Red Hat Linux Networking

Fundamental Network Configuration and Troubleshooting

Pete Nesbitt
March 2006

The information presented should act as a guide to Red Hat Linux networking. It is intended to be accompanied with training and self-study. To access most of these items you will need to have root access, either directly or through 'sudo su -', or via sudo exploits we won't discuss:-) Some items are left for you to review while others are covered in more detail on the following pages.

The descriptions are brief and are meant to be starting points. Use 'man <command or filename>' for help pages. Be aware that man pages expect a certain level of technical knowledge and may require several visits to grasp the information. Making use of multiple windows when working on a system, be it local or remote, allows you to view a man page while trying the command or editing the file in a different window.

Before this information can be useful, you should be able to use the following tools:

- *vi*
  - file editor common an all Linux and Unix systems
- *cat*
  - concatenate and view files
- *less*
  - file pager with many features
- *grep*
  - extract information based on keywords
- *man*
  - -use '*man -k <word>*' to search man pages for a keyword, or try '*man man*'
  - always look at the 'see also' section near the bottom
- basic command line navigation, command execution, and output manipulation
- understand environmental variables
  - it is valuable to understand system variables, although they may not come into play often
- bash shell scripting
  - should be able to at least follow the flow and process of a bash script
- basic network service clients such as ssh and ftp
  - use to test to local host, for confirming if it is a service problem or a network problem
- *tee*
  - use to send output to the monitor as well as a file for later review

Red Hat Linux Networking

Fundamental Network Configuration and Troubleshooting

Pete Nesbitt
March 2006

Configuration Files:

- /etc/hosts
  - used primarily to define static IP to hostname mappings.
- /etc/resolv.conf
  - defines name servers for DNS resolution.
- /etc/host.conf
  - defines parameters, the most common being search order, for name resolution. The *order* keyword may include *hosts*, *bind*, and *nis*.
- /etc/sysconfig/network
  - defines network settings common to all interfaces such as *hostname* and *default gateway*
- /etc/sysconfig/network-scripts/
  - ifcfg-lo ifcfg-eth*n*
  - define the interface parameters such as *IP address* and *network mask*
- /etc/services
  - defines known *port numbers* to *service names*. Commands, services, and utilities often display the service name in their output. Ports undefined in /etc/services should always be defined by *port number*.
- /etc/rc.local
  - commands to run after the init scripts. You may want a static routing entry in here after the network is set up.
- /proc pseudo-filesystem
  - allows you to read and in some cases write kernel variables.
  - look in /proc/sys/net/ for starters. Files show zero size but are populated when accessed.
- /etc/sysctl.conf
  - file you can use to control /proc file entries. This allows changes to survive a reboot.
- /etc/sysconfig/hwconf
  - information on system hardware used for auto-detection of new hardware
  - can edit to force re-detection if troubleshooting a network card. File is rarely of interest.
- /etc/modprobe.conf
  - nic drivers and options are defined here

Access Control Files:

- /etc/hosts.allow /etc/hosts.deny
  - controls TCPwrappers utility (see *tcpd* below)
  - view *man page* with '*man 5 hosts_access*' [note the 5]
- /etc/init.d/iptables (alias of /etc/rc.d/init.d/iptables)
  - control and configuration of Iptables/NetFilter Linux Firewall

Log Files:

Logs can provide valuable troubleshooting information but are too often overlooked.
dmesg falls into the logs category and should be reviewed, particularly if there are local problems as opposed to remote host problems.

- look in /var/log/
- to view startup messages use '*dmesg|less*'

Commands and Utilities:

- arp
  - display or set arp table entries
- dig
  - query name servers
- dmesg
  - display ring buffer information from system startup
- ethereal
  - not installed by default
- ethtool
- host
  - query name servers
- hostname
  - display or set hostname
  - changes do not survive a reboot
- ifup
  - bring an interface up
- ifdown
  - shut down an interface
- ifconfig

  - view or set interface parameters
  - changes do not survive a reboot
- ip
- iptables
  - user space tools to manipulate netfilter kernel based firewall
- iptraf
  - displays network statistics
  - not installed by default
- netstat
  - use to display routing, open (listening) ports, etc
- ping
  - send a n icmp packet to an IP address or hostname
- route
  - display or manipulate routing tables
  - changes do not survive a reboot
- service
  - use to start, stop or restart services via supported init scripts
- tcpd (tcpwrappers)
  - access control for supported services
- tcpdump
  - view network traffic
- telinit (and runlevel)
  - use to change runlevels, use *runlevel* to view current and previous runlevel
  - not usually involved in networking
- traceroute
  - displays the route and statistics for UDP or ICMP packets as they traverse a network.

Network Services:

Starting and stopping of network services can be automated through:

- init scripts
  - symlinks to various scripts used to manage most system daemons
- /etc/rc.local file
  - commands to run after the init scripts. You may want a static routing entry in here after the

network is set up.

- xinetd system
  - see /etc/xinetd.conf and files in /etc/xinetd.d/
  - replaces inetd and incorporates TCPwrapers

Environmental Variables:

These commands are for the bash shell, the default shell for most Linux distributions including Red Hat. Other shells may use different methods.

- View current settings
  - display with '*env*' for all or '*echo $<var-name>'*
- Define new or change existing variables
  - set variables with '*<var-name>=”<value>”*'

Virtual Network Interfaces:

- A single interface can easily be configured to answer multiple IP addresses. Virtual Interfaces are defined as eth*n*:*m* as are the configuration files. (0:0 is the physical nic, 0:1 and higher are virtual, any unique number can be used.
- Create a configuration file /etc/sysconfig/network-scripts/ifcfg-eth*n*:*m* then restart the network or use ifconfig to bring up the new interface.
- When working with iptables be aware it does not distinguish between real and virtual interfaces
- A Virtual Interface configuration file may look like:

```
DEVICE=eth0:1
BOOTPROTO=static
BROADCAST=192.168.1.255
IPADDR=192.168.1.41
NETWORK=192.168.1.0
ONBOOT=yes
```

Troubleshooting Steps:

These are very general steps used for isolating basic connectivity problems.

- ping 127.0.0.1
  - if fail here, network service may not be running. Start network, check logs and runlevel
- ping <localhost ip addr>
  - if fails, confirm IP address and interface state with *ifconfig* command

- ping <default gw ip>
  - if fails, try other local hosts and confirm physical cabling, check arp table
- ping <default gw hostname>
  - if first failure here, look at name resolution. Try *dig* or *host* and *ping* the name server.
- ping <destination hostname>
  - try another remote hostname, check name resolution, try pinging IP address
- traceroute <destination hostname>
  - this will show each step of the trip and indicate where along the path it fails
- tcpdump
  - view or save network traffic.
  - Learn to use the filters such as '*tcpdump -i eth1 host 192.168.1.254 and not port ssh*'

Common Tasks:

- change the systems hostname
  - use '*hostname <newname.domain.com>*' to set the hostname for the running system.
  - To change the hostname permanently, edit /etc/sysconfig/network and restart the network (or combine the two methods)
- change an interface IP address
  - temporarily set the IP address via chkconfig or ip commands.
  - For a permanent change, edit etc/sysconfig/network-scripts/ifcfg-eth*n* and restart the network (or combine the two methods)

More Detail on Commands and Files :

- **ifcfg-eth*n* files**
  - The directory /etc/sysconfig/network-scripts contains, among other related files, the configuration files for each physical and virtual interface.
  - The configuration entries are pretty self-explanatory.
    Typical entries are shown below (not all entries are always required):
    DHCP:

    DEVICE=eth0
    BOOTPROTO=dhcp
    HWADDR=00:11:D8:44:12:19
    ONBOOT=yes

TYPE=Ethernet

Static IP:

DEVICE=eth0
BOOTPROTO=static
NETWORK=192.168.1.0
BROADCAST=192.168.1.255
IPADDR=192.168.1.40
HWADDR=00:11:D8:44:12:19
ONBOOT=yes
TYPE=Ethernet

- If you want to disable or save a copy of an ifcfg file, you should remove it from the directory since a simple appending of _orig (or something similar) will still leave it as a target for the network services to use. Depending on other details this may harmlessly fail or may cause much disruption when the network services are (re)started.

- **ifconfig command**
  - This command will allow you to view or temporarily modify many aspects of the network settings. With no arguments '*ifconfig*' displays information on all active interfaces. Information includes IP, broadcast, network mask, and throughput information amongst other things.
  - You can change much of the network setup with this command. It is typically used to either change something temporarily (e.g. for testing) or to bring a single interface up or down, which can be done with ifup and ifdown using less typing:). One benefit of using ifconfig to change an IP address is that you do not need to start and stop the network. You can even change the MAC address using this command.
- **ip command**
  - Very similar to *ifconfig* this command can display or manipulate network settings. Although some information overlaps these two commands, '*ip*' deals with IP, network address, routing and ARP information.
  - The different output formats can work for you when creating scripts to report or manipulate a network. A prime example is how the IP address and Network Mask are displayed:

    # ifconfig eth0|grep "inet a"

```
    inet addr:192.168.1.40  Bcast:192.168.1.255  Mask:255.255.255.0
# ip addr show dev eth0|grep "inet "
    inet 192.168.1.40/24 brd 192.168.1.255 scope global eth0
```

iptables utility

- The Linux kernel based Firewall. We do not typically use the Linux Firewall but you should know how to check it. The firewall is comprised of NetFilter, the heart of the system which is part of the Linux kernel, and IPtables which is the user space tools and configurations you use to control the firewall. The firewall is always running and is controlled by rules (or chains) the user defines. In what is referred to as *stopped* the firewall is actually running but has no blocking rules. You can display the current state of the firewall with the command '*iptables -L*'. You can flush all rules and in essence '*stop*' a firewall with the command '*service iptables stop*' (see *service* command detailed below).
- A non-blocking firewall will display the following:

```
[root@cosmo /]# iptables -L
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
[root@cosmo /]#
```

- **tcpdump command**
  - Displays packet headers from an interface. This command offers many ways to manipulate the display. The most common is to add simple filters to the command line.
  - A typical remote monitor of web traffic might look like this (wrapped lines are indented):

```
[root@d207-216-142-87 ~]# tcpdump -i eth1 port http
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 96 bytes
16:35:38.976614 IP d207-216-142-87.bchsia.telus.net.50436 > nfl.bodog.com.http: S
    1104743985:1104743985(0) win 5840 <mss 1460,sackOK,timestamp 307804218
    0,nop,wscale 2>
```

16:35:38.986239 IP nfl.bodog.com.http > d207-216-142-87.bchsia.telus.net.50436: S 78647379:78647379(0) ack 1104743986 win 5792 <mss 1380,sackOK,timestamp 744177594 307804218,nop,wscale 0>
16:35:38.986447 IP d207-216-142-87.bchsia.telus.net.50436 > nfl.bodog.com.http: . ack 1 win 1460 <nop,nop,timestamp 307804221 744177594>
16:35:38.989257 IP d207-216-142-87.bchsia.telus.net.50436 > nfl.bodog.com.http: P 1:462(461) ack 1 win 1460 <nop,nop,timestamp 307804222 744177594>
16:35:39.004945 IP nfl.bodog.com.http > d207-216-142-87.bchsia.telus.net.50436: . ack 462 win 6432 <nop,nop,timestamp 744177596 307804222>

- When logged in to a remote system via ssh and want to run tcpdump on the same remote interface, always include a '*not ssh*' filter or every bit of tcpdump info sent to your display will be caught by tcpdump causing a massive ssh loop.
- You should be aware that tcpdump captures packets at the interface level and iptables handles are passed through to the network IP stack, so incoming traffic is captured by tcpdump prior to iptables rules being applied but outbound traffic is passed through iptables rules on it's way to the interface.

- **network file**
  - The file /etc/sysconfig/network defines network settings common to all interfaces.
  - A typical file may look like this:

    NETWORKING=yes
    HOSTNAME=nova3.linux1.ca
    GATEWAY=192.168.101.1
    NOZEROCONF=yes
  - The '*NOZEROCONF=yes*' removes an unnecessary routing entry.
- **chkconfig command**
  - This command is used to simplify the management of service daemons. It works in conjunction with the '*service*' command. It sets what runlevels a services will be started and stopped at.
  - Any init script that contains the required lines can be managed by chkconfig.
  - The following two keyword based entries allow an init script to fall under the control of the *chkconfig* command:

    # chkconfig: 2345 08 92
    # description: Automates a packet filtering firewall with iptables.

Use '*man chkconfig'* for details about the init script entries as well as how to add, delete, or modify a services behavior.

- The most common use, '*chkconfig –list*' (two dashes), is to display what services will be running at runlevel 3 and 5.
- You cannot change the current state of a daemon with this command. Use *service* instead.

- **service command**
  - Where chkconfig is used to report or set the planed states of daemons, 'service' is used to check or change the state of a service daemon.
  - For services that use the 'case' statement, the service commands is used to start, stop, restart or get the status of a daemon. If you drop the action argument the valid choices are displayed.
  - Typical usage, say to troubleshoot flaky ssh connection, could look something like this:

```
[root@cosmo /]# service sshd
Usage: /etc/init.d/sshd {start|stop|restart|reload|condrestart|status}
[root@cosmo /]#
[root@cosmo /]# service sshd status
sshd (pid 23710) is running...
[root@cosmo /]#
[root@cosmo /]# service sshd restart
Stopping sshd:                                             [  OK  ]
Starting sshd:                                             [  OK  ]
[root@cosmo /]#
```

  - See the comment under “Notes and Tips” regarding restarting a remote network.
- **ps command**
  - Although this is a very basic command, I want to touch on it's importance. When troubleshooting network services, you need to confirm the daemon is actually running. It is possible for the *service* command to report a daemon is running when in fact it has crashed and left behind PID and lock files.
  - This command has many options but for network troubleshooting you don't need to know them all, however it would be prudent to spend some time and get comfortable with some of the output formats.
  - A simple check to see if java is running could be '*ps -e|grep java*'
  - See the item under “Notes and Tips” for some further actions

Red Hat Linux Networking

Fundamental Network Configuration and Troubleshooting

Pete Nesbitt
March 2006

Note and Tips:

- When working on a remote machine, always use '*service network restart*' never try and stop then start the network in two steps or you will be disconnected. Same thing with ssh.
- When using tcpdump remotely, be sure to filter out '*ssh*' packets. If you wonder why, try ssh'ing into a remote system and run '*tcpdump*' with no arguments.
- If IP moves from one system to another it will confuse some layer 4 switches, ping out from new machine to tell switches where you are. Otherwise, the new system will not be available until the switch flushes it's tables.
- If ps does not show a process running or 'netstat -l' does not show a listening port, but service reports it is running or reports a related error, you may need to remove old PID or lock files. These are located in /var/run/ and /var/lock/subsys/ respectably.
- If ps reports a daemon has a PID but it is not functioning or will not stop or start, you may need to kill the defunct process. To do this, see the kill and killall man pages.

EOF