UbuntuBonding

Bonding, also called **port trunking** or **link aggregation** means combining several network interfaces (NICs) to a single link, providing either high-availability, load-balancing, maximum throughput, or a combination of these. See Wikipedia for details.

目录

Installation

```
sudo apt-get install ifenslave
```

Interface Configuration

Step 1: Ensure kernel support

Before Ubuntu can configure your network cards into a NIC bond, you need to ensure that the correct kernel module `bonding` is present, and loaded at boot time.

Edit your /etc/modules configuration:

```
sudo vi /etc/modules
```

Ensure that the `bonding` module is loaded:

```
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.

loop
lp
rtc
bonding
```

Step 2: Configure network interfaces

Ensure that your network is brought down:

```
sudo stop networking
```

Then load the bonding kernel module:

```
sudo modprobe bonding
```

Now you are ready to configure your NICs.

A general guideline is to:

1. Pick which available NICs will be part of the bond.
2. Configure all other NICs as usual
3. Configure all bonded NICs:
    1. To be manually configured
    2. To join the named bond-master
4. Configure the bond NIC as if it were a normal NIC
5. Add bonding-specific parameters to the bond NIC as follows.

Edit your interfaces configuration:

```
 sudo vi /etc/network/interfaces
```

For example, to combine eth0 and eth1 as slaves to the bonding interface bond0 using a simple active-backup setup, with eth0 being the primary interface:

```
#eth0 is manually configured, and slave to the "bond0" bonded NIC
auto eth0
iface eth0 inet manual
bond-master bond0
bond-primary eth0

#eth1 ditto, thus creating a 2-link bond.
auto eth1
iface eth1 inet manual
bond-master bond0

# bond0 is the bonding NIC and can be used like any other normal NIC.
# bond0 is configured using static network information.
auto bond0
iface bond0 inet static
address 192.168.1.10
gateway 192.168.1.1
netmask 255.255.255.0
bond-mode active-backup
bond-miimon 100
bond-slaves none
```

The `bond-primary` directive, if needed, needs to be part of the slave description (eth0 in the example), instead of the master. Otherwise it will be ignored.

As another example, to combine eth0 and eth1 using the IEEE 802.3ad LACP bonding protocol:

```
#eth0 is manually configured, and slave to the "bond0" bonded NIC
auto eth0
iface eth0 inet manual
bond-master bond0

#eth1 ditto, thus creating a 2-link bond.
auto eth1
iface eth1 inet manual
bond-master bond0

# bond0 is the bonded NIC and can be used like any other normal NIC.
# bond0 is configured using static network information.
```

```
auto bond0
iface bond0 inet static
address 192.168.1.10
gateway 192.168.1.1
netmask 255.255.255.0
# bond0 uses standard IEEE 802.3ad LACP bonding protocol
bond-mode 4
bond-miimon 100
bond-lacp-rate 1
bond-slaves eth0 eth1
```

For `bonding`-specific networking options please consult the documentation available at BondingModuleDocumentation.

Finally, bring up your network again:

```
sudo start networking
```

Checking the bonding interface

Link information is available under `/proc/net/bonding/`. To check `bond0` for example:

```
# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v3.5.0 (November 4, 2008)

Bonding Mode: IEEE 802.3ad Dynamic link aggregation
Transmit Hash Policy: layer2 (0)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0

802.3ad info
LACP rate: fast
Aggregator selection policy (ad_select): stable
bond bond0 has no active aggregator

Slave Interface: eth1
MII Status: up
Link Failure Count: 0
Permanent HW addr: 00:0c:29:f5:b7:11
Aggregator ID: N/A

Slave Interface: eth2
MII Status: up
Link Failure Count: 0
Permanent HW addr: 00:0c:29:f5:b7:1b
Aggregator ID: N/A
```

Bringing up/down bonding interface

To bring the bonding interface, run

```
ifup bond0
```

To bring down a bonding interface, run

```
ifdown bond0
```

Ethernet Bonding modes

Ethernet bonding has different modes you can use. You specify the mode for your bonding interface in `/etc/network/interfaces`. For example:

```
bond-mode active-backup
```

Descriptions of bonding modes

Mode 0

balance-rr

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

active-backup

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

balance-xor

XOR policy: Transmit based on selectable hashing algorithm. The default policy is a simple source+destination MAC address algorithm. Alternate transmit policies may be selected via the xmit_hash_policy option, described below. 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:

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

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.

Prerequisites:

- 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.

Descriptions of balancing algorithm modes

The balancing algorithm is set with the **xmit_hash_policy** option.

Possible values are:

layer2 Uses XOR of hardware MAC addresses to generate the hash. This algorithm will place all traffic to a particular network peer on the same slave.

layer2+3 Uses XOR of hardware MAC addresses and IP addresses to generate the hash. This algorithm will place all traffic to a particular network peer on the same slave.

layer3+4 This policy uses upper layer protocol information, when available, to generate the hash. This allows for traffic to a particular network peer to span multiple slaves, although a single connection will not span multiple slaves.

encap2+3 This policy uses the same formula as layer2+3 but it relies on skb_flow_dissect to obtain the header fields which might result in the use of inner headers if an encapsulation protocol is used.

encap3+4 This policy uses the same formula as layer3+4 but it relies on skb_flow_dissect to obtain the header fields which might result in the use of inner headers if an encapsulation protocol is used.

The default value is layer2. This option was added in bonding version 2.6.3. In earlier versions of bonding, this parameter does not exist, and the layer2 policy is the only policy. The layer2+3 value was added for bonding version 3.2.2.

CategoryDocumentation CategoryDocumentation

UbuntuBonding (2015-09-15 15:21:15由bryanquigley @ c-50-182-214-227.hsd1.nj.comcast.net[50.182.214.227]:bryanquigley编辑)