Subject: Re: [Fwd:  Complex routing and bridging with OpenVZ]
Posted by den on Thu, 21 Feb 2008 08:14:26 GMT
View Forum Message <> Reply to Message

OpenVz should support this configuration.

Do I understand correctly that you seen ICMP echo request in VE2 on eth0
and do not see in eth1? If this is true, you should have some counter
visible via 'netstat -s'.

If this will not help, pls notify me. I will check this in more details.

Regards,
   Den

On Thu, 2008-02-21 at 10:32 +0300, Pavel Emelyanov wrote:
>
> -------- Original Message --------
> Subject: [Users] Complex routing and bridging with OpenVZ
> Date: Wed, 20 Feb 2008 16:28:32 +0000
> From: Rob Wilson <roobert@gmail.com>
> Reply-To: users@openvz.org
> To: users@openvz.org
>
> Hi,
>
> Our ultimate goal is to have a machine simulating various different
> real world network environments. Initially we want to simulate:
>
>  * NAT/masquerading configurations (e.g. full cone / symmetric / etc.)
>
>  * Network load/quality simulation (e.g. packet loss, latency, etc.)
>
>
> Initially we have a configuration that is composed of the host machine
> and 3 contexts (this would be expanded later) with 2 virtual ethernet
> interface each. The virtual interfaces are called eth0/eth1 from the
> context's perspective, and veth<N>.0/veth<N>.1 from the host's
> perspective.
>
> The contexts' interfaces are then configured such that they form a
> chain of subnets with each interface in each VE sharing a subnet
> withone other VE or the host machine.
>
> Something like this:
>
> +-----HOST------+ +-----VE2-------+ +-----VE3-------+ +---VE4--------+
> |         .1.1 |-|  .1.2 / .2.2  |-|  .2.3 / .3.3  |-| .3.4 / .4.4  |

> +---------------+ +---------------+ +---------------+ +--------------+
>
> The veth endpoints for VE2.1/VE3.0 and VE3.1/VE4.0 are connected via
> the Linux bridging code. The routing tables are configured
> accordingly: e.g. HOST has routes for networks 192.168.[234] via VE2,
> and VE4 has routes to 192.168.1.0 via VE3, etc. IP forwarding is
> enabled in all the contexts and in the host machine.
>
>
> At this stage we would expect to be able to ping any of the machines
> on any of the bridges from the host machine and receive a ping reply,
> however somehow the ICMP echo is getting dropped between leaving VE2
> and entering thesecond bridge - tcpdump of the second bridge shows no
> traffic at all. However, when pinging from VE3 or VE4 to the HOST
> tcpdump shows the ICMP echo request reaches the HOST along the correct
> route (->VE3->VE2->HOST) and the echo REPLY makes it back as far as
> VE2 but gets dropped at the same point as before (When leaving VE2,
> onto the bridge).
>
> Our question is: are the OpenVZ network stack modifications complete
> enough to allow this kind of complicated setup? And if so, do you have
> any idea what is wrong with our configuration, that would cause
> thetraffic to be discarded after its 1nd hop?
>
> Below is the script which is run on HOST to setup the above configuration.
>
> Thanks
>
> #!/bin/bash
>
> #
> # Test script for OpenVZ bridging configuration
> #
>
> # Add a bridge for each pair of machines
> brctl addbr br0
> brctl addbr br1
> brctl addbr br2
>
> # disable icmp redirects
> echo 0 > /proc/sys/net/ipv4/conf/eth0/accept_redirects
> echo 0 > /proc/sys/net/ipv4/conf/eth0/send_redirects
>
> # recommended in bridge FAQ
> echo 0 > /proc/sys/net/bridge/bridge-nf-call-arptables
> echo 0 > /proc/sys/net/bridge/bridge-nf-call-iptables
> echo 0 > /proc/sys/net/bridge/bridge-nf-filter-vlan-tagged
>

>
> # Create OpenVZ instances and assign virtual ethernet devices
> sh /OpenVZ/bin/vz.sh create 4 # this creates VE2, VE3 and VE4
> vzctl set 2 --netif_add eth0,00:60:00:00:01:01,veth2.0,00:60:00:00:02:01 --save
> vzctl set 2 --netif_add eth1,00:60:00:00:01:02,veth2.1,00:60:00:00:02:02 --save
>
> vzctl set 3 --netif_add eth0,00:60:00:00:01:03,veth3.0,00:60:00:00:02:03 --save
> vzctl set 3 --netif_add eth1,00:60:00:00:01:04,veth3.1,00:60:00:00:02:04 --save
> vzctl set 4 --netif_add eth0,00:60:00:00:01:05,veth4.0,00:60:00:00:02:05 --save
>
> vzctl set 4 --netif_add eth1,00:60:00:00:01:06,veth4.1,00:60:00:00:02:06 --save
> sh /OpenVZ/bin/vz.sh start # this starts VE2, VE3 and VE4
>
> # Create the bridges
> # Host and VZ2
> # for [host:br0 vz2:eth0] bridge
> brctl addif br0 veth2.0
> # for [host:eth0 vz2:eth0] bridge
> #brctl addif br0 eth0 veth2.0
>
> # VZ2 and VZ3
> brctl addif br1 veth2.1 veth3.0
>
> # VZ3 and VZ4
> brctl addif br2 veth3.1 veth4.0
>
> # Bring the bridges up
> ip link set br0 up
> ip link set br1 up
> ip link set br2 up
>
> # Make sure all virtual ethernet devices are up
> ip link set veth2.0 up
> ip link set veth2.1 up
> ip link set veth3.0 up
> ip link set veth3.1 up
> ip link set veth4.0 up
> ip link set veth4.1 up
>
> # Give the Host an address on the same subnet as A and configure routing
> # for [host:br0 vz2:eth0] bridge
> ip a a 192.168.1.1/32 dev br0
> ip r a 192.168.1.0/24 dev br0
> ip r a 192.168.2.0/24 via 192.168.1.2 dev br0
> ip r a 192.168.3.0/24 via 192.168.1.2 dev br0
> # for [host:eth0 vz2:eth0:0] bridge
> #ip a a 192.168.1.1/32 dev eth0
> #ip r a 192.168.1.0/24 dev eth0

> #ip r a 192.168.2.0/24 via 192.168.1.2 dev eth0
> #ip r a 192.168.3.0/24 via 192.168.1.2 dev eth0
>
> # Configure VZ2
> vzctl exec 2 ip a a 192.168.1.2/32 dev eth0
> vzctl exec 2 ip a a 192.168.2.2/32 dev eth1
> vzctl exec 2 ip link set eth0 up
> vzctl exec 2 ip link set eth1 up
> vzctl exec 2 ip r a 192.168.1.0/24 dev eth0
> vzctl exec 2 ip r a 192.168.2.0/24 dev eth1
> vzctl exec 2 ip r a 192.168.3.0/24 via 192.168.2.3 dev eth1
> vzctl exec 2 ip r a 192.168.4.0/24 via 192.168.2.3 dev eth1
>
> # Configure VZ3
> vzctl exec 3 ip a a 192.168.2.3/32 dev eth0
> vzctl exec 3 ip a a 192.168.3.2/32 dev eth1
> vzctl exec 3 ip link set eth0 up
> vzctl exec 3 ip link set eth1 up
> vzctl exec 3 ip r a 192.168.2.0/24 dev eth0
> vzctl exec 3 ip r a 192.168.3.0/24 dev eth1
> vzctl exec 3 ip r a 192.168.1.0/24 via 192.168.2.2 dev eth0
> vzctl exec 3 ip r a 192.168.4.0/24 via 192.168.3.3 dev eth1
>
> # Configure VZ4
> vzctl exec 4 ip a a 192.168.3.3 dev eth0
> vzctl exec 4 ip a a 192.168.4.2 dev eth1
> vzctl exec 4 ip link set eth0 up
> vzctl exec 4 ip link set eth1 up
> vzctl exec 4 ip r a 192.168.3.0/24 dev eth0
> vzctl exec 4 ip r a 192.168.4.0/24 dev eth1
> vzctl exec 4 ip r a 192.168.1.0/24 via 192.168.3.2 dev eth0
> vzctl exec 4 ip r a 192.168.2.0/24 via 192.168.3.2 dev eth0