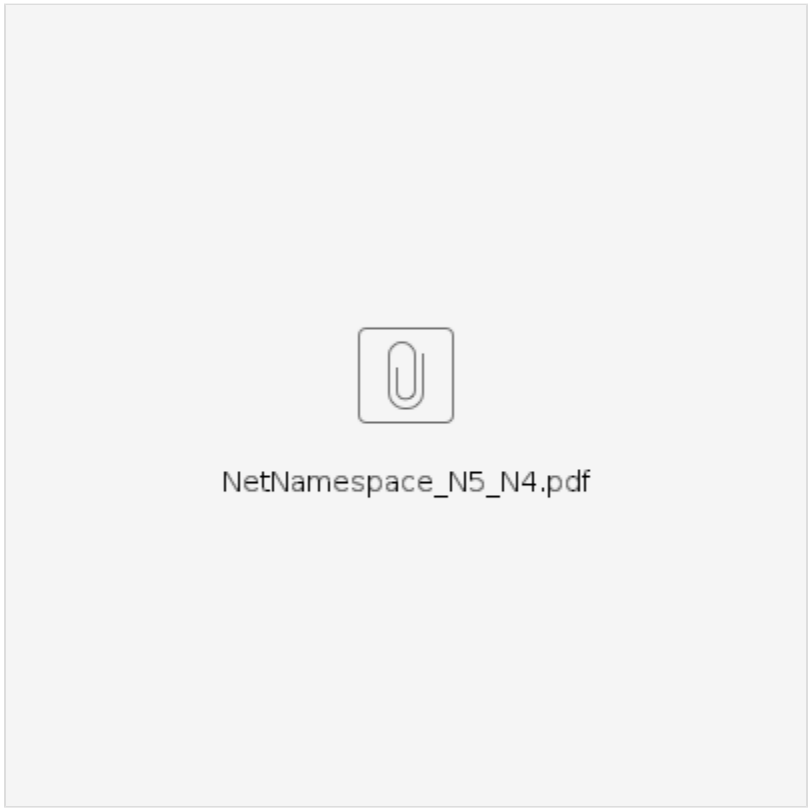
Alt.Speed: Alternative Tests and Tools Used at the Network "Edge"

Introduction

As VSPREF methods are applied to systems in Edge Cloud or Edge Compute use cases, we will begin to see **overlap** with methods that have been employed for Internet Access Speed and Performance testing. MANY methods have been used for Internet performance measurement, and there are some general approaches which were successful when access speeds were <50Mbps. But no widely accepted Standard for metrics and methods has emerged, and now the challenge for Internet methods is to make measurements when access speeds top >900Mbps, and Latency is also a critical benchmark for Edge technologies (5G and its Applications). VSPERF methods routinely work at very high speed and low latency. It is worthwhile for us to compare VSPERF methods and those that use TCP or state-full traffic with multiple simultaneous connections (to account for TCP flow control), because this comparison will be inevitably occur in deployed systems, and made by others without our benchmarking background. It is also valuable to **calibrate** our current methods against known capacity limits, which we can implement using tc qdisc on non-DPDK interfaces.

Development and Configuration

In order to use some different, non-DPDK test tools with Sender and Receiver on a single Node, we determined a Network Namespace Configuration that isolates Client and Server processes and assigns a single external interface NIC port for their use. This is necessary to prevent the host recognizing that both interface ports are local, and routing traffic through the kernel instead of the external ports. The Figure below illustrates the configuration used between Nodes 4 and 5 on Intel Pod 12.



Here is the working net-namespace configuration, and the lines to delete once done:

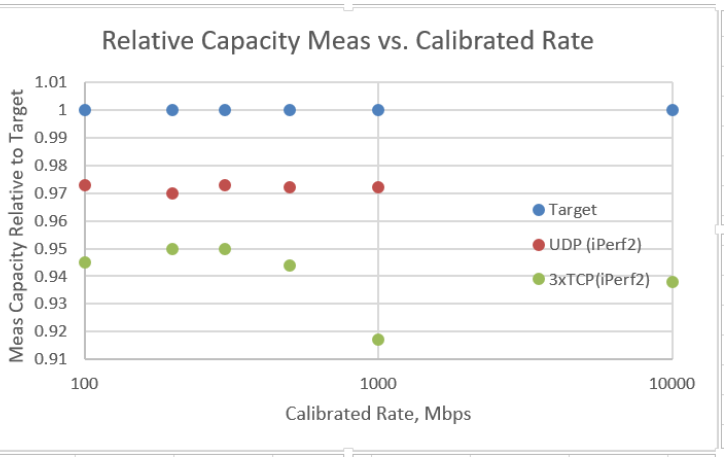
```
# Node 5 root
ip netns list
# Working NetNS config:
ip netns add iperf-server
ip netns add iperf-client
ip link set ens801f1 netns iperf-server
ip link set ens801f0 netns iperf-client
ip netns exec iperf-server ip link set dev lo up
ip netns exec iperf-client ip link set dev lo up
ip netns exec iperf-server ip addr add dev ens801f1 10.10.122.25/24
ip netns exec iperf-client ip addr add dev ens801f0 10.10.124.25/24
ip netns exec iperf-client ip link set dev ens801f0 up
ip netns exec iperf-server ip link set dev ens801f1 up
ip netns exec iperf-server route add default gw 10.10.122.25
ip netns exec iperf-client route add default gw 10.10.124.25

# Delete
```

```
ip netns delete iperf-server
ip netns delete iperf-client
```

Initial results for tests with this configuration (using the iPerf 2 tool) indicate that UDP testing is a more reliable assessment of the calibrated bit rate (after correcting for ETH, IP, and UDP header overhead, the values reported below are for protocol payloads of a single UDP stream or 3 TCP connections).

	C-Rate, M	Target	UDP (iPerf	3xTCP(iPe	Delay, ms
29-Nov	100	1	0.973	0.945	40
29-Nov	200	1	0.97	0.95	40
29-Nov	300	1	0.973	0.95	40
29-Nov	500	1	0.972	0.944	40
24-Jan	1000	1	0.972	0.917	20
11-Dec	10000	1		0.9379	0
			2-Jan		

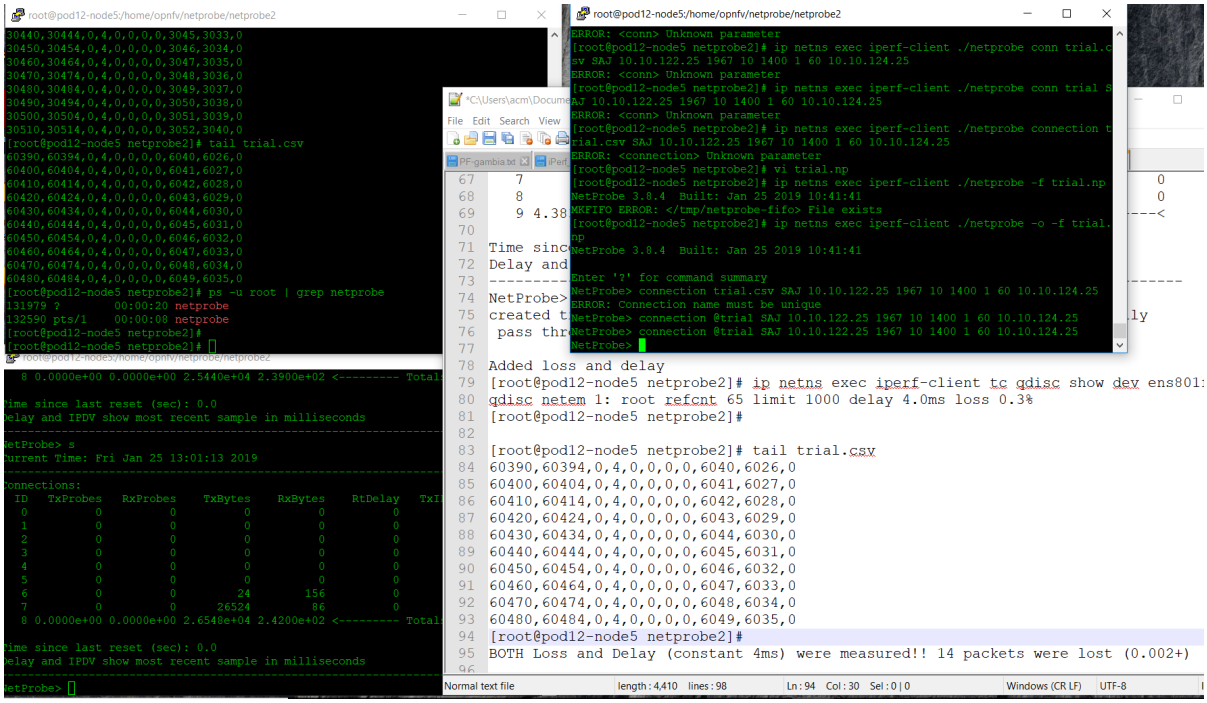


The value plotted for 3 TCP connections at 10000 Mbps uses no tc qdisc, just the 10GE links. A similar value was obtained for 5 TCP connections.

Recent testing (January 24 and 25), has produced two important findings:

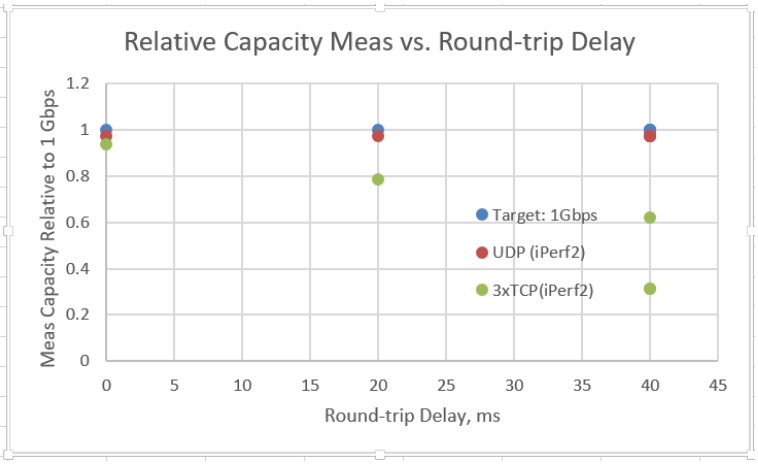
1. iPerf 2 has an unexpected dependency on the Units in the UDP -b Bandwidth #[KM] option configuration on the Client.
    a. the -b 972000000 and -b 972M options produce different sending rates. -b 972000000 seems to be correctly sending the target bandwidth.
2. Neither iPerf 2 or iPerf3 can measure one-way and RT delay, and delay is a both critical measurement metric and a demanding requirement of most Edge Applications.
    a. the measurement tool "netprobe" can measure both loss and delay, and can supply measurements on every packet received. netprobe architecture is similar to iPerf, in that individual test streams can be launched from the Client to the Server (running as a daemon, if desired). netprobe is installed on Node 5 /home/opnfv/netprobe/netprobe2

Here are some windows with netprobe in operation, showing results:



We have also conducted more tests investigating the effects of RT Delay on Measurement (an impairment which should only affect TCP, of course). This cross-section was conducted with a 1Gbps rate limit implemented in a Token bucket filter on the path to the iPerf 2 Server.

	Delay, ms	Target: 1G	UDP (iPerf	3xTCP(iPe	f2)
27-Jan	40	1	0.972	0.62	
27-Jan	40	1	0.972	0.312	
27-Jan	40	1	0.972	0.313	
27-Jan	40	1	0.972		
27-Jan	20	1	0.972	0.786	
11-Dec	0	1	0.972	0.9379	



Tests with concurrent iPerf 2 and Netprobe test streams have revealed the expected outcomes:

1. Parallel TCP stream measurements are very sensitive to the background traffic, with the peak capacity only reached after 10 seconds, and only then account for the CBR Netprobe rate.
2. UDP stream measurements indicate reduced rate equivalent to the CBR Netprobe stream.
3. One-way Delay measurements possible with Netprobe confirm the delay introduced by the Token bucket filter (4ms max), and illustrate the delay variability present when 3 TCP streams compete for capacity.