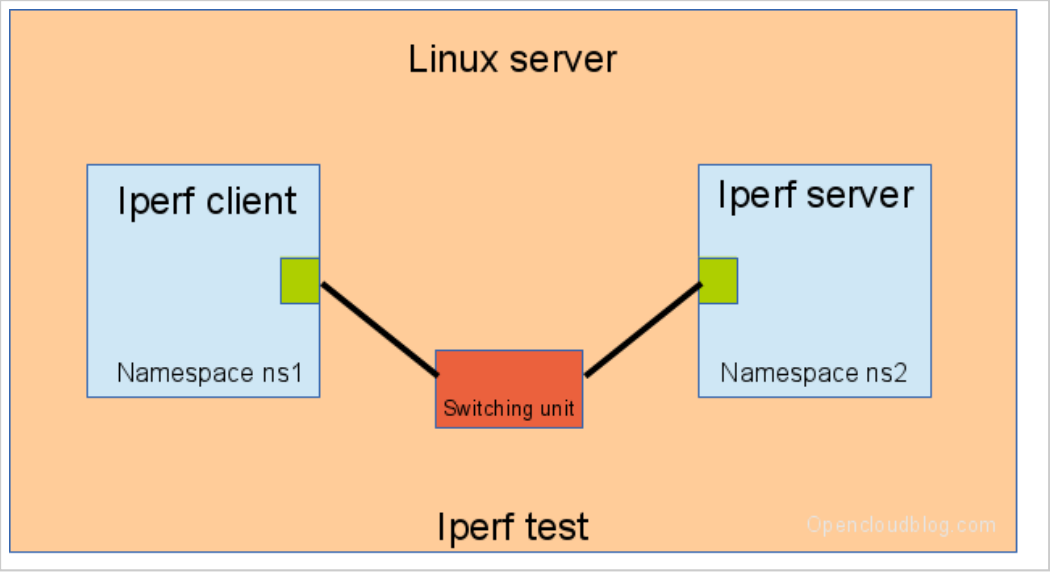
Switching Performance – Connecting Linux Network Namespaces

18/09/2013 Linux Networking, Openstack, Openvswitch

In a previous article I have shown several solutions to interconnect Linux network namespaces (http://www.opencloudblog.com). Four different solutions can be used – but which is the best solution with respect to performance and resource usage? This is quite interesting when you are running Openstack Networking Neutron together with the Openvswitch. The systems used to run these tests have Ubuntu 13.04 installed with kernel 3.8.0.30 and Openvswitch 1.9. The MTU of the IP interfaces is kept at the default value of 1500.

A test script has been used to collect performance numbers. The test script creates 2 namespaces, connects these namespaces using different Linux network technologies and measures the throughput and CPU usage using the software iperf. The iperf server process is started in one namespace, the client iperf process is started in the other namespace.

IMPORTANT REMARK: The results are confusing – a deeper analysis shows, that the configuration of the virtual network devices has a major impact on the performance. The settings TSO and GSO play a very important role when using network devices. I'll show an analyses in an upcoming article.



Perf test setup

Single CPU (i5-2550 [3.3 GHz]) Kernel 3.13

The test system has the Desktop CPU i5-2500 CPU @ 3.30GHz providing 4 CPU cores and 32 GByte DDR3-10600 RAM providing around 160 GBit/s RAM throughput.

The test system is running Ubuntu 14.04 with kernel 3.13.0.24 and Openvswitch 2.0.1

The results are shown in the table below. iperf has been running with one, two and four threads. At the end the limiting factor is CPU usage. The column „efficiency" is defined as network throughput in GBit/s per Gigahertz available on CPUs.

Switch and Connection type	no of iperf threads	throughput [GBit/s] tso gso lro gro on	Efficiency [GBit/s per CPUGHZ]	throughput [GBit/s] tso gso lro gro off
one veth pair	1	37.8	6.3	3.7
one veth pair	2	65.0	5.4	7.9
one veth pair	4	54.6	4.4	11.0
one veth pair	8	40.7	3.2	11.4
one veth pair	16	37.4	2.9	11.7
linuxbridge with two veth pairs	1	33.3	5.5	2.7
linuxbridge with two veth pairs	2	54.3	4.4	5.6
linuxbridge with two veth pairs	4	43.9	3.4	6.9
linuxbridge with two veth pairs	8	32.1	2.5	7.9
linuxbridge with two veth pairs	16	34.0	2.6	7.9
openvswitch with two veth pairs	1	35.0	5.9	3.2
openvswitch with two veth pairs	2	51.5	4.2	6.7
openvswitch with two veth pairs	4	47.3	3.8	8.7
openvswitch with two	8	36.0	2.8	7.5

veth pairs				
openvswitch with two veth pairs	16	36.5	2.8	9.4
openvswitch with two internal ovs ports	1	37.0	6.2	3.3
openvswitch with two internal ovs ports	2	65.6	5.5	6.4
openvswitch with two internal ovs ports	4	74.3	5.7	6.3
openvswitch with two interval ovs ports	8	74.3	5.7	10.9
openvswitch with two internal ovs ports	16	73.4	5.6	12.6

The numbers show an drastic effect, if TSO … are switches off. TSO perfformes the segmentation at the NIC level. In this software NIC only environment no segmentation is done by the NIC, the large packets (average 40 kBytes) are sent to the other side as one packet. The guiding factor is the packet packet rate.

Single CPU (i5-2550 [3.3 GHz]) Kernel 3.8

The test system is running Ubuntu 13.04 with kernel 3.8.0.30 and Openvswitch 1.9 .

The results are shown in the table below. iperf has been running with one, two and four threads. At the end the limiting factor is CPU usage. The column „efficiency“ is defined as network throughput in GBit/s per Gigahertz available on CPUs.

Switch and Connection type	**no of iperf threads**	**throughput [GBit/s]**	**Efficiency [GBit/s per CPUGHZ]**

one veth pair	1	7.4	1.21
one veth pair	2	13.5	1.15
one veth pair	4	14.2	1.14
one veth pair	8	15.3	1.17
one veth pair	16	14.0	1.06
linuxbridge with two veth pairs	1	3.9	0.62
linuxbridge with two veth pairs	2	8.5	0.70
linuxbridge with two veth pairs	4	8.8	0.69
linuxbridge with two veth pairs	8	9.5	0.72
linuxbridge with two veth pairs	16	9.1	0.69
openvswitch with two veth pairs	1	4.5	0.80
openvswitch with two veth pairs	2	9.7	0.82
openvswitch with two veth pairs	4	10.7	0.85
openvswitch with two veth pairs	8	11.3	0.86
openvswitch with two veth pairs	16	10.7	0.81
openvswitch with two internal ovs ports	1	41.9	6.91
openvswitch with two internal ovs ports	2	69.1	5.63
openvswitch with two internal ovs ports	4	75.5	5.74
openvswitch with two interval ovs	8	67.0	5.08

ports			
openvswitch with two internal ovs ports	16	74.3	5.63

The results show a huge differences. The openvswitch using two internal openvswitch ports has the best throughput and the best efficiency.

The short summary is:

- Use Openvswitch and Openvswitch internal ports – in the case of one iperf thread you get 6.9 GBit/s throughput per CPU Ghz. But this solution does not provide any iptables rules on the link.
- If you like the old linuxbridge and veth pairs you get only 0.7 GBit/s per CPU Ghz throughput. With this solution it's possible to filter the traffic on the network namespace links.

The table shows some interesting effects, e.g.:

- The test with the ovs and two ovs ports shows a drop in performance between 4 and 16 threads. The CPU analysis shows, that in the case of 8 threads, the CPU time used by softirqs doubled in comparison to the case of 4 threads. The softirq time used by 16 threads is the same as for 4 threads.

Openstack

If you are running Openstack Neutron, you should use the Openvswitch. Avoid linuxbridges. When connecting the Neutron networking Router/LBaas/DHCP namespaces DO NOT enable ovs_use_veth.

Updated: 26/04/2014 — 15:25