计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(6):240−243 [doi: 10.15888/j.cnki.csa.006388]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 DPDK 的高速数据包捕获方法①

任昊哲, 年　梅

(新疆师范大学 计算机科学技术学院, 乌鲁木齐 830054)
通讯作者: 任昊哲, E-mail: renhaozhe2010@qq.com

摘　要: 不断增大的网络带宽给人们提供了丰富的网络应用和服务的同时, 也给传统的数据俘获系统带来了挑战. 本系统基于 Intel 数据平面开发套件 (Data Plane Development Kit, DPDK) 设计了高速网络数据包捕获软件. 能够更好适应目前校园网高速网络数据包捕获的要求, 为网络数据包分析提供技术支持. 最后, 本文对基于 DPDK 的数据包捕获系统和传统 Libpcap 进行了实验结果对比, 表明基于 DPDK 的数据包捕获系统能够明显提升高速网络出口数据俘获的性能.

关键词: 数据平面开发套件; 数据包捕获; 网络数据包捕获函数包

引用格式: 任昊哲,年梅.基于 DPDK 的高速数据包捕获方法.计算机系统应用,2018,27(6):240–243. http://www.c-s-a.org.cn/1003-3254/6388.html

Method for High Speed Network Packet Capture Based on DPDK

REN Hao-Zhe, NIAN Mei

(College of Computer Science and Technology, Xinjiang Normal University, Urumqi 830054, China)

Abstract: As the bandwidth of network become higher, it can provide a lot of web applications and services. But it also adds challenges to traditional system of data capture. In this study, we develop a high speed network packet capture software based on Intel Data Plane Development Kit (DPDK). This software can be used to capture the data of the high-speed campus network and provides technical support for network data package analysis. Lastly, we test this system and compare it with the packet capture system based on Libpcap. Experimental results show that this packet capture system can improve the performance of capturing high speed network data.

Key words: DPDK; data capture; Libpcap

引言

网络技术的快速发展, 使网络应用越来越广泛, 学生的学习和娱乐离不开校园网的支持, 也使得校园网出口数据流量越来越大. 为了保障校园网系统的顺利运行、准确分析网络内容的安全, 需要对校园网进行内容监测和审计. 网络监测和审计实现的前提是对校园网络出口进行数据俘获, 现在各高校出口流量达到万兆以上, 高速准确的数据包俘获系统对校园网管理的作用越来越重要. 目前数据包俘获系统主要有两种[1], 一种是定制硬件设备的包捕获系统, 使用专有设备, 性能强大、丢包率低, 但部署价格昂贵, 功能单一、扩展性差, 不适合研究人员使用. 另一种是软件包俘获系统, 系统成本低、灵活性高, 可以灵活部署俘获所需的数据包, 从而得到了研究者的青睐, 成为目前网络内容监测、网络安全等研究中广泛使用的技术.

传统软件数据包捕获通常采用基于操作系统内核的旁路机制, 仅在系统内核协议栈处理数据包时捕获数据包, 整个数据包捕获都或多或少的依赖操作系统内核协议栈, 而操作系统内核收发包需要首先由网卡触发中断, CPU 将数据包从网卡缓存拷贝到内核内存

① 基金项目: 赛尔网络下一代互联网技术创新项目(NGII20160604); 国家自然科学基金(61163064); 教育部人文社会科学工程科技人才培养专项(15JDGC022); 新疆师范大学“数据安全”重点实验室,新疆师范大学“计算机应用技术”重点学科资助
收稿时间: 2017-10-08; 修改时间: 2017-11-01; 采用时间: 2017-11-06; csa 在线出版时间: 2018-05-28

空间, 经过内核协议栈处理后, 再将数据包拷贝到用户态内存空间. 此过程中需要消耗大量 CPU 资源用于处理中断、多次内存拷贝以及系统调用[2]; 此外系统普通内存页只有 4 KB, 内存访问速度慢, 协议栈处理也将造成大量的性能消耗[3]. 以上原因导致软件数据包俘获中消耗了大量资源, 使得网络高负载时系统资源被耗尽造成了数据包捕获时大量丢包.

为了提高软件数据包捕获的效率解决软件数据俘获系统中的丢包问题, 研究人员针对性的进行了优化并设计出了不少高速数据包俘获框架, McCanne 与 Jacobson 提出了一种基于 Unix 内核的伯克利封包过滤器 BPF 的数据包过滤机制, BPF 可按照规则过滤无用数据包以提升数据包捕获性能[4]. tcpdump、Libpcap 都使用到了这种机制, 但这种机制没有改进捕获流程中系统资源消耗过大的问题.

为了解决系统资源大量被中断处理消耗的问题, Linux 在 2.5 内核之后开始使用 NAPI(NEW API) 来处理接收到的数据包, 其主要进行了以下改进[5]:

(1) 第一个报文到达时触发网络适配器硬件中断, 将该适配器放入轮询表并关闭中断请求.

(2) 系统激活一个软中断, 在处理函数中对轮询表上的设备进行轮询, 处理数据包.

(3) 直到本次处理时间片用完或者报文接收处理完毕, 重新开启该网络适配器中断请求. NAPI 可以明显减少硬中断, 提高高速流量负载网络下的报文捕获性能. Luca Deri 开发了数据包捕获函数库 pf_ring. 它将网卡接收的数据包储存在内核层的一个环状缓存中, 网卡通过驱动程序支持的 NAPI 像缓存中写入数据. 应用程序通过 MMAP 直接读取缓存数据, 消除了数据包从内核态到用户态的内存拷贝, 极大的提高了数据包捕获的效率.

但是, 以上方法主要是针对软件数据俘获中某一个方面的缺陷进行了改进, 随着校园网出口带宽的不断增长, 对出口数据俘获的速度增长要求越来越强烈, 以上措施依然无法完全解决丢包的问题, 还需要针对软件解包中的各种问题进行解决, 从而最大限度体提升数据俘获的性能, 满足高速网络出口数据的获取需要.

1 Intel DPDK 的基本原理

为了更全面地解决软件方式的数据包转发和捕获效率低下的问题, 6WIND, Intel 等多家公司, 针对 Intel 的 CPU 和网卡开发了数据包转发处理套件 DPDK. DPDK 是一套强大、高度优化的用于数据包处理的函数库和驱动集合, 可以帮助用户将控制面和数据面平台进行整合, 从而能有效地执行数据包处理[6], 可以极大地提高数据处理性能和吞吐量并提高效率. 和传统的网络数据包俘获方式相比 DPDK 主要进行了以下改进:

(1) DPDK 使用轮询模式驱动 (Poll Mode Drivers, PMD) 代替了传统模式通过中断的网卡接收和发送数据包的工作方式, 将收到的数据包通过直接内存存取模式 (Direct Memory Access, DMA) 传输到内存中并直接交由应用程序处理从而实现了零拷贝, 极大地提升了收发包的性能.

(2) 运行在用户空间的 I/O 技术 (UIO): 使用 UIO 机制使网卡驱动程序运行在用户态, 将原本在内核态的处理的工作直接交由用户态应用程序处理, 避免了不必要的内核态和用户态之间的系统调度, 提高了执行效率.

(3) 大内存页面技术: DPDK 通过绑定 2 MB 或者是 1 GB 的 huge 内存页来代替传统的 4 KB 普通页, 提高内存使用效率让程序尽量独占内存防止内存换出, 扩大页表提高 hash 命中率, 提升数据俘获中页面查找的速率.

(4) CPU 亲和性: 利用 CPU 亲和性主要是将控制面线程以及各个数据面线程绑定到不同的 CPU 内核, 省却了反复调度的性能消耗, 同时支持 NUMA 架构尽量访问本端内存.

(5) DPDK 使用了 rte_mbuf 结构来存储数据包, 将数据结构体部分和数据部分合在一起, 因此只需要分配一次内存即可, 进一步节省了分配内存开销, 提高了数据接收和存储的速度.

综上所述, DPDK 针对传统软件俘获数据包存在的问题, 全面地提出了相应的解决方案. 此外, 从而能够通过 DPDK 开发高效的数据包捕获软件系统, 解决传统网络数据包俘获中丢包问题, 实现准确高速网络信息的获取.

2 基于 DPDK 数据包俘获软件系统的设计与实现

DPDK 提供了 x86 平台下的报文数据包处理库和驱动集合, 包括数据包的接收和发送等模块, 为开发高

速数据包俘获系统提供了必须的接口, 其包含的模块集合如图 1 所示主要的数据包接收模块.

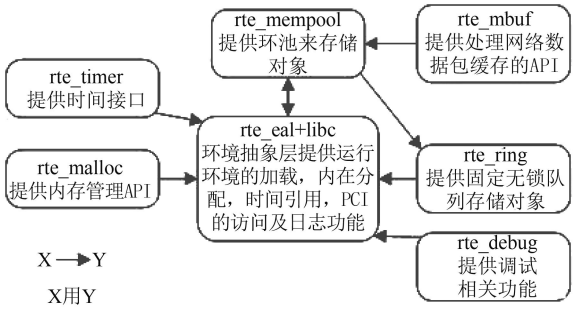

图 1 DPDK 模块集合

利用 DPDK 提供的模块实现网络数据捕获首先需要进行 CPU 和网卡的绑定, 接着申请大内存页, 然后使用接收数据包读取网卡上接收的数据, 最后将数据进行保存. 具体实现流程图如图 2 所示, 主要包括了以下几个步骤:

(1) 初始化环境抽象层 (Environment Abstraction Layer, EAL): 使用 rte_eal_init(argc, argv) 模块, 获取系统可用 CPU 的数量、建立日志文件、查询可用 PCI 网卡设备、挂载巨页内存并申请可用内存、创建 lcore 主线程绑定 CPU 并最终完成 EAL 的初始化;

(2) 创建网络数据包的缓存池队列以及存储结构: 使用 init_mbuf 相关 API 初始化 pools 以及 mbuf;

(3) 初始化抓包端口进行并配置: 使用 init_port (ports->id[i]) 查看和配置对应端口的状态;

(4) 读取接收到的数据包: 使用 rte_eth_rx_burst 接收数据;

(5) 使用自定义函数保存 PCAP 文件: 调用自定义函数赋值 PCAP 结构体构造 PCAP 文件, 完成后调用写盘接口将文件写入磁盘, 完成数据包捕获.

至此, 该系统将所有发送到该网卡的网络数据包抓取并保存在一个标准 PCAP 文件中, 实现了数据包的捕获和保存工作.

3 实验分析

为了测试系统性能, 本文分别对使用 DPDK 和 LIBPCAP 搭建的网络数据俘获系统进行了平行实验测试. 首先选择了两台联想 PC 机, 均配备了万兆网卡, 一台作为数据包发送机, 一台为数据包抓取机, 在实验中分别发送 64 B、512 B 和 1500 B 的数据包进行测试. 实现对不同捕包平台和不同大小的数据包、丢包率进行比较. 其中收包率是指实验中实际收到的数据包所占发出的数据包的比例. 测试的结果分别如图 3 和图 4 所示.

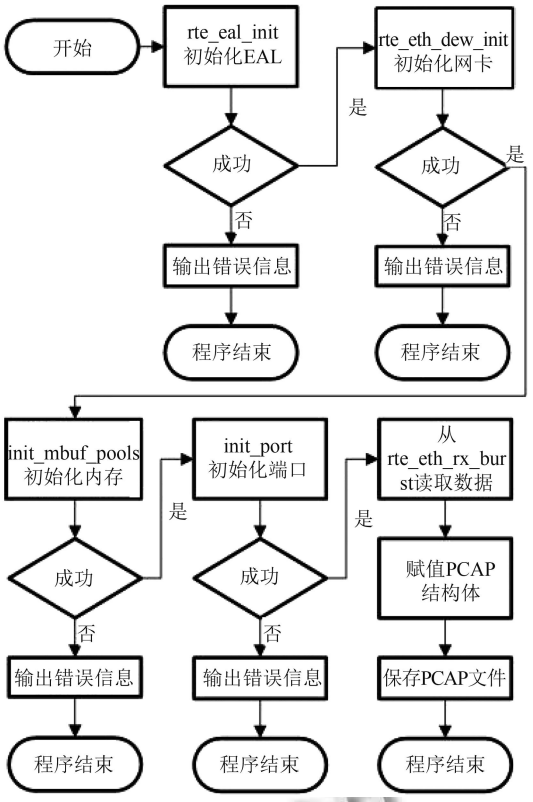

图 2 数据包捕获流程图

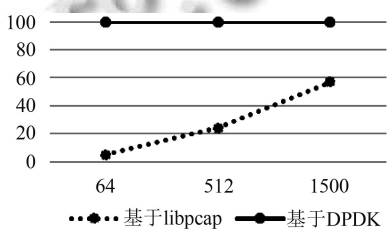

图 3 千兆带宽下的数据包捕获率

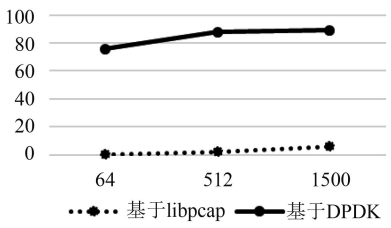

图 4 万兆带宽下的数据包捕获率

由以上实验结果可以看出传统模式下基于 Libpcap 的数据包捕获系统在千兆网络下已经有了大量的丢包,

而基于 DPDK 的数据包捕获系统则能在千兆网络下达到线性速度, 在万兆网络下除了 64 B 数据包捕获率为 75.6%, 512 B 和 1500 B 的数据包捕获率都接近 90%, 分析其原因主要在于 Libpcap 工具存在大量的系统调用和进程上下文切换开销, 大部分系统资源都浪费在两次内存拷贝和系统调用中. 经过测试基于 DPDK 的数据包捕获系统, 极大地提升了数据包捕获的性能, 并且能够按照应用的要求进行网络应用数据采集的扩展和配置.

4 结束语

本文针对高速大容量网络出口数据俘获的需要, 并在对 DPDK 的相关技术特点以及软件数据包捕获机制进行分析的基础上, 设计实现了基于 DPDK 的软件数据包俘获系统, 通过与传统数据包俘获软件进行实验对比证明, 该系统在数据包捕获率方面有较大的性能提升, 系统虽然最终受磁盘写入性能以及实验机的 CPU 核心数的限制并没有达到万兆线性速度, 但也为后续优化并达到万兆网络线性速度数据包捕捉提供了思路, 同时也为高速数据过滤提供了技术参考.

参考文献

1 徐慧, 姜恒, 杨林. PF_RING 高效数据包捕获技术研究与设计. 计算机科学, 2012, 39(10S): 88–89, 114.
2 王佰玲, 方滨兴, 云晓春. 零拷贝报文捕获平台的研究与实现. 计算机学报, 2005, 28(1): 46–52.
3 王佰玲, 方滨兴, 云晓春. 传统报文捕获平台性能影响因素分析. 计算机工程与应用, 2003, (22): 151–152. [doi: 10.3321/j.issn:1002-8331.2003.22.049]
4 杨铭. 伯克利数据包过滤器的探索与研究. 科技创新与应用, 2014, (33): 92.
5 张楠. 基于 IP 网络的通用数据采集系统的设计与实现[硕士学位论文]. 北京: 北京邮电大学, 2015.
6 赵宁, 谢淑翠. 基于 dpdk 的高效数据包捕获技术分析与应用. 计算机工程与科学, 2016, 38(11): 2209–2215. [doi: 10.3969/j.issn.1007-130X.2016.11.008]