**Linux Basics: Create Network Bonding On CentOS 6.5**

by SK

Please share

**What is network bonding?**

**Network bonding** is a method of combining (joining) two or more network interfaces together into a single interface. It will increase the network throughput, bandwidth and give redundancy. If one interface is down or unplugged, the other one will keep the network traffic alive. Network bonding can be used in situations wherever you need redundancy, fault tolerance or load balancing networks.

Linux allows us to bond multiple network interfaces into single interface using a special kernel module named **bonding**. The **Linux bonding driver** provides a method for combining multiple network interfaces into a single logical **“bonded”** interface. The behavior of the bonded interfaces depends upon the mode; generally speaking, modes provide either hot standby or load balancing services. Additionally, link integrity monitoring, may be performed.

**Types of Network Bonding**

According the to the official documentation, here is the types of network bonding modes.

**mode=0** (balance-rr)

**Round-robin policy:** It the default mode. It transmits packets in sequential order from the first available slave through the last. This mode provides load balancing and fault tolerance.

**mode=1** (active-backup)

**Active-backup policy:** In this mode, only one slave in the bond is active. The other one will become active, only when the active slave fails. The bond’s MAC address is externally visible on only one port (network adapter) to avoid confusing the switch. This mode provides fault tolerance.

**mode=2** (balance-xor)

**XOR policy:** Transmit based on [(source MAC address XOR'd with destination MAC address) modulo slave count]. This selects the same slave for each destination MAC address. This mode provides load balancing and fault tolerance.

**mode=3** (broadcast)

**Broadcast policy:** transmits everything on all slave interfaces. This mode provides fault tolerance.

**mode=4** (802.3ad)

IEEE 802.3ad Dynamic link aggregation. Creates aggregation groups that share the same speed and duplex settings. Utilizes all slaves in the active aggregator according to the 802.3ad specification.

Prerequisites:

- Ethtool support in the base drivers for retrieving the speed and duplex of each slave.  
- A switch that supports IEEE 802.3ad Dynamic link aggregation. Most switches will require some type of configuration to enable 802.3ad mode.

**mode=5** (balance-tlb)

**Adaptive transmit load balancing:** channel bonding that does not require any special switch support. The outgoing traffic is distributed according to the current load (computed relative to the speed) on each slave. Incoming traffic is received by the current slave. If the receiving slave fails, another slave takes over the MAC address of the failed receiving slave.

Prerequisite:

- Ethtool support in the base drivers for retrieving the speed of each slave.

**mode=6** (balance-alb)

**Adaptive load balancing:** includes balance-tlb plus receive load balancing (rlb) for IPV4 traffic, and does not require any special switch support. The receive load balancing is achieved by ARP negotiation. The bonding driver intercepts the ARP Replies sent by the local system on their way out and overwrites the source hardware address with the unique hardware address of one of the slaves in the bond such that different peers use different hardware addresses for the server.

**Setup Network Bonding On CentOS / RHEL / Scientific Linux 6.5**

In this handy tutorial let us see how to setup network bonding on CentOS 6.5. Though it was tested on CentOS, it should work on RHEL and Scientific Linux 6.x versions. I have three network interfaces, namely **eth0**, **eth1** and **eth2** in my CentOS 6.5 system. Let us combine two NICs (**eth1** and **eth2**) and make them into one NIC named **bond0**.

**Configure Bond0 Interface**

First, let us create a **bond0** configuration file as shown below.

Go to the directory where CentOS stores the network configuration files. By default RHEL and its clones such as CentOS, Scientific Linux stores the network configuration files under **/etc/sysconfig/network-scripts/** directory.

Create **bond0** configuration file under the above mentioned directory.

# vi /etc/sysconfig/network-scripts/ifcfg-bond1

Add the following lines.

DEVICE=bond1

BOOTPROTO=none

ONBOOT=yes

IPADDR=10.144.40.70

NETWORK=10.144.40.0

NETMASK=255.255.255.192

USERCTL=no

BONDING\_OPTS="mode=1 miimon=100"

**Note:** Here **BONDING\_OPTS** describes the bonding mode. In our case, we will be configuring **mode1**(active-backup). Save and close file. **192.168.1.200** is bond0 IP address.

Next we have to load up the bond0 interface into the kernel. To do that, create a new file **/etc/modprobe.d/bonding.conf**,

# vi /etc/modprobe.d/bonding.conf

Add the following line in it.

alias bond1 bonding

Save and close the file.

**Configure Network interfaces**

Now we should modify both(**eth1 & eth2**) configuration files as shown below. First, let us start from **eth1**.

Edit file **/etc/sysconfig/network-scripts/ifcfg-eth1**,

# vi /etc/sysconfig/network-scripts/ifcfg-ens3f0

Modify the file as shown below.

DEVICE=ens3f0

MASTER=bond1

SLAVE=yes

USERCTL=no

ONBOOT=yes

BOOTPROTO=none

Then Edit file **/etc/sysconfig/network-scripts/ifcfg-eth2**,

# vi /etc/sysconfig/network-scripts/ifcfg-ens3f1

Modify the file as shown below.

DEVICE=eth2

MASTER=bond0

SLAVE=yes

USERCTL=no

ONBOOT=yes

BOOTPROTO=none

Save and close the files.

Enter the following command to load the bonding module.

# modprobe bonding

Restart network service to take effect the changes.

# service network restart

**Test Network Bonding**

Now enter the following command to check whether the bonding interface bond0 is up and running:

# cat /proc/net/bonding/bond0

Sample output:

Ethernet Channel Bonding Driver: v3.6.0 (September 26, 2009)

Bonding Mode: fault-tolerance (active-backup)

Primary Slave: None

Currently Active Slave: eth1

MII Status: up

MII Polling Interval (ms): 100

Up Delay (ms): 0

Down Delay (ms): 0

Slave Interface: eth1

MII Status: up

Speed: 1000 Mbps

Duplex: full

Link Failure Count: 0

Permanent HW addr: 08:00:27:fe:6f:bf

Slave queue ID: 0

Slave Interface: eth2

MII Status: up

Speed: 1000 Mbps

Duplex: full

Link Failure Count: 0

Permanent HW addr: 08:00:27:34:17:c0

Slave queue ID: 0

As you see in the above output, the bond0 interface is up and running and it is configured as **active-backup(mode1)** mode. In this mode, only one slave in the bond is active. The other one will become active, only when the active slave fails.

To view the list of network interfaces and their IP address, enter the following command:

# ifconfig

Sample output:

bond0     Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          inet addr:192.168.1.200  Bcast:192.168.1.255  Mask:255.255.255.0

          inet6 addr: fe80::a00:27ff:fefe:6fbf/64 Scope:Link

          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1

          RX packets:379 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:32354 (31.5 KiB)  TX bytes:24078 (23.5 KiB)

eth0      Link encap:Ethernet  HWaddr 08:00:27:BE:25:49

          inet addr:192.168.1.101  Bcast:192.168.1.255  Mask:255.255.255.0

          inet6 addr: fe80::a00:27ff:febe:2549/64 Scope:Link

          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

          RX packets:1402 errors:0 dropped:0 overruns:0 frame:0

          TX packets:904 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:1000

          RX bytes:134823 (131.6 KiB)  TX bytes:124938 (122.0 KiB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1

          RX packets:285 errors:0 dropped:0 overruns:0 frame:0

          TX packets:156 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:1000

          RX bytes:24746 (24.1 KiB)  TX bytes:22956 (22.4 KiB)

eth2      Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1

          RX packets:95 errors:0 dropped:0 overruns:0 frame:0

          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:1000

          RX bytes:7674 (7.4 KiB)  TX bytes:1364 (1.3 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: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)

As per the above output, **bond0** is configured as **master**; **eth1** and **eth2** are configured as a **slave**.

**Configure multiple IP addresses for bond0**

I want to assign multiple IP addresses to bond0 interface. What should i do? Very simple, just create an alias for the bond0 interface and assign multiple IP addresses.

Let me make it more clear. Say for example we want to assign IP address **192.168.1.201** to **bond0**. To create an alias for bond0, copy the existing configuration file(**ifcfg-bond0**) to a new configuration file(**ifcfg-bond0:1**).

# cp /etc/sysconfig/network-scripts/ifcfg-bond0 /etc/sysconfig/network-scripts/ifcfg-bond0:1

Then edit the alias file **/etc/sysconfig/network-scripts/ifcfg-bond0:1**,

# vi /etc/sysconfig/network-scripts/ifcfg-bond0:1

Modify the device name and IP address as shown below.

DEVICE=bond0:1

BOOTPROTO=none

ONBOOT=yes

IPADDR=192.168.1.201

NETWORK=192.168.1.0

NETMASK=255.255.255.0

USERCTL=no

BONDING\_OPTS="mode=1 miimon=100"

Save and close the file. Restart network service to take effect the saved changes.

# service network restart

Now list out the network interfaces and their IP address using the command:

# ifconfig

Sample output:

bond0     Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          inet addr:192.168.1.200  Bcast:192.168.1.255  Mask:255.255.255.0

          inet6 addr: fe80::a00:27ff:fefe:6fbf/64 Scope:Link

          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1

          RX packets:1048 errors:0 dropped:0 overruns:0 frame:0

          TX packets:590 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:0

          RX bytes:88622 (86.5 KiB)  TX bytes:84340 (82.3 KiB)

bond0:1   Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          inet addr:192.168.1.201  Bcast:192.168.1.255  Mask:255.255.255.0

          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1

eth0      Link encap:Ethernet  HWaddr 08:00:27:BE:25:49

          inet addr:192.168.1.101  Bcast:192.168.1.255  Mask:255.255.255.0

          inet6 addr: fe80::a00:27ff:febe:2549/64 Scope:Link

          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1

          RX packets:1422 errors:0 dropped:0 overruns:0 frame:0

          TX packets:916 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:1000

          RX bytes:136317 (133.1 KiB)  TX bytes:126150 (123.1 KiB)

eth1      Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1

          RX packets:942 errors:0 dropped:0 overruns:0 frame:0

          TX packets:581 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:1000

          RX bytes:80036 (78.1 KiB)  TX bytes:84266 (82.2 KiB)

eth2      Link encap:Ethernet  HWaddr 08:00:27:FE:6F:BF

          UP BROADCAST RUNNING SLAVE MULTICAST  MTU:1500  Metric:1

          RX packets:111 errors:0 dropped:0 overruns:0 frame:0

          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0

          collisions:0 txqueuelen:1000

          RX bytes:8916 (8.7 KiB)  TX bytes:1492 (1.4 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: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)

You should see the alias bond0:1 has been created and up.