



网络功能虚拟化中的网络转发性能优化技术研究

唐 宏, 欧 亮

(中国电信股份有限公司广东研究院 广州 510630)

摘 要: 网络功能虚拟化(NFV)采用软件转发技术实现,其网络转发性能受到各类运营商关注。重点分析了NFV技术中影响转发性能的关键因素,讨论了多项开源转发性能优化技术实现方法,提出了NFV中性能优化整体解决方案,最后测试验证了部分开源技术对转发性能的提升能力,为NFV技术推广提供了重要参考。

关键词: 网络功能虚拟化;软件定义网络;网络虚拟化;性能优化

doi: 10.3969/j.issn.1000-0801.2014.11.023

Research on Forwarding Performance for Network Function Virtualization

Tang Hong, Ou Liang

(Guangdong Research Institute of China Telecom Co., Ltd., Guangzhou 510630, China)

Abstract: The key affecting factors of forwarding performance for NFV scenario were discussed, after parts of open source technologies on forwarding performance optimization were introduced briefly. An integrated NFV system end-end optimization scheme was put forward by means of serial open source software suite. Moreover, experiment reached to evaluate DPDK technology in single server case was did. It shows a remarkable increase on forwarding performance in single server achieved by open source suite. Studies provide a valuable reference NFV system design for both SDN appliance developer and network operators.

Key words: network function virtualization, software defined networking, network virtualization, performance optimization

1 引言

软件定义网络(software defined networking, SDN)技术对未来IP网络演进和运营方式将带来深刻的变化,对网络运营商而言,承载与控制分离不仅为实现智能管道提供了高效的承载控制手段,同时也通过运用网络虚拟化技术,增加了业务部署的灵活性。全球主流运营商先后提出了大量基于网络虚拟化的SDN实现思路,其中,以ETSI组织的网络功能虚拟化(network function virtualization, NFV)项目最为突出。

作为一种更现实的SDN应用场景,NFV提出将专用

的网络业务功能以软件实现,运行在基于x86架构的虚拟化基础设施上,并采用控制器实现网络资源与计算资源的协同调度。典型的应用场景如业务组链(service chaining)技术,它不仅支持IP网络中NAT、DPI、防火墙等各类业务功能的广域灵活部署,也支持PCC(policy and charging control)、GGSN(gateway GPRS support node,网关GPRS支持节点)等移动网元的弹性部署。

NFV设计的初衷是针对部分低转发流量类业务功能,x86服务器在配备高速网卡(10 Gbit/s)后,业务应用不经特殊优化,基本可以满足大多数低速率转发业务的处理要求。然而,随着SDN技术的推动,各类高速率转发和会话



控制类网络业务功能,如 vBNG、vPGW(40 Gbit/s 以上)等,也逐步加入 NFV 的应用行列。

通用服务器在处理 NFV 的高速率转发业务时,面临着严重的性能瓶颈,需要有针对性地从事务链组网、服务器虚拟化环境和业务应用 3 个层面进行性能优化,才能达到各类网络业务的线速转发要求。

本文首先分析影响 NFV 转发性能的主要因素,并介绍几类主流的优化技术。随后,结合开源技术提出一种 NFV 性能优化方案,最后,通过实验测试,验证了性能优化技术对 NFV 服务器转发效率的提升作用。

2 转发性能影响因素与优化技术

2.1 性能影响因素

NFV 中的网络业务应用运行于服务器的虚拟化环境中,单个应用业务流量的收发要经过虚拟化层、服务器 I/O 通道、内核协议栈等多个处理流程,而多个应用业务之间又可以用复杂的物理或虚拟网络连接,被称为业务组链技术。因此,NFV 系统的整体性能取决于单服务器转发性能与业务组链转发性能两个方面。

由于采用软件转发和交换技术,单服务器内部转发性能是 NFV 系统的主要瓶颈。如图 1 所示,业务应用流量的收发 I/O 通道依次包括物理网卡、虚拟交换机、虚拟网卡 3 个环节(见图 1 左半部分),从软件结构上看,报文的收发需要经过物理网卡驱程、宿主机内核网络协议栈、内核态虚拟交换层、虚拟机(VM)网卡驱程、虚拟机内核态网络协议栈、虚拟机用户态应用等多个转发通道,存在着海量系统中断、内核上下文切换、内存复制、虚拟化封装/解封等大量 CPU 费时操作过程。

(1)网卡硬件中断

目前大量流行的 PCI/PCIe 网卡在收到报文后,一般采

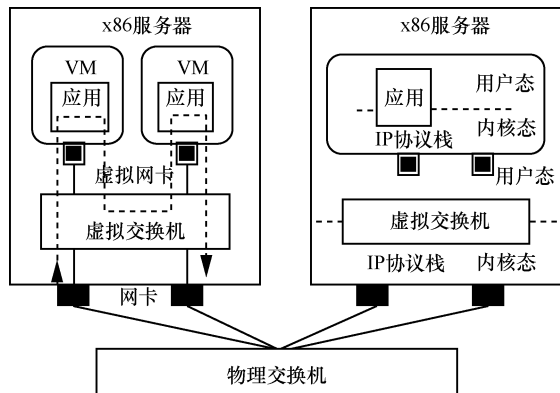


图 1 IP 网络流量流向分析与预测方案

用 DMA(direct memory access,直接存储器存取)方式直接写入内存并产生 CPU 硬件中断,当网络流量激增时,CPU 大部分时间阻塞于中断响应。在多核系统中,可能存在多块网卡绑定同一个 CPU 核的情况,使其占用率达到 100%。因此,网卡性能改进一般采用减少或关闭中断、多核 CPU 负载均衡等优化措施。

(2)内核网络协议栈

Linux 或 FreeBSD 系统中,用户态程序调用系统套接字进行数据收发时,会使用内核网络协议栈。这将产生两方面的性能问题:一是系统调用导致的内核上下文切换,会频繁占用 CPU 周期;二是协议栈与用户进程间的报文复制是一种费时的操作。NFV 系统中,业务应用报文处理从物理网卡到业务应用需要完成收发操作各 1 次,至少经过 4 次上下文切换(宿主机 2 次以及 VM 内 2 次)和 4 次报文复制。将网络协议栈移植到用户态是一种可行思路,但这种方法违反了 GNU 协议(GNU 是 GNU GPL 的简称, Linux 内核受 GNU GPL 保护,内核代码不能用于 Linux 内核外)。因此,弃用网络协议栈以换取转发性能,是唯一可行的办法,但需要付出大量修改业务应用代码的代价。

(3)虚拟化层的封装效率

业务应用中存在两类封装:服务器内部的 I/O 封装和网络层对流量的虚拟化封装。前者是由于 NFV 的业务应用运行于 VM 中,流量需要经历多次封装/解封过程:宿主机 Hypervisor 对 VM 的 I/O 封装(如 QEMU)、虚拟交换机对端口的封装(如 OVS 的 tap)、云管理平台对虚拟网络端口的封装(如 OpenStack 中的 vnet);后者是为实现 NFV 用户隔离,在流量中添加的用户标识,如 VLAN、VxLAN 等。这两类封装/解封均要消耗 CPU 周期,会降低 NFV 系统的转发效率。

(4)业务链网络转发效率

NFV 的业务链存在星型和串行两种组网方式,如图 2 所示。星型连接依赖物理网络设备的硬件转发能力,整体转发性能较优,但当应用(运行于 x86 服务器)的数量较大时,会消耗大量昂贵的网络设备端口。因此,在业务链组网范围不大时,如在 IDC 内部,为简化组网和节约端口,更多地采用串行连接。

当串行连接时,NFV 控制器需要在多个业务应用中,选择合适位置的应用进程或进程组来处理流量,以均衡各应用负荷并兼顾业务链网络性能。不合适的负载均衡算法,会造成流量在不同进程组的上下行链路之间反复穿

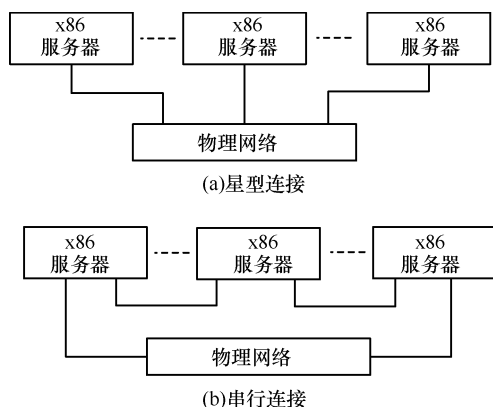


图2 两种常见的NFV组网连接方法

越,严重降低业务链网络的带宽利用率。

2.2 转发性能提升方法

针对上述转发性能影响因素,目前业界已提出多类解决方法。对网卡频繁中断问题,提出DMA方式和设备轮询机制(NAPI),其中NAPI采用“中断+轮询”方式工作,网卡驱程在中断量较大(设定阈值)时会关闭中断,并自动采用轮询方式收发报文,释放CPU的中断响应操作。

针对网络协议栈多次复制和内核上下文切换问题,提出了大量零复制(zero-copy)技术,其主要思想是:用户态进程直接读取网卡缓冲区,旁路网络协议栈。其中,UIO(userspace I/O framework)技术通过设置内核态UIO驱动,不仅可关闭中断,还可通过内核的mmap(映射虚拟内存页)功能将内核空间映射至用户空间,用户通过标准的read/write实现对网卡缓冲的零复制收发;另一类称为PF_ring技术,它在内核定义一个定长的环形(ring)缓冲,并创建了一类特殊的PF_socket,用户态进程通过PF套接字,直接访问内核中的ring缓冲区,避免了标准内核协议栈的处理。

此外,部分开源项目提供了基于上述技术的综合解决方案,典型方案以Netmap和Intel DPDK(data plane development kit,数据平台开发套件)为代表,可大幅提升单服务器内的报文转发性能。两者思路相似,如图3所示,Netmap采用了UIO的零复制、设备轮询、定长环形队列、多核负载分担等多项技术,在内核开辟环形缓冲,以mmap技术实现用户进程、内核线程和网卡之间的共享内存,以旁路协议栈处理,用户进程通过调用Netmap API,即可实现对环形队列和网卡I/O通道的高效读写,单核(1 Gbit/s主频)转发能力高达14 Mpps(million packet per second,百万分组数/s)。

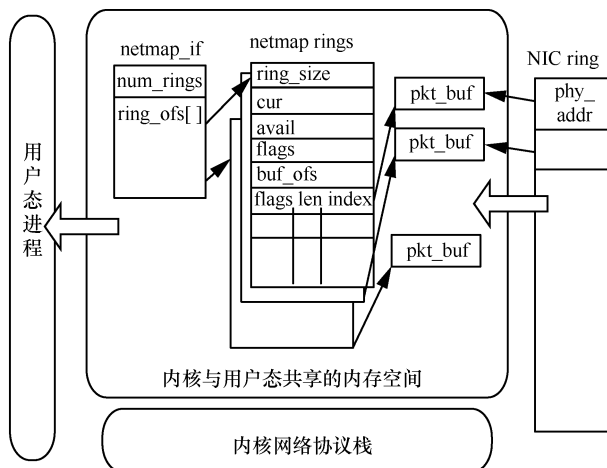


图3 Netmap的内部转发结构

更进一步地,DPDK不仅实现了网卡缓冲区到用户空间的零复制,还提供虚拟环境下的虚拟接口、兼容OpenvSwitch的虚拟交换机、专为短小报文提供的hugepage访问机制等实用技术,可以达到单核20 Mpps的处理能力。

上述开源方案能很好满足NFV中DPI、防火墙、CGN等无需协议栈的网络业务功能,但存在着大量改写原有业务应用套接字的问题,应用中需要在性能提升与代码改动之间进行取舍。

3 NFV转发性能综合提升方案

3.1 方案思路

从NFV系统的组网和转发性能要求来看,需要考虑服务器宿主OS的I/O通道、业务应用及其VM转发通道、虚拟交换机、业务链组网效率以及NFV控制器等多个方面的性能优化,才能实现系统性能的整体提升,目前业界缺乏对应解决方案,要解决这些问题,需要考虑以下几点。

(1) 优化网卡驱动

采用应用层或内核轮询机制,关闭网卡中断,释放CPU中断处理时间。

(2) 旁路内核协议栈,减少内核上下文切换

数据通道应采用UIO技术,用户态直接收发网卡缓冲区,减少报文处理中的内核系统调用;但同时应保留协议栈功能供管理网络使用。

(3) 提供虚拟网络的性能优化

虚拟交换机容易导致性能瓶颈,选择用户态虚拟交换机有利提升性能和进行故障定位;方案应支持OpenFlow功能,满足各类复杂的转发要求;应可关闭不必要的虚拟化端口和封装,减少CPU对报文的封装开销。



(4) 提供 VM 内部 I/O 优化能力

优化方案应能提升 VM 内部的虚拟网卡及其业务应用转发通道的性能。

(5) 与 OpenStack 类云管理平台的无缝对接

优化技术不能影响 OpenStack 相关网络插件和虚拟网络的管理。

(6) 支持 CPU 与网络资源的负载分担

利用 CPU 亲和性技术, 可按需调整和定制多核 CPU 的负载分布; 利用 NFV 控制器的全局拓扑, 提供业务组链的链路负荷优化和路由优化。

3.2 基于开源技术的性能优化解决方案

根据前面的分析, 提出一种基于 DPDK 的串行 NFV 业务链性能优化方案。

如图 4 所示, 系统由 NFV 调度器、NFV 服务器、NFV 业务应用(动态创建于 VM 中)、OpenStack 管理平台、虚拟/物理交换机 5 个部分组成。其中, NFV 调度器用于均衡分配业务应用的 CPU 开销和业务链流量, 它可以基于 Open Daylight (ODL) 或 OpenContrail 等开源 SDN 控制器实现; OpenStack 用于创建业务应用所需的 VM 环境和镜像激活以及 VM 所需的虚拟网络创建; 物理交换机用于连接 NFV 服务器, 软件虚拟交换机用于 NFV 服务器内部组网, 后者包括数据转发网络与管理网络。

该方案对网络转发性能的提升原理如下。

整体业务链网络的转发效率由 NFV 调度器智能控制, 它通过采用特有的业务路径优化算法, 可以根据 OpenStack 和 SDN 控制器对 CPU 和网络利用率的监测, 动态定位业务应用的位置, 为每条到达的网络流量计算最优

路径。调度器的另一个作用是简化 OpenStack 内部虚拟网络定义和防火墙配置, 减少不必要的虚拟化封装和设置。

通过在 NFV 服务器中配置 DPDK 工具包和 OVDK 虚拟交换机, 可大幅提升单服务器的转发性能; NFV 服务器中的宿主 OS 和 VM 操作系统中加载 DPDK 模块后, 用户态的业务应用可以直接访问 VM 虚拟网卡, 无需经过 VM 协议栈处理, 加快了 VM 内部的转发能力; 其次, 宿主机的网卡中断将被关闭, 网卡缓冲区通过 UIO 方式映射至用户态, 用户态进程与网卡缓冲区共同轮询环形队列, 形成生产与消费关系。在服务器加载 OVDK 虚拟交换机以后, OVDK 进程就成为 DPDK 的用户进程, 因此 OVDK 是一个纯用户态交换机, 它的虚拟端口可以挂接 VM 的虚拟网卡。

在多核 CPU 负载均衡方面, 由于 DPDK 可以灵活指定单个或多个 CPU 处理环形队列中的报文, 通过方案中的 NFV 控制器, 用户可以按需或者自动调整业务应用对 CPU 核的占用。

因此, 本方案通过融合开源 SDN 控制器和 DPDK 技术, 实现了从物理网卡到业务应用的端到端高速转发, 中间未经任何内核协议栈处理。此外, 由于 OVDK 虚拟交换机还可支持 OpenStack Neutron 插件、兼容 OpenvSwitch 指令以及支持 OpenFlow 1.0 等多项功能, 保证了方案具有良好的扩展性和标准化支持能力。

4 实验验证

下面进行 DPDK 技术的网络转发性能实验。实验目的在于比较 Linux 内核与 DPDK 两种模式下的 L2/L3 转发性能。由于 NFV 整系统性能测试存在较多不稳定因素, 限于

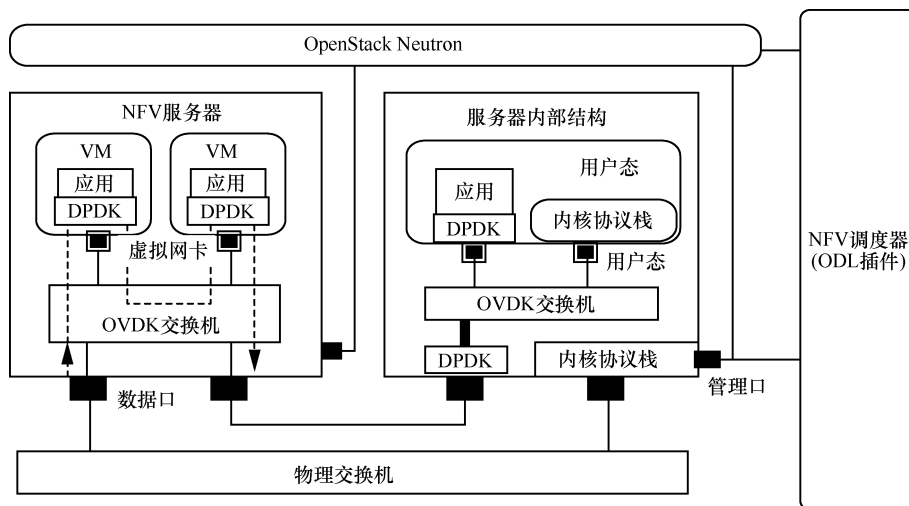


图 4 基于 DPDK 的串行业务链优化方案

篇幅以及结论清晰起见,仅介绍非虚拟化状态下的单机实验结果。

实验环境由待测服务器、10 Gbit/s 网卡、测试仪组成,如图 5 所示。为减少软件测量误差,测试环境中不采用 Netperf 等软件发送分组工具,而由 TestCenter 测试仪控制和统计收发分组过程,服务器配备双 10 Gbit/s 网卡,网卡具有独立 IP 地址,并在服务器内部构成环回通道。

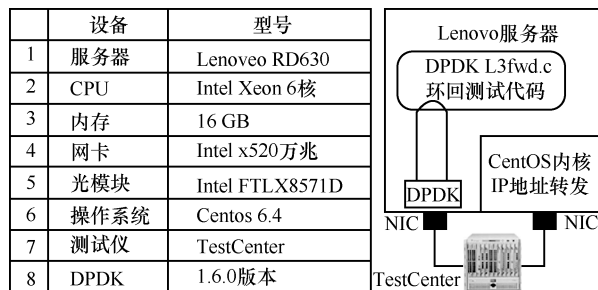


图 5 试验设备与测试环境

环回通道分为 L2 和 L3 两种,普通 Linux 的 L2 环回为内核态的 Linux bridge 设备,L3 环回为内核 IP 协议栈,L3 测试需要开启 IP 地址转发功能打开网络协议栈。DPDK 软件包带有测试专用的环回测试代码,非常容易分别实现对 L2 和 L3 转发性能的测试。

实验比较 Linux 内核与 DPDK 1.6.0 软件包的 L2 和 L3 转发性能时,主要考察单位时间接收分组率(Mpps)、平均时延(μ s)、分组丢失率 3 项指标。

测试方法如下:服务器在 CentOS 开启 bridge 时,DPDK 加载 L2 环回;而在 CentOS 开启 IP 地址转发功能时,DPDK 开启 L3 测试例程,DPDK 选用单个 CPU 核;从 1 Gbit/s 至 10 Gbit/s 逐步提升 TestCenter 发送速率,统计两种方案收到的报文状况。

测试中发现,对于 64 byte 的短小分组,Linux 内核在 2 Gbit/s 时已经出现分组丢失现象,而在单核情况下,DPDK 在达到 7 Gbit/s 时才会出现性能下降。因此,7 Gbit/s 时的性能比较更具有代表性。

图 6 说明测试仪以 7 Gbit/s 速率发送分组时,两种方案的接收分组的性能差异,DPDK 始终能维持线速转发,超过 10 Mpps,仅在 64 byte 小分组时出现千分之一的分组丢失,而 Linux 协议栈的处理性能始终维持在 0.4 Mpps 的较低水平,其中 64 byte 小分组的分组丢失率达到 92%,两者性能相差 20 倍。两者的平均转发性能相差 5~10 倍。

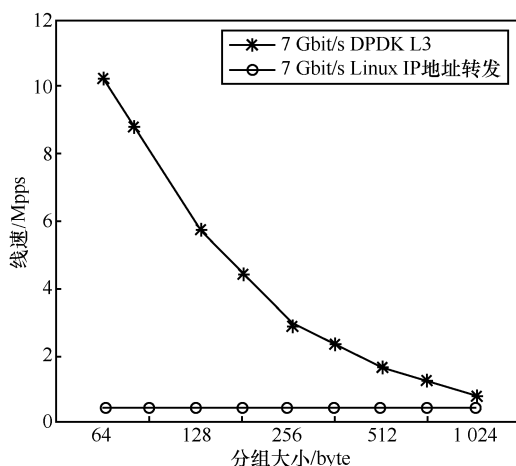


图 6 DPDK 与 Linux L3 转发比较(7 Gbit/s)

图 7 中 DPDK 的转发时延始终维持在 250 μ s 左右的较低水平,而 Linux 内核协议栈的平均处理时延到达数毫秒,同时呈现随分组长度增加而递减现象,说明单位时间内到达的报文越多,在内核中的缓冲时间则越长。

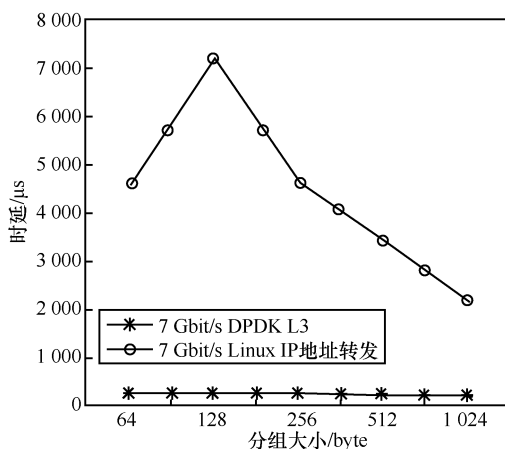


图 7 DPDK 与 Linux 的 L3 转发平均时延比较(7 Gbit/s)

因此,就单核性能而言,各项实验结果说明 DPDK 技术具有更良好的转发效率,相比普通 Linux 内核协议栈,转发性能和平均时延两项指标提升了 5~10 倍。

最后,还对 DPDK 进行了多核并行转发性能测试,证明了 CPU 亲和性技术可以进一步提升转发效率,不再赘述。

5 结束语

本文详细分析了 NFV 系统中对转发性能的影响因素,介绍了 UIO 等多类优化方法以及 Netmap 等开源优化方案原理。针对 NFV 系统的组网效率和单机性能瓶颈,本

(下转第 172 页)