Linux学习之CentOS--Linux网卡高级命令、IP别名及多网卡绑定

本篇随笔将详细讲解Linux系统的网卡高级命令、IP别名以及Linux下多网卡绑定的知识

**一、网卡高级命令**

在之前的一篇随笔里[Linux学习之CentOS(九)--Linux系统的网络环境配置](http://www.cnblogs.com/xiaoluo501395377/archive/2013/04/05/3000859.html)，详细讲解了Linux系统下的网络环境配置等知识，我们了解了一些关于网络配置的一些基本命令。在这里将补充一些Linux系统下有关网卡的一些高级命令。

①**mii-tool** 命令

mii-tool命令我们可以用来查看网卡状态信息，包括了以太网连接是否正常，使用的是哪种型号的网卡等等

[root@xiaoluo ~]# mii-tool eth0

eth0: no autonegotiation, 100baseTx-FD, link ok

我们从输出的信息可以看到，当前以太网连接正常，以太网的速率是100M每秒，通过这个命令我们可以看到网卡底层的一些信息，当有时我们在上层通过ping、traceroute等命令进行网络排错时，如果发现网络还是不能连接畅通，这时就可以使用这个命令来查看是否是以太网连接出问题了

②**ethtool** 命令

通过mii-tool命令可以查看网卡的一些底层信息，但是如果我们想要查看网卡的更详细信息，就可以使用ethtool 这个命令，它可以将网卡的所有信息都详细列出来

[root@xiaoluo ~]# ethtool eth0

Settings for eth0:

Supported ports: [ TP ]

Supported link modes: 10baseT/Half 10baseT/Full

100baseT/Half 100baseT/Full

1000baseT/Full

Supported pause frame use: No

Supports auto-negotiation: Yes

Advertised link modes: 10baseT/Half 10baseT/Full

100baseT/Half 100baseT/Full

1000baseT/Full

Advertised pause frame use: No

Advertised auto-negotiation: Yes

Speed: 1000Mb/s

Duplex: Full

Port: Twisted Pair

PHYAD: 0

Transceiver: internal

Auto-negotiation: on

MDI-X: Unknown

Supports Wake-on: umbg

Wake-on: d

Current message level: 0x00000007 (7)

drv probe link

Link detected: yes

包括网卡支持的链接速率、双工模式等等信息都能查看到

如果说我们想要查看网卡的驱动信息，可以在 ethtool 命令后加上 -i 参数，就能显示网卡的驱动信息了

[root@xiaoluo ~]# ethtool -i eth0

driver: e1000

version: 7.3.21-k8-NAPI

firmware-version:

bus-info: 0000:00:03.0

supports-statistics: yes

supports-test: yes

supports-eeprom-access: yes

supports-register-dump: yes

supports-priv-flags: no

如果说我们还需要查看网卡当前底层工作状态信息，可以在命令后加上 -S 参数

[root@xiaoluo ~]# ethtool -S eth0

NIC statistics:

rx\_packets: 3255

tx\_packets: 898

rx\_bytes: 477575

tx\_bytes: 2228309

rx\_broadcast: 2190

tx\_broadcast: 198

rx\_multicast: 0

tx\_multicast: 6

rx\_errors: 0

tx\_errors: 0

tx\_dropped: 0

multicast: 0

collisions: 0

rx\_length\_errors: 0

rx\_over\_errors: 0

rx\_crc\_errors: 0

rx\_frame\_errors: 0

rx\_no\_buffer\_count: 0

rx\_missed\_errors: 0

tx\_aborted\_errors: 0

tx\_carrier\_errors: 0

tx\_fifo\_errors: 0

tx\_heartbeat\_errors: 0

tx\_window\_errors: 0

tx\_abort\_late\_coll: 0

tx\_deferred\_ok: 0

tx\_single\_coll\_ok: 0

tx\_multi\_coll\_ok: 0

tx\_timeout\_count: 0

tx\_restart\_queue: 0

rx\_long\_length\_errors: 0

rx\_short\_length\_errors: 0

rx\_align\_errors: 0

tx\_tcp\_seg\_good: 263

tx\_tcp\_seg\_failed: 0

rx\_flow\_control\_xon: 0

rx\_flow\_control\_xoff: 0

tx\_flow\_control\_xon: 0

tx\_flow\_control\_xoff: 0

rx\_long\_byte\_count: 477575

rx\_csum\_offload\_good: 0

rx\_csum\_offload\_errors: 0

alloc\_rx\_buff\_failed: 0

tx\_smbus: 0

rx\_smbus: 0

dropped\_smbus: 0

包括网卡发送了多少数据包、接受了多少数据包等信息都能看到。

所以通过 mii-tool 和 ethtool 这两个命令，我们可以看到网卡更多的底层信息以及网卡的物理信息等等，这可能会为我们在进行网络排错时提供便利

**二、IP别名**

在Linux系统中，我们可以在一块物理网卡上配置多个IP地址，以此来实现类似子接口的功能，我们称之为IP别名。当然在其他操作系统上，比如windowns、mac os上也支持在一块物理网卡配置多个IP地址。比如说我们的Linux主机作为一个DHCP服务器，它可能要为不同的网段分配多个IP地址，在Linux系统中，我们通过IP别名来实现在一块物理网卡上配置多个IP地址。

【**注意：**】**要使用IP别名来配置多个IP地址，我们首先需要关闭NetworkManager服务**

在CentOS或者RHEL上，都默认会启用NerworkManager这个服务来对网卡进行管理，NetworkManager服务其实就是我们操作系统右上方那两个小电脑图标，我们在图形化界面可以点击那个来指定使用哪块网卡，但是我们如果需要使用IP别名，就必须要禁用掉这个服务

[root@xiaoluo ~]# service NetworkManager stop

Stopping NetworkManager daemon: [ OK ]　　// 停掉我们的NetworkManager服务，此时我们可以发现右上角那个图标就不见了

[root@xiaoluo ~]# chkconfig NetworkManager off　　//　　NetworkManager这个服务默认会开机启动，所以我们将其设置成开机不启动

[root@xiaoluo ~]# chkconfig --list | grep NetworkManager

NetworkManager 0:off 1:off 2:off 3:off 4:off 5:off 6:off

在停掉NetworkManager服务以后我们就可以使用IP别名技术来配置多个IP地址了，这里我们使用 **ip** 这个命令，命令语法格式如下：

ip addr add 172.25.215.40/24 dev etho label eth0:0  
  
**后面的eth0:0 表示，我们给eth0这块网卡增加一个IP别名，后面那个0表示别名号，第二个别名就可以写成 eth0:1**

我们首先通过 ifconfig 命令来查看我们当前网卡的信息

[root@xiaoluo ~]# ifconfig

eth0 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

inet addr:172.25.215.40 Bcast:172.25.215.255 Mask:255.255.255.0

inet6 addr: fe80::a00:27ff:fed7:f884/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:5801 errors:0 dropped:0 overruns:0 frame:0

TX packets:2011 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:819876 (800.6 KiB) TX bytes:2288399 (2.1 MiB)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:16436 Metric:1

RX packets:167 errors:0 dropped:0 overruns:0 frame:0

TX packets:167 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:14861 (14.5 KiB) TX bytes:14861 (14.5 KiB)

此时我们的eth0网卡信息就会显示出来，我们此时通过ip命令来给eth0添加一个IP别名

[root@xiaoluo ~]# ip addr add 172.25.215.31/24 dev eth0 label eth0:0

[root@xiaoluo ~]# ifconfig

eth0 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

inet addr:172.25.215.40 Bcast:172.25.215.255 Mask:255.255.255.0

inet6 addr: fe80::a00:27ff:fed7:f884/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:6323 errors:0 dropped:0 overruns:0 frame:0

TX packets:2011 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:914440 (893.0 KiB) TX bytes:2288399 (2.1 MiB)

eth0:0 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

inet addr:172.25.215.31 Bcast:0.0.0.0 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:16436 Metric:1

RX packets:167 errors:0 dropped:0 overruns:0 frame:0

TX packets:167 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:14861 (14.5 KiB) TX bytes:14861 (14.5 KiB)

我们看到，此时我们对出来了一块 eth0:0 网卡，其实这个就是eth0网卡的一个IP别名，eth0上就有两个IP地址了，此时我们通过外面的主机ping这两个IP地址都是可以ping通的，如果说我们需要永久保存这个信息，我们就需要在 **/etc/sysconfig/network-scripts** 这个文件下建立一个名字为 ifcfg.eth0:0 这样的配置文件，新建的IP别名是什么，那么就在这个目录下建立一个与该名字对应的文件即可

[root@xiaoluo ~]# cd /etc/sysconfig/network-scripts/

[root@xiaoluo network-scripts]# ls

ifcfg-eth0 ifdown-isdn ifup-aliases ifup-plusb init.ipv6-global

ifcfg-lo ifdown-post ifup-bnep ifup-post net.hotplug

ifdown ifdown-ppp ifup-eth ifup-ppp network-functions

ifdown-bnep ifdown-routes ifup-ippp ifup-routes network-functions-ipv6

ifdown-eth ifdown-sit ifup-ipv6 ifup-sit

ifdown-ippp ifdown-tunnel ifup-isdn ifup-tunnel

ifdown-ipv6 ifup ifup-plip ifup-wireless

[root@xiaoluo network-scripts]# vi ifcfg-eth0:0

DEVICE=eth0:0　　　　// 设备名字

IPADDR=172.25.215.31　　//　　IP地址  
PREFIX=24　　// 前缀是24，其实就是相当于子网掩码有24位

ONPARENT=yes　　//　　eth0:0是eth0的IP别名，yes表示附属于eth0

通过添加上述的配置文件，我们的IP别名信息就能永久保存了。

**三、多网卡绑定**

为什么要使用多网卡绑定技术呢？我们知道如果使用一块网卡进行网络数据传输时其速度是有限的，即使是千兆网卡，其速度也是有限制的，我们为了提高带宽，我们可以将多块物理网卡绑定成一块逻辑网卡，这样网卡的速率就是多块网卡的速率了。

Linux支持将多块物理网卡绑定成一块逻辑网卡，绑定后的逻辑网卡可以并行使用组成其的所有物理网卡，通过这样的方式可以提高带宽以及网路的稳定性

我们在将多块物理网卡绑定成一块逻辑网卡以后，我们的IP地址就要求配置在这块逻辑网卡上了，而非多块物理网卡

Linux下支持三种模式的网卡绑定：

**①模式0**：**平衡轮训**　　使用这种模式来进行多网卡绑定时我们可以提高网络的带宽，其流量是从绑定的多块网卡上平均分配的

**②模式1**：**主动备份**　　使用这种模式来进行多网卡绑定时我们可以提高网络的稳定性，这种模式不会提高网络的带宽，每次只有一块网卡在走流量，只有当这块网卡发生故障时，绑定在一起的其它物理网卡才会工作

**③模式3**：**广播模式**　　这种模式一般都不用

我们如果想通过多网卡绑定来提升网络的带宽，就选择模式0，如果想提高网络的稳定性，则选择模式1

下面我们来看看如何进行网卡绑定的配置

①网卡绑定以后我们需要给我们的逻辑网卡取一个名字，通常格式为 bondn，n代表的是编号，如 /dev/bond0, /dev/bond1

②然后我们需要在 /etc/sysconfig/network-scripts 这个目录下创建该逻辑网卡的配置文件，如：

/etc/sysconfig/network-scripts/ifc-bond0

[root@xiaoluo network-scripts]# vi ifcfg-bond0

DEVICE=bond0　　//　　设备名字

IPADDR=172.25.215.200　　//　　网卡ip地址

PREFIX=24　　//　　子网掩码24位

ONBOOT=yes　　//　　开机启动yes

BOOTPROTO=none　　//　　启动协议，因为我们是手动配置IP，所以设置成none，如果是通过DHCP动态分配，则设置成dhcp

USERCTL=no　　//　　关闭掉用户控制

BONDING\_OPTS="mode=1 miimon=5"　　//　　**这个是bond网卡最主要的配置，我们的bond网卡其实是根据一个叫做bonding的驱动来配置的，后面跟配置参数，这里我们使用的是模式1(主动备份)**

因为我们要使用多网卡绑定，所以我们这里再添加一块网卡，然后根据提示就可以添加一块新网卡了，我们这里添加一块网卡eth1，使用ifconfig查看一下：

[root@xiaoluo network-scripts]# ifconfig

bond0 Link encap:Ethernet HWaddr 00:00:00:00:00:00

inet addr:172.25.215.200 Bcast:172.25.215.255 Mask:255.255.255.0

UP BROADCAST RUNNING MASTER MULTICAST MTU:1500 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

eth0 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

inet6 addr: fe80::a00:27ff:fed7:f884/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:1285 errors:0 dropped:0 overruns:0 frame:0

TX packets:6 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:221706 (216.5 KiB) TX bytes:468 (468.0 b)

eth1 Link encap:Ethernet HWaddr 08:00:27:F4:7D:CC

inet6 addr: fe80::a00:27ff:fef4:7dcc/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:1196 errors:0 dropped:0 overruns:0 frame:0

TX packets:95 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:216126 (211.0 KiB) TX bytes:4806 (4.6 KiB)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:16436 Metric:1

RX packets:73 errors:0 dropped:0 overruns:0 frame:0

TX packets:73 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:6330 (6.1 KiB) TX bytes:6330 (6.1 KiB)

此时我们计算机里面就有两块网卡了，因为我们需要将这两块网卡绑定到bond0这个逻辑网卡中，所以我们要在两块网卡的配置文件里面添加两条语句：

MASTER=bond0　　// 将这两块网卡绑定到bond0这块逻辑网卡上

SLAVE=yes　　//　　SLAVE=yes表示这两块网卡附属于bond0这块网卡

[复制代码](javascript:void(0);)

[root@xiaoluo network-scripts]# vi ifcfg-eth0

DEVICE=eth0

HWADDR=08:00:27:D7:F8:84

TYPE=Ethernet

UUID=2877e96e-6ddd-443b-a337-bafa7c77c6e5

ONBOOT=no

NM\_CONTROLLED=yes

BOOTPROTO=none

IPV6INIT=no

USERCTL=no

MASTER=bond0

SLAVE=yes

[root@xiaoluo network-scripts]# vi ifcfg-eth1

DEVICE=eth1

BOOTPROTO=none

TYPE=Ethernet

PEERDNS=yes

IPV6INIT=no

ONBOOT=no

USERCTL=no

HWADDR=08:00:27:f4:7d:cc

MASTER=bond0

SLAVE=yes

之前说过bond0这块网卡其实是通过一个叫做 bonding 的驱动来配置的，所以我们还需要为bond0这块网卡添加驱动支持，将这个驱动添加到 /etc/modprobe.d/ 这个目录下

[root@xiaoluo network-scripts]# cd /etc/modprobe.d/

[root@xiaoluo modprobe.d]# ls

anaconda.conf blacklist-visor.conf dist.conf openfwwf.conf

blacklist.conf dist-alsa.conf dist-oss.conf

这个文件夹下放的都是我们系统所需驱动的配置文件，我们这时需要给我们的bonding添加一个名字为 bonding.conf 的配置文件

[root@xiaoluo modprobe.d]# vi bonding.conf

alias bond0 bonding

此时配置好以后，我们就可以通过 ifup bond0 来启动我们的bond0网卡

[root@xiaoluo network-scripts]# ifup bond0

Active connection state: activated

Active connection path: /org/freedesktop/NetworkManager/ActiveConnection/2

此时我们可以通过ifconfig命令来查看一下当前网卡信息：

[root@xiaoluo network-scripts]# ifconfig

bond0 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

inet addr:172.25.215.200 Bcast:172.25.215.255 Mask:255.255.255.0

inet6 addr: fe80::a00:27ff:fed7:f884/64 Scope:Link

UP BROADCAST RUNNING MASTER MULTICAST MTU:1500 Metric:1

RX packets:8175 errors:0 dropped:0 overruns:0 frame:0

TX packets:117 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:1417884 (1.3 MiB) TX bytes:6230 (6.0 KiB)

eth0 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

UP BROADCAST RUNNING SLAVE MULTICAST MTU:1500 Metric:1

RX packets:4131 errors:0 dropped:0 overruns:0 frame:0

TX packets:22 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:711693 (695.0 KiB) TX bytes:1424 (1.3 KiB)

eth1 Link encap:Ethernet HWaddr 08:00:27:D7:F8:84

UP BROADCAST RUNNING SLAVE MULTICAST MTU:1500 Metric:1

RX packets:4044 errors:0 dropped:0 overruns:0 frame:0

TX packets:95 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:706191 (689.6 KiB) TX bytes:4806 (4.6 KiB)

lo Link encap:Local Loopback

inet addr:127.0.0.1 Mask:255.0.0.0

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:16436 Metric:1

RX packets:81 errors:0 dropped:0 overruns:0 frame:0

TX packets:81 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:7226 (7.0 KiB) TX bytes:7226 (7.0 KiB)

此时我们通过 ping 命令来ping我们外面的主机，发现已经可以正常ping通的

[root@xiaoluo network-scripts]# ping 172.25.215.25

PING 172.25.215.25 (172.25.215.25) 56(84) bytes of data.

64 bytes from 172.25.215.25: icmp\_seq=1 ttl=64 time=1.78 ms

64 bytes from 172.25.215.25: icmp\_seq=2 ttl=64 time=1.10 ms

64 bytes from 172.25.215.25: icmp\_seq=3 ttl=64 time=1.08 ms

64 bytes from 172.25.215.25: icmp\_seq=4 ttl=64 time=0.811 ms

64 bytes from 172.25.215.25: icmp\_seq=5 ttl=64 time=1.17 ms  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

这个时候我们的bond0网卡就已经配置成功了，它是由两块物理网卡组成的，我们可以根据需要，比如说要提高网络带宽或者提高网络稳定性来设置bond0的网卡绑定模式即可。