒绝所有的数据包，再允许需要的,如只做 WEB 服务器.还是推荐三个链都是 DROP)
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT
(注:意直接设置这三条完已经掉线了)
开启 22 端口
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
如果 OUTPUT 设置成 DROP 的，要写上下面一条
iptables -A OUTPUT -p tcp --sport 22 -j ACCEPT
(注:,不写导致无法 SSH.其他的端口一样,OUTPUT 设置成 DROP 的话,也要添加一条链)
如果开启了 web 服务器,OUTPUT 设置成 DROP 的话,同样也要添加一条链
iptables -A OUTPUT -p tcp --sport 80 -j ACCEPT
做 WEB 服务器,开启 80 端口 ,其他同理
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
做邮件服务器,开启 25,110 端口
iptables -A INPUT -p tcp --dport 110 -j ACCEPT
iptables -A INPUT -p tcp --dport 25 -j ACCEPT
允许 icmp 包通过,允许 ping
iptables -A OUTPUT -p icmp -j ACCEPT (OUTPUT 设置成 DROP 的话)
iptables -A INPUT -p icmp -j ACCEPT (INPUT 设置成 DROP 的话)
允许 loopback!(不然会导致 DNS 无法正常关闭等问题)
IPTABLES -A INPUT -i lo -p all -j ACCEPT (如果是 INPUT DROP)
IPTABLES -A OUTPUT -o lo -p all -j ACCEPT(如果是 OUTPUT DROP)

==========

---

4 服务管理

启动 sendmail 服务
./sendmail start
/etc/init.d/sendmail start

关闭 sendmail 服务
./sendmail stop

查看 sendmail 服务当前状态
./sendmail status

发送邮件
echo 内容| /bin/mail -s "标题" 收件箱

打开服务列表 需要打*
ntsysv

让某个服务不自动启动 35 指的是运行级别

httpd：chkconfig --level 35 httpd off

让某个服务自动启动
httpd：chkconfig --level 35 httpd on

查看所有服务的启动状态
chkconfig --list

查看某个服务的启动状态
chkconfig --list |grep httpd

查看服务的状态
chkconfig –-list [服务名称]

设置非独立服务启状态
chkconfig 服务名 on|off|set

开启 mysql 后台运行
/usr/local/mysql/bin/mysqld_safe --user=mysql &

开机启动执行
vi /etc/rc.d/rc.local

开机启动和关机关闭服务连接
/etc/rc.d/rc3.d/S55sshd
S 开机 start K 关机 stop 55 级别 后跟服务名
ln -s -f /usr/local/httpd/bin/apachectl /etc/rc.d/rc3.d/S15httpd

==========

---

5 网络管理

##########
本机网络配置文件
vi /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE=eth0
BOOTPROTO=none
BROADCAST=192.168.1.255
HWADDR=00:0C:29:3F:E1:EA
IPADDR=192.168.1.55
NETMASK=255.255.255.0
NETWORK=192.168.1.0
ONBOOT=yes
TYPE=Ethernet
GATEWAY=192.168.1.1
##########
增加逻辑 IP 地址
ifconfig eth0:0 192.168.1.221 netmask 255.255.255.0

查看路由表
route

添加路由表
route add default gw 192.168.1.1 dev eth0

设置 DNS
vi /etc/resolv.conf

禁用网卡
ifconfig eth0 down

启用网卡
ifconfig eth0 up
ifup eth0:0 up

测试跳数
traceroute www.baidu.com (linux)
tracert www.baidu.com (windows)

根据 IP 和主机最大数计算掩码
ipcalc -m "$ip" -p "$num"

用 wget 的递归方式下载整个网站
wget --random-wait -r -p -e robots=off -U Mozilla www.example.com

通过 DNS 来读取 Wikipedia 的 hacker 词条
dig +short txt hacker.wp.dg.cx
host -t txt hacker.wp.dg.cx

rz ssh 小文件上传
sz ssh 小文件下载

从 linux ssh 登录另一台 linux
ssh -p 22 wang@192.168.1.209

利用 ssh 操作远程主机
ssh -p 22 root@192.168.1.209 环境变量中脚本

把本地文件拷贝到远程主机
scp -P 22 文件 root@ip:/目录

ssh 连接不提示 yes
ssh -o StrictHostKeyChecking=no 192.168.70.130

把远程指定文件拷贝到本地
scp root@192.168.1.209:远程目录 本地目录

通过 SSH 挂载远程主机上的文件夹
sshfs name@server:/path/to/folder /path/to/mount/point

卸载的话使用 fusermount 命令
fusermount -u /path/to/mount/point

用 DIFF 对比远程文件跟本地文件
ssh user@host cat /path/to/remotefile | diff /path/to/localfile -

用 SSH 创建端口转发通道
ssh -N -L2001:remotehost:80 user@somemachine

嵌套使用 SSH
ssh -t host_A ssh host_B

密钥信任
ssh-keygen –t rsa
vi 用户/.ssh/authorized_keys

远程关掉一台 WINDOWS 机器
net rpc shutdown -I IP_ADDRESS -U username%password

禁 ping
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

####################################

先 ping 在扫描主机开放端口
nmap -PT 192.168.1.1-111

扫描出系统内核版本
nmap -O 192.168.1.1

扫描端口的软件版本
nmap -sV 192.168.1.1-111

半开扫描（通常不会记录日志）
nmap -sS 192.168.1.1-111

不支持 windows 的扫描（判断是否是 windows）
nmap -sF 192.168.1.1-111
nmap -sX 192.168.1.1-111
nmap -sN 192.168.1.1-111

不 ping 直接扫描
nmap -P0 192.168.1.1-111

详细信息
nmap -d 192.168.1.1-111

无法找出真正扫描主机（隐藏 IP）
nmap -D 192.168.1.1-111

端口范围
nmap -p 20-30,139,60000-
表示：扫描 20 到 30 号端口，139 号端口以及所有大于 60000 的端口

组合扫描（不 ping、软件版本、内核版本、详细信息）
nmap -P0 -sV -O -v 192.168.30.251

###################################

==========

---
6 磁盘管理

查看硬盘容量
df -h

查看磁盘分区格式
df -T

修改只读文件系统为读写
mount -o remount,rw /

查看卷标
e2label /dev/sda5

创建卷标
e2label /dev/sda5 new-label

NTFS 添加卷标
ntfslabel -v /dev/sda8 new-label

ext2 分区转 ext3 分区
tune2fs -j /dev/sda

格式化分区
mkfs -t ext3 /dev/hda3

指定索引块大小
mke2fs -b 2048 /dev/sda5

对文件系统修复
fsck -y /dev/sda6

查看超级块的信息
dumpe2fs -h /dev/sda5

查看当前系统支持文件系统
cat /proc/filesystems

检测硬盘状态
smartctl -H /dev/sda

检测硬盘信息
smartctl -i /dev/sda

检测所有信息
smartctl -a /dev/sda

检测目录下所有文件大小
du -h 目录

显示当前目录中子目录的大小
du -h --max-depth=1

挂载光驱
mount -t iso9660 /dev/dvd /mnt

挂载镜像文件
mount -o loop /software/rhel4.6.iso /mnt/

##################################

磁盘 grub 开机引导项添加
/etc/grub.conf
title ms-dos
rootnoverify (hd0,0)
chainloader +1

##################################

建立软 RAID1
两块硬盘分区一样，分别新建分区文件类型为 software RAID
创建两个就点击 raid 合并为 RAID1，这里的挂载点为正常 linux 目录
查看 raid 工作状态
cat /proc/mdstat
两个盘都加 grub 引导
查看
cat /boot/grub/grub.conf
grub
root (hd0,0)
setup (hd0)
root (hd1,0)
setup (hd1)
修改 grub 启动项配置文件
vi /boot/grub/grub.conf
复制 title CentOS 到最后的四行
在下面粘贴修改为刚才新加的引导(hd1,0)。
查看分区
sfdisk -d /dev/hda
导出 A 盘分区列表

sfdisk -d /dev/hda > fq.hda
将分区列表导入到新加的硬盘
sfdisk /dev/hdb < fq.hda
恢复新挂载的硬盘分区
mdadm /dev/md1 -a /dev/hdb5
raid5 可利用 lvm 扩容

##################################

新硬盘挂载
fdisk /dev/sdc
p　　#　打印分区
d　　#　删除分区
n　　#　创建分区，（一块硬盘最多 4 个主分区，扩展占一个主分区位置。p 主分区 e 扩展）
w　　#　保存退出
mkfs -t ext3 -L 卷标　/dev/sdc1　　#　格式化相应分区
mount /dev/sdc1　/mnt　　#　挂载

添加开机挂载分区
vi /etc/fstab
用卷标挂载
LABEL=/data　　/data　　ext3　　defaults　　1 2
用真实分区挂载
/dev/sdb1　　/data4　　ext3　　defaults　　1 2

第一个数字"1"该选项被"dump"命令使用来检查一个文件系统应该以多快频率进行转储，若不需要转储就设置该字段为 0
第二个数字"2"该字段被 fsck 命令用来决定在启动时需要被扫描的文件系统的顺序，根文件系统"/"对应该字段的值应该为 1，其他文件系统应该为 2。若该文件系统无需在启动时扫描则设置该字段为 0

##################################

==========

---
7 用户管理

建立用户
useradd xuesong

修改密码
passwd 用户
echo "xuesong" | passwd xuesong --stdin

查找用户显示信息
finger

添加组

groupadd

修改文件拥有者(R 递归)
chown -R

修改所有者用户中包含点"."
chown y\.li:mysql

修改用户组
chgrp

修改用户名
usermod -l 新用户名 老用户名

修改用户所属组
usermod -g user group

修改用户家目录
usermod -d 目录 -m 用户

将用户添加到附加组
usermod -G user group

删除帐号及家目录
userdel -r

指定组并不允许登录的用户
useradd -g www -M -s /sbin/nologin www

切换用户执行
su - user -c "
#命令 1
"
==========

---
8 脚本相关

##########

正则表达式

^ 行首定位
$ 行为定位
. 匹配除换行符以外的任意字符
* 匹配 0 或多个重复字符
+ 重复一次或更多次
? 重复零次或一次
[] 匹配一组中任意一个字符

[^]　匹配不在指定组内的字符
\　　用来转义元字符
\<　　词首定位符(支持 vi 和 grep)
\<love
\>　　词尾定位符(支持 vi 和 grep)
love\>
x\{m\}　　重复出现 m 次
x\{m,\}　　重复出现至少 m 次
x\{m,n\}　　重复出现至少 m 次不超过 n 次
X?　匹配出现零次或一次的大写字母 X
X+　匹配一个或多个字母 X
(abc|def)+ 匹配一连串的（最少一个） abc 或 def；abc 和 def 将匹配
\d　　匹配任意一位数字
\D　　匹配任意单个非数字字符
\w　　匹配任意单个字母数字字符，同义词是 [:alnum:]
\s　　匹配任意的空白符
\b　　匹配单词的开始或结束
[:alpha:]　代表所有字母不论大小写
[:lower:]　表示小写字母
[:upper:]　表示大写字母
[:digit:]　表示数字字符
()　　括号内的字符为一组

##################################

　　基本流程结构

```
if [ $a == $b ]
then
echo "等于"
else
echo "不等于"
fi
##################################
case $xs in
0) echo "0" ;;
1) echo "1" ;;
*) echo "其他" ;;
esac
##################################
num=1
# while true　等同　　while :
while [ $num -lt 10 ]
do
echo $num
((num=$num+2))
done
##################################
grep a　a.txt | while read a
do
```

```
        echo $a
done
###################################
w=`awk -F ":" '{print $1}' c`
for d in $w
do
        $d
done
###################################
for ((i=0;i<${#o[*]};i++))
do
echo ${o[$i]}
done
###################################
until 循环#    当 command 不为 0 时
until command
do
        body
done
###################################
流程控制
break N         #    跳出几层循环
continue N    #    跳出几层循环，循环次数不变
continue        #    重新循环次数不变
###################################

        变量

将  变量 A  赋值为字串
A="a b c def"
A=`命令`
间接调用
eval a=\$$a
将  变量 A  定义为組数
A=(a b c def)
set a1 a2 a3 a4
$1    $2    $*
在子 shell 中运行
(a=bbk)

        定义变量类型

declare 或者 typeset
-r  只读(readonly 一样)
-i  整形
-a  数组
-f  函数
-x export
declare -i n=0

env                             #    查看环境变量
```

```
env | grep "name"        #   查看定义的环境变量
set                      #   查看环境变量和本地变量
read name                #   输入变量
readonly name            #   把 name 这个变量设置为只读变量,不允许再次设置
readonly                 #   查看系统存在的只读文件
export name              #   变量 name 由本地升为环境
export name="RedHat"  #   直接定义 name 为环境变量
export Stat$nu=2222     #   变量引用变量赋值
unset name               #   变量清除
export -n name           #   去掉只读变量
shift                    #   用于移动位置变量,调整位置变量,使$3 的值赋给$2.$2 的值赋予$1
name + 0                 #   将字符串转换为数字
number " "               #   将数字转换成字符串
```

常用系统变量

```
$0     #   脚本启动名(包括路径)
basename $0 #   只取脚本名
$n     #   当前程式的第 n 个参数,n=1,2,…9
$*     #   当前程式的所有参数(不包括脚本本身)
$#     #   当前程式的参数个数(不包括脚本本身)
$$     #   当前程式的 PID
$!     #   执行上一个指令的 PID
$?     #   执行上一个指令的返回值
```

变量引用技巧 ${}

```
${name:+value}          #   如果设置了 name,就把 value 显示,未设置则为空
${name:-value}          #   如果设置了 name,就显示它,未设置就显示 value
${name:?value}          #   未设置提示用户错误信息 value
${name:=value}          #   如未设置就把 value 设置并显示<写入本地中>,
${#A}                    #   可得到变量中字节
${#A[*]}                 #   数组个数
${A[2]}                  #   脚本的一个参数
${A:4:9}                 #   取变量中第 4 位到后面 9 位
${A/www/http}            #   取变量并且替换每行第一个关键字
${A//www/http}           #   取变量并且全部替换每行关键字
```

定义了一个变量为：

```
file=/dir1/dir2/dir3/my.file.txt
${file#*/}       #   去掉第一条 / 及其左边的字串：dir1/dir2/dir3/my.file.txt
${file##*/}      #   去掉最后一条 / 及其左边的字串：my.file.txt
${file#*.}       #   去掉第一个 .  及其左边的字串：file.txt
${file##*.}      #   去掉最后一个 .  及其左边的字串：txt
${file%/*}       #   去掉最后条 / 及其右边的字串：/dir1/dir2/dir3
${file%%/*}      #   去掉第一条 / 及其右边的字串：(空值)
${file%.*}       #   去掉最后一个 .  及其右边的字串：/dir1/dir2/dir3/my.file
${file%%.*}      #   去掉第一个 .  及其右边的字串：/dir1/dir2/dir3/my
```

是去掉左边(在键盘上 # 在 $ 之左边)
% 是去掉右边(在键盘上 % 在 $ 之右边)
单一符号是最小匹配；两个符号是最大匹配

##################################

test 条件判断

符号 [ ] 等同 test

expression 为字符串操作

-n str # 字符串 str 是否不为空
-z str # 字符串 str 是否为空

expression 为文件操作

-b # 是否块文件
-p # 文件是否为一个命名管道
-c # 是否字符文件
-r # 文件是否可读
-d # 是否一个目录
-s # 文件的长度是否不为零
-e # 文件是否存在
-S # 是否为套接字文件
-f # 是否普通文件
-x # 文件是否可执行，则为真
-g # 是否设置了文件的 SGID 位
-u # 是否设置了文件的 SUID 位
-G # 文件是否存在且归该组所有
-w # 文件是否可写，则为真
-k # 文件是否设置了的粘贴位
-t fd # fd 是否是个和终端相连的打开的文件描述符（fd 默认为 1）
-O # 文件是否存在且归该用户所有
! # 取反

expression 为整数操作

expr1 -a expr2 # 如果 expr1 和 expr2 评估为真，则为真
expr1 -o expr2 # 如果 expr1 或 expr2 评估为真，则为真

两值比较

整数 字符串
-lt <: # 小于
-gt >: # 大于
-le <=: # 小于或等于
-ge >=: # 大于或等于

-eq　　==：　　#　等不等
-ne　　!=：　　#　不等于

判断大小,0 为真,1 为假
test 10 -lt 5
echo $?
1
判断字符串长度是否为 0
test -n "hello"
0

####################################

重定向

#　标准输出 stdout 和 标准错误 stderr　标准输入 stdin
cmd 1> fiel　　#　把 标准输出 重定向到 file 文件中
cmd > file 2>&1　　#　把 标准输出 和 标准错误 一起重定向到 file 文件中
cmd 2> file　　#　把 标准错误 重定向到 file 文件中
cmd 2>> file　　#　把 标准错误 重定向到 file 文件中(追加)
cmd >> file 2>&1　　#　把 标准输出 和 标准错误 一起重定向到 file 文件中(追加)
cmd < file >file2　　#　cmd 命令以 file 文件作为 stdin(标准输入)，以 file2 文件作为 标准输出
cat <>file　　#　以读写的方式打开 file
cmd < file cmd　　#　命令以 file 文件作为 stdin
cmd << delimiter
cmd; #从 stdin 中读入，直至遇到 delimiter 分界符。
delimiter

>&n　　#　使用系统调用 dup (2) 复制文件描述符 n 并把结果用作标准输出
<&n　　#　标准输入复制自文件描述符 n
<&-　　#　关闭标准输入（键盘）
>&-　　#　关闭标准输出
n<&-　　#　表示将 n 号输入关闭
n>&-　　#　表示将 n 号输出关闭

####################################

运算符

$[]等同于$(())　#　$[]表示形式告诉 shell 求中括号中的表达式的值
~var　　#　按位取反运算符,把 var 中所有的二进制为 1 的变为 0,为 0 的变为 1
var\<<str　　#　左移运算符,把 var 中的二进制位向左移动 str 位,忽略最左端移出的各位,最右端的各位上补上 0 值,每做一次按位左移就有 var 乘 2
var>>str　　#　右移运算符,把 var 中所有的二进制位向右移动 str 位,忽略最右移出的各位,最左的各位上补 0,每次做一次右移就有实现 var 除以 2
var&str　　#　与比较运算符,var 和 str 对应位,对于每个二进制来说,如果二都为 1,结果为 1.否则为 0
var^str　　#　异或运算符,比较 var 和 str 对应位,对于二进制来说如果二者互补,结果为 1,

否则为 0
var|str　　#　或运算符,比较 var 和 str 的对应位,对于每个二进制来说,如二都该位有一个 1 或都是 1,结果为 1,否则为 0

运算符的优先级

级别	运算符	说明
1	=,+=,-=,/=,%=,*=,&=,^=,|=,<<=,>>==	# 赋值运算符
2	||	# 逻辑或 前面不成功执行
3	&&	# 逻辑与 前面成功后执行
4	|	# 按位或
5	^	# 按异位与
6	&	# 按位与
7	==,!=	# 等于/不等于
8	<=,>=,<,>	# 大于或等于/小于或等于/大于/小于
9	\<<,>>	# 按位左移/按位右移 (无转意符号)
10	+,-	# 加减
11	*,/,%	# 乘,除,取余
12	! ,~	# 逻辑非,按位取反或补码
13	-,+	# 正负

##################################

数学运算

$(( )) 整数运算符号大致有这些：
+ - * / **：分别为 "加、减、乘、除、密运算"
& | ^ !：分别为 "AND、OR、XOR、NOT" 运算
% ：余数运算

运算
let
let x=16/4
let x=5**5

手工命令行计数器
expr
SUM=`expr 2 \* 3`

计算字串长度
expr length "bkeep zbb"

抓取字串
expr substr "bkeep zbb" 4 9

抓取第一个字符数字串出现的位置
expr index "bkeep zbb" e

整数运算
expr 14 % 9

expr 30 / 3 / 2　　#　运算符号有空格

增量计数(加循环即可)
LOOP=0
LOOP=`expr $LOOP + 1`

数值测试(如果试图计算非整数，则会返回错误。)
rr=3.4
expr $rr + 1
expr: non-numeric argument
rr=5
expr $rr + 1
6

模式匹配(可以使用 expr 通过指定冒号选项计算字符串中字符数)
.* 意即任何字符重复 0 次或多次
expr bkeep.doc : '.*'

在 expr 中可以使用字符串匹配操作，这里使用模式抽取.doc 文件附属名。
expr bkeep.doc : '\(.*\).doc'

次方计算
echo "m^n"|bc

####################################

grep 用法

-c 显示匹配到得行的数目，不显示内容
-h 不显示文件名
-i 忽略大小写
-l 只列出匹配行所在文件的文件名
-n 在每一行中加上相对行号
-s 无声操作只显示报错，检查退出状态
-v 反向查找
-e 使用正则表达式
-A1 打印匹配行和下一行
-wc 匹配出现次数
grep 可用于 if 判断，找到$?为 then

过滤关键字符行
grep -v "a" txt

精确匹配字符串
grep 'a\>' txt

大小写敏感
grep -i "a" txt

同时匹配大小写

grep "a[bB]" txt

查找 0-9 重复三次的所在行
grep '[0-9]\{3\}' txt

任意条件匹配
grep -E "word1 | word2 | word3" file

同时匹配三个
grep word1 file | grep word2 |grep word3

##################################

tr 用法

-c 用字符串 1 中字符集的补集替换此字符集，要求字符集为 ASCII。
-d 删除字符串 1 中所有输入字符。
-s 删除所有重复出现字符序列，只保留第一个；即将重复出现字符串压缩为一个字符串。

[a-z] a-z 内的字符组成的字符串。
[A-Z] A-Z 内的字符组成的字符串。
[0-9] 数字串。
\octal 一个三位的八进制数，对应有效的 ASCII 字符。
[O*n] 表示字符 O 重复出现指定次数 n。因此[O*2]匹配 OO 的字符串。
tr 中特定控制字符的不同表达方式
速记符含义八进制方式
\a Ctrl-G 铃声\007
\b Ctrl-H 退格符\010
\f Ctrl-L 走行换页\014
\n Ctrl-J 新行\012
\r Ctrl-M 回车\015
\t Ctrl-I tab 键\011
\v Ctrl-X \030

将所有大写转换成小写字母
tr A-Z a-z

将空格替换为换行
tr " " "\n"

删除空行
tr -s "[\012]" < plan.txt
tr -s ["\n"] < plan.txt

删除文件中的^M，并代之以换行
tr -s "[\015]" "[\n]" < file
tr -s "[\r]" "[\n]" < file

替换 passwd 文件中所有冒号，代之以 tab 键
tr -s "[:]" "[\011]" < /etc/passwd

tr -s "[:]" "[\t]" < /etc/passwd

增加显示路径可读性
echo $PATH | tr ":" "\n"

tr 在 vi 内使用，在 tr 前加处理行范围和感叹号('$'表示最后一行)
1,$!tr -d '\t'

Mac -> UNIX
tr "\r" "\n"<macfile > unixfile

UNIX -> Mac
tr "\n" "\r"<unixfile > macfile

DOS -> UNIX
Microsoft DOS/Windows 约定，文本的每行以回车字符(\r)并后跟换行符(\n)结束
tr -d "\r"<dosfile > unixfile

UNIX -> DOS：在这种情况下，需要用 awk，因为 tr 不能插入两个字符来替换一个字符
awk '{ print $0"\r" }'<unixfile > dosfile

####################################

awk 用法

\b 退格
\f 换页
\n 换行
\r 回车
\t 制表符 Tab
\c 代表任一其他字符
-F 改变 FS 值(分隔符)

~ 匹配
= 赋值
== 匹配
+= 叠加

[:alnum:] 字母数字字符
[:alpha:] 字母字符
[:cntrl:] 控制字符
[:digit:] 数字字符
[:graph:] 非空白字符(非空格、控制字符等)
[:lower:] 小写字母
[:print:] 与[:graph:]相似，但是包含空格字符
[:punct:] 标点字符
[:space:] 所有的空白字符(换行符、空格、制表符)
[:upper:] 大写字母
[:xdigit:] 十六进制的数字(0-9a-fA-F)

内建变量

$n	当前记录的第 n 个字段，字段间由 FS 分隔
$0	完整的输入记录
ARGC	命令行参数的数目
ARGIND	命令行中当前文件的位置 (从 0 开始算)
ARGV	包含命令行参数的数组
CONVFMT	数字转换格式 (默认值为 %.6g)
ENVIRON	环境变量关联数组
ERRNO	最后一个系统错误的描述
FIELDWIDTHS	字段宽度列表 (用空格键分隔)
FILENAME	当前文件名
FNR	同 NR ，但相对于当前文件
FS	字段分隔符 (默认是任何空格)
IGNORECASE	如果为真（即非 0 值），则进行忽略大小写的匹配
NF	当前记录中的字段数(列)
NR	当前行数
OFMT	数字的输出格式 (默认值是 %.6g)
OFS	输出字段分隔符 (默认值是一个空格)
ORS	输出记录分隔符 (默认值是一个换行符)
RLENGTH	由 match 函数所匹配的字符串的长度
RS	记录分隔符 (默认是一个换行符)
RSTART	由 match 函数所匹配的字符串的第一个位置
SUBSEP	数组下标分隔符 (默认值是 /034) 。
BEGIN	先处理(可不加文件参数)
END	结束时处理

[[:digit:][:lower:]]	数字和小写字母(占一个字符)
-F"[ ]+|[%]+"	已多个空格或多个%为分隔符
[a-z]+	多个小写字母
[a-Z]	代表所有大小写字母(aAbB...zZ)
[a-z]	代表所有大小写字母(ab...z)

打印匹配到得行
awk '/Tom/' 文件

如果第三个字段值小于 4000 才打印
awk '$3 <4000' 文件

匹配 Tom 开头的行 打印第一个字段
awk '/^Tom/{print $1}'

显示所有第一个字段不是以 ly 结尾的行
awk '$1 !~ /ly$/' 文件

条件判断(如果$1 大于$2,max 值为为$1,否则为$2)
awk '{max=($1 > $2) ? $1 : $2; print max}' 文件
awk '{print ($1>$2)?"第一排"$1:"第二排"$2}' e.txt

(括号代表 if 语句判断 "?"代表 then ":"代表 else)

算术运算(第三个字段和第四个字段乘积大于 500 则显示)
awk '$3 * $4 > 500' 文件

打印 tom 到 suz 之间的行
awk '/tom/,/suz/' 文件

去掉前三行
awk '{$1="";$2="";$3="";print}' a.sh

将 date 值赋给 d，并将 d 设置为数组 mon，打印 mon 数组中第 2 个元素
awk 'BEGIN{ "date" | getline d; split(d,mon) ; print mon[2]}' 文件

以空格、:、制表符 Tab 为分隔符
awk -F'[:\t]' '{print $1,$2}'

取关键字下第几行
awk '/关键字/{a=NR+2}a==NR {print}'

awk 中引用变量
a=22aa &&echo 88:99:44|awk -F":" '{print $1,""$a"",$2,$3}'

指定类型(%d 数字,%s 字符)
/sbin/ifconfig |awk '{printf("line %d,%s\n",NR,$0)}'
awk -v RS=# 'NF{for(i=1;i<=NF;i++)if($i) printf $i;print ""}' 文件

awk 加 if 判断
awk '{if ($6 > 50) print $1 " Too high" ;\
else print "Range is OK"}' file
awk '{if ($6 > 50) { count++;print $3 } \
else { x+5; print $2 } }' file

awk 加循环
awk '{i = 1; while (i <= NF) { print NF, $i ; i++ } }' file
awk '{ for (i = 1; i <= NF; i++) print NF,$i }' file

提取时间，空格不固定
ll | awk -F'[ ]+|[ ][ ]+' '/^$/{print $8}'

查看磁盘空间
df|awk -F"[ ]+|%" '$5>14{print $5}'

取出第四列等于 90 的第五列
awk '$4==90{print $5}'

打印所有以模式 no 或 so 开头的行
awk '/^(no|so)/' test

排列打印

```
awk 'END{printf "%-10s%-10s\n%-10s%-10s\n%-10s%-10s\n","server","name","123","12345","234","1234"}' txt
awk 'BEGIN{printf "|%-10s|%-10s|\n|%-10s|%-10s|\n|%-10s|%-10s|\n","server","name","123","12345","234","1234"}'
awk 'BEGIN{
print "     *** 开 始 ***     ";
print "+-----------------+";
printf "|%-5s|%-5s|%-5s|\n","id","name","ip";
}
$1!=1 && NF==4{printf "|%-5s|%-5s|%-5s|\n",$1,$2,$3" "$11}
END{
print "+-----------------+";
print "     *** 结 束 ***     "
}' txt
```

awk 中计算(求余数)

```
echo list|awk '{ i=($1%100);if ( $i >= 0 ) {print $0,$i}}'
```

sub 匹配第一次出现的符合模式的字符串，相当于 sed 's//'

```
awk '{sub(/Mac/,"Macintosh");print}' urfile
#用 Macintosh 替换 Mac
```

gsub 匹配所有的符合模式的字符串，相当于 sed 's//g'

```
awk '{sub(/Mac/,"MacIntosh",$1); print}' file
#第一个域内用 Macintosh 替换 Mac
```

处理 sql 语句

```
cat 1.txt|awk -F" # " '{print "insert into user (user,password,email)values("""\""$1""\"\,"""\""$2""\"\,"""\""$3""\"\)\;"'}' >>insert_1.txt
```

引用外部变量

```
awk '{print """$a""","""$b"""}'
```

在 END 块里面把数组内容写到文件

```
awk -F: '{name[x++]=$1};END{for(i=0;i<NR;i++)print i,name[i]}' data >tmp
```

将$1 的值叠加后赋给 sum

```
awk 'sum+=$1{print sum}' <<EOF
1 11 111
2 22 222
8 33 333
10 44 444
EOF
```

加入 END 只打印最后的结果

```
awk '{sum2+=$2;count=count+1}END{print sum2,sum2/count}' <<EOF
1324236000: 4.8726625090e+06 1.4806911317e+07
1324236300: 3.1952608823e+05 1.3144868085e+07
1324236600: 5.0792587262e+05 1.4931600767e+07
EOF
#结果中第一个值是$2 的总和 第二个值是$3 的总和 第三个值是$2 总和除个数（平均值） 第
```

四个值是$3 总和除个数（平均值）
#e+06 是科学计数法，表示乘以 10 的 6 次方

```
awk '{sum2+=$2;count=count+1}END{print sum2,sum2/count}' <<EOF
1: 3 30
2: 6 60
3: 9 90
EOF
```

列求和
```
cat txt |awk '{a+=$1}END{print a}'
```

列求平均值
```
cat txt |awk '{a+=$1}END{print a/NR}'
```

列求最大值
```
cat txt |awk 'BEGIN{a=0}{if ($1>a) a=$1 fi}END{print a}'
```
#设定一个变量开始为 0，遇到比该数大的值，就赋值给该变量，直到结束

求最小值
```
cat txt |awk 'BEGIN{a=11111}{if ($1<a) a=$1 fi}END{print a}'
```

判断$1 是否整除(awk 中定义变量引用时不能带 $)
```
cat txt |awk '{ i=$1%10;if ( i == 0 ) {print i}}'
```

##################################

sed 用法

-n 输出由编辑指令控制(取消默认的输出,必须与编辑指令一起配合)
-i 直接对文件操作
-e 多重编辑

p 打印
d 删除
s 替换
g 配合 s 全部替换
i 行前插入
a 行后插入
r 读
y 转换
q 退出

\(..\) 保存..作为标签 1(\1)
& 代表查找的串内容
* 前导符
.* 匹配任意多个字符

模式空间(两行两行处理)

N 将第二行追加到第一行 将换行符\n 替换空极为一行
n 将第二行覆盖到第一行
h 把模式空间里的行拷贝到暂存空间
H 把模式空间里的行追加到暂存空间
g 用暂存空间的内容替换模式空间的行
G 把暂存空间的内容追加到模式空间的行后
x 将暂存空间的内容于模式空间里的当前行互换
！ 对所选行以外的所有行应用命令。
注意：暂存空间里默认存储一个空行。

sed 命令替换并打印出替换行(-i 改变文本)
sed -n -e "{s/文本(正则表达式)/替换的文本/p}"

打印并删除正则表达式的那部分内容
sed -n -e "{s/^ *[0-9]*//p}"

删除含关键的一行
sed -i "/^$f/d" a

直接对文本替换
sed -i "s/=/:/" c

找到 pearls 开头在行尾加 jcui
sed -i "/^pearls/s/$/jcui/" ab.txt

标签(保存\(mar\)作为标签 1)
sed -n 's/\(mar\)got/\1ianne/p' 文件
echo "margot"|sed -n 's/\(mar\).*\(t\)/\1\2/p'
sed -e 's/^\([a-zA-Z]\+\) \([a-zA-Z]\+\)\(.*\)/\2 \1\3/g' file

在以[0-9][0-9]结尾的行后加 5
sed 's/[0-9][0-9]$/&5' 文件

打印从第 5 行到以 no 开头行之间的所有行
sed -n '5,/^no/p' 文件

修改 west 和 east 之间的所有行，在结尾处加*VACA*
sed '/west/,/east/s/$/*VACA*/' 文件

多重编辑(先删除 1-3 行，在将 1 替换成 2)
sed -e '1,3d' -e 's/1/2/' 文件

找到含 suan 的行，在后面加上读入的文件内容
sed '/suan/r 读入文件' 文件

找到含 no 的行，写入到指定文件中
sed -n '/no/w 写入文件' 文件

取出第一组数字，并且忽略掉开头的 0

```
sed    's/[^1-9]*\([0-9]\+\).*/\1/'
```

打印 1 和 3 之间的行
```
sed '/1/,/3/p' file
```

取出指定行
```
sed -n '1p'  文件
```

在第 5 行之前插入行
```
sed '5i\aaa' file
```

在第 5 行之后抽入行
```
sed '5a\aaa' file
```

在匹配行前插入一行
```
echo a|sed -e '/a/i\b'
```

在匹配行后插入一行
```
echo a|sed -e '/a/a\b'
echo a|sed 's/a/&\nb/g'
```

引用外部变量
```
sed -n "$a',10p'
sed -n ""$a",10p"
```

取用户家目录(匹配不为/的字符和匹配:到结尾的字符全部删除)
```
sed -n '{s/^[^\/]*//;s/\:.*//;p}' /etc/passwd
```

将换行符转换为换行
```
echo abcd\\nabcde |sed 's/\\n/@/g' |tr '@' '\n'
```

删除掉@后面所有字符，和空行
```
sed -e 's/@.*//g' -e '/^$/d'
```

将几行合并成一行
```
echo $(cat file)|sed 's/ //g'
```

取一列最大值
```
cat tmp|awk '{print $1}'|sort -n|sed -n '$p'
```

查找属主权限为 7 的文件
```
ls -l|sed -n '/^.rwx.*/p'
```

在 a 前面加#号
```
echo a|sed -e 's/a/#&/g'
```

```
####################################
```

图形 dialog

多选界面[方括号]

dialog --title "Check me" --checklist "Pick Numbers" 15 25 3 1 "one" "off" 2 "two" "on" 3 "three" "off" 2>tmp

多选界面(圆括号)
dialog --title "title" --radiolist "checklist" 20 60 14 tag1 "item1" on tag2 "item2" off 2>tmp

单选界面
dialog --title "title" --menu "MENU" 20 60 14 tag1 "item1" tag2 "item2" 2>tmp

进度条
dialog --title "Installation" --backtitle "Star Linux" --gauge "Linux Kernel" 10 60 50

选择 yes/no
dialog --title "标题" --backtitle "Dialog" --yesno "说明" 20 60

公告
dialog --title "公告标题" --backtitle "Dialog" --msgbox "内容" 20 60

显示讯息后立即离开
dialog --title "hey" --backtitle "Dialog" --infobox "Is everything okay?" 10 60

输入对话框
dialog --title "hey" --backtitle "Dialog" --inputbox "Is okay?" 10 60 "yes"

显示文档内容
dialog --title "Array 30" --backtitle "All " --textbox /root/txt 20 75

多条输入对话框
dialog --title "Add a user" --form "Add a user" 12 40 4 "Username:" 1 1 "" 1 15 15 0 "Full name:" 2 1 "" 2 15 15 0 2>tmp

星号显示输入--insecure
dialog --title "Password" --insecure --passwordbox "请输入密码" 10 35

选择日期
dialog --stdout --title "日历" --calendar "请选择" 0 0 9 1 2010

#####################################

脚本头 # sh 为当前系统默认 shell,可指定为 bash 等
#!/bin/sh

临时文件定义
errtmp=/tmp/$$`date +%s%N`.errtmp
outtmp=/tmp/$$`date +%s%N`.outtmp
true > $outtmp

随机数
$RANDOM

进程号

$$

纳秒
date +%N

在当前 bash 环境下读取并执行 FileName 中的命令
source file-name　　#　等同　　. FileName

间隔睡眠 5 秒
sleep 5

在接收到信号后将要采取的行动
trap

当前目录
$PWD

之前一个目录的路径
$OLDPWD

返回上一个目录路径
cd -

重复打印
yes

自动回答 y 或者其他
yes |rm -i *

查看目录所有文件夹
ls -p /home

查看匹配完整路径
ls -d /home/

命令替换 xargs
#将前面的内容，作为后面命令的参数
find / -perm +7000 | xargs ls -l

不换行执行下一句话
echo -n aa;echo bb

使转义生效
echo -e "s\tss\n\n\n"

取字符串中字元
echo $a | cut -c2-6

排列组合(括号内一个元素分别和其他括号内元素组合)
echo {a,b,c}{a,b,c}{a,b,c}

返回目录名
```
pwd | awk -F/ '{ print "目录名:" ,$2 }'
```

替换上一条命令中的(foo)一个短语(bar)
```
^foo^bar^
!!:s/foo/bar/
!!:gs/foo/bar
```

##################################

shell 例子

判断参数是否为空-空退出并打印 null
```
#!/bin/sh
echo $1
name=${1:?"null"}
echo $name
```

##################################

循环数组
```
for ((i=0;i<${#o[*]};i++))
do
echo ${o[$i]}
done
```

##################################

判断路径

```
if [ -d /root/Desktop/text/123 ];then
        echo "找到了 123"
             if [ -d /root/Desktop/text ]
             then echo "找到了 text"
             else echo "没找到 text"
             fi
else echo "没找到 123 文件夹"
fi
```

##################################

匹配替换密码

```
#!/bin/sh
cat mailpassword > temp.txt
sed -i "s/:/ = /" temp.txt
w=`awk -F " = " '{print $1}' temp.txt`
for d in $w
do
             grep -w $d svnpassword >/dev/null
             if [ $? -eq 0 ]
             then
```

```
        sed -i "/^$d/d" svnpassword
        grep "^$d" temp.txt >> svnpassword
#替换到转义就出问题
#sed -i "/^$d/c    $(grep "^$d" temp.txt)" svnpassword
            fi
done
rm temp.txt

###################################
```

多行合并

```
将两行并为一行(去掉换行符)
sed '{N;s/\n//}' file

将 4 行合并为一行(可扩展)
awk '{if (NR%4==0){print $0} else {printf"%s ",$0}}' file
将所有行合并
awk '{printf"%s ",$0}'

###################################
```

shift 用法

```
./cs.sh 1 2 3
#!/bin/sh
until [ $# -eq 0 ]
do
echo "第一个参数为: $1  参数个数为: $#"
#shift  命令执行前变量  $1  的值在 shift 命令执行后不可用
shift
done

###################################
```

给脚本加参数 getopts

```
#!/bin/sh
while getopts :ab: name
do
case $name in
a)    aflag=1
            ;;
b)    bflag=1
bval=$OPTARG
            ;;
\?) echo "USAGE:`basename $0` [-a] [-b value]"
exit    1
            ;;
esac
done
if [ ! -z $aflag ] ; then
```

```
echo      "option -a specified"
echo    "$aflag"
echo "$OPTIND"
fi
if [ ! -z $bflag ] ; then
echo    "option -b specified"
echo    "$bflag"
echo    "$bval"
echo    "$OPTIND"
fi
echo "here    $OPTIND"
shift $(($OPTIND -1))
echo "$OPTIND"
echo " `shift $(($OPTIND -1))`    "

#####################################
```

判断脚本参数是否正确

```
./test.sh    -p 123 -P 3306 -h 127.0.0.1 -u root
#!/bin/sh
if [ $# -ne 8 ];then
echo "USAGE: $0 -u user -p passwd -P port -h host"
exit 1
fi

while getopts :u:p:P:h: name
do
case $name in
u)
mysql_user=$OPTARG
;;
p)
mysql_passwd=$OPTARG
;;
P)
mysql_port=$OPTARG
;;
h)
mysql_host=$OPTARG
;;
*)
echo "USAGE: $0 -u user -p passwd -P port -h host"
exit 1
;;
esac
done

if [ -z $mysql_user ] || [ -z $mysql_passwd ] || [ -z $mysql_port ] || [ -z $mysql_host ]
then
echo "USAGE: $0 -u user -p passwd -P port -h host"
exit 1
fi
```

```
echo $mysql_user $mysql_passwd $mysql_port    $mysql_host
#结果  root 123 3306 127.0.0.1

###################################

        dialog 取值接值退出

outtmp=/tmp/outtmp
dialog --no-shadow --backtitle "雪松制作" --title "请选择" --ok-label "提交" --cancel-label "取消"
--menu "请选择" 10 60 0 \
1               "[目录]" \
q               "[退出]" \
2> $outtmp
#判断提交取消
if [ $? == "1" ]
then
        break
        #continue
fi
#只判断退出
pt=`awk '{print}' $outtmp`
if [ $pt == q ]
then
break
fi
#判断操作
case $pt in
1)
#操作
;;
q)
exit #break
;;
esac

###################################

        dialog 输入对话框加判断

while :
do
outtmp=/tmp/outtmp
dialog --title "雪松制作" --form "请输入" 20 60 10    "用户:" 1    2 "" 1    15    35    0 "ID:" 2    2 "" 2
15    35    0    "" 6    2 "" 6    0    0    0 "输入格式如下" 7    2 "" 7    0    0    0 "说明" 8    2 "" 8    0    0
0 2> $outtmp
yh=`sed -n '1p' $outtmp`
id=`sed -n '2p' $outtmp`
if echo $yh | grep -e '^[1-9][0-9]\{0,2\}' > /dev/null 2>&1
then
        if echo $id | grep -e '^[1-9][0-9]\{0,12\}' > /dev/null 2>&1
        then
```

```
            break
        else
            echo "您输入的不合法，请从新输入！"
        fi
else
        echo "您输入的不合法，请从新输入！"
fi
done

###################################
```

打印表格

```
#!/bin/sh
clear
awk 'BEGIN{
print "+--------------------+--------------------+";
printf "|%-20s|%-20s|\n","Name","Number";
print "+--------------------+--------------------+";
}'
a=`grep "^[A-Z]" a.txt |sort +1 -n |awk '{print $1":"$2}'`
#cat a.txt |sort +1 -n |while read list
for list in $a
do
name=`echo $list |awk -F: '{print $1}'`
number=`echo $list |awk -F: '{print $2}'`
awk 'BEGIN{printf "|%-20s|%-20s|\n","'"$name"'","'"$number"'";
print "+--------------------+--------------------+";
}'
done
awk 'BEGIN{
print "                        *** The End ***                        "
print "                                                                "
}'

###################################
```

判断日期是否合法

```
#!/bin/sh
while read a
do
   if echo $a | grep -q "-" && date -d $a +%Y%m%d > /dev/null 2>&1
   then
        if echo $a | grep -e '^[0-9]\{4\}-[01][0-9]-[0-3][0-9]$'
        then
            break
        else
            echo "您输入的日期不合法，请从新输入！"
        fi
   else
      echo "您输入的日期不合法，请从新输入！"
```

```
   fi
done
echo "日期为$a"

###################################

        打印日期段所有日期

#!/bin/bash
qsrq=20010101
jsrq=20010227
n=0
>tmp
while :;do
current=$(date +%Y%m%d -d"$n day $qsrq")
if [[ $current == $jsrq ]];then
        echo $current >>tmp;break
else
        echo $current >>tmp
        ((n++))
fi
done
rq=`awk 'NR==1{print}' tmp`

###################################

        打印提示
cat <<EOF
#内容
EOF

        登陆远程执行命令（特殊符号需要\转义）
ssh root@ip << EOF
#执行命令
EOF

        登陆 mysql 执行命令
mysql -uroot -ppasswd -S mysql.sock --default-character-set=gbk -A<<EOF>outtmp
#mysql 命令
EOF

###################################

        数学计算的小算法

#!/bin/sh
A=1
B=1
while [ $A -le 10 ]
do
SUM=`expr $A \* $B`
echo "$SUM"
```

```
if [ $A = 10 ]
then
B=`expr $B + 1`
A=1
fi
A=`expr $A + 1`
done
```

#####################################

竖行转横行(三种)

```
cat file|tr '\n' ' '
echo $(cat file)
#!/bin/sh
for i in `cat file`
do
        a=${a}" "${i}
done
echo $a
```

#####################################

取用户的根目录

```
#! /bin/bash
#showhome.sh
IFS=:
while read name pass uid gid gecos home shell
do
echo $home
done
```

执行./showhome.sh < /etc/passwd

#####################################

把汉字转成 encode 格式：

```
echo  论坛  | tr -d "\n" | xxd -i | sed -e "s/ 0x/%/g" | tr -d " ,\n"
%c2%db%cc%b3
```

大写的：

```
echo  论坛  | tr -d "\n" | xxd -i | sed -e "s/ 0x/%/g" | tr -d " ,\n" | tr "[a-f]" "[A-F]"
%C2%DB%CC%B3
```

#####################################

把带有大写字母的文件名改为全部小写

```
#!/bin/bash
for f in *;do
mv $f `echo $f |tr "[A-Z]" "[a-z]"`
```

```
done
###################################
```

查找连续多行，在不连续的行前插入

```
#/bin/bash

lastrow=null
i=0
cat incl|while read line
do
i=`expr $i + 1`
if echo "$lastrow" | grep "#include <[A-Z].h>"
then
        if echo "$line" | grep -v    "#include <[A-Z].h>"
        then
                sed -i "$i'i\\/\/All header files are include' incl
                i=`expr $i + 1`
        fi
fi
lastrow="$line"
done

###################################
```

==========

---

9 服务配置

---

samba

修改配置文

/etc/samba/smb.conf

```
[global]
workgroup = WORKGROUP
server string = sambaserver
security=share
hosts allow=网络主机
[file]
comment=home directories
path=/etc/samba/xuesong
public=yes
writable=yes
browseable=yes
```

重启

```
service smb restart
```

linux 访问 windows 共享

```
smbclient -L //ip 地址或计算机名
```

##################################

DNS 服务器

DNS 主配置文件，moon.com 域名，zone.moon.com 域的配置文件
vi /etc/named.conf
```
zone "moon.com" IN {
        type master;
        file "zone.moon.com";
          allow-update { none; };
};
```
测试主配置文件
named-checkconf
新建域的配置文件 zone.moon.com
vi /var/named/chroot/var/named/zone.moon.com
```
$TTL 86400
@               IN      SOA xs.      quanzhou722.163.com. (
                                                                20102014
                                                                3H
                                                                15M
                                                                1W
                                                                1D
                                                               )
                IN    NS    xs.
                IN    MX 5 mail.
www             IN    A      192.168.1.209
ftp             IN    A      192.168.1.209
mail            IN    A      192.168.1.209
```
测试 moon.com 域配置文件
named-checkzone moon.com   /var/named/chroot/var/named/zone.moon.com
建立域配置文件的软连接
ln -s /var/named/chroot/var/named/zone.moon.com   /var/named/zone.moon.com
重启 named 服务
service named   restart
设置 DNS
vi /etc/resolv.conf
测试解析是否成功
host www.moon.com
nslookup www.moon.com

##################################

ftp 配置

查看 vsftp 是否安装
rpm   -qa | grep vsftp
主配置文件
vi /etc/vsftpd/vsftpd.conf
启动 VSFTPD 服务
service vsftpd restart
查看 21 端口是否开放

netstat -atln |grep 21
1、匿名用户是否允许访问
anonymous_enable=YES
anon_mkdir_write_enable=YES
2、本地用户管理
local_enable=YES
chroot_list_enable=YES
chroot_list_file=/etc/vsftpd.chroot_list
将用户添加到 vsftpd.chroot_list 文件中允许访问
cat /etc/passwd |grep ":5[0-9][0-9]:" | awk -F : '{print $1}' > /etc/vsftpd.chroot_list
3、虚拟用户管理
（1）要用到文件数据库 db4
rpm -qa |grep db4
（2）将帐户放入文本：（奇数行）用户名 （偶数行）密码
vi zhanghu.dat
（3）生成帐户的文件数据库
db_load -T -t hash -f zhanghu.dat /etc/vsftpd/vsftpd_zhanghu.db
（4）修改文件数据库访问权限
chmod 600 /etc/vsftpd/vsftpd_zhanghu.db
（5）手工建立虚拟用户所需的 PAM 配置文件
vi /etc/pam.d/vsftpd.zhanghu
auth required /lib/security/pam_userdb.so db=/etc/vsftpd/vsftpd_zhanghu
account required /lib/security/pam_userdb.so db=/etc/vsftpd/vsftpd_zhanghu
（6）添加一个虚拟用户的入口帐户
useradd -d /var/ZhanDian -s /sbin/nologin XuNiRuKou
（7）在主配置文件中添加虚拟用户的配置内容
虚拟用户的新添加配置
chroot_local_user=YES
guest_enable=YES
guest_username=XuNiRuKou
pam_service_name=vsftpd.zhanghu
（8）在 vsftpd.conf 配置文件中指定用户配置文件的目录
user_config_dir=/etc/vsftpd/vsftpd_zhandian
（9）建立配置文件中指定用户配置文件的目录
mkdir /etc/vsftpd/vsftpd_zhandian
（10）因为入口帐户是一个本地帐户，所以要指定虚拟用户具有本地帐户的权限，才能读写 自己空间中的文件
虚拟用户就可以就拥有写权限
virtual_use_local_privs=YES
（11）
touch /etc/vsftpd/vsftpd_zhandian/xuesong
echo "local_root=/var/ZhanDian/xuesong" >> /etc/vsftpd/vsftpd_zhandian/xuesong
（12）
mkdir /var/ZhanDian/xuesong
（13）
chown -R XuNiRuKou:XuNiRuKou /var/ZhanDian/
（14）重启服务
/etc/init.d/vsftpd restart
（15）客户端虚拟账号测试

建立用户脚本。最好在加上判断用户名是否存在

```
#!/bin/sh
echo  请输入用户名
read y
echo $y >> zhanghu.dat
echo  请输入密码
read m
echo $m >> zhanghu.dat
db_load -T    -t hash    -f zhanghu.dat /etc/vsftpd/vsftpd_zhanghu.db
chmod 600    /etc/vsftpd/vsftpd_zhanghu.db
touch /etc/vsftpd/vsftpd_zhandian/$y
echo "local_root=/var/ZhanDian/$y" >> /etc/vsftpd/vsftpd_zhandian/$y
mkdir /var/ZhanDian/$y
chown -R XuNiRuKou:XuNiRuKou /var/ZhanDian/$y
/etc/init.d/vsftpd restart
```

####################################

httpd 安装

```
tar zxvf httpd-2.2.12.tar.gz
cd httpd-2.2.12
./configure    --prefix=/usr/local/httpd    \
--enable-rewrite
make    &&      make    install
ls    /usr/local/
groupadd www
useradd -g www -M    -s /sbin/nologin    www
cd /usr/local/httpd/
vi conf/httpd.conf
```

服务器开启在软件安装目录的 bin 中

```
./apachectl start
```

网页放在的 htdocs 中

开机启动

```
vi /etc/rc.d/rc.local
/usr/local/httpd/bin/apachectl start
```

如果启动提示

httpd: Could not reliably determine the server fully qualified domain name, using 127.0.0.1 for ServerName

修改/etc/hosts 计算机名

在 httpd.conf 中开启

```
ServerName 127.0.0.1:80
```

编码设置

```
AddDefaultCharset UTF-8
AddDefaultCharset GB2312
```

####################################

mysql 源码安装

```
groupadd mysql
```

useradd mysql -g mysql -M -s /bin/false
tar zxvf mysql-5.0.22.tar.gz
cd mysql-5.0.22
./configure --prefix=/usr/local/mysql \
--with-client-ldflags=-all-static \
--with-mysqld-ldflags=-all-static \
--with-mysqld-user=mysql \
--with-extra-charsets=all \
--with-unix-socket-path=/var/tmp/mysql.sock
make && make install
生成 mysql 用户数据库和表文件，在安装包中输入。
scripts/mysql_install_db --user=mysql
vi ~/.bashrc
export PATH="$PATH: /usr/local/mysql/bin"
配置文件,有 large,medium,small 三个，根据机器性能选择
cp support-files/my-medium.cnf /etc/my.cnf
cp support-files/mysql.server /etc/init.d/mysqld
chmod 700 /etc/init.d/mysqld
cd /usr/local
chmod 750 mysql -R
chgrp mysql mysql -R
chown mysql mysql/var -R
cp /usr/local/mysql/libexec/mysqld mysqld.old
ln -s /usr/local/mysql/bin/mysql /sbin/mysql
ln -s /usr/local/mysql/bin/mysqladmin /sbin/mysqladmin
ln -s -f /usr/local/mysql/bin/mysqld_safe /etc/rc.d/rc3.d/S15mysql5
ln -s -f /usr/local/mysql/bin/mysqld_safe /etc/rc.d/rc0.d/K15mysql5

##################################

mysql rpm 安装

到 http://www.mysql.com/downloads/mysql 下载 mysql RPM（service 和 client），
MySQL-server-community-5.1.51-1.rhel5.i386.rpm 安装服务端
MySQL-client-community-5.1.51-1.rhel5.i386.rpm 安装客户端
MySQL-devel-community-5.1.51-1.rhel5.i386.rpm 用于 php 的 mysql 安装
rpm –ivh MySQL-server-community-5.1.51-1.rhel5.i386.rpm
rpm –ivh MySQL-client-community-5.1.51-1.rhel5.i386.rpm
rpm –ivh MySQL-devel-community-5.1.51-1.rhel5.i386.rpm
/etc/init.d/mysqld start
mysqladmin -u root password 'xuesong'
mysql -u root -S /var/lib/mysql/mysql.sock -p
数据目录/var/lib/mysql
chkconfig --list |grep mysqld
chkconfig --level 3 mysqld on
配置文件(日志\pid)
/etc/my.cnf
ll /usr/bin/mysql*

##################################

mysql 常用命令

启动 mysql 服务
./mysql/bin/mysqld_safe --user=mysql &

登录 mysql 的命令
格式： mysql -h 主机地址 -u 用户名 -p 用户密码
mysql -h110.110.110.110 -P3306 -uroot -p
mysql -uroot -p -S /data1/mysql5/data/mysql.sock -A --default-character-set=GBK

刷新
flush privileges;

显示所有数据库
show databases;

打开数据库
use dbname;

显示选中数据库中所有的表
show tables;

查看表结构
desc tables;

删除数据库
drop database name;

删除表
drop table name;

创建数据库
create database name;
mysqladmin -h myhost -u root -p create dbname

创建表
create table ka(ka_id varchar(6),qianshu int);

插入字段
alter table name add column accountid(列名) int(11) NOT NULL(字段不为空);

不登陆 mysql 插入字段
mysql -uroot -p -S mysql.sock -e "use db;alter table gift add column accountid int(11) NOT NULL;flush privileges;"

查询
select 列名称 from 表名称;
select host,user,password from user;

备份数据库
mysqldump -h host -u root -p dbname >dbname_backup.sql

mysqlhotcopy 只能备份 MyISAM 引擎
/bin/mysqlhotcopy -u root -p

还原数据库
mysql -h host -u root -p dbname < dbname_backup.sql

登陆 mysql 后还原 sql 文件
source 路径.sql

赋予查询权限(没有用户，直接创建)
grant select on db.* to log@"%" Identified by "123456";

赋予所有权限(没有用户，直接创建)
grant all privileges on db.* to log@"IP" identified by '123456' with grant option;

更改密码
update user set password=password('新密码') where user='root'
mysqladmin -u root password 'xuesong'

mysql 查看所有用户的语句
select user();

删除用户
delete from user where user='sss' and host='localhost' ;

查看主从
show slave status\G;

查看主从复制进程
show processlist;

添加索引 解决停 mysql 速度慢
ALTER TABLE player ADD INDEX weekcredit_faction_index (weekcredit, faction);

创建主从复制用户
grant replication slave on *.* to '用户'@'%' identified by '密码';

备份表
mysqldump -u root -p -S mysql.sock --default-character-set=utf8 --hex-blob db table1 table1 > /data/db.sql

查看系统的字符集和排序方式的设定
SHOW VARIABLES LIKE 'character_set_%';

查看 mysql 所有参数
show variables;

删除空用户
delete from user where user='';

创建表指定存储引擎的类型:

CREATE TABLE innodb (id int, title char(20)) ENGINE = INNODB
CREATE TABLE myisam (id int, title char(20)) ENGINE = MyISAM

改变现有的表使用的存储引擎:
ALTER TABLE mytable ENGINE = MyISAM

查询表引擎
SHOW TABLE STATUS from　数据库库名　where Name='表名';
SHOW TABLE STATUS from db where Name='innodb';

查看 mysql 变量
SHOW VARIABLES;

查询用户权限
select host,user,password from user;

查看表的引擎状态
show table status

####################################

gd 安装

分别安装组件 libxml、zlib、freetype、libpng、jpegsrc、gd
libxml2-2.6.24.tar.bz2
zlib-1.2.3.tar.gz
freetype-2.2.1.tar.gz
libpng-1.2.10.tar.gz
jpegsrc.v6b.tar.gz
gd-2.0.33.tar.gz

libxml 安装
tar xzvf libxml2-2.6.24.tar.bz2
tar xvf libxml2-2.6.24.tar
cd libxml2-2.6.24
./configure
make
make　install

zlib 安装
tar xzvf zlib-1.2.3.tar.gz
cd zlib-1.2.3
./configure
make
make　install

freetype 安装
tar xzvf freetype-2.2.1.tar.gz
cd freetype-2.2.1
./configure --prefix=/usr/local/php/modules/freetype
make
make install

libpng 安装
tar xzvf libpng-1.2.10.tar.gz
cd libpng-1.2.10
./configure --prefix=/usr/local/php/modules/libpng

jpeg 安装
mkdir /usr/local/php/modules/jpeg6
mkdir /usr/local/php/modules/jpeg6/bin
mkdir /usr/local/php/modules/jpeg6/lib
mkdir /usr/local/php/modules/jpeg6/include
mkdir /usr/local/php/modules/jpeg6/man
mkdir /usr/local/php/modules/jpeg6/man/man1
一句创建
mkdir -pv /usr/local/jpeg/{,bin,lib,include,man/man1,man1}
tar zxvf jpegsrc.v6b.tar.gz
cd jpeg-6b/
./configure --prefix=/usr/local/php/modules/jpeg6 \
--enable-shared \
--enable-static
make
make install

gd 安装
tar zxvf gd-2.0.33.tar.gz
cd gd-2.0.33
./configure --prefix=/usr/local/gd2 \
--with-jpeg=/usr/local/php/modules/jpeg6 \
--with-png=/usr/local/php/modules/libpng \
--with-zlib \
--with-freetype=/usr/local/php/modules/freetype
修改 Makefile 文件 231 行，重复的第二个 freetype 改为 libpng
make
make install

php 安装
tar xvf php-5.1.6.tar
cd php-5.1.bz2.6
./configure --prefix=/usr/local/php \
--sysconfdir=/etc \
--with-apxs2=/usr/local/httpd/bin/apxs \
--with-config-file-path=/usr/local/php/lib \
--with-libxml-dir=/usr/local/libxml2 \
--with-zlib \
--with-jpeg-dir=/usr/local/php/modules/jpeg6 \
--with-freetype-dir=/usr/local/php/modules/freetype \
--with-gd=/usr/local/gd2 \
--with-openssl \
--with-regex=php \
--with-gnu-ld \
--with-tsrm-pthreads \
--with-bz2 \

```
--with-gettext \
--with-db4=/usr \
--enable-ftp \
--enable-bcmath \
--enable-dba \
--enable-sockets \
--enable-soap \
--enable-zend-multibyte \
--enable-mbstring \
--with-curlwrappers \
--with-curl=/usr/local/curl \
--with-pdo-mysql=/usr/local/mysql \
--with-mysql=/usr/local/mysql \
--with-mysqli=/usr/local/mysql/bin/mysql_config
make
make install
cp php.ini-recommended /usr/local/php/lib/php.ini
vi /usr/local/php/lib/php.ini
```
在 PHP.IN 添加
```
[eaccelerator]
extension=/usr/local/php/lib/php/extensions/no-debug-non-zts-20060613/eaccelerator.so
eaccelerator.shm_size="32"
eaccelerator.cache_dir="/usr/local/cache/eaccelerator"
eaccelerator.enable="1"
eaccelerator.optimizer="1"
eaccelerator.check_mtime="1"
eaccelerator.debug="0"
eaccelerator.filter=""
eaccelerator.shm_max="0"
eaccelerator.shm_ttl="0"
eaccelerator.shm_prune_period="0"
eaccelerator.shm_only="0"
eaccelerator.compress="1"
eaccelerator.compress_level="9"
```
完成后在/usr/local/httpd/conf/httpd.conf 文件上添加如下一行：
```
AddType application/x-httpd-php .php .php5
ln -s /usr/local/php/bin/php /sbin/php
vi index.php
<? php
phpinfo();
?>
```
重启 httpd
```
/usr/local/httpd/bin/apachectl restart
```

最后一步重新启动 apache 报如下错误:
httpd: Syntax error on line 53 of /usr/local/apache/conf/httpd.conf: Cannot load /usr/local/apache/modules/libphp5.so into server: /usr/local/apache/modules/libphp5.so: cannot restore segment prot after reloc: Permission denied

不关闭 SELINUX 的方法:
```
# setenforce 0
# chcon -c -v -R -u system_u -r object_r -t textrel_shlib_t /usr/local/apache/modules/libphp5.so
# service httpd restart
```

setenforce 1

###################################

LINUX 集群

LVS-- ipvsadm 命令参考
ipvsadm 的用法和格式如下： ipvsadm -A|E -t|u|f virutal-service-address:port [-s scheduler] [-p [timeout]] [-M netmask]
ipvsadm -D -t|u|f virtual-service-address
ipvsadm -C ipvsadm -R ipvsadm -S [-n]
ipvsadm -a|e -t|u|f service-address:port -r real-server-address:port [-g|i|m] [-w weight] ipvsadm -d -t|u|f service-address -r server-address
ipvsadm -L|l [options]
ipvsadm -Z [-t|u|f service-address]
ipvsadm --set tcp tcpfin udp ipvsadm --start-daemon state [--mcast-interface interface] ipvsadm --stop-daemon
ipvsadm -h

命令选项解释：有两种命令选项格式，长的和短的，具有相同的意思。在实际使用时，两种都可以。
-A --add-service 在内核的虚拟服务器表中添加一条新的虚拟服务器记录。也就是增加一台新的虚拟服务器。
-E --edit-service 编辑内核虚拟服务器表中的一条虚拟服务器记录。
-D --delete-service 删除内核虚拟服务器表中的一条虚拟服务器记录。
-C --clear 清除内核虚拟服务器表中的所有记录。
-R --restore 恢复虚拟服务器规则
-S --save 保存虚拟服务器规则，输出为-R 选项可读的格式
-a --add-server 在内核虚拟服务器表的一条记录里添加一条新的真实服务器记录。也就是在一个虚拟服务器中增加一台新的真实服务器
-e --edit-server 编辑一条虚拟服务器记录中的某条真实服务器记录
-d --delete-server 删除一条虚拟服务器记录中的某条真实服务器记录
-L|-l --list 显示内核虚拟服务器表

其他的选项:-t --tcp-service service-address 说明虚拟服务器提供的是 tcp 的服务[vip:port] or [real-server-ip:port]
-u --udp-service service-address 说明虚拟服务器提供的是 udp 的服务[vip:port] or [real-server-ip:port]
-f --fwmark-service fwmark 说明是经过 iptables 标记过的服务类型。
-s --scheduler scheduler 使用的调度算法，有这样几个选项 rr|wrr|lc|wlc|lblc|lblcr|dh|sh|sed|nq,-p --persistent [timeout] 持久稳固的服务。这个选项的意思是来自同一个客户的多次请求，将被同一台真实的服务器处理。timeout 的默认值为 300 秒。
-M --netmask netmask persistent granularity mask
-r --real-server server-address 真实的服务器[Real-Server:port]
-g --gatewaying 指定 LVS 的工作模式为直接路由模式（也是 LVS 默认的模式）
-i --ipip 指定 LVS 的工作模式为隧道模式
-m --masquerading 指定 LVS 的工作模式为 NAT 模式
-w --weight weight 真实服务器的权值
--mcast-interface interface 指定组播的同步接口

-c --connection 显示 LVS 目前的连接 如：ipvsadm -L -c

##################################

VS-NAT 模式

内容服务器的容量 10～20
使用保留地址的 LAN
双向重写

安装 ipvsadm 软件
内核做个连接
ln -s /usr/src/kernels/2.6.18-53.e15-i686/ /usr/src/linux
安装软件
查看 ipvsadm 版本
ipvsadm --version

建立脚本/etc/rc.d/lvs
echo "1" > /proc/sys/net/ipv4/ip_forward
#清除 ipvsadm 一系列规则
ipvsadm -C
ipvsadm -A -t 外网 IP:端口 -s rr
ipvsadm -a -t 外网 IP:端口 -r 内网 1 服务器 IP:端口 -m -w 1
ipvsadm -a -t 外网 IP:端口 -r 内网 2 服务器 IP:端口 -m -w 1

修改内网服务器 ip 和网关，开启 httpd，建立不同页面测试

缺点不能排除宕机的 web 服务器，可使用 LVS+Keepalived 组合
##################################

heartbeat 心跳检测（双机热备）

安装 heartbeat 的 rpm 包
需要先装 heartbeat-pils 和 heartbeat-stonith 两个包
主节点的配置
安装完解压配置文件
rpm -q heartbeat -d
cd /etc/ha.d
cp /usr/share/doc/heartbeat-2.1.3/ha.cf .
cp /usr/share/doc/heartbeat-2.1.3/authkeys .
cp /usr/share/doc/heartbeat-2.1.3/haresources .
查看计算机名，记录下来
uname -n
vi ha.cf
打开日志记录文件
logfile /var/log/ha-log
打开心跳检测
keepalive 2
死亡时间
deadtime 30

warntime 10
initdead 120
用什么协议端口（udp）
udpport 694
使用哪个网卡作为心跳检测
bcast eth1
备用的 IP
ucast eth1 192.168.1.20
主节点服务器
node 主节点计算机名
node 另一个计算机名
ping 外网测试
ping 一个外网的 IP
配置验证模式
vi authkeys
打开
auth1
1 crc
配置虚拟服务
vi haresources
加上一条名称和 IP
计算机名 IP httpd
sev1.example.com 222.90.88.88 httpd
hosts 文件要配置上两个机器的机器名和 IP
vi /etc/hosts
222.90.88.10 sev1.example.com sev2
192.168.1.10 sev1
222.90.88.20 sev1.example.com sev2
192.168.1.20 sev1
chmod 600 authkeys
service heartbeat start
在配置备用节点
直接拷贝主节点的文件修改
安装完进入安装目录
cd /etc/ha.d
用 sftp 登陆主机
sftp 222.90.88.10
cd /etc/ha.d
get ha.cf
get authkeys
get hare*
bye
只需要修改 ha.cf 文件
vi ha.cf
ucast eth1 192.168.1.10
启动备用节点
service heartbeat start

测试
主节点和负节点建立不同 index 页面

主节点虚拟出外网 ip eth0:0
停掉主节点任务
重新访问
启动后在访问

##################################

DR 直接路由模式

同一网段的 LAN
内容服务器的容量 20 以上

降低 LVF 的负载
VIP 服务器的配置脚本
vi /etc/init.d/vip.sh
ifconfig eth0:0 222.90.88.88 broadcast 222.90.88.88 netmask 255.255.255.255 up
route add -host 222.90.88.88 dev eth0:0
ipvsadm -C
ipvsadm -A -t 222.90.88.88:80 -s rr
ipvsadm -a -t 222.90.88.88:80 -r 222.90.88.10 -g
ipvsadm -a -t 222.90.88.88:80 -r 222.90.88.20 -g
ipvsadm
chmod 777 /etc/init.d/vip.sh
/etc/init.d/vip.sh

服务器配置脚本
vi /etc/init.d/lvs.sh
ifconfig lo:0 222.90.88.88 broadcast 222.90.88.88 netmask 255.255.255.255 up
route add -host 222.90.88.88 dev lo:0
echo "1" > /proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" > /proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" > /proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" > /proc/sys/net/ipv4/conf/all/arp_announce
sysctl -p
chmod 777 /etc/init.d/lvs.sh
/etc/init.d/lvs.sh
第二台服务器
sftp 222.90.88.10
get /etc/init.d/lvs.sh
bye
mv lvs.sh /etc/init.d/
/etc/init.d/lvs.sh

测试两个服务器测试页面写入不同内容
分别启动 httpd
真是 IP 可以分别访问不同服务器，访问虚拟 IP88，则互相伦叫

##################################

IP 隧道模式

WAN（或 LAN 也可）
内容服务器的容量能到 100

vip-server
vi /etc/init.d/vip.sh
ifconfig tunl0 222.90.88.88 broadcast 222.90.88.88 netmask 255.255.255.255 up
route add -host 222.90.88.88 dev tunl0
ipvsadm -C
ipvsadm -A -t 222.90.88.88:80 -s rr
ipvsadm -a -t 222.90.88.88 :80 -r 222.90.88.10 -i
ipvsadm -a -t 222.90.88.88 :80 -r 222.90.88.20 -i
ipvsadm
chmod -R 777 /etc/init.d/vip.sh
/etc/init.d/vip.sh

1web 服务器
ifconfig runl0 222.90.88.88 broadcast 222.90.88.88 netmask 255.255.255.255 up
route add -host 222.90.88.88 dev tunl0
echo "1" > /proc/sys/net/ipv4/conf/tunl0/arp_ignore
echo "2" > /proc/sys/net/ipv4/conf/tunl0/arp_annaunce
echo "1" > /proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" > /proc/sys/net/ipv4/conf/all/arp_annaunce
sysctl -p
chmod 777 /etc/init.d/lvs.sh
/etc/init.d/lvs.sh
2web 服务器
sftp 222.90.88.10
get /etc/init.d/lvs.sh
bye
mv lvs.sh /etc/init.d/
/etc/init.d/lvs.sh
测试两个服务器测试页面写入不同内容
分别启动 httpd
真是 IP 可以分别访问不同服务器，访问虚拟 IP88，则互相伦叫

##################################

sendmail 邮件服务器

配置 DNS 服务器为 mail 服务器 IP 解析
在配置文件中添加
IN　MX 5 mail.
mail　　IN　A　　192.168.1.209
重启服务 service named　restart
测试
ping mail.xuesong.com
nslookup mail.xuesong.com
mail 服务器上设置 DNS 服务器
vi /etc/resolv.conf
安装软件
sendmail-8.13.8-8.el5.i386.rpm

sendmail-cf-8.13.8-8.el5.i386.rpm
sendmail-doc-8.13.8-8.el5.i386.rpm
yum install dovecot
m4
sendmail 配置文件目录：/etc/mail
vi local-host-names
添加 xuesong.com
vi sendmai.mc
去掉前面两个 dnl
dnl TRUST_AUTH_MECH('EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define('confAUTH_MECHANISMS', 'EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
修改 IP
DAEMON_OPTIONS('Port=smtp,Addr=192.168.1.209, Name=MTA')dnl
生成 sendmail 配置文件 sendmail.cf
m4 sendmail.mc > sendmail.cf
查看启动级别
chkconfig --list |grep sas
开启启动级别 3
chkconfig --level 3 saslauthd on
makemap hash access.db < access
vi /etc/dovecot.conf
打开下面一句话
protocols = imap imaps pop3 pop3s
service sendmail restart
service dovecot restart
group mail
useradd -g mail -M -s /sbin/nologin 用户名

####################################

JDK 安装

rpm -qa|grep jdk
确认 gcj 的版本号
rpm -qa | grep gcj
卸载 gcj
yum -y remove java-1.4.2-gcj-compat
从 SUN 下载 jdk-1_5_0_14-linux-i586-rpm.bin 或 jdk-1_5_0_14-linux-i586.bin
mkdir /usr/java
cd /usr/java
1. jdk-1_5_0_14-linux-i586-rpm.bin 文件安装
chmod 777 jdk-1_5_0_14-linux-i586-rpm.bin
rpm -ivh jdk-1_5_0_14-linux-i586.rpm
2. jdk-1_5_0_14-linux-i586.bin 文件安装
chmod u+x jdk-1_5_0_14-linux-i586.bin
./jdk-1_5_0_14-linux-i586.bin
添加环境变量
vi /etc/profile
export JAVA_HOME=/usr/local/java/jdk1.6.0_23
export

CLASSPATH=.:$JAVA_HOME/jre/lib/rt.jar:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin
reboot　　重启生效

####################################

　　xen 管理

查看虚拟主机列表
xm list
使用文本方式连接虚拟系统
xm console test
启动虚拟主机
xm create test
虚拟机和主机切换
ctrl+]
关闭虚拟主机
xm shutdown 主机名
虚拟机克隆(关闭后克隆)
virt-clone -o test -n test1 -f /data/xen/test1.img
虚拟机增加硬盘
dd if=/dev/zero of=/data/xen/test.img bs=1M count=8
编辑原来配置文件，添加新磁盘文件
vi /etc/xen/vm11
name = "vm11"
uuid = "91b0200b-eb41-46d3-1e11-ee2a87035f24"
maxmem = 128
memory = 128
vcpus = 1
bootloader = "/usr/bin/pygrub"
on_poweroff = "destroy"
on_reboot = "restart"
on_crash = "restart"
vfb = ["type=vnc,vncunused=1,keymap=en-us"]
disk = ["tap:aio:/data/vm11.img,xvda,w", "tap:aio:/data/vm11b.img,xvdb,w"　]
vif = ["mac=00:16:36:75:8a:ea,bridge=virbr0,script=vif-bridge"]
启动 XEN 虚拟机并登陆测试新加的磁盘
增加虚拟机网卡
cat /etc/xen/vm11
name = "vm11"
uuid = "91b0200b-eb41-46d3-1e11-ee2a87035f24"
maxmem = 512
memory = 128
vcpus = 1
bootloader = "/usr/bin/pygrub"
on_poweroff = "destroy"
on_reboot = "restart"
on_crash = "restart"
vfb = ["type=vnc,vncunused=1,keymap=en-us"]
disk = ["tap:aio:/data/vm11.img,xvda,w"]
vif = ["mac=00:16:36:75:8a:ea,bridge=xenbr0,script=vif-bridge",
"mac=00:16:36:75:8a:eb,bridge=xenbr0,script=vif-bridge"]