

Microsoft Network Load Balancing and Cisco Catalyst Configuration

OVERVIEW	2
UNICAST MODE.....	2
MULTICAST MODE.....	2

OVERVIEW

Microsoft's Network Load Balancing software allows a collection of Windows boxes to share a virtual IP address and to distribute the work of handling incoming and outgoing traffic amongst the members of the cluster. Deploying this product has implications for nearby switches and routers; in this document, I outline the mechanics of configuring Catalyst gear to support this product.

UNICAST MODE

In Unicast Mode, the sys admin clicks the 'unicast' button in the MS NLB configuration GUI. This choice instructs the cluster members to respond to ARP queries for their virtual address using MAC address *abc* ... but to source the packets they emit using MAC address *xyz*. The local Ethernet switch will populate its port-to-MAC-address table (called a 'CAM' table or a 'mac-address-table' in Cisco-speak) using MAC address *xyz* ... and will never know which port connects to MAC address *abc*.

Thus, when local hosts, including the local router, want to send a packet to the virtual IP address, they will send that packet using MAC address *abc*. The switch won't know which port holds MAC address *abc* and will therefore flood the packet to all ports.

When this happens, the cluster gets what it wants: every member receives the incoming packet. The cluster members use some internal algorithm to determine which one of them will process the packet and, possibly, respond to it.

While the cluster is getting what it wants, every other station on this subnet (broadcast domain and VLAN) is also receiving this packet. I call this behavior *anti-social*, a term which I define specifically as meaning that the node is optimizing for its own benefit at the expense of its neighbors. As long as in-bound cluster traffic remains low, I have trouble imagining a scenario under which this behavior would disrupt service to the neighbors.

Nevertheless, I don't like taking these kinds of risks – designs which add a subnet-wide failure mode seem undesirable to me.

One way to restrict NLB's anti-social behavior is to assign the cluster members to their own VLAN, or even to a private VLAN. Another way is to insert static entries into the switch's mac-address-table. I leave these configurations as an exercise to the reader.

MULTICAST MODE

In Multicast Mode (aka IGMP Mode), the sys admin clicks the 'IGMP Multicast' button in the MS NLB configuration GUI. This choice instructs the cluster members to respond to ARPs for their virtual address using a multicast MAC address, call it *pdq* and to emit IGMP Membership Report packets.

If the local switch also speaks IGMP, it will then populate mac-address-table appropriately, associating the multicast MAC address *pdq* with each port feeding a cluster member. In this way, when a local station ARPs for the cluster's virtual IP address, the cluster will respond with *pdq*, the local station will address the packet to *pdq*, the local switch will forward the packet out each of the ports feeding cluster members, and everybody is happy: the cluster members get what they want (every single packet destined for the virtual IP address) and the neighbors don't have to hear about it. In our Catalyst-based environment, this approach works seamlessly: our Catalysts shipped with IGMP enabled, no need for configuration changes there.

However, cluster's virtual IP address becomes unreachable from outside the local subnet.

Poking around, I found that the local router (a Layer 3 switch: Catalyst 6500 w/Sup 720) was ARPing for the cluster's virtual IP address, the cluster members were replying ... but the Cat6500 didn't seem to hear. This ARP entry never appeared in its ARP table; it eventually quit ARPing and returned an ICMP Host unreachable to my test station (located outside the cluster's subnet).

Here's the story as I understand it. Microsoft thinks that associating a unicast IP address with a multicast MAC address is hunky-dory. Cisco thinks that doing this violates an RFC and therefore ignores ARP responses containing such a mapping. I poked briefly at some of the multicast RFCs and wasn't able to resolve the issue. If you can find a reference, drop me a note.

If you want override Cisco's default behavior, here's what you do:

```
arp 10.1.2.15 0100.5301.0200 ARPA
```

This static ARP entry populates the router's ARP table, the router doesn't bother to ARP but just constructs and forwards the packet. Connectivity restored.

However, you aren't out of the woods yet. Via IGMP, the Cat6500 is hearing from the cluster members about their use of multicast address *pdq*. In theory, this would allow the Cat6500 to populate its mac-address-table. And it does. But, curiously enough, the Cat6500 ignores this entry and process-switches each cluster-bound packet. I'm unclear why, but perhaps again it considers a unicast IP address to multicast MAC address mapping to be invalid. This lead to high CPU utilization on our Cat6500 (spikes to 100% during the day). So, I inserted a static mac-address-table entry, and the Cat6500 started switching cluster-bound packets in hardware once more.

```
mac-address-table static 0100.5301.0200 vlan 2 interface GigabitEthernet3/1 disable-snooping
```

The 'disable-snooping' parameter is essential; without it, the statement does not affect behavior.

In this example:

Cluster virtual IP address:	10.1.2.15
Cluster multicast MAC address:	0100.5301.0200
VLAN on which cluster is located:	2
Interface servicing VLAN 2:	GigabitEthernet3/1

ANALYSIS

CPU Utilization

To identify major contributors to CPU utilization:

```
Router#show processes cpu | exclude 0.00
CPU utilization for five seconds: 91%/50%; one minute: 89%; five minutes: 47%
  PID Runtime(ms)   Invoked    uSecs   5Sec   1Min   5Min  TTY Process
    5      881160      79142     11133  0.49%  0.19%  0.16%  0 Check heaps
   98      121064     3020704         40 40.53% 38.67% 20.59%  0 IP Input
  245      209336      894828         233  0.08%  0.05%  0.02%  0 IFCOM Msg Hdlr
```

[Cribbed from

http://www.cisco.com/en/US/customer/products/hw/switches/ps708/products_tech_note09186a00804916e0.shtml, a link which also describes, in greater detail that I display below, how to sniff on Route Processor traffic.]

The 'IP Input' line indicates that the Route Processor is spending much of its time process-switching transit packets.

Capture Packets

To watch your C6K process-switch packets, attach a sniffer to a 'shutdown' interface and configure a SPAN port to forward Route Processor traffic to that shutdown interface. In the following example, Gi3/15 is an administratively 'shutdown' interface, and the sniffer is plugged into Gi3/16.

```
Router# config t
Router(config)# monitor session 1 source interface Gi3/15
Router(config)# monitor session 1 destination interface Gi3/16
Router(config)# exit
Router# remote login switch
Trying Switch ...
Entering CONSOLE for Switch
Type "^C^C^C" to end this session
```

```
Router-sp#test monitor add 1 rp-inband both
Router-sp#test monitor show session 1
Session [1] Info
-----
asic_session    = 0
session_type    = 0
session_status  = 0
dst_ltl_idx     = 0x10000
decap_index     = -1
```

Source Port-VLAN Info

Ingress Source Ports: 3/15 15/1
Egress Source Ports : 3/15 15/1
Ingress Source Vlans: <null>
Egress Source Vlans : <null>
Ingress Filter Vlans : <null>
Egress Filter Vlans : <null>
Exclude Filter Vlans : <empty>
Exclude Alt Filter Vlans : <empty>
Ingress Filter Vlan Count: 0
Egress Filter Vlan Count : 0
Exclude Filter Vlan Count: 0
Exclude Alt Vlan Count : 0

Destination ports: 3/16

Router-sp#