TCP 加速技术研究综述*

王 圣+, 苏金树

(国防科学技术大学 计算机学院,湖南 长沙 410073)

A Survey of Technology for TCP Acceleration

WANG Sheng+, SU Jin-Shu

(School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: Phn: +86-731-4515374, E-mail: jackyws@163.net

Received 2004-06-11; Accepted 2004-09-07

Wang S, Su JS. A survey of technology for TCP acceleration. *Journal of Software*, 2004,15(11):1689~1699.
http://www.jos.org.cn/1000-9825/15/1689.htm

Abstract: With the rapid increment of network bandwidth, the overheads of protocol on host have become the bottleneck of system performance. A lot of work including optimization for protocol mechanism, implementation and architecture have been developed in the fields of protocol acceleration. After analyzing overhead and performance of the protocols, it is attempted to survey the developments of TCP (transfer control protocol) accelerating on mechanism, implementation and architecture. Finally, some future directions are discussed.

Key words: protocol; performance; overhead; acceleration; mechanism; implementation; architecture

摘 要: 随着网络带宽的迅速增长,主机协议处理开销已经成为系统整体性能的瓶颈.近年来,在协议加速领域开展了大量的研究工作,主要分为协议处理机制、实现以及结构的优化.在分析了协议处理的开销与性能之后,从协议机制、实现和处理结构这 3 方面对 TCP(transmission control protocol)加速技术的研究现状进行了总结,并分析了进一步的研究方向.

关键词: 协议;性能;开销;加速;机制;实现;结构

中图法分类号: TP393 **文献标识码**: A

随着网络技术,尤其是光纤技术的快速发展,光纤通信网络正迅速成为主要的网络传输手段,使网络带宽不断提升.网络应用的性能需求表现为高吞吐量、低延迟、高带宽、低主机开销和低存储开销等特点[1].由于处理 1 位的网络信息需要 1Hz 的 CPU 性能,因此,当 100Mbps 以上网络出现后,网络协议处理一直占用相当多的 CPU 能力.对于万兆以太网[2]或 Infiniband[3],主机系统上的协议处理更是成为网络可靠传输的性能瓶颈,所以 TCP(transfer control protocol)加速技术一直是近年来的研究热点.

本文主要综述国内外有关 TCP 加速技术的研究成果.第 1 节论述系统 TCP 处理的主要开销.第 2 节是关于 TCP 性能的理论分析和实验验证.第 3 节阐述通过优化 TCP 处理机制进行加速的技术.第 4 节重点讨论 TCP 实

* Supported by the National Natural Science Foundation of China under Grant No.90104001 (国家自然科学基金)

作者简介: **王圣**(1977－),男,江苏句容人,博士生,主要研究领域为计算机网络,信息安全;**苏金树**(1962－),男,博士,教授,博士生导师,主要研究领域为计算机网络,信息安全.

现的优化方式.第 5 节研究从结构上进行 TCP 优化调整.最后进行小结.

1 TCP 处理主要开销

高速网络中的 TCP 性能通常受限于发送与接收主机,而不是网络硬件或协议本身的实现[4].TCP 处理的主要开销分为中断操作、数据复制和协议处理这 3 个方面.

1.1 中断操作

磁盘传输延时可以有比较精确的估计.与磁盘 IO 不同,由于网络传输延时的不确定性,系统与 NIC(network interface controller)之间一般采用异步中断的通信方式[5].

网络驱动程序每次和内核交互一个报文,这样的处理使得驱动程序不需要关心协议的具体实现,而协议也不需要关心数据的物理传输[6],但是产生了较多的中断开销.由于是在内核执行,中断处理代码要尽可能少[7].当数据到达 NIC,硬件中断内核时,内核只是将数据简单移出 NIC,而后迅速恢复以前的运行状态,其余的操作将在软中断中处理.但是,将中断处理分段进行,必然会引入更大的开销.

将中断处理分段进行,可以防止硬件中断占用过多的 CPU 时间,将更多的操作留到 CPU 空闲的时候处理.但是,频繁的上下文切换以及相应的软中断控制增加了额外的开销.标准 NIC 为每一个到达的报文产生一个中断信号,触发系统运行设备驱动程序进行报文的接收操作.在网络带宽较小的时候,这种设计对系统的影响并不明显.当网络带宽达到 Gbit 时,频繁的中断操作将严重影响系统的整体性能.

1.2 数据复制

在传统的数据复制中,收、发各需要经历 4 个步骤[8].接收方的 4 个步骤是:(1) 将数据从 NIC 存储器复制到内核空间.如果 NIC 没有提供 DMA(direct memory access)方式,则复制操作由 CPU 执行,数据将两次经过存储总线;(2) 对数据进行校验和计算,此操作涉及到所有的传输数据,数据将再次经过存储总线;(3) 将数据从内核复制到用户空间,此操作必须由 CPU 执行,数据跨越内核与用户空间,经过两次存储总线;(4) 应用从用户空间获取数据.与接收方相对,发送方也需要类似的 4 个步骤.显然,在整个网络数据接收或发送过程中,数据最多会 6 次经过存储总线,存储总线需要提供 6 倍于网络数据流的带宽.

如此复杂的复制操作源于 TCP 具有可靠保序的性质[9].该特性使 TCP 向上层应用提供透明的可靠传输.为了保证报文传输可靠且有序,TCP 需要提供重传机制,在出错或超时的情况下重新传输,而重传报文只能在内核空间保持.应用将数据传输给内核之后,则认为报文已经正确发送,立即返回,并释放或重复使用用户空间的缓冲.在接收方,TCP 收到的可能是错误或乱序的数据,必须等待发送方的重传或者在本地重新排序,然后通知应用,等待应用接收.

1.3 协议处理

协议处理主要分为链接的管理、数据传输、定时器管理和错误与拥塞控制这 4 个部分[10].其处理开销主要包括:(1) 应用数据被分割成最适合发送的数据块;(2) 每次发送数据后,启动重传定时器;(3) 收到数据后,发送延时确认;(4) 对所有数据进行校验和计算;(5) 如果报文乱序,将对乱序报文重新排序;(6) 检测并丢弃重复报文;(7) 提供流量控制,防止拥塞.

在具体实现 TCP 协议时,其开销主要分为 3 个部分[4]:(1) 报文相关的开销,包括协议处理开销以及报文所产生的中断开销;(2) 与数据相关的开销,包括数据复制以及校验和计算开销;(3) 操作系统相关的开销,包括上下文切换以及存储管理开销.实验数据表明[3],在传统的 TCP 协议处理中,有 700Mbps 的 TCP 流量就可以将运行于 500MHz 的 Alpha CPU 处理能力消耗完.

2 TCP 性能的理论分析与实验验证

为了分析 TCP 的性能,人们建立了许多 TCP 模型,从理论上进行论证,并通过实验对模型进行调整,使模型与实际更为接近,同时利用实验结果对 TCP 性能进行定量的分析.

2.1 性能评价标准

人们一般从吞吐量、CPU 使用率和延时这 3 个方面评价 TCP 的处理性能[11].系统处理能够完成的带宽是性能评价最基本的标准.在传统的协议处理中,CPU 使用开销基本上是 1Hz/bit,对于接近 10Gbps 或者更高的网络带宽,CPU 显然不堪重负.因此,CPU 的使用率将是评价协议处理性能的一个重要标准.面对越来越快的网络速度,经过系统 IO 总线和存储总线的事务越来越频繁,造成的延时严重影响了性能,所以,延时降低的程度将对性能的提升起重要的作用.

上述 3 个标准从不同的角度评价协议处理性能.吞吐量与 CPU 使用率主要是针对系统处理,属于整体性能指标,而延时主要针对具体的应用数据流,属于局部范围的性能评价标准.有时候需要在这些标准之间进行一定的权衡.例如,当利用流水线技术增加 TCP 的整体处理带宽时,由于流水段之间的交互,必然会影响单个数据流的延时.

2.2 性能分析模型假定

分析模型基本假定一般有两点[12]:(1) 发送方采用 TCP Reno 算法进行拥塞控制.由于模型主要针对 TCP 性能,所以忽略了调度或缓冲带来的延时,同时,发送方将始终在拥塞窗口所允许的范围内以最快的速度持续发送最大限度的报文.(2) 针对 TCP 性能进行测试的模型基本上都采用往返时间作为度量.在不同的往返中,丢失行为互不影响;但是在同一个往返过程中,一旦出现丢失行为,将造成后续所有报文的丢失.同时,还假定报文丢失概率与窗口的大小无关.由于在传输过程中应答的丢失对窗口造成的影响很小,所以假定报文丢失都发生在有效数据传输的时候.

2.3 性能分析模型

Padhye[12]提出了一个基于 TCP Reno 的稳定状态模型,将吞吐量表示成丢失率的函数,重点考虑了超时情况下引起丢失的协议行为,并证明了模型在不同丢失概率范围内均有效.通过分析,发现超时对 TCP 性能具有非常大的影响,而模型能够很好地反映这种影响,特别适合于丢失率比较高的情况.另外,还对 TCP 带宽的表达式进行了简化,可作为模型在大多数情况下的近似结果.

Cardwell[13]针对 TCP 建立链接与 TCP 慢启动提出了新的模型,对稳定状态模型进行了扩展,可以分析没有报文丢失下的延时.通过模拟发现,链接建立模型能够比较精确地预测分析延时,同时,扩展的数据传输模型能够很好地描述在不同丢失情况下不同长度报文流的特性.

为了预测延时与吞吐量,可以对随机 TCP 模型进行扩展[14],为拥塞窗口具有不连续变化特性的慢启动阶段构建新的模型,同时将它与稳定状态的 TCP 模型结合起来,以获取更好的性能预测.通过分析,得出了发送速率与吞吐量相对于丢失率与往返时间的函数关系.模拟结果表明,模型能够比较精确地预测大容量数据传输以及短期 TCP 链接的性能.

在传统流控机制中,拥塞发生前数据传输速率以线性方式增长,Altman[15]对此作了改进,重点考虑了报文丢失的突发性.模型假定突发丢失时为 Bad 状态,丢失很少时为 Good 状态,信道在两个状态上持续的时间呈几何或指数分布,并且允许在两个状态下都可以丢失报文,用 Markov 链过程对突发性的丢失行为进行分析.结果指出,在一定的丢失率下,如果丢失总是出现在突发性传输时,则性能可以得到很大改善.

Mitzenmacher[16]研究了在 TCP 传输时间的累积分布下,不同丢失模型所产生的影响.模拟结果表明,当丢失模型的选择对实际分布函数具有显著影响的时候,在不同的模型中,函数的表达形式也将具有很大的差异性.由于在所有的丢失模型中,传输时间与超时的数量具有近似的线性关系,而超时分布呈现为不同参数的常态分布,所以修正过程具有一定的有效性.文献[16]还对传输时间的分布与丢失应答之间的关系进行了研究,模拟显示,应答丢失仍然会在很大程度上降低传输的速度.

为了弥补 PRAM 模型[17]与 BSP 模型[18]的不足,进一步研究并行计算互连网络中的算法,Culler[19]提出了基于点到点互连通信网络的模型 LogP. Keeton 等人[20]将 LogP 模型应用到具有较低延时的局域网中,指出开销、延时、网络拓扑、通信模式、报文大小、竞争和拥塞都是评价局域网条件下 TCP 性能的重要因素.

Shivam 等人[21]建立了 LAWS 模型,为协议卸载的实验结果提供了进行合理解释的基础,其中 L 是 CPU 处理速度与 NIC 处理速度的比值,A 是每单元网络带宽所花费的应用处理开销与参考主机处理每单元网络带宽数据所需的通信开销之间的比值,W 是主机处理的饱和带宽与网络峰值带宽的比值,S 是通过复制避免技术不能消除的开销占所有开销的比例.分析结果指出,对高速网络中通信占主导的应用来说,性能可以获得 0.5~2 倍的增益.

2.4 分析结果

Keeton[20]对几个商业性局域网的往返延时与传输吞吐量进行了实验.结果表明,相对于主机系统的处理开销,网络的传输交换开销几乎可以忽略不计.同时,当网络带宽增加时,软件的处理开销也会增加,这对主机系统的应用性能会造成显著的影响,因此,只有降低主机处理开销,才能实现 TCP 性能与网络带宽的同比增长.

Clark[22]通过计算执行过程中的指令数进行性能分析.为了避免 TCP 处理受到操作系统的干扰,修改了相应代码.实验对处理路径作了一定的限制,重点分析正常路径,将一些异常处理排除在外.结果表明,代码的执行并不是影响性能的关键因素,数据传输才是主要原因,其次是与操作系统相关的其他操作,包括中断处理以及缓冲的分配与释放.Pink[23]的分析结果表明,性能瓶颈不在于 TCP 协议本身,协议的具体实现、运行时缺乏的资源、数据的传输以及不完善的硬件设计,才是性能提升的主要障碍.

然而,在低速的环境下却并非如此[24].由于协议处理的过程中加入了协议调度、存储管理、中断管理等操作系统的干预,协议的执行很难用精确的数学公式描述.实验在内核代码中加入了 10 多个探测点,在 100Mbps 的环境下进行测试,结果表明,此时 TCP 协议处理的主要开销表现在协议本身,而内核的存储器复制、校验和计算以及系统调用三者在发送和接收时仅分别占总开销的 22%和 16.7%.

NIC 是整个协议处理的重要部分,实现高性能 NIC 的关键因素可归纳为两类参数[25]:数据传输和缓冲.(1) 数据传输:单个报文长度的增加可以使传输更加有效,节省控制开销;处理器在数据传输过程中的参与程度对性能有着直接的影响;报文传输过程中,源和目的存储器的物理技术以及数据是否直接从源传输到目的存储器会严重影响性能.(2) 缓冲:系统需要大容量的缓冲来匹配应用、处理器与 NIC 之间的速度;NIC 缓冲不足会经常迫使处理器频繁查询 NIC 状态,将报文从 NIC 存储器移出来;运行多道程序时,NIC 必须为每一个程序分割一块缓冲.此外,不可靠的网络需要 NIC 执行一些流控操作,为了避免阻塞网络,很多 NIC 都需要使用大容量缓冲.

3 TCP 处理机制优化技术

Buzzard[26]指出,在网络底层提供有序、容错的报文传输,并避免死锁出现的前提下,可以通过 TCP 处理机制的优化措施来减少主机开销,以获取更高的处理带宽.

3.1 校验和计算

校验和计算本身并不复杂,但由于计算操作涉及到所有的传输数据,计算开销相对来说就显得很大.Borman[27]对原始的校验和计算进行了 4 个方面的优化,包括延迟进位、循环展开、结合校验和的复制操作、递增的校验和更新优化.后两种优化在某些系统中没有实现.文献[28,29]进一步作了改进,允许改变部分数据,并在不重新检查所有字节的情况下更新校验和.

Kleinpaste[30]通过修改数据复制的操作语义优化处理,将数据与控制信息分开传输,对校验和计算作相应的变动.协议数据头部的校验和由主机软件计算,并将结果通知硬件,而用户数据的校验和则由硬件计算.

上述研究对校验和计算作了很好的优化,但是对类似于万兆以太网的高速网络来说,开销仍然太大.Henriksson[31]等人设计了 TUCFP 芯片,可以并行处理 32 位校验和数据.它可以较容易地与通用处理器集成.

3.2 协议处理加速

对于 TCP 的性能,每年都有若干专家提出许多方法进行加速优化,但都有一个最基本的原则,就是对大概率事件进行优化处理.TCP 中一个比较典型的实现就是采用高速缓存实现 PCB(protocol control block)[32].实验表明,高速缓存命中率可以达到 80%.

Jacobson 提出的首部预测也颇具代表性[32].它基于两种常见现象:(1) 如果 TCP 发送数据,连接上等待接收的下一个报文是对已经发送数据的应答;(2) 如果 TCP 接收数据,连接上等待接收的下一个报文是顺序到达的报文.经过实验,当网络跨越 LAN 传输时,利用首部预测处理的报文可占 97%以上;当跨越 WAN 传输时,比例有所降低,在 83%~99%之间.

3.3 拥塞控制

由于 TCP 是一种可靠的传输协议,所以拥塞控制在 TCP 协议处理中相当重要.尽量减少或者避免拥塞也会对协议处理产生加速作用.许多人对这个问题进行了仔细的研究,例如,TCP 传输的关键路径分析[33]、TCP 窗口的精确采样分析[34]、非连续网络拥塞控制系统的模型与优化[35]、长期与短期 TCP 流响应时间的可预测性[36]、报文传输错误与拥塞丢失下的 TCP 吞吐量分析[37]、大容量缓冲对 TCP 排队行为的影响[38]等.

为了弥补现有 TCP MIB(management information base)的不足,对 TCP 性能有更好的指示,并对形成 TCP 拥塞瓶颈的成因有准确、实时的描述,人们开发了 Web100[39].在修改后的 TCP MIB 信息中,用户进程可以针对具体的 TCP 流对许多变量进行调整,例如动态调整 TCP 缓冲的大小.允许用户进程对 TCP 参数进行调节有着巨大的优点:降低了改动 TCP 核心代码的风险;简化了算法的测量,能够在资源需求与限制上进行更好的平衡;可以在用户空间实施约束自动调节的复杂策略.

4 TCP 实现优化技术

由于 TCP 在实现的时候所采取的不同方式对 TCP 的性能有很大的影响,所以在实现方式上进行优化一直是 TCP 加速技术研究的重点.

4.1 减少复制

网络速度越快,引发的数据传输量自然越多,相应的复制操作开销就越大,这使得存储器带宽日益成为约束性能的瓶颈.因此,必须寻找有效方法,以避免多次复制操作.

(1) 应用直接访问 NIC

操作系统将 NIC 的存储空间预先映射到用户或者内核空间,应用直接访问 NIC 存储空间[9].网络输入的数据一直保留在 NIC 存储器内,直到应用对报文进行访问处理,数据才会越过存储总线.这种技术要求 NIC 具有足够的智能,将数据正确引导到 NIC 上相应的用户映射区域.为了同时支持多个应用程序,NIC 需要具有足够的存储空间缓存数据.传统的系统调用也要作相应的修改.

(2) 内核与 NIC 共享存储区域

为了减少 NIC 的复杂性,可以由内核管理 NIC 存储区域,使用 DMA 或者 PIO(programmable IO)在 NIC 存储器与应用缓冲之间移动数据.这种设计方法在 Socket 层上保持了原来的操作语义,原先的应用可以不用经过任何修改而直接运行.Dalton 在实验中使用了该技术[40].

(3) 应用与内核共享存储区域

系统在用户空间与内核空间里分别定义了一些具有共享语义的 API,使用 DMA 在 NIC 与共享存储区域里移动数据.Druschel[41]使用这种技术实现了快速缓冲.但这种技术的明显缺点是应用缺乏兼容性,所有的应用必须使用改进后的 API.另外,对共享存储区域的分配与管理也需要应用、协议软件与 NIC 三者之间进行复杂、密切的协作.

(4) 基于 COW(copy-on-write)技术的内核到用户空间的页映射

可以利用已经装载到 MMU(memory management unit)的信息直接提供用户空间到内核的映射机制,从而避免复制操作[9].所有的缓冲都在用户空间里,NIC 存储器与主机之间通过 DMA 相连.这种方法不仅可以减少复制开销,而且发送方利用 COW 技术还保留了传统的复制语义.实验结果表明,这种技术使得协议处理很少接触传输数据,所引起的 Cache 失效要少得多,性能有了明显的提升.

(5) 控制与数据分离

通过对原来的协议处理进行修改,可以支持新的复制操作语义[30],即协议在处理过程中携带的只是协议数

据以及用户数据的索引,而真正的用户数据一直放在外部空间,例如用户空间或 NIC 存储空间,只有当数据最终传输的时候,才进行复制操作.

(6) 基于发送方的存储管理

网络拥塞或者接收方缓冲溢出会造成报文丢失.Buzzard 等人[26]提出一种机制,可以避免因接收方缓冲溢出而造成报文丢失.该机制采用基于发送方的存储管理,由发送方在发送数据之前决定数据在接收方的最终位置.

(7) 提前置入缓冲

对于传统的 TCP 发送处理,在传输方进行改进比较容易,NIC 可以直接访问用户空间传输应用数据.但是,对于接收方,由于数据到达的延时与传输顺序不确定,内核必须对进入的数据进行缓存,应用在得到内核的通知后才分配相应的空间准备接收.可以采用提前置入缓冲的方法对 TCP 的处理进行有效的加速[42].

(8) RDMA(remote direct memory access)协议

RDMA[43]在报文中携带数据的目的地存储位置信息,在接收端由 NIC 直接将数据放置到用户空间里,以避免数据的多次复制.

由于复制操作与操作系统的存储管理密切关联,所以任何减少复制的措施都会增加存储管理开销.由于 RDMA 协议利用报文传输地址信息,因此操作系统不需要进行复杂的地址映射管理,同时在应用接口层保持了原来的复制语义.当 RDMA 协议与具有 TOE(TCP offload engine)的 NIC 结合起来时,可以有效地实施快速避免复制的协议处理.在最近的研究中,RDMA 已经成为最主要的减少复制技术.

4.2 减少中断

如何减少中断的次数一直是人们研究的热点,主要集中在以下几个方面:

(1) 将异步触发变为轮询

Rangarajan 等人[44]在 TCP 处理的卸载实验中,将 TCP 的功能完全独立出来,用一台设备作为 TCP 服务器进行专门处理.由于 TCP 服务器只专注于 TCP/IP 协议的处理,所以中断处理相对比较容易,可以用轮询替换.由于上层协议的处理将使轮询时间的间隔不可预知,所以必须仔细控制轮询的频率,过于频繁将导致总线拥塞,速度过慢将导致系统不能及时处理所有的事件.

(2) 中断合并

为了消除与报文相关的开销,可以采用中断合并技术[4].当报文到达 NIC 时,NIC 只是将报文暂时缓存,并不立即向主机发出中断.当报文达到一定数量或 NIC 缓存使用率达到一定阈值时,NIC 才向主机发出中断.主机在一次中断中对所有报文进行处理,从而减小中断开销.但是,集中处理带来了单个报文延迟时间不确定的问题,从而造成单个报文端到端的延时不确定.同时,在中断处理中一次性处理多个报文也使得应用程序的调度优先级变相降低.

(3) 增加单个报文的长度

减少中断数量有一个简单而有效的方法,就是增加单个报文的数据传输长度(例如,每次都以 MTU(maximum transfer unit)的大小传输).Juan[45]对数据传输过程中浪费掉的中断数目进行了研究.如果上层协议数据以小于 MTU 的长度进行分段,那么传输报文时将使产生的中断数目增加.

(4) 报文过滤

对于某些局域网内部产生的广播报文以及某些 UDP(user datagram protocol)报文,应该由 NIC 直接处理,实现对上层协议的报文过滤,这样可以有效减少主机的中断数目[45].显然,这种方法对 NIC 的智能程度要求较高.

中断是硬件与软件进行数据交互的有效方式,减小中断开销的唯一方法就是减少中断数目.增加单个报文的长度是一种简单、可行的有效方法.它对原来的中断处理行为没有任何影响,但是需要上层协议的支持.由于硬件技术的飞速发展,利用智能 NIC 对某些报文进行自主处理以减少中断数目的技术越来越受到重视.

4.3 Cache优化

延时是协议处理性能的关键指标,可以通过修改编译器来增加指令 Cache 的效率[46].在对指令不作任何修

改的情况下,减少每一条指令的执行开销.

(1) 集中处理高频度代码

在原来的执行代码中,每一条指令的处理频度不可能一样,有些代码(例如错误处理或初始化代码)执行的概率相对较小.可以在编译器中加一些选项,通过 C 的预编译处理,将执行频度较小的代码集中在程序或函数的尾部.这种技术能够有效地减少跳转,进而降低指令 Cache 的失效率.

(2) 延时克隆

延时克隆尽量将代码克隆的时机推迟,将多个程序段频繁执行的代码克隆到一起,减少许多调用与跳转,进一步减少指令 Cache 的失效.例如,在 TCP 处理中,等到 TCP 链接完成之后再克隆,能够获取的信息就比较多,许多状态信息变成常量.同时,如果调用指令与被调用函数相距很近,可以将跳转指令转变成与 PC 相关的分支指令,避免装载一些无用指令,减少指令 Cache 的失效.

(3) 路径内联

最后,将需要经常处理的函数内联到处理代码中.这样实现可以消除大量的调用开销,并且可以使编译器能够获得更多的上下文,对代码进行更为充分的优化.实验测试了大量乒乓报文,结果表明,这种方法很好地改善了端到端的延时.

4.4 延时隐藏

我们可以采用硬件流水与软件多线程技术实现延时隐藏.

(1) 硬件流水

Yocum[47]采用截断式 DMA 的策略对报文进行流水化传输.原来的 DMA 传输总是采用存储转发策略,只有报文完全进入主存或 NIC 存储器,才初始化相应的 DMA 进行传输.在截断式策略中,只要满足两个条件就可以进行传输:DMA 引擎空闲,而且进入 DMA 缓冲的数据达到某个阈值.在实验中,当报文长度超过 1K 时,截断式 DMA 策略表现出了很好的性能.

为了进一步提高性能,可以在硬件中设置多个协议引擎 PPE(parallel protocol engine)并行处理[48].实验对 802 的 LLC(logic link control)处理进行了测量,结果显示,PPE 可以较明显地提高性能.但由于多个引擎间的负载均衡开销,所以性能无法与流量的增长呈线性关系.

Nordqvist 等人[49]提出采用 FP(functional pages)机制在 SOC(system on chip)上对数据进行流水处理.其中,FP 是一些数据处理加速器,主要包括 CRC(cyclic redundancy check)校验单元、XAC(extract and compare)单元以及加法器,并利用微控制器对 FP 进行编程控制.

(2) 多线程技术

通常,多个 TCP 流之间表现出较弱的相关性,因此可以采用多线程处理技术对不同的 TCP 流进行处理[8].多线程并发技术可以使存储访问的延时在 TCP 流的重叠处理中进行很好的隐藏.

5 TCP 处理结构优化技术

可以对 TCP 的处理结构进行优化,主要包括在用户空间实现传输协议,将协议处理卸载到硬件执行以及 RDMA 协议.

5.1 用户级传输协议设计

由于传统的协议处理是在内核空间进行,多次复制与多次切换造成了很大的开销.许多研究开始尝试将协议处理放置在用户空间,主要方法包括:

(1) 提升处理路径

Mapp[50]实现了用户空间传输协议 A1.有别于 TCP 的字节流,A1 主要基于块传输,采用有选择性的重传数据策略,进行显式的 RTT(round trip time)测量,其流控是基于端到端的,并支持组播.结果表明,相对于传统的 TCP 实现,A1 的吞吐量可以获得明显的提升.

Thekkath[51]在应用与内核之间设置了若干模块,以实现用户级传输协议.主要模块有:(1) 协议库,包含典型

的协议处理实现,例如重传、校验和计算,流控等;(2) 注册服务器,它是一个运行在特权级别的进程,代表应用对端系统通信细节进行处理;(3) 网络接口,为应用提供有效安全的数据传输.结果表明,内核中各种减少开销的措施同样适用于用户空间,并且更容易维护与调试.

Edwards[52]在内核单复制的基础上实现了用户级的 TCP 协议.实验结果表明,只要底层提供合适的接口,就可以在用户空间实现一个性能可以接受的传输协议.结论还表明,虽然可以将复杂协议的部分实现转移到用户空间,但是对于一些复杂的操作,例如协议数据分离以及缓冲管理等,还是需要留在内核空间执行.

(2) 分离处理路径

为了将处理从内核的关键路径中剥离,Shivam 等人[53]实现了 EMP 系统.系统将描述符管理、虚拟存储管理和 NIC 初始化等操作转移到用户空间,并在 NIC 上实现少量描述符高速缓冲以及部分协议处理.实验进行了大量乒乓报文测试,结果表明,在延时与带宽方面,EMP 的性能比传统 TCP 有明显提升.

5.2 TOE

由于硬件具有执行速度快的优点,所以可以采用硬件实现协议处理,以获得最大的性能提升,这就是 TOE[11].在具体实现 TOE 时,可以使用可编程器件或通用处理器,以充分获取灵活性,或者使用 ASIC 设计[54],以获取更高的性能.

Henriksson[55]提出了一种新型体系结构,为报文处理设计了协议处理器,将某些处理独立出来,并在上面运行实时操作系统.通过模拟分析,如果采用先进的 CMOS 工艺,可以线性处理 10Gbps 的数据流.

Ang[56]实现了一个 TOE 系统.在 NIC 上,用 10MHz 的 i960 作为控制处理器,NIC 带有 16M 字节,66MHz 的 SDRAM.i960 与 MAC 芯片间的 PCI 总线为 32 位,66MHz,而 i960 与主机间通过 33MHz 的 32 位总线相连.主机上运行实时系统 RTX,在传输路径与接收路径上均作了优化,避免多次复制.结果表明,主机 CPU 的占用率在传输大容量报文时明显有所降低.实验同时表明,开销比预期设想的要高很多,每个到达的报文将花费上万个系统时钟周期.同时,结果还指出,硬件设计与缓存管理是影响性能的主要瓶颈.

但是,TOE 也有其缺点[57]:(1) 在 NIC 上实现传输层远比只在 NIC 上实现数据链路层处理复杂,因为摩尔定律的存在,专用而复杂的 NIC 会很快落后于通用 CPU 的性能,TOE 很可能会再次变成瓶颈;(2) 协议卸载后,硬件与主机之间接口设计相当困难;(3) TOE 可以将数据直接放在存储器中,但是需要上层协议的协同处理;(4) TOE 必须为每一个 TCP 链接维护状态,同时与主机保持一致;(5) 软件的不成熟可以依靠打补丁来改进,而硬件更新困难较大;(6) 由于 TCP 协议的复杂性,TOE 设计难度很大,设计错误很难定位;(7) 由于 TOE 的协议管理接口与原来的接口不兼容,导致操作系统很难实现与主机 TCP 相同的可见性状态管理.虽然这些问题最终都能解决,但是它们确实弱化了协议卸载的优点.

5.3 RDMA

为了更好地实现 0 复制 TCP 协议栈,同时避免往 NIC 中加入各种不同的应用协议,人们提出 RDMA 作为一种通用的方法.RDMA 在传输中包含一个唯一的 46 位的 ID,用来标志应用缓冲区,或者在 TCP 数据流中定义封装 RDMA 信息的格式,避免对 TCP 协议本身作修改.

但是,RDMA 也有缺点[57].RDMA 是显式的性能优化,如果不考虑具体操作系统结构,也不会获得很好的性能与灵活性.同时,RDMA 引入了许多与缓冲管理等相关的问题,如何解决好这些问题还缺乏成熟的研究成果;操作系统与 RDMA 之间的接口以及应用程序与 RDMA 之间的接口设计也十分困难;在安全方面,以前总是假设系统处于封闭的安全网络中,但是基于 IP 的 RDMA 应用往往将系统公开面对整个网络.

Romanow 等人[58]提出 RDMA 协议的功能实现框架 RDDP(remote direct data placement).RDMA 协议放置于 TCP 上,通过类似于信令的消息,与远程对等端协商源和目的缓冲的地址信息.对等端发送数据之前,在每一个数据报文中添加目的缓冲的基地址与相应的偏移量.当数据到达目的端时,由 NIC 根据报文中的地址与偏移量直接将数据移动到相应的缓冲.由于 TCP 是可靠保序的字节流传输协议,基于 TCP 的 RDMA 性能必然会受到 TCP 的影响,因此,Erdogan 等人[59]提出并分析了另一种基于 IP 的 RDMA 实现框架.

6 结束语

本文主要分析了 TCP 协议处理的开销和性能,从协议处理机制、实现方式以及处理结构等方面对加速技术的研究现状进行了总结.这些技术可以总结为以下几大类:

(1) 对协议处理本身的研究,包括 PCB 高速缓冲、首部预测以及优化校验和计算.

(2) 对处理代码的研究,包括 Cache 优化与延时隐藏.

(3) 对协议实现方式的研究,包括减少中断与避免复制.

(4) 对协议实现结构的研究,包括用户级传输协议、TOE 与 RDMA 协议.

近年来,许多研究指出,应该将 RDMA 与 TOE 进行结合,这样可以有效地将协议卸载与复制避免技术结合起来,对性能提升起着非常重要的作用.对 RDMA 与 TOE 结合之后的性能测量与分析,还有待于进一步深化.

References:

[1] Recio RJ. Server I/O networks past, present, and future. In: Proc. of the ACM SIGCOMM Workshop on Network-I/O Convergence: Experience, Lessons, Implications. New York: ACM Press, 2003. 163~178.

[2] Intel in Communications. 10 gigabit Ethernet technology overview. White Paper, 2003. http://www.intel.com/network/connectivity/resources/doc_library/white_papers/pro10gbe_lr_sa_wp.pdf

[3] InfiniBand: The next step in high performance computing. Voltaire Inc., 2002. http://www.hitachi-hitec.com/jyouhou/network/@gif/voltaire_20hpc.pdf

[4] Chase JS, Gallatin AJ, Yocum KG. End-System optimizations for high-speed TCP. IEEE Communications Magazine, 2001,39(4): 68~74.

[5] Mukherjee SS, Hill MD. A survey of user-level network interfaces for system area networks. Technical Report, 1340, Computer Sciences Department, University of Wisconsin-Madison, 1997. http://citeseer.ist.psu.edu/mukherjee97survey.html

[6] Rubini A, Corbet J. Linux Device Driver. O'Reilly & Associations, Inc., 2001.

[7] Bovet DP, Cesati M. Understanding the Linux Kernel. O'Reilly & Associations, Inc., 2001.

[8] Minturn D, Regnier G, Krueger J, Iyer R, Makineni S. Addressing TCP/IP processing challenges using the IA and IXP processors. Intel Technology Journal, 2003,7(4):39~50.

[9] Chu HKJ. Zero-Copy TCP in Solaris. In: Proc. of the USENIX 1996. http://citeseer.ist.psu.edu/chu96zerocopy.html

[10] Stevens RW. TCP/IP Illustrated, Volume 1: The Protocol. Addison Wesley, 1993.

[11] Yeh E, Chao H, Mannem V, Gervais J, Booth B. Introduction to TCP/IP offload engine. 2002. http://www.10gea.org/SP0502IntroToTOE_F.pdf

[12] Padhye J, Firoiu V, Towsley D, Kurose J. Modeling TCP throughput: A simple model and its empirical validation. University of Massachusetts Technical Report, CMPSCI TR 98-008. http://citeseer.ist.psu.edu/padhye98modeling.html

[13] Cardwell N, Savage S, Anderson T. Modeling TCP latency. In: Proc. of IEEE Infocom. 2000. http://citeseer.ist.psu.edu/context/1219150/273307

[14] Zheng D. On the modeling of TCP latency and throughput [MS. Thesis]. Mississippi State University, 2002.

[15] Altman E, Avrachenkov K, Barakat C. TCP in presence of bursty losses. In: Proc. of the 2000 ACM SIGMETRICS Int'l Conf. on Measurement and Modeling of Computer Systems. New York: ACM Press, 2000. 124~133.

[16] Mitzenmacher M, Rajaramany R. Towards more complete models of TCP latency and throughput. Journal of Supercomputing, 2001, 20(2):137~160.

[17] Fortune S, Wyllie J. Parallelism in random access machines. In: Proc. of the 10th Annual Symp. on Theory of Computing. New York: ACM Press, 1978. 114~118.

[18] Valiant LG. A bridging model for parallel computation. Communications of the Association for Computing Machinery, 1990,33(8): 103~111.

[19] Culler D, Karpy R, Patterson D, Sahay A, Schauser KE, Santos E, Subramonian R, von Eicken T. LogP: Towards a realistic model of parallel computation. In: Proc. of the 4th ACM SIGPLAN Symp. on Principles and Practice of Parallel Programming. New York: ACM Press, 1993. 1~12.

[20] Keeton KK, Anderson TE, Patterson DA. LogP quantified: The case for low-overhead local area networks. In: Proc. of the Hot Interconnects III. 1995. http://www.cs.washington.edu/homes/ tom/pubs/logp.html

[21] Shivam P, Chase JS. On the elusive benefits of protocol offload. In: SIGCOMM 2003 Workshop on Network I/O Convergence: Experience, Lessons, Implications (NICELI). New York: ACM Press, 2003. 179~184.

[22] Clark DD, Jacobson V, Romkey J, Salwen H. An analysis of TCP processing overhead. IEEE Communications Magzine, 1989, 27(6):23~29.

[23] Pink S. TCP/IP on Gigabit Networks. Swedish Institute of Computer Science, 1995.

[24] Guo CX, Zheng SR. Analysis and evaluation of the TCP/IP protocol stack of LINUX. 2000. http://www.ifip.or.at/con2000/icct2000/icct452.pdf

[25] Mukherjee SS, Hill MD. The impact of data transfer and buffering alternatives on network interface design. In: HPCA, ed. Proc. of the 4th Int'l Symp. on High-Performance Computer Architecture (HPCA). Washington: IEEE CS Press, 1998. 207~218.

[26] Buzzard G, Jacobson D, Mackey M, Marovich S, Wilkes J. An implementation of the Hamlyn sender-managed interface architecture. In: Proc. of the 2nd Symp. on Operating Systems Design and Implementation. New York: ACM Press, 1996. 245~259.

[27] Borman D, Research C, Partridge C. Computing the Internet checksum. RFC1071, 1988. http://www.faqs.org/rfcs/ rfc1071.html

[28] Mallory T, Kullberg A. Incremental updating of the Internet checksum. RFC1141, 1990. http://www.faqs.org/rfcs/rfc1141.html

[29] Rijsinghani A. Computation of the Internet checksum via incremental update. RFC1624, 1994. http://www.faqs.org/rfcs/rfc1624.html

[30] Kleinpaste K, Steenkiste P, Zill B. Software support for outboard buffering and checksumming. ACM SIGCOMM Computer Communication, 1995,25(4):87~98.

[31] Henriksson T, Persson N, Liu DD. VLSI implementation of Internet checksum calculation for 10 gigabit Ethernet. http://www.da.isy.liu.se/pubs/tomhe/DDECS2002_checksum.pdf

[32] Wright GR, Stevens WR. TCP/IP Illustrated, Volume 2: The Implementation. Addison Wesley, 1995.

[33] Barford P, Crovella M. Critical path analysis of TCP transactions. ACM SIGCOMM Computer Communication, 2001,31(2): 80~102.

[34] Goel A, Mitzenmacher M. Exact sampling of TCP window states. 2002. http://citeseer.ist.psu.edu/502712.html

[35] Xiong Y, Liu JC, Shin KG, Zhao W. On the modeling and optimization of discontinuous network congestion control systems. In: IEEE INFOCOM 2004. 2004. http://www.ieee-infocom.org/2004/Papers/58_1.PDF

[36] Avrachenkov K, Ayesta U, Brown P, Nyberg E. Differentiation between short and long TCP flows: Predictability of the response time. In: IEEE INFOCOM 2004. 2004. http://www.ieee-infocom.org/2004/Papers/16_2.PDF

[37] Baccelli F, Kim KB. TCP throughput analysis under transmission error and congestion losses. In: IEEE INFOCOM 2004. 2004. http://www.ieee-infocom.org/2004/Papers/58_3.PDF

[38] Sun JS, Zukerman M, Ko KT, Chen GR, Chan S. Effect of large buffers on TCP queueing behavior. In: IEEE INFOCOM 2004. 2004. http://www.ieee-infocom.org/2004/Papers/16_1.PDF

[39] Web100 Project. WEB100: Facilitating effective and transparent network use. 2000. http://www.web100.org/docs/statement_of_work.php

[40] Dalton C, Watson G, Banks D, Calamvokis C, Edwards A, Lumley J. Afterburner. IEEE Network, 1993,7(4):36~43.

[41] Druschel P, Peterson L. Fbufs: A high-bandwidth cross-domain transfer facility. In: Proc. of the 14th ACM Symp. on Operating Systems Principles. New York: ACM Press, 1993. 189~202.

[42] Rodrigues SH, Anderson TE, Culler DE. Highperformance local-area communication with fast sockets. In: USENIX, ed. Proc. of 1997 Annual Technical Conf. 1997. Berkeley: USENIX, 1997. 257~274.

[43] Culley P, Garcia D, Hilland J. An RDMA protocol specification. IETF Internet-Draft, 2003. http://www.ietf.org/proceedings/03mar/I-D/draft-ietf-rddp-rdmap-00.txt

[44] Rangarajan M, Bohra A, Banerjee K, Carrera EV, Bianchini R. TCP servers: Offloading TCP processing in Internet servers. Design, implementation, and performance. Technical Report, Department of Computer Science, Rutgers University, 2002. http://citeseer.ist.psu.edu/rangarajan02tcp.html

[45] Solá-Sloan JM. UDP, TCP, and IP fragmentation analysis and its importance in TOE devices. 2003. http://mayaweb.upr.clu.edu/crc/crc2003/papers/JuanSola.pdf

[46] Mosberger D, Peterson LL, Bridges PG, O'Malley S. Analysis of techniques to improve protocol processing latency. In: Proc. of ACM SIGCOMM'96. ACM, 1996. 73~84. http://citeseer.ist.psu.edu/mosberger96analysis.html

[47] Yocum KG, Anderson DC, Chase JS, Gadde S, Gallatin AJ, Lebeck AR. Balancing DMA Latency and Bandwidth in a High Speed Network Adapter. Technical Report, CS-1997-20, Department of Computer Science, Duke University, 1997.

[48] Kaiserswerth M. The parallel protocol engine. IEEE/ACM Trans. on Networking, 1993,1(6):650~663.

[49] Nordqvist U, Liu DK. A comparative study of protocol processors. In: Proc. of CCSSE. 2002. http://www.da.isy.liu.se/pubs/ulfnor/ulfnor-ccsse2002.pdf

[50] Mapp G, Pope S. The design and implementation of a high-speed user-space transport protocol. In: Proc. of the 1st Int'l Workshop on High Performance Protocol Architectures. 1994. http://citeseer.ist.psu.edu/metzler94design.html

[51] Thekkath CA, Nguyen TD, Moyt E, Lazowska ED. Implementing network protocols at user level. IEEE/ACM Trans. on Networking, 1993,1(5):554~565.

[52] Edwards A, Muir S. Experiences implementing a high performance TCP in user-space. In: Proc. of SIGCOMM'95. New York: ACM Press, 1995. 196~205.

[53] Shivam P, Wyckoff P, Panda D. EMP: Zerocopy OS-bypass NIC-driven gigabit Ethernet message passing. In: Proc. of the 2001 ACM/IEEE Conf. on Supercomputing. New York: ACM Press, 2001. 57~57.

[54] Adaptec Corporation. Advantages of a TCP/IP offload ASIC. 2004. http://graphics.adaptec.com/pdfs/tcpio_adv_wp.pdf

[55] Henriksson T, Nordqvist U, Liu DK. Embedded protocol processor for fast and efficient packet reception. 2002. Technical Report, http://www.da.isy.liu.se/pubs/tomhe/ICCD2002.pdf

[56] Ang BS. An evaluation of an attempt at offloading TCP/IP protocol processing onto an i960RN-based iNIC. 2001. http://www.hpl.hp.com/techreports/2001/HPL-2001-8.pdf

[57] Mogul JC. TCP offload is a dumb idea whose time has come. In: HotOS 2003. 2003. http://www.usenix.org/events/hotos03/tech/full_papers/mogul/mogul_html/

[58] Romanow A, Bailey S. An overview of RDMA over IP. Technical Report, 2003. http://datatag.web.cern.ch/datatag/pfldnet2003/papers/romanow.pdf

[59] Erdogan O, Patel PK. Design and implementation of RDMA as a best-efforts service and providing reliability over it. Technical Report, 2003. http://www.stanford.edu/~priyank9/projects/