高技术通讯 2010 年 第 20 卷 第 10 期:991 ~ 995

doi:10.3772/j.issn.1002-0470.2010.10.001

网络处理器和通用处理器相结合的流量识别系统①

李云春② 秦先龙③ 王 啸

(北京航空航天大学计算机学院 北京 100191)

摘 要 根据网络处理器(NP)与通用处理器(GPP)的特点,提出了一种结合 NP 和 GPP 的流量管理系统结构,基于此系统结构,设计了一种流量识别系统,并描述了系统中核心部分——NP 与 GPP 的通信机制的设计和实现。在这种系统中,NP 作为快速数据通信进行数据包收发、流存储等处理,并根据需要将部分数据包通过 21555 非透明桥传给 GPP;GPP 作为慢速数据通道使用基于深度包检测的 L7-filter 模块对流量进行识别,并将识别的结果传给 NP,NP 根据结果对流量进行控制管理等操作。

关键词 流量识别,网络处理器,PCI 通信,深度包检测,流量管理

0 引 言

随着网络的迅速发展,网络流量表现出多样化,这对网络流量管理提出了新的挑战。例如,P2P 应用的出现,使流量识别变得越来越困难。流量识别的基本识别方法有基于端口的识别方法与基于有效负载的识别方法。基于端口的识别方法通过检查数据包的端口号,看其是否是特定协议的端口来进行识别,对于现有的一些网络流量,尤其是 P2P 流量,应用这种方法不再可靠;基于有效负载的识别方法是对数据包中的固定字段和内容进行匹配[1],以实现对流的识别,但是这种方法只对一些流量有效,对一些加密或不断改变协议特征字段没有作用。此外,有学者提出基于统计的流量识别方法和基于协议特征的流量识别方法,但这些方法还处于研究阶段,在实际应用中还存在一些问题。针对这些方法的缺点,近年来,有学者提出深度包检测技术进行流量识别,这种技术不但能用于流量识别,还可用于病毒及有害代码的检测。其本质是基于特征的识别方法,但它使用的特征不是简单的固定字段,而是通过正则表达式表示的复杂特征,由于正则表达式具有灵活性的特点,对于多变的网络流量,能够准确地进行识别。但是,这种方法由于要对正则表达式进行解析,对硬件的性能要求比较高。

流量识别实现的硬件平台大体上分为三种:通用处理器(GPP),专用集成电路(ASIC)以及网络处理器(NP)。GPP 由于其采用通用操作系统,操作系统调度的任务繁多,限制了其计算性能。ASIC 具有很好的计算性能,但由于其开发难度大,开发周期长,可扩展性差,限制了其广泛应用。ASIC 虽然可以达到很大的吞吐量,但针对应用层,如统一威胁管理(unified threat management,UTM)设备的某些智能功能,ASIC 难以实现,即使是 ASIC 芯片融合部分应用层安全威胁的控制,也不能做到快速地扩充和升级。NP 介于这两者之间,具有接近于 ASIC 的高性能,同时也具备 GPP 的可编程性。但是,由于 NP 没有用专门的硬件单元进行正则表达式解析,因而只能用软件的办法进行正则表达式解析,而要在 NP 上使用软件进行正则表达式解析非常复杂,且对 NP 的资源占用较大(如占用一个或多个核进行处理),这样会影响 NP 的效率。因此,本文结合 NP 与 GPP 的特点,利用 GPP 来解析正则表达式,对流量进行识别,将识别的结果传递给 NP,对报文进行快速处理。

1 相关研究

早期的流量识别方法是根据端口号进行识别,但随着 P2P 等软件采用随机端口号进行通信,该识别方法受到了一定的局限。基于有效负载的方法准确性较高,并且可以用于实时的流分类系统,因而是

① 863 计划(2009AA01A129)资助项目。
② 男,1972 年生,博士,副教授;研究方向:网络管理,网络测量,自主计算;E-mail: lych@buaa.edu.cn
③ 通讯作者,E-mail: rainconut@gmail.com
(收稿日期:2009-06-15)

目前绝大多数流量管理系统选择使用的方法[2,3]。针对这种方法,一些P2P软件的开发者使用变动有效负载的方法来防止被识别,且针对一些私有协议采用加密的方式进行传输,使这种方法失去作用。基于统计的流量识别方法,如Moore采用有监督的朴素贝叶斯(Naïve Bayesian)分类方法进行流量分类与应用识别[4]的方法(NB方法),首先把网络流量数据手动进行分类,确定流量的具体应用类型,并把流量数据分成训练集和测试集。为了评估NB方法的性能,每个数据集依次作为训练集输入到朴素贝叶斯分类器中,其他的数据集作为测试集进行评估,获得的平均分类准确性超过了83%。Roughan采用最近邻和线性判别分析的方法[5],以连接持续时间和平均包的大小作为流量分类的特征向量,仍然采用贝叶斯的方法进行分类,但是,只采用两个属性的统计信息并不能区分所有的应用类别,因此获得的准确度很低。

学术界和工业界对网络处理器(NP)进行了大量的研究。学术界较有代表性的研究有Princeton大学的可扩充路由器VERA[6]。国内外的主要网络公司也较多地使用NP开发产品,如思科7XXX系列路由器,华为Quidway NE5000E核心路由器以及NE40、NE80路由器,联想Super V-5000防火墙等。随着网络的发展,只使用某种单一的硬件架构已不能满足多方面的需求,需要采用混合结构,常用的有结合ASIC与NP,ASIC与GPP,但ASIC架构比较复杂。文献[7]提出了一种结合Intel IXP系列NP与GPP的架构,这种架构结合了NP与GPP的优点,具有一定的优势。

2 结合NP和GPP的流量管理系统结构

结合NP和GPP的流量识别方法,结合NP数据包处理效率高与GPP可扩展性强的优点来实现网络流量识别。NP形成快速数据处理通道,GPP形成慢速数据处理通道。利用NP接发网络包,并将数据流进行流分类,每条数据流开始的少量数据包(如前10个数据包)通过21555非透明桥芯片传到GPP;由于连接NP和GPP的PCI总线带宽为66MHz/64Bit,若传送过多的数据包会成为一个瓶颈,因此传送到GPP的流量为每条数据流开始的前几个数据包,而不是数据流中所有数据包。GPP在接收到传来的数据包后,对数据包进行深度包检测识别流量的类型,并将识别出的流量类型结果传递到NP上,以便NP对数据流进行处理和控制。

为了提高系统的效率,本文基于流量转发与识别分离的机制,提出一种结合NP和GPP的流量管理的系统结构。一般而言,数据包转发与识别是串行的,串行处理的优点是结构简单,而且可以对一条流的所有数据包进行控制,但也很明显,串行会对转发速度产生影响,影响的程度取决于识别的效率,例如,采用基于深度包检测的应用层识别方法的识别速度会大大小于转发速度,这也是应用层识别的问题所在。基于这点,本文将转发与识别两个过程分离,其设计思想是:在识别一条流的前N个数据包时,并不等待识别结果后再进行转发,而是直接转发,直到识别出流量类型后,再根据相应的配置策略进行控制或整形。这种方法主要针对于P2P流量的管理,而这种流量的存活期较长,因此在识别出结果后再对其进行控制或整形,不会影响管理的效果。基于流量转发与识别分离的机制的结合NP和GPP的流量管理系统结构如图1所示。

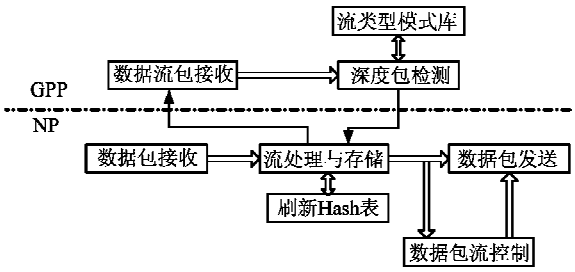

图1 结合NP和GPP的流量管理系统结构图

NP部分实现数据包接收模块、数据包发送模块、刷新Hash表模块、流处理与存储模块、数据包流控制模块,GPP部分实现数据流包接收模块、深度包检测模块、流类型模式库。在这个系统结构中对流量识别的步骤如下:

(1)NP中的流处理与存储模块一方面从数据包接收模块中获取数据包的五元组流信息,并将五元组流信息进行Hash存储,另一方面向GPP中的数据流包接收模块发送流的前$N(N = 10)$个数据包。

(2)GPP中的深度包检测模块对数据流包接收模块接收到的数据包依据流类型模式库中的模式进行深度包检测处理,并根据需要与否,将流处理结果传输给NP中的流处理与存储模块。

(3)NP中的数据包流控制模块根据流处理与存储模块输出的数据包信息进行流量控制,且该流量经数据包发送模块向外发送。

NP 部分分为两层:数据层和控制层。数据层包含 8 个微引擎,负责对数据包的接收、发送、其它处理等。控制层采用 Xscale 处理器,负责对数据层进行初始化、配置,处理数据层传来的异常数据包,如地址解析协议(ARP)包、网际控制报文协议(ICMP)包等。NP 和 GPP 的通讯通过 21555 非透明桥芯片采用 PCI 协议进行通信。流量识别算法使用开源的 L7-filter 软件。L7-filter 基于 Linux 的 Netfilter 机制,使用深度包检测(deep packet inspection, DPI)技术,它对数据内容根据确定的模式(模式用正则表达式表示)进行匹配。

3 NP 与 GPP 的通信设计

结合 NP 和 GPP 的系统结构的一个关键问题是 NP 与 GPP 之间的通信,主要包括如何收发控制消息与数据报文。本文设计的系统中通过 21555 非透明桥芯片连接 NP 和 GPP 两端的 PCI 总线。非透明桥的主要特点在于它使得其两端的系统内存互见,这样为数据的传输提供了便利。

3.1 消息机制

如图 2 所示,为了完成消息与数据交互,需要使用 21555 桥上的两类寄存器:32 位中断寄存器 IRQ 和 8 个 32 位 Scratch 寄存器。中断寄存器用于消息中断、报文接收、报文发送中断等。8 个 Scratch 寄存器用于消息的传送,共分为两组,每组 4 个作为 NP 到 X86 或 X86 到 NP 的通信消息区。

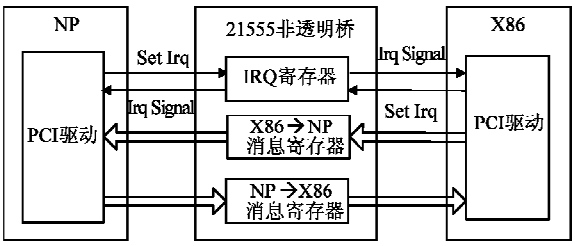

图 2 PCI 通信消息机制结构图

消息机制的关键是使用 Scratch 寄存器定义消息的类型及相关数据。消息类型包括测试消息 ECHO/ECHOREPLY 消息、握手消息 HELLO/HELLOACK 消息、窗口地址映射消息等。每种消息有对应的 ID 号,如用 0×1 表示 HELLO 消息,0×2 表示 HELLOACK 消息。在消息传递时,ID 号写在对应 4 个 Scratch 寄存器中的第 1 个寄存器内容,另外 3 个寄存器传送具体消息内容,对于 ECHO、ECHOACK、HELLO、HELLOACK 这 4 种消息,具体消息内容为空。

21555 上的 Scratch 寄存器是 X86 和 NP 都可以访问的用户自定义寄存器组,双方可以通过此进行通信,传递彼此的状态、地址等信息给对方使用,本文利用这组寄存器传递各种消息,包括控制消息、数据消息等。Scratch 寄存器有 8 个寄存器,每个寄存器是 32 位。X86 端使用其中的前 4 个寄存器,可读可写,通过这 4 个寄存器放置需要传递给 NP 端的各种消息,而 NP 通过读取前 4 个寄存器获得需要的信息,并且置位某些寄存器位表明读写结果;NP 端使用其中的后 4 个寄存器,可读可写,通过这 4 个寄存器放置需要传递给 X86 端的各种消息,而 X86 通过读取后 4 个寄存器获得需要的信息,并且通过置位某些寄存器位传递读写结果。消息用于在 NP 和 X86 之间传递彼此的信息以及自己的参数等给对方,消息通过 Scratch 寄存器以及中断机制传递。

3.2 报文收发机制

报文收发机制建立在消息机制之上,首先要利用消息机制进行报文窗口地址映射。由于 21555 桥芯片的非透明性,X86 与 NP 可以互见对方的内存空间,这样 NP 和 GPP 两端只需要将各自的地址预留空间通用消息机制通知对方即可实现数据报文的传输。

以 X86 为 HOST 端,报文收发包括报文接收与报文发送两种。对于报文接收,即 NP 端需要将数据包往 X86 端传,通过 21555 中断寄存器发出报文中断,通知 X86 端接收报文。X86 收到报文接收中断后,从相应的报文缓冲区中依次取出报文。显然,为了在 NP 和 X86 端传送报文,需要设立报文缓冲区和对应的描述符和报文缓冲区。报文传输有两种方式:普通传输方式和直接存储器访问(DMA)传输。普通传输方式直接将报文拷贝到报文窗口,21555 桥芯片根据映射信息将报文传输到 X86 端,DMA 是为了提高数据的传输效率。两种方法传输数据具有不同的效率,根据实验测试的结果,DMA 对于大于 1024 字节的数据传输较普通传输方式效率高,而对于小于 1024 字节的数据传输不如普通方式效率高。根据这个测试结果,在数据传输时,系统根据传输数据的大小灵活采用两种方式进行传输,以提高整体的传输效率。

3.3 虚拟网卡驱动

网络处理器与通用处理器通信通过 PCI 总线,总线工作在 66bit/66MHz 下,中间通过非透明桥 Intel

21555 进行数据通信，通过非透明桥，NP 与 GPP 两端的内存可以通过映射彼此互见。如前所述，NP IXP2400 硬件包含 Xscale 和微引擎两部分。Xscale 运行 vxWorks 操作系统，通用 PCI 驱动对 21555 桥芯片进行初始化设置。

对于 GPP 部分，运行 Linux 操作系统；PCI 驱动不进行 21555 芯片的初始化设置，它只进行数据的接收与发送等操作。PCI 驱动与应用模块的交互可以直接进行，但根据需要，数据包需要 Linux 网络协议栈的处理，例如，对流量识别，可以利用 Linux 下的开源 L7-filter 软件对数据包进行识别。这是结合 NP 和 GPP 的目的之一，对不需实时处理的报文，可以在 GPP 上进行处理，利用通用操作系统资源丰富、便利、可移植等优点，提高系统的可扩展性。在这里需要设计虚拟网卡驱动，虚拟体现在没有用到的网卡，将 PCI 驱动接收到的数据传输到 Linux 的网络协议栈中。另外，中断可以用 21555 的中断寄存器进行虚拟，即当 NP 发送完数据后，发出中断，虚拟网卡驱动对此中断进行注册，从而可以接收到中断，进而调用中断处理程序，完成报文的处理。

数据包的接收与发送是由 21555 驱动与虚拟网卡驱动共同完成的，如图 3 所示。21555 实现了芯片中断处理函数，当 NP 要上传数据包时，首先发出一个中断，21555 驱动接收到中断后触发中断处理函数根据中断号调用相应的处理函数，如数据传输中断对应 pci _ data _ rx _ task 函数，该函数调用虚拟网卡驱动提供的数据包接收函数 vnetdev _ skb _ recv，该函数申请内存空间保存数据包，并解析数据包为何种协议（以太网或其它，主要讨论以太网）并将数据包提交给 Linux 内核网络协议栈，通过调用netif _ rx 函数完成。

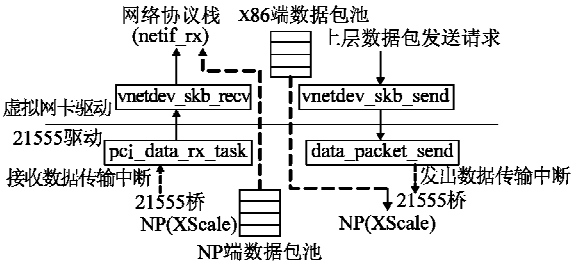

图 3 虚拟网卡数据包收发流程

当 Linux 上层需要发送数据包到 NP 时，首先调用虚拟网卡提供的包发送函数 vnetdev _ skb _ send，它将数据包内容放到 X86 端数据包池中，并调用 21555 桥驱动中的 data _ packet _ send 函数发送数据包，该函数发出数据传输中断给 21555 桥，21555 桥收到中断后通知 NP 端接收数据。

4 实验与结果

实验共分为两部分：系统的吞吐量以及识别的准确率。系统的吞吐量主要是网络处理器数据层处理数据包的速率。根据测试结果，在使用 Intel 的原始软件开发工具箱（SDK）不添加其它模块的情况下，IXP2400 所能达到的最大速率为 2.5Gbps。本文介绍的系统在数据层添加的模块主要有流处理与存储模块、Hash 表刷新模块、数据流控制模块。由于 Hash 表刷新模块是每过一段时间（默认 5 分钟）刷新 Hash 表，而且它使用独立的线程去执行，所以在测试时它的执行时间相当于没有记录。数据流控制模块根据通用处理器的识别结果以及手工配置的控制速率对数据流进行流控，所以在测试时该模块暂时不用，因而系统加入的模块主要是流处理与存储模块。

测试采用 SmartBit 600B 测试仪，由于测试吞吐量持续时间长，需要大量的数据，所以只有测试准确率时采用准备的已知数据进行测试，测试吞吐量采用 SmartBit 自身产生的数据进行。测试端口有 4 个，即将 SmartBit 的 4 个端口与我们系统的 4 个端口相连，同时发送数据，结果为 4 个端口之和。实验结果如图 4 所示。

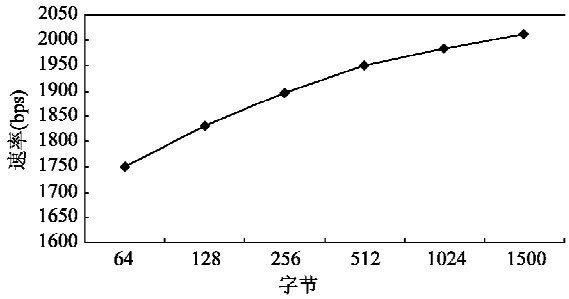

图 4 系统吞吐量

从图 4 中可以看出，当数据包大小为 64 字节时，性能为 1.7Gbps 左右；当数据包大小达到 1000 字节以上时，性能大约在 2Gbps 左右。即满足一般的中大型网络的性能要求。

系统的准确率测试使用已知的数据集进行测试，数据是在我们所在的网络环境捕获的，然后通过查看端口、IP 等方式将流类型得出，将无法辨认的

流剔除,这样得到一组已知的数据流。共200条流,其中有58条HTTP、12条FTP(使用Filezilla)、22条迅雷流、40条BT流、20条EMAIL流、18条DNS流、15条QQ流、15条eMule流。测试使用L7-filter模块,所用深度包检测的匹配模式从L7-filter的网站[8]上下载最新的匹配模式。所有200条已知流采用重放的方式传给L7-filter模块,均全部正确判出。从实验结果可见,结合NP和GPP的系统结构可在保持流量识别精度的条件下,保证流量识别的高效率。

5 结 论

本文通过结合NP和GPP的优势,提出一种基于数据流转发与识别分离机制的结合NP和GPP的系统结构,并对其关键的通信机制进行了设计和实现。并据此系统结构设计了一种流量识别与控制系统。通过实验证明,使用该系统结构实现的流量识别系统在数据包处理性能上具有较大的优势。该系统结构还适用于其它网络应用系统,如入侵检测系统等。类似于本文的实现方法,使用GPP上运行的开源入侵检测系统Snort软件,结合NP的快速数据转发,可提高开源软件Snort的效率。由于流量识别采用深度包检测的方法,本文采用GPP上运行软件的方法实现正则表达式的解析,虽然根据本文所述的转发和识别分离思想,识别过程并不会太影响系统的转发效率,但这只针对一般的P2P流量识别有效,对于其它系统如入侵检测系统则并不一定有效。因此可以考虑采用硬件的方法,如基于Cavium公司设计的正则表达式解析器RegExp芯片来加速正则表达式的解析,提高系统的运行效率。

参考文献

[1] Sen S, Spatscheck O, Wang D. Accurate, scalable In-Network Identification of P2P Traffic Using Application Signatures. In: Proceedings of the 13th International Conference, New York, USA: ACM Press, 2004. 512-521

[2] Dews C, Wichmann A, Feldmann A. An analysis of internet chat systems. In: Proceedings of the 3rd ACM SIGCOMM Conference, New York, USA: ACM Press, 2003. 51-64

[3] Haffner P, Sen S, Spatscheck O, et al. Automated construction of application signatures. In: Proceedings of SIGCOMM'05 MineNet Workshop, New York, USA: ACM Press, 2005. 197-202

[4] Moore A, Zuev D. Internet traffic classification using bayesian analysis techniques. In: Proceedings of the 2005 ACM SIGMETRICS International Conference, New York, USA: ACM Press, 2005. 50-60

[5] Roughan M, Sen S, Spatscheck O, et al. Class-of-service mapping for QoS: a statistical signature-based approach to IP traffic classification. In: Proceedings of the IMC'04, Taormina, Sicily, Italy. 2004. 135-148

[6] Karlin S, Peterson L. VERA: An extensible router architecture. *Computer Networks*, 2002, 38(3): 277-293

[7] Kristen A, Tony B, Frank H, et al. Network processor acceleration for a Linux * netfilter firewall. In: Proceedings of the Symposium on Architecture for Networking and Communications Systems, New Jersey, USA: ACM Press, 2005. 115-123

[8] L7-filter. http://l7-filter.sourceforge.net/,2009

A traffic identification system based on integration of NP with GPP

Li Yunchun, Qin Xianlong, Wang Xiao

(Computer Science Department, Beihang University, Beijing 100191)

Abstract

According to the characteristics of network processors (NP) and general purpose processors (GPP), the authors proposed an extensible architecture for network traffic management, and based on this architecture, designed and implemented a traffic identification system. The key module, the communication mechanism between NP and GPP is described in detail in the paper. In this system, NP as the fast data plane receives, stores packet flow and sends the needed packets to the GPP through the 21555 non-transparent bridge, and the GPP as the slow data plane uses the deep packet inspection based on L7-filter module to identify traffic types.

Key words: traffic identification, network processor, PCI communication, deep packet inspection, traffic management