极速前进！DPDK GRO/GSO的转发性能提升实例

原创：王艺楠 DPDK与SPDK开源社区 2018-12-19

作者简介

王艺楠，软件测试工程师，主要从事DPDK虚拟化相关方向的测试工作。

通常，以太网的MTU是1500B，除去TCP/IP的协议首部，TCP的MSS(Max Segment Size)大小是1460B。一般情况下，协议栈会对超过1460B的TCP payload进行切片，保证生成的IP包不超过MTU的大小，但对于支持TSO的网卡，我们可以把最多64KB大小的TCP payload直接往下传给协议栈，此时IP层也不会进行segmentation，一直会传给网卡驱动，支持TSO的网卡会自己进行TCP切片，这样可以减少很多协议栈上的数据包处理数量。切片、checksum计算等原本靠CPU来做的工作都转移给了网卡，从而提高网络性能。相对应的，LRO(Large Receive Offload)是在接收方向上，通过将接收到的多个TCP数据聚合成一个大的数据包，然后传递给网络协议栈处理，以减少上层协议栈处理的开销。当然，如果都是小包，那么功能基本就作用不大了。

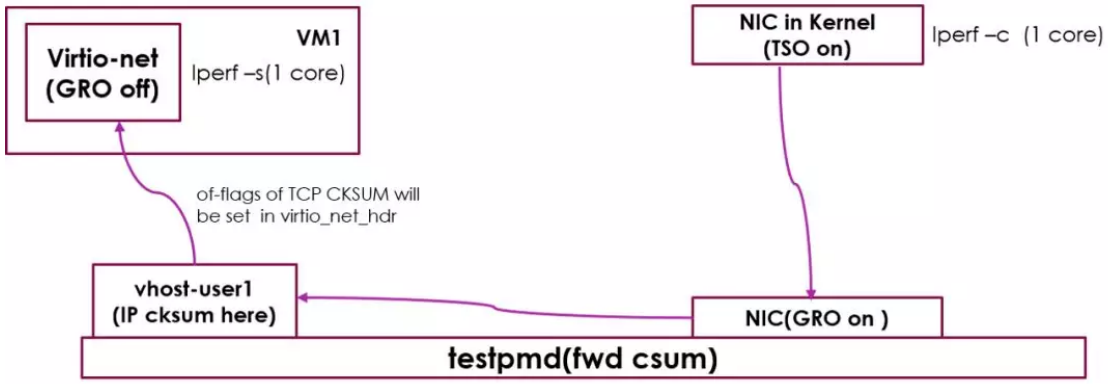
TCP Segmentation Offload （ TSO ），UDP Fragmentation Offload （ UFO ）和Large Receive Offload （ LRO ）技术基于网卡特性，可在网卡上进行包合并和拆分，减轻CPU负荷。其中，TSO是针对TCP的拆包，UFO是针对UDP的拆包，LRO是针对TCP的合包。然而，LRO、TSO和UFO只能处理TCP和UDP包，而且并非所有的网卡都支持这些特性。而软件包合并（ Generic Receive Offload，GRO ）和包拆分（ Generic Segmentation Offload，GSO ）在网卡不支持分片、重组offload能力（ 如TSO、UFO、LRO ）的情况下，GSO推迟数据分片直至数据发送到网卡驱动之前进行分片后再发往网卡，GRO将大量的小报文合并为少量的大报文，再将合并后的大报文提交给OS协议栈处理。同时GSO、GRO不仅支持TCP和UDP包，还可支持vxlan和gre。

本文将介绍DPDK GRO和GSO的测试方法并与Kernel GSO/GRO等进行网络应用性能的对比。

> 测试环境信息：

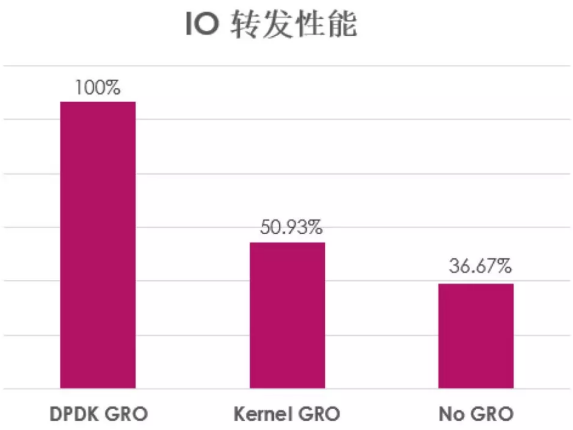
Item	Description
Server Platform	Intel® Server Board S2600GZ Intel® Server Board S2600GZ Family
CPU	Intel(R) Xeon(R) Platinum 8180 (38.5M L3 Cache, 2.50 GHz) Number of cores 56, Number of threads 112.
Memory	Total 96GB over 8 channels, DDR4 @2666 Mhz
PCIe	1 x PCIe Gen3 x8
NICs	Intel® Ethernet Converged Network Adapter XL710-QDA2 (2x40G)
BIOS	SE5C620.86B.01.00.0013
Microcode version	0x2000043
Host Operating System	Ubuntu 18.04 LTS
Host Linux kernel version	4.15.0-20-generic
Host GCC version	gcc (Ubuntu 7.3.0-16ubuntu3) 7.3.0
Host DPDK version	18.05

DPDK GRO测试拓扑图



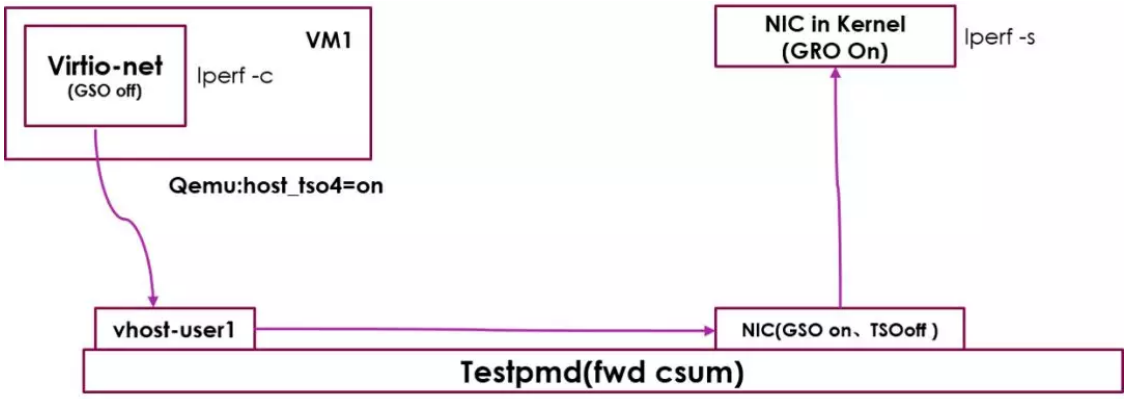
以DPDK 18.05 为例，在IO转发TCP数据包的配置下，不同包合并方式的转发性能比较如下(以DPDK GRO为基准)：

DPDK GRO	Qemu:host_tso4=on,guest_tso4=on; virtio-net GRO off ; NIC GRO on
Kernel GRO	Qemu:host_tso4=on,guest_tso4=on; virtio-net GRO on, NIC GRO off
No offload	Qemu:host_tso4=on,guest_tso4=on; virtio-net GRO off , NIC GRO off



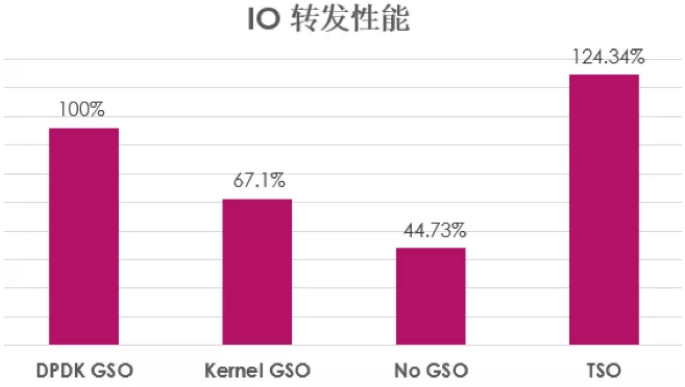
可以看到，DPDK GRO的吞吐量较KernelGRO和No GRO offload分别高出49%和63%。

DPDK GSO测试拓扑图



以DPDK 18.05 为例，在IO转发TCP数据包的配置下，对比以下不同拆包方式的转发性能(以DPDK GSO为基准)：

DPDK GSO	Qemu:host_tso4=on; virtio-net GSO off ; NIC TSO off, GSO on
Kernel GSO	Qemu:host_tso4=off; virtio-net GSO on ; NIC TSO off, GSO off
No GSO offload	Qemu:host_tso4=off; virtio-net GSO off ; NIC TSO off, GSO off
TSO	Qemu:host_tso4=on; virtio-net GSO off ; NIC TSO on



可以看到，TSO 硬件拆包具有最好的吞吐量，而软件拆包中DPDK GSO的吞吐量较Kernel GSO 和No GSO offload分别高出33%和55%。

推荐阅读

怎么提高网络应用性能？让DPDK GRO和GSO来帮你

一文读懂SPDK加速关键应用：解析SPDK Perf应用

下一代存储技术的先行: NVDIMM 你了解吗?

VPP环境配置指南