# Yum安装使用

Yellow Dog Update Mange

1、删除redhat原有的yum源

# rpm -aq | grep yum|xargs rpm -e --nodeps

2、下载新的yum安装包

这里我们使用CentOS的yum源

wget http://centos.ustc.edu.cn/centos/6.1/os/i386/Packages/python-iniparse-0.3.1-2.1.el6.noarch.rpm

wget http://centos.ustc.edu.cn/centos/6.1/os/i386/Packages/yum-metadata-parser-1.1.2-16.el6.i686.rpm

wget http://centos.ustc.edu.cn/centos/6.1/os/i386/Packages/yum-3.2.29-17.el6.centos.noarch.rpm

wget http://centos.ustc.edu.cn/centos/6.1/os/i386/Packages/yum-plugin-fastestmirror-1.1.30-6.el6.noarch.rpm

3、安装：

rpm -ihv yum\*.rpm

4、配置：

cd /etc/yum.repos.d/

下载配置文件：wget http://mirrors.163.com/.help/CentOS6-Base-163.repo

将文件里面的$releasever全部替换为具体的操作系统版本号

5、清理yum缓存

# yum clean all

# yum makecache #将服务器上的软件包信息缓存到本地,以提高搜索安装软件的速度

# yum install vim\*#测试域名是否可用

参考：<http://www.myhack58.com/Article/48/66/2012/32803.htm>

# iptables

iptables使用简介(区分大小写)：

iptables [-t table] command [match] [target|jump]

**table:[filter|nat|mangle]**

filter:信息包过滤：包含INPUT,OUPUT,FORWARD链

nat:转发信息包：PREROUTING,OUTPUT,POSTROUTING链

mangle:高级路由包

command

-A|--append：添加规则附加到链尾

-D|--delete: 从链中删除规则

-L|--list: 列出链规则

-I|--insert: 在链中插入规则,默认为第一条

-F|--flush: 删除链中所有规则

match:

-p|--proto:协议tcp|udp

-s|--source:源地址

-d|--destination:目标地址

-j|--jump:跳转

--sport:源端口

--dport:目标端口

保存规则：

[root@host70-155 init.d]# service iptables save

Saving firewall rules to /etc/sysconfig/iptables: [ OK ]

打开内核IP转发：

vi /etc/sysctl.conf

# Controls IP packet forwarding

net.ipv4.ip\_forward = 1

示例：

iptables -t nat -A PREROUTING -p tcp -d 218.206.70.155 --dport 80 -j DNAT --to-destination 218.206.70.151:80

-t nat 指定要在nat表中添加一条PREROUTING链

-p tcp -d 15.45.23.67 --dport 80 所有要送往202.123.45.67的80端口的数据包

-j DNAT --to-destination 192.168.10.8:80 被DNAT转向192.168.10.8:80端口（也就是LAN内的web服务器上）

iptables -t nat -A POSTROUTING -p tcp -d 218.206.70.151 --dport 80 -j SNAT --to-source 218.206.70.155

把即将要流出本机的数据的source ip address修改成为218.206.70.155

云华示例：

iptables -t nat PREROUTING -I -d 192.168.1.107 -p tcp --dport 1234 -j DNAT --to 203.88.193.109:8089

iptables -I POSTROUTING -t nat -d 203.88.193.109 -j MASQUERADE

MASQUERADE：ＩＰ地址伪装。

连接107服务器1234窗口的数据会被转移到203.88.193.109:8089

电信示例：

iptables -t nat -A PREROUTING -p tcp -d 211.147.224.30 --dport 80 -j DNAT --to 124.172.234.192:9002

iptables -t nat -A POSTROUTING -p tcp -d 124.172.234.192 –dport 9002 -j SNAT --to-source 211.147.224.30

iptables -t nat –A PREROUTING -p tcp -d 211.147.224.30 --dport 9081 -j DNAT --to 118.85.200.55:9081

iptables -t nat –A POSTROUTING -p tcp -d 118.85.200.55 --dport 9081 -j SNAT --to-source 211.147.224.30

移动示例：

iptables -t nat -A PREROUTING -p tcp -d 203.88.193.223 --dport 7891 -j DNAT --to 211.136.221.168:7891

iptables -t nat -A POSTROUTING -p tcp -d 211.136.221.168 --dport 7891 -j SNAT --to-source 203.88.193.223

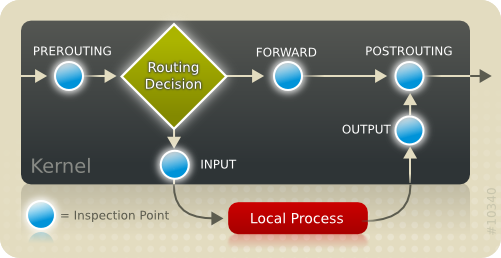
iptables –t nat –L查看规则

禁止PING:

iptables –A INPUT –i eth0 –p icmp –icmp-type 8 –j DROP

　　在filter表的INPUT链的最后追加一条规则——丢弃通过eth0传入的序号为8的icmp协议数据包，即ping request，这样其他主机将不能成功的ping通本机了。

数据包在iptables中要经过的链（chain）：

[](http://blog.myhnet.cn/wp-content/uploads/2009/01/netfilter_packet_flow.png)

图中正菱形的区域是对数据包进行判定转发的地方。在这里，系统会根据IP数据包中的destination ip address中的IP地址对数据包进行分发。如果destination ip adress是本机地址，数据将会被转交给INPUT链。如果不是本机地址，则交给FORWARD链检测。

NAT（Network Address Translation，网络地址转换）是将IP 数据报报头中的IP 地址转换为另一个IP 地址的过程。

在实际应用中，NAT 主要用于实现私有网络访问公共网络的功能。

DNAT/SNAT

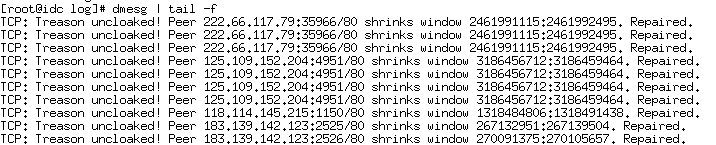
DNAT（Destination Network Address Translation,目的地址转换) 通常被叫做目的映谢。

SNAT（Source Network Address Translation，源地址转换）通常被叫做源映谢。

# dmesg监控系统

显示内核中的ring buffer内容。内核将开机信息存放在ring buffer(环状链表)中，用户通过查看ring buffer信息，可以了解系统开机时所发生的事件。用户也可能查看/var/log/dmesg 文件获得开机信息。通过dmesg –c 可以清除ring buffer数据．

同时一些系统的错误(如网络错误，内存错误)也会填入ring buffer中去．如：



结论：

The remote host decided to shrink the TCP window size without negotiating such with your Linux box. The message is of the informational level, meaning Linux doesn't like what it is seeing but will cope with it and carry on.

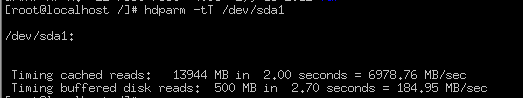
意思是：在远程client和服务器通信的过程中，由于client没有和服务器协商，就减少了 TCP window的长度，也就是包的大小变小了，因此有这类的提示信息，此时服务器依然处理该类请求，但是处理的速度会比之前较差。

出现这种情况的时候，最好看看你的服务器流量监控情况，看看是否有流量异常，如果带宽被占满，有可能被电信或者联通进行了带宽限制，最好的方法是花费银子买带宽，或者看看你的web服务器有没有可优化的地方，例如gzip是否开启，响应时间是合适等等，这些手段比被电信限制你的龟速访问的体验好一些，是没钱人的垂死挣扎的手段，不是最终的方法。

此类问题，增加带宽或许是最好的方式。

<http://hi.baidu.com/zheng918/blog/item/d86079f456ac94d1f2d3858b.html>

# hdparm硬盘速度



数据库应采用raid0+1方式提高硬盘写速度，raid5读速度不错，但写速度相对要差，检验信息计算需要花费太多时间，简单的raid1做数据备份性能最高。

最简单的技术处理也就是最实在的解决方案。

详细raid描述可见:

<http://zh.wikipedia.org/wiki/Raid>

查看当前硬盘写Cache状态

root@ubuntu:~/hdparm-9.37# hdparm -W  /dev/sda

关闭硬盘的写Cache

root@ubuntu:~/hdparm-9.37# hdparm -W  0 /dev/sda

**打开硬盘的写Cache**

root@ubuntu:~/hdparm-9.37# hdparm -W  1 /dev/sda

# FIO

wget <http://fossies.org/linux/privat/fio-2.1.6.1.tar.gz>

bs=16k           单次io的块文件大小为16k

rw=randwrite        测试随机写的I/O  
rw=randrw          测试随机写和读的I/O

numjobs=30         本次的测试线程为30.

runtime=1000        测试时间为1000秒

fio -readonly -name iops -rw=randread -bs=512 -runtime=10 -iodepth 1 -direct=1 -filename=/dev/mapper/VolGroup-lv\_home

fio -readonly -name iops -rw=read -bs=512 -runtime=20 -iodepth 1 -filename=/dev/sda1 -direct=1

fio -name iops -rw=write -bs=4k -runtime=20 -iodepth 1 -filename /dev/sda1 -direct=1

# iostat

iostat -d -k 1 10

参数 -d 表示，显示设备（磁盘）使用状态；-k某些使用block为单位的列强制使用Kilobytes为单位；1 10表示，数据显示每隔1秒刷新一次，共显示10次。

iostat –x –k 1 100

await：每一个IO请求的处理的平均时间（单位是微秒毫秒）。这里可以理解为IO的响应时间，一般地系统IO响应时间应该低于5ms，如果大于10ms就比较大了。

%util：在统计时间内所有处理IO时间，除以总共统计时间。例如，如果统计间隔1秒，该设备有0.8秒在处理IO，而0.2秒闲置，那么该设备的%util = 0.8/1 = 80%，所以该参数暗示了设备的繁忙程度。一般地，如果该参数是100%表示设备已经接近满负荷运行了（当然如果是多磁盘，即使%util是100%，因为磁盘的并发能力，所以磁盘使用未必就到了瓶颈）。



设备号：

/dev/sda 第1个 scsi 磁盘(整个磁盘)

/dev/sdb 第2个 scsi 磁盘(整个磁盘)

/dev/sdc 第3个 scsi 磁盘(整个磁盘)

fdisk -l

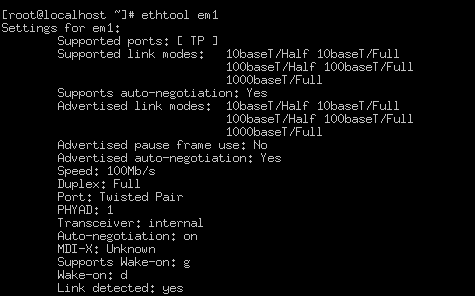
/dev/sda1 第1个分区

/dev/sda2 第2个分区

/dev/md-0 第1组 metadisk (raid)设备

/dev/md-1 第2组 metadisk (raid)设备

# ethtool网卡测试



# Screen

使用 screen 很方便，有以下几个常用选项：

* screen -list 来列出所有会话。
* screen -r *session name* 来重新连接指定会话。
* screen –S session name 新建会话
* 用快捷键CTRL-a d 来暂时断开当前会话。
* C-a K (kill) Destroy current window.

# Traceroute Ping

**Traceroute的工作原理：**

Traceroute最简单的基本用法是：traceroute hostname

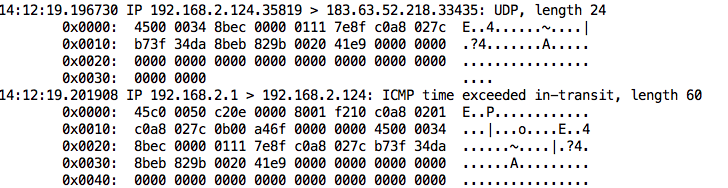
traceroute 的原理是试图以最小的TTL发出探测包来跟踪数据包到达目标主机所经过的网关，然后监听一个来自网关ICMP的应答。发送数据包的大小默认为 38个字节。

Traceroute程序的设计是利用ICMP及IP header的TTL（Time To Live）栏位（field）。首先，traceroute送出一个TTL是1的IP datagram（其实，每次送出的为3个40字节的包，包括源地址，目的地址和包发出的时间标签）到目的地，当路径上的第一个路由器（router）收到这个datagram时，它将TTL减1。此时，TTL变为0了，所以该路由器会将此datagram丢掉，并送回一个「ICMP time exceeded」消息（包括发IP包的源地址，IP包的所有内容及路由器的IP地址），traceroute 收到这个消息后，便知道这个路由器存在于这个路径上，接着traceroute 再送出另一个TTL是2 的datagram，发现第2 个路由器...... traceroute 每次将送出的datagram的TTL 加1来发现另一个路由器，这个重复的动作一直持续到某个datagram 抵达目的地。当datagram到达目的地后，该主机并不会送回ICMP time exceeded消息，因为它已是目的地了，那么traceroute如何得知目的地到达了呢？

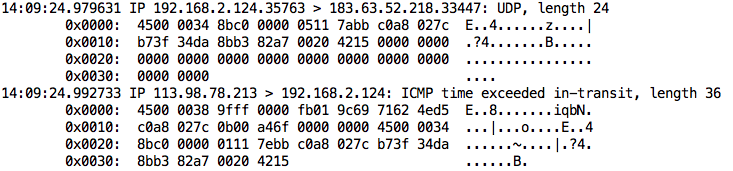
Traceroute在送出UDP datagrams到目的地时，它所选择送达的port number 是一个一般应用程序都不会用的号码（30000 以上），所以当此UDP datagram 到达目的地后该主机会送回一个「ICMP port unreachable」的消息，而当traceroute 收到这个消息时，便知道目的地已经到达了。所以traceroute 在Server端也是没有所谓的Daemon 程式。

Traceroute提取发 ICMP TTL到期消息设备的IP地址并作域名解析。每次 ，Traceroute都打印出一系列数据,包括所经过的路由设备的域名及 IP地址,三个包每次来回所花时间。

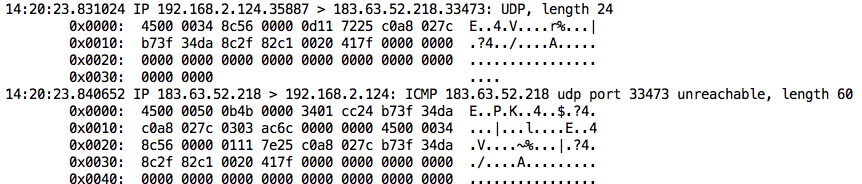
第一跳：



第五跳：



最后一跳：



记录按序列号从1开始，每个纪录就是一跳 ，每跳表示一个网关，我们看到每行有三个时间，单位是 ms，其实就是-q的默认参数。探测数据包向每个网关发送三个数据包后，网关响应后返回的时间；如果您用 traceroute -q 4 www.58.com ，表示向每个网关发送4个数据包。

有时我们traceroute 一台主机时，会看到有一些行是以星号表示的。出现这样的情况，可能是防火墙封掉了ICMP的返回信息，所以我们得不到什么相关的数据包返回数据。

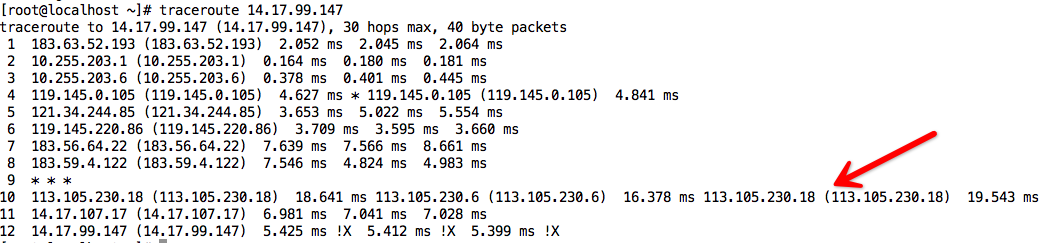
有时我们在某一网关处延时比较长，有可能是某台网关比较阻塞，也可能是物理设备本身的原因。当然如果某台DNS出现问题时，不能解析主机名、域名时，也会 有延时长的现象；您可以加-n 参数来避免DNS解析，以IP格式输出数据。

traceroute可以在ip包中放三种数据：

1) 使用UDP包(默认选项是－U)

2）使用TCP包 选项是－T

3）使用ICMP包 选项是－I



双线保护

如果在局域网中的不同网段之间，我们可以通过traceroute 来排查问题所在，是主机的问题还是网关的问题。如果我们通过远程来访问某台服务器遇到问题时，我们用到traceroute 追踪数据包所经过的网关，提交IDC服务商，也有助于解决问题；但目前看来在国内解决这样的问题是比较困难的，就是我们发现问题所在，IDC服务商也不可能帮助我们解决。

Traceroute Hop Information

Flag Description

!H ICMP host unreachable

!N ICMP network unreachable

!P ICMP protocol unreachable

!S Source route failed

!F Fragmentation needed

!X Communication administratively prohibited

!# ICMP unreachable code #

Ping TTL通常linux设置位64， win则为128

Ping返回55，则路由跳数位64-55=9 hop

网络丢包测试：ping –f 192.168.10.1

批量发送ICMP ECHO\_REQUEST请求，每个请求打印一个点，每个响应删除一个点。如果网络存在丢包，那么会呈现出一长串不断增加的点，简单易用.

<http://www.cnblogs.com/peida/archive/2013/03/07/2947326.html>

每天一个linux命令（55）：traceroute命令