By Chris Stuttler
Published: 2007-02-13 18:48

How to Set up Network Bonding in Ubuntu 6.10 Why you may want to do this:

Network Bonding, otherwise known as port trunking allows you to combine multiple network ports into a single group, effectively aggregating the bandwidth of multiple interfaces into a single connection. For example, you can aggregate two gigabyte ports into a two-gigabyte trunk port. Bonding is used primarily to provide network load balancing and fault tolerance. First, we will run two different network tools to check for network connectivity and capability. Run mii-tool to check your interfaces for connectivity:

```
mii-tool
```

For our purposes, we will assume you have three interfaces. The result of the mii-tool command is listed below:

```
eth0: negotiated 100baseTx-HD, link ok
eth1: negotiated 100baseTx-HD, link ok
eth2: negotiated 100baseTx-HD, link ok
```

Next run ethtool for each interface to check to see what capabilities:

```
ethtool eth0 && ethtool eth1 && ethtool eth3
```

The result of the ethtool command is listed below:

```
Settings for eth0:
        Supported ports: [ TP MII ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Supports auto-negotiation: Yes
```

```
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
        Advertised auto-negotiation: Yes
        Speed: 100Mb/s
        Duplex: Half
        Port: MII
        PHYAD: 1
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: g
        Wake-on: g
        Current message level: 0x00000007 (7)
        Link detected: yes
Settings for eth1:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised auto-negotiation: Yes
        Speed: Unknown! (65535)
        Duplex: Unknown! (255)
        Port: Twisted Pair
        PHYAD: 0
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: umbg
        Wake-on: d
        Current message level: 0x00000007 (7)
        Link detected: no
```

```
Settings for eth3:
        Supported ports: [ TP ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Advertised auto-negotiation: Yes
        Speed: Unknown! (65535)
        Duplex: Unknown! (255)
        Port: Twisted Pair
        PHYAD: 0
        Transceiver: internal
        Auto-negotiation: on
        Supports Wake-on: umbg
        Wake-on: d
        Current message level: 0x00000007 (7)
        Link detected: no
```

Next, we need to install ifenslave. It(TM)s a simple install:

```
apt-get update && apt-get install ifenslave
```

Options for mode types:

You can set up your bond interface according to your needs. In order to do this, you simply change the mode type depicted in the examples below (mode=X). There are seven mode types available. They are as follows:

mode=0

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

mode=1

This mode uses an Active-backup policy: Only one slave in the bond is active. A different slave becomes active if, and only if, 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. The primary option affects the behavior of this mode.

mode=2

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

mode=4

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.

*Pre-requisites:

1. Ethtool support in the base drivers for retrieving the speed and duplex of each slave.

2. 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

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

Adaptive load balancing: includes balance-transmit load balancing plus receive load balancing 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.

Now append the following items to your aliases file:

```
pico /etc/modprob.d/aliases
```

```
# Append to the bottom of this file:
alias bond0 bonding
alias eth0 e100
alias eth1 e100
alias eth2 e100
options bonding mode=0 miimon=100
```

Next, append the following items to your i386 file:

```
pico /etc/modprob.d/arch/i386
```

```
# Append to the bottom of this file:
alias bond0 bonding
options bonding mode=0 miimon=100 downdelay=200 updelay=200
```

Now we have to modify the interface file. Start off by commenting out any information on the physical interfaces, eth0, eth1, etc, and create a virtual interface such as bond0, configure it similar to below, and be sure to choose a unique hwaddress. Be sure to leave the loopback interface configuration intact.

```
pico /etc/network/interfaces
```

It should look something like this:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
# The loopback network interface
auto lo
iface lo inet loopback
# The primary network interface
#auto eth0
#iface eth0 inet static
#   address 192.168.0.120
#   netmask 255.255.255.0
#   network 192.168.0.0
#   broadcast 192.168.0.255
#   gateway 192.168.0.1
auto bond0
```

```
iface bond0 inet static
  address 192.168.0.120
  netmask 255.255.255.0
  network 192.168.0.0
  broadcast 192.168.0.255
  gateway 192.168.0.1
  hwaddress ether 00:03:B3:48:50:2C
  post-up ifenslave bond0 eth0 eth1
```

Save the file and then reboot the system:

```
shutdown -r now
```