网络测试命令：

ping 测试网络连通性

-c ping 的个数

-t TTL值

-s ping包大小

-i ping的间隔

traceroute 用于追踪数据包在网络上的传输时的全部路径，它默认发送的数据包大小是40字节

tracepath 用来追踪并显示报文到达目的的主机所经过的路由信息

host 是常用的分析域名查询工具，可以用来测试域名系统工作是否正常

格式：host [option] domain [server]

-a 显示详细的DNS信息

例如：host www.baidu.com 8.8.8.8

dig 是比nslookup和host更强大的DNS查询工具。

格式：dig [options] domain [@server]

+trace —— 输出域名解析过程中详细的debug信息

-t type —— 制定查询的DNS记录类型，例如A记录、CNAME记录以及NS记录等等

-x —— 从ip反解析域名

+short —— 精简地输出dig结果

nslookup

nslookup支持交互式和非交互式两种查询。当直接在命令提示符后输入nslookup命令时，就进入了nslookup的交互式操作。否则直接返回给用户解析结果

arp 显示和修改操作系统的ARP缓存表。这个程序已经被ip neigh替代

-a<主机>：显示arp缓冲区的所有条目；

-d<主机>：从arp缓冲区中删除指定主机的arp条目；

-e：以Linux的显示风格显示arp缓冲区中的条目；

-s<主机><MAC地址>：设置指定的主机的IP地址与MAC地址的静态映射；

-n：以数字方式显示arp缓冲区中的条目；

-v：显示详细的arp缓冲区条目，包括缓冲区条目的统计信息；

netstat

netstat命令是一个监控TCP/IP网络的非常有用的工具，它可以显示路由表、实际的网络连 接以及每一个网络接口设备的状态信息。

-a或--all：显示所有连线中的Socket；

-l或--listening：显示监控中的服务器的Socket；

-n或--numeric：直接使用ip地址，而不通过域名服务器；

-p或--programs：显示正在使用Socket的程序识别码和程序名称；

-r或--route：显示Routing Table；

-s或--statistice：显示网络工作信息统计表；

-t或--tcp：显示TCP传输协议的连线状况；

-u或--udp：显示UDP传输协议的连线状况；

-v或--verbose：显示指令执行过程；

-V或--version：显示版本信息；

ss

ss命令用于显示socket状态. 他可以显示PACKET sockets, TCP sockets, UDP sockets, DCCP sockets, RAW sockets, Unix domain sockets等等统计. 它比其他工具展示等多tcp和state信息. 它是一个非常实用、快速、有效的跟踪IP连接和sockets的新工具。

ss -l 显示本地打开的所有端口

ss -pl 显示每个进程具体打开的socket

ss -t -a 显示所有tcp socket

ss -u -a 显示所有的UDP Socekt

ss -o state established '(dport = :smtp or sport = :smtp)' 显示所有已建立的SMTP连接

ss -o state established '(dport = :http or sport = :http)' 显示所有已建立的HTTP连接

ss -x src /tmp/.X11-unix/* 找出所有连接X服务器的进程

ss -s 列出当前socket详细信息:

ss做地址筛选

ss src ADDRESS_PATTERN

src：表示来源

ADDRESS_PATTERN：表示地址规则

如下：

ss src 120.33.31.1 # 列出来之20.33.31.1的连接

列出来至120.33.31.1,80端口的连接

ss src 120.33.31.1:http

ss src 120.33.31.1:80

ss做端口筛选

ss dport OP PORT

OP:是运算符

PORT：表示端口

dport：表示过滤目标端口、相反的有sport

运算符如下：

<= or le : 小于等于 >= or ge : 大于等于

== or eq : 等于

!= or ne : 不等于端口

< or lt : 小于这个端口 > or gt : 大于端口

具体实例：

ss sport = :http 也可以是 ss sport = :80

ss dport = :http

ss dport \> :1024

ss sport \> :1024

ss sport \< :32000

ss sport eq :22

ss dport != :22

ss state connected sport = :http

ss \(sport = :http or sport = :https \)

ss -o state fin-wait-1 \(sport = :http or sport = :https \) dst 192.168.1/24

iperf

一个 TCP/IP 和 UDP/IP 的性能测量工具，能够提供网络吞吐率信息，以及震动、丢包率、最大段和最大传输单元大小等统计信息；从而能够帮助我们测试网络性能，定位网络瓶颈。

TCP测试：

服务器命令：iperf3 -s -i 1

客户端命令：iperf3 -c 192.168.5.187 -p 5201 -i 1 -t 100

UDP测试：

服务器命令：iperf3 -s -i 1

客户端命令：iperf3 -c 192.168.5.187 -p 5201 -u -i 1 -t 100

网络配置命令：

ifconfig

ifconfig eth0 查看eth0 ip信息

ifconfig eth0 192.168.1.10 netmask 255.255.255.0 临时设置IP信息

ifconfig eth0 down

ifconfig eth0 up

ifup

ifdown

nmcli

RHEL7里面的网卡命名方式从eth0,1,2的方式变成了enoXXXXX的格式。 en 代表的是enthernet (以太网），o 代表的是onboard (内置），那一串数字是主板的某种索引编号自动生成，以便保证其唯一性。和原先的命名方式对比，这种新的方式比较长，难以记忆，不过优点在于编号唯一，做系统迁移的时候不容易出错。

说明：RHEL 7.0默认安装好之后是没有自动开启网络连接的！

方法1、进入网络置配文件目录，修改网卡配置文件，如下图所示：

TYPE="Ethernet|Bridge"网络接口类型

BOOTPROTO="static | none" #启用静态IP地址(BOOTPROTO=dhcp启用动态获取IP)

DEFROUTE="yes"

IPV4_FAILURE_FATAL="no"

IPV6INIT="yes|no" 是否支持IPV6

USERCTL="yes|no" 是否允许普通用户控制此接口

PEERDNS = "yes|no" 是不是接受DHCP服务器指派的DNS服务器地址

IPV6_AUTOCONF="yes"

IPV6_DEFROUTE="yes"

IPV6_FAILURE_FATAL="no"

NAME="eno16777736"

UUID="8071cc7b-d407-4dea-a41e-16f7d2e75ee9" 设备标识号

ONBOOT="yes" #开启自动启用网络连接（默认值为no）

IPADDR0="192.168.21.128" #设置IP地址 只有在BOOTPROTO={none|static}设置才有效

PREFIX0="24" #设置子网掩码 此设置也可用 PREFIX=n (n为掩码位数)

GATEWAY0="192.168.21.2" #设置网关 要与IP地址属于同一网段

DNS1="8.8.8.8" #设置主DNS

DNS2="8.8.4.4" #设置备DNS

HWADDR="00:0C:29:EB:F2:B3"

IPV6_PEERDNS="yes"

IPV6_PEERROUTES="yes"

service network restart #重启网络

查看ip信息：

Ifconfig命令或ip addr show 或ip addr show dev 网络设备名或ip addr查看所有网卡信息。

#ip addr show dev eno16777736

```
2: eno16777736: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 00:0c:29:24:ec:72 brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.100/24 brd 192.168.10.255 scope global eno16777736
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe24:ec72/64 scope link
       valid_lft forever preferred_lft forever
```

1.接口状态

2.硬件

3.IPv4地址与子网

4.广播地址，范围，设备名称

5.IPv6信息

ip 命令显示关于网络性能的统计信息，接收（RX）发送（TX）数据包。

```
[root@server1 ~]# ip -s link show eno16777736
2: eno16777736: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT q
len 1000
    link/ether 00:0c:29:24:ec:72 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    11868      129      0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    4054       28       0       0       0       0
```

用nmcli connection show网卡信息

```
[root@server1 ~]# nmcli connection show
NAME         UUID                                   TYPE            DEVICE
eno16777736  a460f7fb-9236-4b4d-b037-0382d1b382e7   802-3-ethernet  eno16777736
eno33554992  0885a5e8-269f-4306-ab63-cbd2f41392e8   802-3-ethernet  eno33554992
```

测试网络是否正常：

ping www.baidu.com #测试网络是否正常

traceroute 和 mtr 显示出我们到达一个网络所经过的路由信息

yum -y install traceroute

yum -y install mtr

traceroutewww.baidu.com

mtr是Red Hat自带的工具，结合了"traceroute"和"ping"功能于一身诊断工具

```
[root@server1 ~]# mtr -r 192.168.53.5
Start: Thu Apr 30 15:50:32 2015
HOST: server1.benet.com           Loss%   Snt   Last   Avg  Best  Wrst StDev
  1.|-- 192.168.53.5               0.0%    10    0.4   0.6   0.2   1.9   0.0
```

第一列:显示的是IP地址和本机域名，这点和tracert很像

第二列:是显示的每个对应IP的丢包率

第三列:snt设置每秒发送数据包的数量，默认值是10 可以通过参数 -c来指定

第四列:显示的最近一次的返回时延

第五列:是平均值这个应该是发送ping包的平均时延

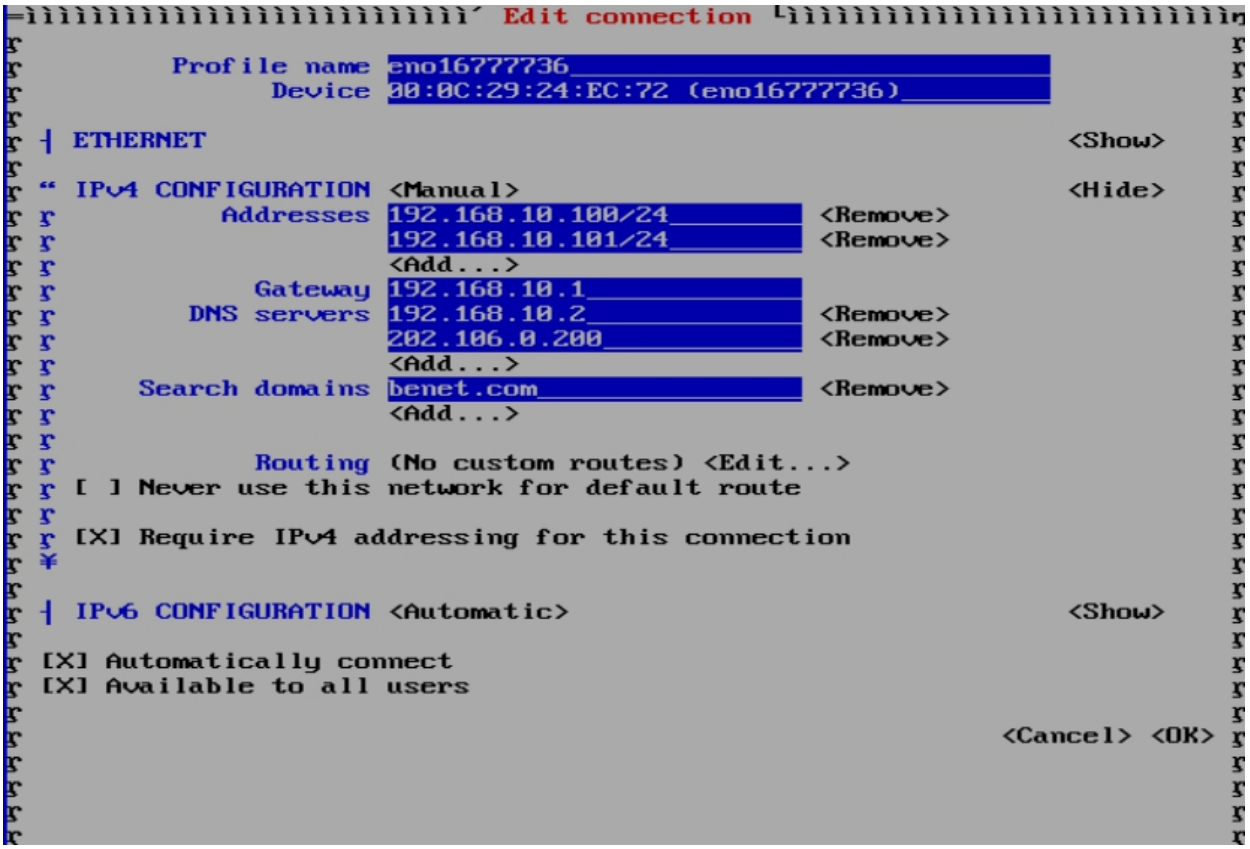
第六列:是最好或者说时延最短的

第七列:是最差或者说时延最常的

第八列:是标准偏差

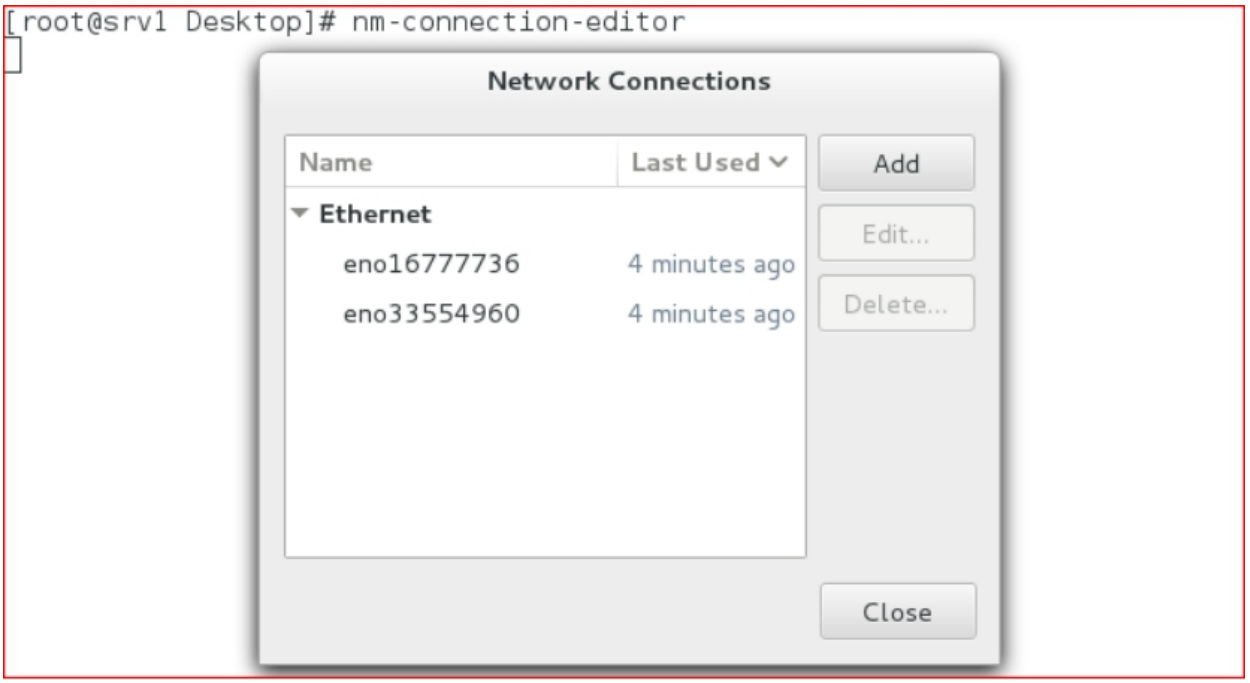
方法2：执行nmtui命令

使用nmtui命令会弹出一工具界面



重启network服务。

方法3：通过nm-connection-editor来配置连接（注：必须在图形界面下运行）



配置好了，重启网络服务，执行ifconfig或ip addr发现已经获取了新的地址。

一旦该连接建立成功，在/etc/sysconfig/networks-scripts 下面就会自动创建同名的配置文件

方法4：nmcli命令

nmcli是个很强大的命令，后面一大堆选项和对象可以配置。看看帮助文档，对象可以是综合信息，网络，信号和连接。

地址配置工具：nmcli

nmcli – command-line tool for controlling NetworkManager

命令语法：

nmcli[OPTIONS] OBJECT { COMMAND | help }

OBJECT和COMMAND可以用全称也可以用简称，最少可以只用一个字母，建议用头三个字母。OBJECT里面我们平时用的最多的就是connection和device，这里需要简单区分一下connection和device。

device叫网络接口，是物理设备

device -show and manage network interfaces

nmcli device help

connection是连接，偏重于逻辑设置

connection -start, stop, and manage network connections

nmcli connection help

多个connection可以应用到同一个device，但同一时间只能启用其中一个connection。这样的好处是针对一个网络接口，我们可以设置多个网络连接，比如静态IP和动态IP，再根据需要up相应connection

```
[root@server1 ~]# nmcli --help
Usage: nmcli [OPTIONS] OBJECT { COMMAND | help }

OPTIONS
  -t[erse]                                  terse output
  -p[retty]                                 pretty output
  -m[ode] tabular|multiline                 output mode
  -f[ields] <field1,field2,...>|all|common  specify fields to output
  -e[scape] yes|no                          escape columns separators in values
  -n[ocheck]                                don't check nmcli and NetworkManager versions
  -a[sk]                                    ask for missing parameters
  -w[ait] <seconds>                         set timeout waiting for finishing operations
  -v[ersion]                                show program version
  -h[elp]                                   print this help

OBJECT
  g[eneral]       NetworkManager's general status and operations
  n[etworking]    overall networking control
  r[adio]         NetworkManager radio switches
  c[onnection]    NetworkManager's connections
  d[evice]        devices managed by NetworkManager
```

如何配置连接，还是先看看帮助，他后面可以跟show，up, down, add, modify, edit, delete, reload,

```
[root@server1 ~]# nmcli connection --help
Usage: nmcli connection { COMMAND | help }

COMMAND := { show | up | down | add | modify | edit | delete | reload | load }

  show [--active] [[id | uuid | path | apath] <ID>] ...

  up [[id | uuid | path] <ID>] [ifname <ifname>] [ap <BSSID>]

  down [id | uuid | path | apath] <ID>

  add COMMON_OPTIONS TYPE_SPECIFIC_OPTIONS IP_OPTIONS

  modify [--temporary] [id | uuid | path] <ID> ([+|-]<setting>.<property> <value>)+

  edit [id | uuid | path] <ID>
  edit [type <new_con_type>] [con-name <new_con_name>]

  delete [id | uuid | path] <ID>

  reload

  load <filename> [ <filename>... ]
```

看看device有哪些参数

```
[root@server1 ~]# nmcli device --help
Usage: nmcli device { COMMAND | help }

COMMAND := { status | show | connect | disconnect | wifi }

  status

  show [<ifname>]

  connect <ifname>

  disconnect <ifname>

  wifi [list [ifname <ifname>] [bssid <BSSID>]]

  wifi connect <(B)SSID> [password <password>] [wep-key-type key|phrase] [ifname <ifname>]
                         [bssid <BSSID>] [name <name>] [private yes|no]

  wifi rescan [[ifname] <ifname>]
```

```
[root@server1 ~]# nmcli connection down eno16777736
[root@server1 ~]# nmcli connection show
NAME         UUID                                  TYPE            DEVICE
eno16777736  a460f7fb-9236-4b4d-b037-0382d1b382e7  802-3-ethernet  --
eno33554992  0885a5e8-269f-4306-ab63-cbd2f41392e8  802-3-ethernet  eno33554992
[root@server1 ~]# nmcli device status
DEVICE       TYPE      STATE         CONNECTION
eno33554992  ethernet  connected     eno33554992
eno16777736  ethernet  disconnected  --
lo           loopback  unmanaged     --
```

查看具体的设备信息可以通过 nmcli connection show 设备名来查看

```
[root@server1 ~]# nmcli connection show eno16777736
connection.id:                          eno16777736
connection.uuid:                        a460f7fb-9236-4b4d-b037-0382d1b382e7
connection.interface-name:              --
connection.type:                        802-3-ethernet
connection.autoconnect:                 yes
connection.timestamp:                   1430385092
connection.read-only:                   no
connection.permissions:
connection.zone:                        public
connection.master:                      --
```

添加一个新连接，先先看看帮助

```
nmcli connection add --help
```

例如：

nmcli connection add con-name test-con ipv4.addresses "10.1.1.100/24" ipv4.gateway 10.1.1.1 ipv4.dns 202.106.0.20 ipv4.method manual connection.autoconnect yes type ethernet ifname ens33

修改现有连接，可以先先看看帮助

nmcli connection modify --help

```
[root@server1 ~]# nmcli connection modify eno16777736 ipv4.method manual
[root@server1 ~]# nmcli connection modify eno16777736 ipv4.addresses "192.168.5.20/24"
[root@server1 ~]# service  network restart
Restarting network (via systemctl):                        [  OK  ]
[root@server1 ~]# ifconfig
eno16777736: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.5.20  netmask 255.255.255.0  broadcast 192.168.5.255
        inet6 fe80::20c:29ff:fe24:ec72  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:24:ec:72  txqueuelen 1000  (Ethernet)
        RX packets 2680  bytes 219713 (214.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3173  bytes 236425 (230.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

附：修改RHEL7的网卡名称

RHEL7安装完成之后，默认的网卡名称是eno16777736

输入如下命令，进入对应目录，编辑文件：

vim /etc/sysconfig/grub

然后，往这个文件中添加“net.ifnames=0 biosdevname=0”内容，作用是禁用该可预测命名规则，如下图所示：

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=rhel/root crashkernel=auto  rd.lvm.lv=rhel/swap vconsole.font=latarcyr
heb-sun16 vconsole.keymap=us rhgb quiet net.ifnames=0 biosdevname=0"
GRUB_DISABLE_RECOVERY="true"
```

接着执行下面的命令，效果如下：

```
[root@server1 ~]# grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-123.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-123.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-60e839870acd49b68d569dcbb05142e2
Found initrd image: /boot/initramfs-0-rescue-60e839870acd49b68d569dcbb05142e2.img
[ 1634.511385] end_request: I/O error, dev fd0, sector 0
done
```

然后，重启系统后查看网卡名称

```
[root@server1 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.10.100  netmask 255.255.255.0  broadcast 192.168.10.255
        inet6 fe80::20c:29ff:fe24:ec72  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:24:ec:72  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 119  bytes 10067 (9.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

ethtool

查看网路接口设备本身的属性

ethtool 网卡接口名称

hostname

hostnamectl

rhel7不再使用/etc/sysconfig/network设置主机名，而是使用/etc/hostname文件，下面就介绍有关rhel7设置主机名的几种方法。

方法：1

修改/etc/hostname文件设置主机名。

方法：2

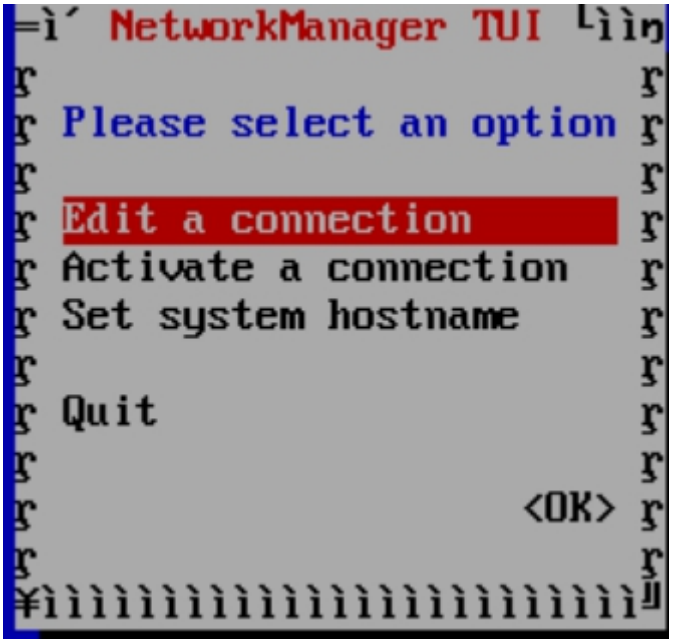
执行hostnamectl命令

使用hostnamectl命令，hostnamectl set-hostname *name*，再通过hostname或者hostnamectl status命令查看更改是否生效。

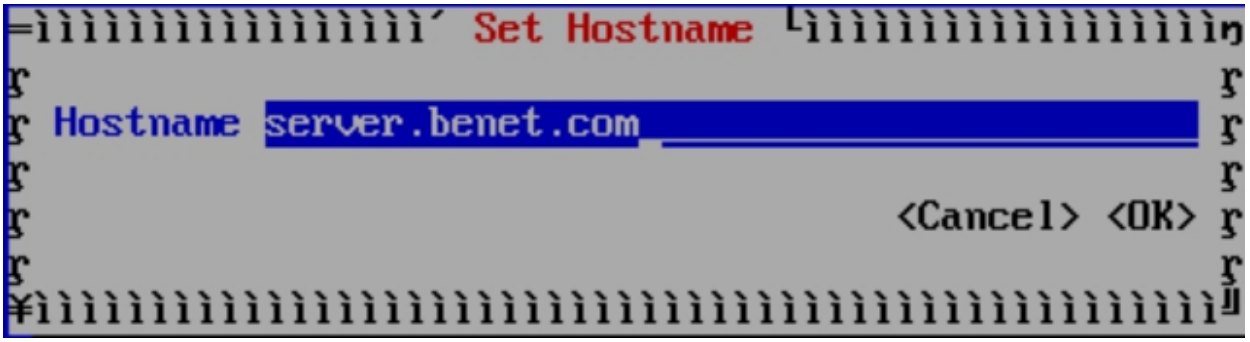
方法：3

执行nmtui命令

使用nmtui命令会弹出一工具界面



选择第三项"set system hostname"修改主机名



修改后选择"ok"退出nmtui工具。

执行**systemctl restart systemd-hostnamed**

执行hostname 或hostnamectl查看。

方法：4

执行nmcli命令

nmcli general hostname查看主机名

nmcli general hostname my-server 设置主机名

执行systemctl restart systemd-hostnamed使修改生效

高级网络命令

ip

Linux的ip命令和ifconfig类似，但前者功能更强大，并旨在取代后者。使用ip命令，只需一个命令，你就能很轻松地执行一些网络管理任务。

设置和删除IP地址

ip addr add 192.168.17.30/24 dev eth0

ip addr del 192.168.17.30/24 dev eth0

列出路由表条目：

ip route show

ip route get 192.168.17.130

添加路由条目：

ip route add default via 192.168.17.3

ip route add 192.168.100.0/24 dev ens33

显示网络统计数据

ip -s link

ip -s link ls ens33

ip -s -s link ls ens33

ARP条目：

ip neigh

激活和停止网络接口：

ip link set eth0 down

ip link set eth0 up

iptraf

IP局域网监控工具，它可以实时地监视网卡流量，可以生成各种网络统计数据，包括TCP信息、UDP统计、ICMP和OSPF信息、以太网负载信息、节点统计、IP校验和错误和其它一些信息。

yum info iptraf-ng -y

iptraf-ng --help 主要参数可以查看帮助

-i iface　网络接口：立即在指定网络接口上开启IP流量监视,iface为all指监视所有的网络接口，iface指相应的interface

-g　立即开始生成网络接口的概要状态信息

-d iface　网络接口：在指定网络接口上立即开始监视明细的网络流量信息,iface指相应的interface

-s iface　网络接口：在指定网络接口上立即开始监视TCP和UDP网络流量信息,iface指相应的interface

-z iface　网络接口：在指定网络接口上显示包计数,iface指相应的interface

-l iface　网络接口：在指定网络接口上立即开始监视局域网工作站信息,iface指相应的interface

-t timeout　时间：指定iptraf指令监视的时间，timeout指监视时间的minute数

主菜单如下：

tcpdump

tcpdump是一个用于截取网络分组，并输出分组内容的工具。tcpdump凭借强大的功能和灵活的截取策略，使其成为类UNIX系统下用于网络分析和问题排查的首选工具。 tcpdump提供了源代码，公开了接口，因此具备很强的可扩展性，对于网络维护和入侵者都是非常有用的工具。

YUM安装即可：yum install tcpdump -y

常用参数介绍：

-i 指定抓包网络接口

-c 在收到指定的数量的分组后，tcpdump就会停止

-v 输出一个稍微详细的信息，例如在ip包中可以包括ttl和服务类型的信息

-vv 输出详细的报文信息

-w 直接将分组写入文件中，而不是不分析并打印出来

-b 在数据-链路层上选择协议，包括ip、arp、rarp、ipx都是这一层的

-A，数据包的内容以 ASCII 显示，通常用来捉取 WWW 的网页数据包资料。

-X，可以列出十六进制 (hex) 以及 ASCII 的数据包内容，对于监听数据包内容很有用

-n 不把网络地址转换成名字

-nn 不进行端口名称的转换

-r 从指定的文件中读取包(这些包一般通过-w选项产生)

-S 将tcp的序列号以绝对值形式输出，而不是相对值

-t 不在每一行中输出时间戳

-tt 在每一行中输出非格式化的时间戳

-ttt 输出本行和前面一行之间的时间差

-tttt 在每一行中输出由date处理的默认格式的时间戳

tcpdump的表达式介绍

表达式是一个正则表达式，tcpdump利用它作为过滤报文的条件，如果一个报文满足表 达式的条件，则这个报文将会被捕获。如果没有给出任何条件，则网络上所有的信息包

将会被截获。

在表达式中一般如下几种类型的关键字：

第一种是关于类型的关键字，主要包括host，net，port，例如 host 210.27.48.2， 指明 210.27.48.2是一台主机，net 202.0.0.0指明202.0.0.0是一个网络地址，port 23 指明端口号是23。如果没有指定类型，缺省的类型是host。

第二种是确定传输方向的关键字，主要包括src，dst，dst or src，dst and src， 这些关键字指明了传输的方向。举例说明，src 210.27.48.2 ，指明ip包中源地址是 210.27.48.2 ， dst net 202.0.0.0 指明目的网络地址是202.0.0.0。如果没有指明 方向关键字，则缺省是src or dst关键字。

第三种是协议的关键字，主要包括fddi，ip，arp，rarp，tcp，udp等类型。Fddi指明是在FDDI (分布式光纤数据接口网络)上的特定的网络协议，实际上它是”ether”的别名，fddi和ether 具有类似的源地址和目的地址，所以可以将fddi协议包当作ether的包进行处理和分析。 其他的几个关键字就是指明了监听的包的协议内容。如果没有指定任何协议，则tcpdump 将会监听所有协议的信息包。

除了这三种类型的关键字之外，其他重要的关键字如下：gateway， broadcast，less， greater， 还有三种逻辑运算，取非运算是 ‘not ' '! ‘， 与运算是’and’，’ &&';或运算是’or’ ，’ ||’ ； 这些关键字可以组合起来构成强大的组合条件来满足人们的需要。

举例说明：

想要截获所有210.27.48.1 的主机收到的和发出的所有的数据包：

#tcpdump host 210.27.48.1

想要截获主机210.27.48.1 和主机210.27.48.2 或210.27.48.3的通信，使用命令： （在命令行中使用 括号时，一定要加斜杠）

#tcpdump host 210.27.48.1 and \(210.27.48.2 or 210.27.48.3 \)

如果想要获取主机210.27.48.1除了和主机210.27.48.2之外所有主机通信的ip包，使用命令：

#tcpdump ip host 210.27.48.1 and ! 210.27.48.2

如果想要获取主机210.27.48.1接收或发出的telnet包，使用如下命令：

#tcpdump tcp port 23 host 210.27.48.1

对本机的udp 123 端口进行监视 123 为ntp的服务端口

tcpdump udp port 123

系统将只对名为hostname的主机的通信数据包进行监视。主机名可以是本地主机，也可以是网络上的任何一台计算机。下面的命令可以读取主机hostname发送的所有数据：

#tcpdump -i eth0 src host hostname

下面的命令可以监视所有送到主机hostname的数据包：

#tcpdump -i eth0 dst host hostname

还可以监视通过指定网关的数据包：

#tcpdump -i eth0 gateway Gatewayname

如果想监视编址到指定端口的TCP或UDP数据包，那么执行以下命令：

#tcpdump -i eth0 host hostname and port 80

如果想要获取主机210.27.48.1除了和主机210.27.48.2之外所有主机通信的ip包,使用命令：

#tcpdump ip host 210.27.48.1 and ! 210.27.48.2

将结果保存为wireshark识别的文件

#tcpdump -i ens33 -nn port 22 and src host 192.168.5.189 -w test.pacpng

过滤源主机192.168.0.1和目的端口不是telnet的报头，并导入到tes.t.txt文件中：

#tcpdump src host 192.168.0.1 and dst port not telnet -l > test.txt

nmap

网络探勘工具与安全/端口扫描器。nmap是不局限于仅仅收集信息和枚举，同时可以用来作为一个漏洞探测器或安全扫描器。它可以适用于winodws,linux,mac等操作系统。Nmap是一款非常强大的实用工具,可用于：

检测活在网络上的主机（主机发现）

检测主机上开放的端口（端口发现或枚举）

检测到相应的端口（服务发现）的软件和版本

检测操作系统，硬件地址，以及软件版本

检测脆弱性的漏洞（Nmap的脚本）

使用实例：

使用主机名扫描：

nmap www.baidu.com

使用IP地址扫描：

nmap 192.168.1.100

使用-v选项，可以提供更详细的输出

nmap -v www.baidu.com

扫描多个主机

nmap -v www.baidu.com www.sina.com.cn

nmap 192.168.0.101,102,103

nmap 192.168.5.*

从一个文件中输入要扫描的主机

nmap -iL ip.txt

[root@cloud002 ~]# cat ip.txt
192.168.5.190
192.168.5.191
192.168.5.192
扫描一个IP地址段
nmap 192.168.0.101-110
排除扫描的地址
nmap 192.168.0.* --exclude 192.168.0.100
探测操作系统、使用脚本扫描和路由追踪
nmap -A www.baidu.com
开启TCP ack 扫描
nmap -sA www.baidu.com
UDP扫描
nmap -sU 192.168.1.1
开启TCP FIN扫描
nmap -sF 192.168.1.1
探测某个网络上的存活的主机
nmap -v -sn 192.168.5.0/24
扫描特定的端口
nmap -p 80,443 192.168.0.101
nmap -p 80-160 192.168.0.101

其它有关网络相关命令

开启Linux主机路由功能：
1、改配置文件
vim /etc/sysctl.conf
net.ipv4.ip_forward = 1 默认值为0
[root@ketang-test ~]# sysctl -p 令修改立刻生效
2、临时开启：
echo 1 > /proc/sys/net/ipv4/ip_forward

查看命令
ip addr 查看地址信息
ip route 查看路由表
route -n 查看路由表
netstat -r 查看路由表

ip rule 查看路由策略

添加路由条目

route add -net 192.168.4.0/24 gw 192.168.1.254 添加具体路由条目，通过下一跳

route add -net 192.168.3.0/24 dev eth0 添加具体路由条目，通过出接口

route add default gw 192.168.1.254 添加默认路由

route del -net 192.168.4.0/24 删除路由表

网卡配置文件目录：

/etc/sysconfig/network-scripts

第一块网卡配置文件:ifcfg-eth0

DEVICE=eth0 设备名

TYPE=Ethernet 设备类型

UUID=52a6bbc3-917c-4828-8ab5-45369ab90d4b 设备唯一标识符

ONBOOT=yes 是否随系统启动而启动

NM_CONTROLLED=yes 是否受NetworkManager服务管理

#BOOTPROTO=dhcp dhcp 自动获取地址

static 静态设置IP地址

none 静态设置IP地址

DEFROUTE=yes

IPADDR=192.168.1.3 设置IP地址

NETMASK=255.255.255.0 子网掩码

GATEWAY=192.168.1.254 网关

DNS1=8.8.8.8 DNS服务器

NAME="System eth0"

PREFIX=24 设置网络前缀长度

HWADDR=00:0C:29:C1:5D:CF 设置MAC地址

设置以后要重启服务才能生效：

service network restart

设置DNS地址：

1、在网卡的配置文件中设置DNS1=8.8.8.8

2、编辑/etc/resolv.conf 更改以后立马生效

nameserver 202.106.0.20