Subject: Re: [PATCHSET] 2.6.20-lxc8
Posted by Daniel Lezcano on Wed, 21 Mar 2007 23:04:07 GMT
View Forum Message <> Reply to Message

Eric W. Biederman wrote:
> Daniel Lezcano <dlezcano@fr.ibm.com> writes:
>
>
>> Hi Herbert,
>>
>> I played with the L2 namespace patchset from Eric Biederman, I did some
>> benchmarking with netperf:
>>
>> With 2 hosts, Intel EM64T bipro HT / 2,4 GHz , 4Go ram and GB network.
>> Host A is running the netserver on a RH4 kernel 2.6.9-42
>> Host B is running the netperf client inside and outside the container with the
>> command:
>> netperf -H HostA -c -l 20 -n 2 -p 12865
>>
>> Results are:
>> inside the container:
>>   Throughput : 940.39 Mbit/s CPU usage : 15.80 %
>>
>> outside the container:
>>   Throughput : 941.34 Mbits/s CPU usage : 5.80 %
>>
>
> Could you clarify a little bit what you mean by inside the container
> and outside the container? In particular was the outside the container case
> in a kernel that had the network namespace compiled in?
>
Sure.
Outside the container means : no container and netperf is ran into a
kernel with the network namespace compiled in.
Inside the container means : a network namespace is created with the
configuration described below and netperf is ran inside this network
namespace.
I ran a netperf on a vanilla kernel too and netperf results are roughly
the same than netperf ran on a network namespace kernel.

Anyway, I want to do some benchmarking on Dmitry's patchset and
Linux-Verser and I will come back with a more complete benchmarking
results in a few days.

> Could you also clarify how you have setup networking inside the container
> and going to the outside world?
>
Ok. Let's assume we have host A which is a RH4 kernel with netperf

server running and with an IP address 1.2.3.100
Host B is the kernel with network namespace compiled in. The objective is to be able to setup the container without doing extra network configuration on host A.
The physical network interface is eth0, we want to have IPs for etun pair-device 1.2.3.4/1.2.3.5, we have 2 shells A and B
So I did the following:

```
    (in shell A)
    # do proxy arp for eth0
    echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp

    # add in the init namespace IP address of the same network than the
containers
    # it is more comfortable. We can add a route -net otherwise
    ifconfig eth0:0 1.2.3.1

    # create the pair device
    echo etun1,etun2 > /sys/module/etun/parameters/newif

    # do proxy arp for etun1
    echo 1 > /proc/sys/net/ipv4/conf/etun1/proxy_arp

    # set ip address for etun1
    ifconfig etun1 1.2.3.4

    # route packet for etun2 IP to etun1
    route add -host 1.2.3.5 dev etun1

    (in shell B)
    # unshare network namespace

    (in shell A)
    # move etun2 to network namespace
    echo pid_of_shell_B > /sys/class/net/etun2/new_ns_pid

    (in shell B)
    # set the etun2 IP address
    ifconfig etun2 1.2.3.5

    # set the loopback up
   ifconfig lo up

    # check ping, 1.2.3.4, 1.2.3.5, 1.2.3.1, 1.2.3.100 is successful
    # run netperf or what ever you want
```

>> I did the test again with 50 containers. I created them one by one having one
>> running netperf and the other being idle.

>> Each time I created a container, I rerun netperf. To be more explicit, I created
>> 1 container, run netperf inside it and blocked it on a fifo reading, I created a
>> second container, run netperf inside it and blocked it, and son on ... to 50
>> containers. The benchmarking result are the same as running one container, so I
>> guess it scales well.
>>
>> There are a lot of scenarii to do for benchmarking, for example, running netperf
>> in each container in the same time and look how it behaves.
>> I am profiling the kernel to look where the cpu overhead is.
>>
>
> Thanks. This is simple enough that at least part of this should be easy
> to reproduce. Once things settle down a little I'm interested in tracking
> the cpu usage if someone else hasn't done it already.
>
Benjamin Thery and I we were looking at this.
For the moment we are investigating if there is IP fragmentation between
the eth0 and the pair devices.
The profiling shows us "pskb_expand_head" and
"csum_partial_copy_generic" functions have the bigger CPU usage when we
are inside a container.

Regards
   -- Daniel

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers