基于 P4 的主动网络遥测机制

刘争争[1,2,3]，毕军[1,3]，周禹[1,2,3]，王旸旸[1,3]，林耘森箫[1,2,3]
(1. 清华大学网络科学与网络空间研究院，北京 100084；2. 清华大学计算机科学与技术系，北京 100084；
3. 北京信息科学与技术国家研究中心，北京 100084)

摘 要：随着以可编程协议无关报文处理语言（P4，programming protocol-independent packet processors）为主要编程语言的可编程数据平面的兴起，给网络遥测领域带来了新的发展机遇。带内网络遥测（INT, In-band network telemetry）就是其中一种具有代表性的能够探测到设备级遥测数据的被动遥测技术。然而 INT 的探测范围受到探测点部署位置的限制，难以获取全局网络视图。同时将遥测指令和数据封装到正常数据分组所带来较高的探测开销以及部署和运维的复杂性导致可扩展性不好。为此，提出了一种基于 P4 的能够覆盖全网且可扩展性强的主动网络遥测平台 NetVision。

关键词：可编程协议无关报文处理语言；可编程数据平面；全局网络视图；可扩展性；主动网络遥测
中图分类号：TP393
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2018181

Paradigm for proactive telemetry based on P4

LIU Zhengzheng[1,2,3],BI Jun[1,3], ZHOU Yu[1,2,3], WANG Yangyang[1,3], LIN Yunsenxiao[1,2,3]
1. Institute for Network Sciences and Cyberspace,Tsinghua University, Beijing 100084, China
2. Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
3. Beijing National Research Center for Information Science and Technology(BNRist), Beijing 100084, China

Abstract: With the rise of programmable data plane with P4 as the main programming language, it brings new opportunities for network telemetry. INT is one of the typical passive telemetry techniques that can detect device-level telemetry data. However, the detection scope of INT is limited by the deploy field of vantage points. Therefore, it is hard to achieve the global network view. At the same time, encapsulating telemetry instructions and data into normal packets brings high telemetry overhead and high operation complexity. As a result, INT has low scalability. For this reason, a proactive network telemetry platform NetVision was proposed based on P4 that can cover the whole network and has good scalability.

Key words: P4, programmable data plane, global network view, scalability,proactive network telemetry

1 引言

随着互联设备和网络协议的激增，互联网变得愈发复杂臃肿。在这样错综复杂的网络环境下必然会频繁出现诸如误配置，硬件故障，软件错误等网络问题。为此网络管理员迫切需要一种快速高效的网络遥测方案，能够利用采集到的实时准确的网络状态信息来快速检测和定位例如高队列延迟[1]，TCP Incast[2]，链路负载不均衡[3]等常见网络故障。

在以可编程协议无关报文处理语言[4]（P4）为主要编程语言的可编程数据平面出现之前，网络管理员一般只能够通过网络边缘的终端设备来间接获取滞后且不准确的网络遥测信息。基于可编程数据平面的带内网络遥测[5]（INT）技术的出现很大

收稿日期：2018−09−08
通信作者：毕军，junbi@tsinghua.edu.cn
基金项目：国家重点研发计划基础前沿类基金资助项目（No.2017YFB0801701）；国家自然科学基金资助项目（No.61472213）；赛尔网络下一代互联网技术创新项目（No.NGII20160123）
Foundation Items: The National Key R&D Program of China (No.2017YFB0801701), The National Science Foundation of China (No.61472213), The Next Generation Internet Technology Innovation Project of CERNET (No.NGII20160123)

程度上缓解了管理员面临的上述困境。INT 利用可编程设备的可定制化特性，能够直接在转发数据分组时获取设备内部更为细粒度和准确的遥测数据。

然而，INT 作为一种被动遥测技术，也存在一些局限性。首先，INT 的探测范围受到探测点部署的位置限制，难以获取全局网络视图。网络管理员需要提前指定需要探测的路径和遥测指标，无法在运行时动态地按需进行调整。因此，INT 只能够探测指定路径上特定的遥测数据。这样，INT 就无法探测到其他路径上出现的网络故障，即探测范围受限。其次，采用将遥测指令和数据封装到正常数据分组中的方式，一方面会降低数据分组的有效载荷比，探测流量占总流量的比例过大，带来较大的探测开销；另一方面遥测的发送端和接收端需要同步和协调正常数据分组中遥测指令的嵌入和遥测数据的提取工作，会占用设备有限的计算和存储资源且操作复杂。基于上述两点，INT 的可扩展性不足。

基于INT存在的覆盖范围有限以及可扩展性不足的问题，本文提出了一种基于 P4 的高效主动网络遥测平台 NetVision，具有覆盖范围广和可扩展性强的特点。本文采用合适数量和特殊格式的探针数据分组来代替正常数据分组进行网络遥测，降低遥测开销，提高了数据分组的有效载荷比。此外，基于段路由[6]（SR, segment routing）具有的简单灵活的路由控制能力，本文可以在运行时通过改变 SR 标签及其排列顺序来动态指定探测路径。本文可以在探针中加入 SR 标签栈来支持获取全面的网络视图。同时通过环形探测路径的方式，单个探测点即可兼具探针发送端和接收端的功能，减少了多个探测点之间同步协调探针等复杂操作。另一方面，为了减少冗余的遥测数据，在探针数据分组中加入遥测数据指示域，指定需要采集的遥测数据。最后，将可编程设备内部的状态信息嵌入到探针中可通过可编程设备的定制化能力自定义数据分组处理逻辑来实现。

2　相关研究背景

2.1　P4 语言概述

斯坦福大学的 Nick McKeown 教授为了充分解放数据平面的编程能力于 2014 年首次设计并提出了数据平面特定领域编程语言 P4，一经提出就得到了学术界和工业界的广泛关注和认可[4]。工业界纷纷跟进并着手研制了一系列高性能的可编程硬件，其中主要包含 Barefoot Tofino[7]，Cavium XPliant[8] 以及 Netronome NICs[9]等。其中 Barefoot Tofino 是目前业界数据分组转发速度最快的可编程硬件，最高可以达到 6.5Tbit/s 的线速数据分组转发速率，性能远超传统交换机[7]。一方面基于可编程设备的可定制化特性能够快速实现和验证一些新型的网络架构、功能和协议，极大加速了网络演进和创新；另一方面基于可编程设备的高性能特性，传统上由灵活但低性能的中间件实现的一些比如防火墙，负载均衡等较为简单的网络功能可以卸载到可编程数据平面上实现来获取可观的性能提升。数据平面特定领域编程语言 P4 具有如下 3 点语言特性。

1) 可重配置性[4]：P4 支持转发逻辑代码经过编译部署到具体平台上之后动态修改报文的处理方式。这样的话，运营商就可以在不更换硬件的前提下灵活定义数据平面的处理行为，极大降低了更换设备的资金成本和等待新设备开发的时间成本。

2) 协议无关性[4]：P4 并不绑定于某个特定的网络协议。开发人员只需根据 P4 语言定义的语法语义要素结合平台的相关特性就可以自定义新协议，同时也能够去除冗余的协议，按需使用协议，降低了额外开销，提高了设备的资源利用率。

3) 平台无关性[4]：开发人员可以独立于特定的底层运行平台来编写数据报文处理逻辑。代码能够通过设备相关的后端编译器快速地在硬件交换机、FPGA、SmartNIC、软件交换机等不同平台之间移植，减轻了开发人员的负担，提高了开发效率。

为此，P4 语言定义了一套抽象转发模型[10]来支撑上诉三点语言特性。如图 1 所示，抽象转发模型包含 3 个主要部分。

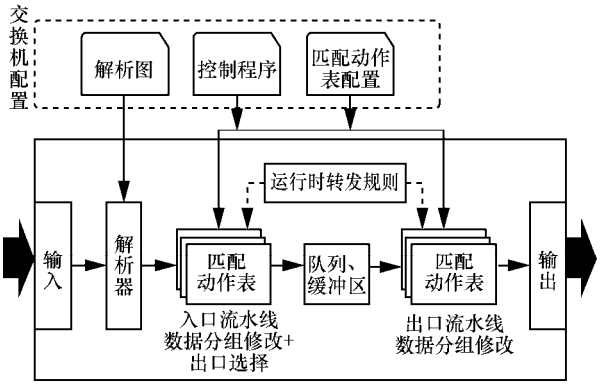

图 1　P4 抽象转发模型

1) 第一部分是可编程的数据报文头部解析器[10]。开发人员在编写 P4 代码时可以自定义报文头部解

析流程，灵活解析不同的数据报文格式，经过编译之后产生类似于图 2 所示的数据报文头部解析状态转移图，在部署时配置到可编程设备的报文头部解析器上。在数据报文进入可编程设备时数据平面先将报文头部和载荷分离，接着根据解析图的状态转移规则解析并保存报文的头部到对应自定义的头部域中，用于流水线中流表的匹配操作。

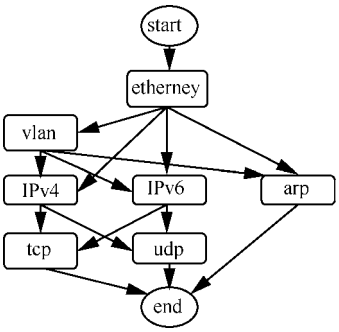

图 2 数据报文头部解析图实例

2) 第二部分是可编程的多阶段流水线[10]。从图 1 中可以看出主要分为入口流水线和出口流水线，其中，入口流水线主要进行数据分组修改以及决定出端口的操作，随之数据分组进入对应出端口的缓存队列中；而出口流水线仅负责数据分组的修改。开发人员可以自定义每张匹配动作表中的匹配头部域，执行动作及其参数，流表的数量等以及各条流水线中每张匹配动作表的执行顺序。P4 代码经过编译之后会产生一张由匹配动作表组成的有向无环图（DAG），即数据平面控制流。运行时数据平面会依据控制流中匹配动作表的顺序依次匹配处理每个数据报文。

3) 最后一部分是控制平面上的控制程序[10]。P4 程序会在编译后生成对应的控制接口，主要负责在运行时向设备的数据平面下发并安装具体的流表匹配规则，配置计数器、寄存器等与平台相关的特定对象以及采集其他运行时的状态统计信息。

2.2 基于 P4 的网络遥测

在可编程数据平面出现之前，传统上网络管理员只能将网络中的各种设备作为黑盒处理，因为设备厂商一般基于商业利益的考虑不会开放设备的网络遥测接口。因此，管理员一般只能通过网络边缘的终端设备来间接地获取滞后且不准确的网络遥测信息。从网络内部的设备上获取实时且准确的网络遥测信息成为网络管理员们实际且迫切的需求。而可编程数据平面的出现很大程度上缓解了上述困境。可编程交换机相较于传统交换机是一种白盒交换机，即交换机像一张白纸一样本身不附带任何数据分组处理逻辑。管理员们可以利用可编程交换机的可定制化特性以及具体平台提供的运行时状态信息，能够支持在网络设备运行时实时获取内部第一手的更为细粒度和准确的遥测数据。

基于可编程数据平面上 P4 语言的被动式网络遥测 INT 技术应运而生。这项新型的被动网络遥测技术能够直接通过数据平面收集和上报网络遥测信息给监控器，而不需要制平面的介入。这在获取了细粒度且准确的遥测数据的同时也简化了网络遥测流程，降低了控制平面与数据平面的通信开销。

INT 被动遥测的基本过程如下：首先由网络边缘的 INT 流量发送端(应用、终端主机协议栈、管理程序、智能网卡，发送端架顶交换机)将被称作遥测指令的特定数据分组匹配字段嵌入到正常数据分组中发送到待探测网络中。紧接着当数据分组按指定路径穿越网络时，该遥测指令字段会告诉支持 INT 的可编程设备需要收集哪些网络遥测信息，并把相关信息封装到正常数据分组中，携带至同样处在网络边缘的 INT 流量接收端。最后在 INT 流量接收端提取封装在正常数据分组中的网络遥测数据，上报给 INT 监控设备进行处理。在整个遥测过程中，INT 流量发送端在发送终端发出的正常数据分组嵌入遥测指令，INT 流量接收端从中提取遥测数据，将原始的正常数据分组发给目的终端。因此发送端和接收端的终端设备看到的都是原始正常的数据分组，对遥测是无感知的。基于 INT 技术，网络管理员能够快速获取探测包在穿越路径上的第一手数据平面遥测信息。

INT 具有广泛的应用场景，可以被用来进行 OAM 操作，实时控制或反馈循环，网络事件探测，网络故障排除和数据平面验证等方面。同时基于 INT 底层遥测技术，学术界也进行了大量的基于 P4 的关于高效网络监控、快速大流检测和通用遥测语言等网络遥测相关的研究。其中，ossRadar[11]采用可逆布隆过滤器计算数据分组摘要，比较进出同一个网络域时数据分组摘要的差异来检测和定位分组丢失，其开销仅正比于分组丢失数目，节省了大量资源，极大提高了监控和处理分组丢失故障的效率。而 HashPipe[12]则是基于改进的 Space Saving 算法[13]设计了由存储大流计数器的散列表组成的流

水线，同时“驱逐”小流的方案。实验结果表明只需占用不到 80 B 内存即可识别出 40 万条流中的95%的大流，内存开销小且准确率比较高。同时为了应对管理员不断变化的性能监控需求，Narayana等[14]设计了 Marple 性能查询语言来屏蔽底层实现的复杂性，只需占用很少硬件资源就可以支持百万量级的查询请求。

但是 INT 也具有一些固有的局限性。由于探测路径和遥测指标需要在部署前由网络管理员提前指定，并且部署之后难以修改。因此，NT 的探测范围受到探测点部署位置的限制。INT 只能够探测指定路径上固定的遥测指标，不能适应变化的网络环境和遥测需求。这样，INT 就无法探测到其他路径上可能出现的网络故障，无法覆盖全网。其次采用在正常数据分组中封装遥测指令和数据的方案，一方面降低了数据分组的有效载荷比，带来了较大的冗余探测开销，消耗了交换机中有限的计算和存储资源；另一方面 INT 的流量发送端和接收端边缘交换机需要同步和协调正常数据分组遥测指令的嵌入和遥测数据的提取工作，部署、运行和维护工作复杂。基于上述两点，INT 的可扩展性不足。

3　系统设计

3.1　系统架构和工作流程

为了保证主动遥测的探测路径在运行时是灵活可控的，本文采用段路由机制来灵活控制探针的探测路径。段路由机制简单易用，不需要额外的协议支持，通过组合一系列简单的网络操作指令就可以完全控制数据分组的转发路径。并且该机制可扩展性较好，每条数据流的状态只需要在段路由域的入口节点上维护，在降低了网络成本的同时也提高了灵活性。最后段路由能够支持增量部署，降低了部署难度，可行性好。段路由能够原生运行在 MPLS 和 IPv6 的数据平面上，只需升级软件系统即可增加段路由功能。本文可以设定探针的探测路径为一个环形，即探针从探测点发送出来，探测一圈后返回原探测点。采用这种探测方式，在待探测网络中只需部署单个探测点负责发送和接收探针数据分组即可完成探测任务。环形探测一方面省去了在大量探测点之间同步和协调探针发送和接收等复杂操作，极大减少了探测点部署和维护的开销；另一方面网络管理员可以灵活定义遥测路径，按需探测可能或已经出现问题的路径，快速定位故障位置。另外我们可以在探针格式中加入指示探测遥测数据类型的字段来支持按需获取遥测数据。同时要保证能够采集到网络设备内部的状态信息等细粒度准确的遥测数据，我们可以通过修改可编程设备的数据平面处理逻辑来区分处理正常数据分组和探针数据分组。对于正常数据分组直接正常转发，而对于探针数据分组匹配其中的路径转发标签以及遥测指令字段，将实时的网络状态信息封装在探针数据分组中。

基于 P4 的主动网络遥测平台 NetVision 的系统架构和工作流程如图 3 所示。网络管理员向

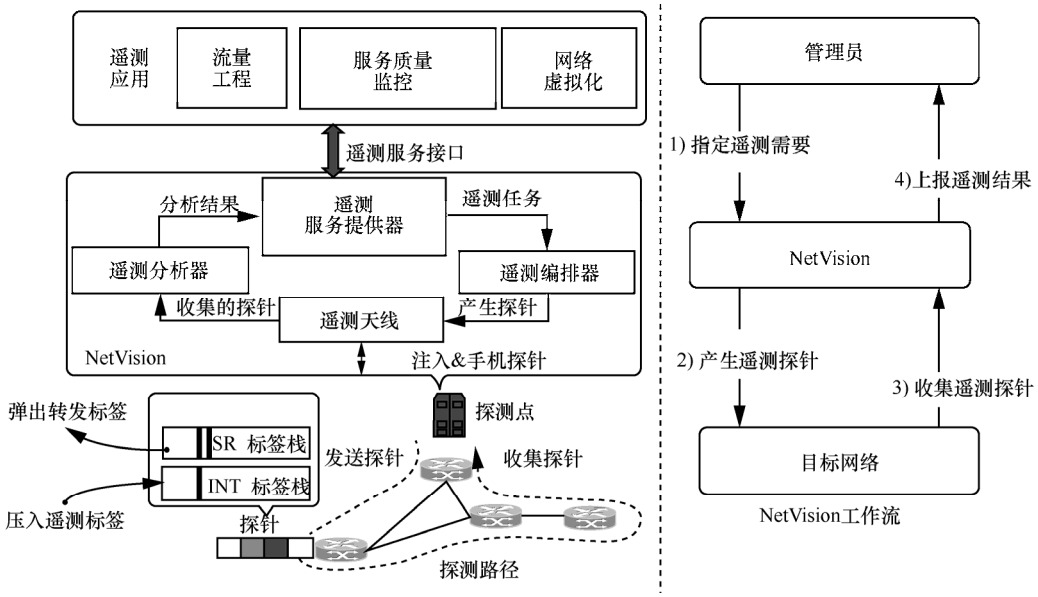

图 3　NetVision 遥测平台系统架构和工作流程

NetVision 遥测平台指定抽象的遥测需求，然后由该遥测平台负责生成，下发和接收对应的探针数据分组，最后向网络管理员返回遥测结果。NetVision 主要由 4 组件构成，分别是遥测天线、遥测编排器、遥测分析器和遥测服务提供器。整个遥测平台的工作流程如下：首先上层例如流量工程和网络虚拟化等网络遥测应用通过由遥测服务提供器开放的网络遥测服务 API 下发高级遥测策略；接着由遥测服务提供器向遥测编排器下发由遥测策略产生的遥测任务，遥测编排器负责产生各个任务中具体的探针数据分组，设置数据分组的内容，数量以及探测路径等必需信息；下一步交给遥测天线将探针数据分组交由底层的探测点发送，并从那里接收完成遥测的探针数据分组，转发给遥测分析器进行分析；最后分析结果由遥测服务提供器反馈给上层遥测应用。

3.2 双栈探针数据分组设计

本文设计了一种在数据平面上灵活可控的双栈探针数据分组格式。其中包括负责转发的段路由转发栈和负责记录转发路径上遥测数据的INT标签栈。如图 4 所示，段路由转发栈中包含出端口标签列表及列表长度。这种路由操作使交换机不需要存储，要通过匹配其他字段来间接获取出端口的额外的匹配动作表，节省了交换机有限的 TCAM 和 SRAM 等资源。而 INT 标签栈同样由 INT 标签列表及列表长度组成。INT 标签由交换机标识符，遥测元数据位图和一系列的遥测元数据组成，其中遥测元数据位图用来指示后面跟着的遥测元数据类型。

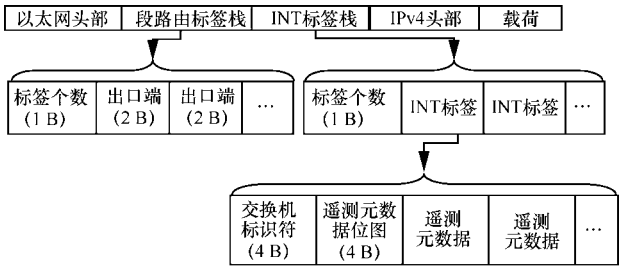

图 4 双栈探针数据分组格式

3.3 网络遥测服务原语

为了简化管理员使用遥测平台的方式同时屏蔽底层实现的复杂性，本文设计了一套简洁实用并且具有丰富表现力的网络遥测服务原语。原语提供了必要的网络遥测元数据和网络遥测查询语句。如表 1 所示，网络遥测原语中包含了设备节点、链路、转发路径等遥测信息。表 2 是提供查询设备节点，链路和转发路径遥测信息相关的查询语句。

如表 3 所示，本文提供一些网络遥测服务原语的应用实例，其中包含了端到端的延迟探测、实时数据分组的传输速率计算以及数据分组黑洞探测等常用网络功能。

表 1 网络遥测元数据

元数据类型	元数据	描述	范围
1	IngressPortID	入端口号	设备节点
2	EgressPortID	出端口号	设备节点
3	IngressTStamp	入时间标签	设备节点
4	EgressTStamp	出时间标签	设备节点
5	IngressCnt	入口流量	设备节点
6	EgressCnt	出口流量	设备节点
7	IngressDrop	入口分组丢失流量	设备节点
8	EgressDrop	出口分组丢失流量	设备节点
9	IngressRate	入口流量速率	设备节点
10	EgressRate	出口流量速率	设备节点
11	QueueID	队列标识符	设备节点
12	InsQueueLength	实时队列长度	设备节点
13	AvQueueLength	平均队列长度	设备节点
14	SwitchID	交换机标识符	设备节点
15	SwitchCnt	交换机总流量	设备节点
16	SwitchDrop	交换机分组丢失流量	设备节点
17	HopLatency	逐跳延迟	设备节点
18	LinkLatency	链路延迟	链路
19	LinkUtilize	链路带宽利用率	链路
20	PathTrace	网络路径	路径
21	PathLength	路径长度	路径
22	PathRTT	路径 RTT	路径

表 2 网络遥测查询语句

查询语句	参数	描述
NodeQuery	设备节点	查询节点信息
PathQuery	源节点和目的节点	查询路径信息
Select	元数据类型	查询具体元数据
Where	过滤语句	过滤元数据
Period	时间段	遥测时间

3.4 探针探测路径生成算法

为了降低探针探测路径冗余带来的额外开销，高效地进行基于 P4 的主动网络遥测，本文需要设计一种控制探针中段路由标签嵌入顺序的探针探测机制。在保证具有探测全网能力的前提下，优化探测路径中存在的冗余。为此，我们将网络中的链

表 3　网络遥测查询语句

应用	遥测服务接口	描述
端到端延迟测量	Q = PathQuery(“1:1”, ”2:1”) .Select(“PathRTT”) .Where(“PathTrace= [1:1,3:1,3:2,2:2,2:1]”)	探测交换机 1 的端口 1 和交换机 2 的端口 1 在遵循路径限制下路径数据分组的往返时间
链路黑洞探测	Q=PathQuery(“*:*”, “*:*”) .Select(“Path”) .Where(“PathLength==1 and PassedPkts==0”) .Period(“1 s”)	为了探测在 1 s 的链路黑洞，本文指定路径长度为 1 且寻找没有通过数据分组的链路
实时数据分组传输速率计算	Q=NodeQuery(“*1:1*”) .Select(“IngressRate”) .Period(“5 s”)	计算交换机 1 上端口 1 在 5 s 之内的平均数据分组接收速率
节点黑洞探测	Q = NodeQuery(“*:*”) .Select(“SwitchId”) .Where(“IngressCnt==0or EgressCnt==0”) .Period(“1 s”)	通过查询交换机端口在 1 s 之内的出入端口经过的流量来定位发生黑洞的交换机标识符

路看成双向的有向边，网络变成了一张有向图。根据欧拉定理，这样的有向图存在欧拉回路。因此本文可以采用 Hierholzer 算法[15]在 $O(E)$时间内计算出从单探测点出发的欧拉回路，作为探测路径。

4　实验测试

4.1　实验环境

实验采用一台戴尔 R730 服务器，硬件参数为 24 个 6 核 Intel(R) Xeon(R) E5-2620 v3 CPU, 128 GB RAM。在服务器上使用 mininet 网络仿真软件模拟主机和 P4 交换机，并搭建了如图 5 所示的 3 层 fat-tree 拓扑。其中，P4 交换机的数据平面处理逻辑使用 $P4_{16}$ 标准定义的语法和语义格式进行编写。数据平面逻辑运行在开源的软件交换机模型 bmv2 的 v1model 架构之上。

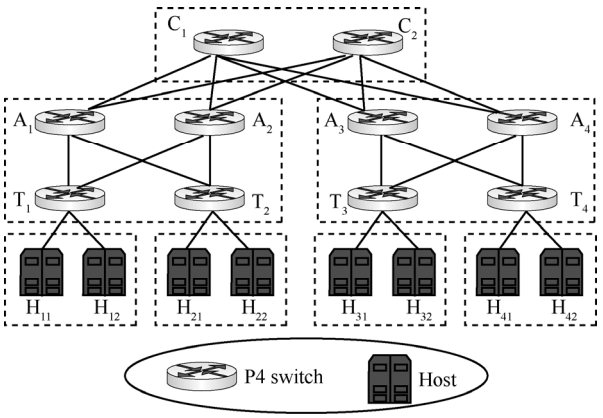

图 5　实验使用的 3 层 fat-tree 拓扑

4.2　HTTP 延迟测试

实验中 H_{22} 作为 HTTP 服务器持续回复来自 H_{11} 的周期性循环的 HTTP 请求。其中，H_{11} 的请求报文沿着 T_1-A_2-T_2 路径到达 H_{22}，H_{22} 的回复报文沿着 T_2-A_2-T_1 路径返回 H_{11}。而 H_{32} 将会向 H_{22} 发送大量突发流量，沿着 T_3-A_3-C_1-A_2-T_2 路径。其中，探针的探测路径为 T_1-A_2-T_2-A_2-T_1。本文可以通过比较探针两次经过 T_1 的时间标签即可求出延时数据。通过这种方式，消除了多台交换机之间时间同步等复杂操作。同时基于段路由的灵活路由能力，本文可以指定来回的路径一致，消除了不对称路由问题带来延迟误差较大的问题。

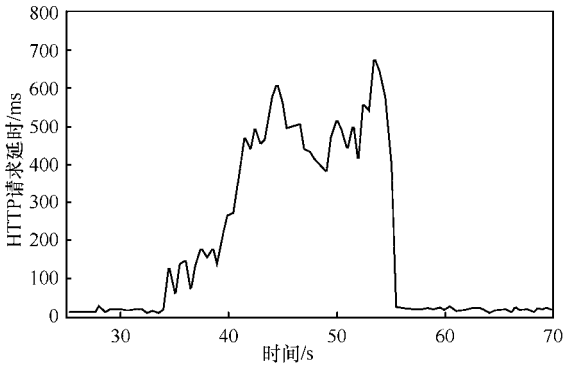

图 6　HTTP 请求延时

图 6 是 HTTP 的请求延迟随时间的变化图，图 7 是交换机 T_1、A_2、T_2 的排队延迟随时间变化。

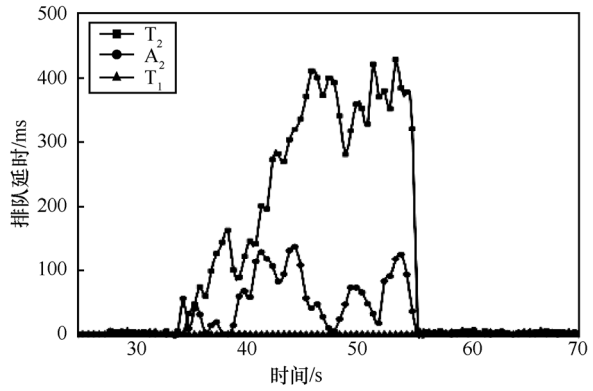

图 7 交换机的排队延时

HTTP 请求延时在 35 s 的时候从正常水平开始上升，在 55 s 的时候下降到正常水平，说明这段时间受到了突发流量的影响。类似的交换机 T_2 的排队延迟大幅度地上升了，A_2 的排队延迟小幅度上升。而 T_1 的排队延迟并没有明显变化。本文可以推断 T_1 并不在阻塞路径上，而 T_2 和 A_2 则在阻塞路径上。网络管理员可以根据延时变化，快速判断和定位出现流量阻塞的位置。

4.3 交换机负载均衡测试

实验中在交换机 T_1 上部署负载均衡器，H_{11} 向 T_1 发送流量，其他 3 个端口作为流量出口。交换机 T_1 上采用基于 IP 五元组的散列选择出端口。本文可以通过对各个出端口通过的流量进行计数，计算每秒钟流量的增量，作为数据分组平均发送速率。

首先 H_{11} 向 T_1 发送具有完全随机 IP 五元组的数据分组，3 个出端口以及理论均值如图 8 所示。benwen 可以看到 3 个出端口的数据分组发送速率均保持 0.15 Mbit·s^{-1} 左右，与理论均值基本一致，说明负载均衡的效果很好。

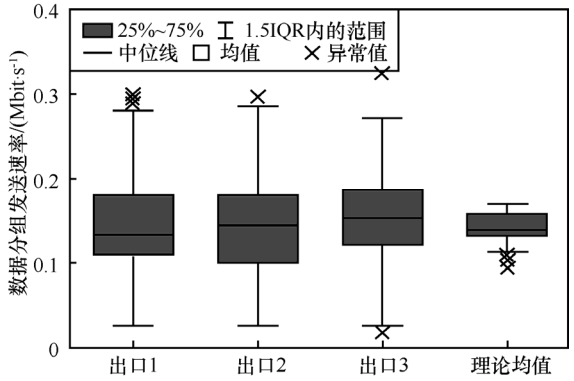

图 8 交换机负载均衡下的数据分组发送速率

其次，H_{11} 向 T_1 发送具有长大象流的非完全随机 IP 五元组的数据分组。此时 3 个出端口以及理论均值如图 9 所示。从图 9 中可以看到出口 1 的平均发送速率在 0.1 Mbit/s 左右，出口 2 的平均发送速率在 0.8 Mbit/s 左右，而出口 3 的平均发送速率在 0.45 Mbit/s 左右。而理论均值则在 0.45 Mbit/s 左右，负载均衡效果不理想。

通过主动网络遥测的方式可以实时获取各端口的瞬时数据分组发送速率，从而可以帮助管理员迅速地发现负载不均衡等网络问题。

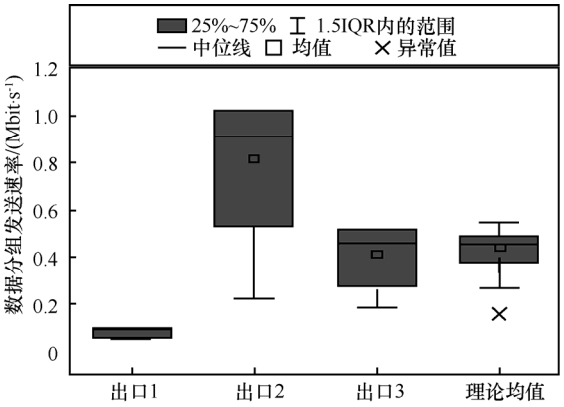

图 9 交换机负载不均衡下的数据分组发送速率

4.4 路由黑洞测试

实验中使用探针测试 T_1 分别连接 A_2 和 A_1 的端口，A_2 连接 T_1 的端口的发送和接收数据分组速率。其中，H_{11} 按 T_1-A_2-T_2 的路径与 H_{22} 以 2 Mbit/s 的速率进行 TCP 通信，而 H_{11} 按 T_1-A_1-C_2-A_3-T_3 的路径与 H_{32} 以 1 Mbit/s 的速率进行 TCP 通信。发包开始后 50 s 左右将 T_1 到 A_2 的路径切断，一段时间后再将路径恢复。

实验结果显示 H_{11} 与 H_{32} 的收发包速率稳定在 1.96 Mbit/s，与理论值基本一致，通信路径上的链路通畅。而 H_{11} 与 H_{22} 的收发包速率一开始在 0.98 Mbit/s，在 53 s 时收包速率迅速下降到 0，发包速率无明显变化，在 102 秒时收包速率恢复至正常速率。A_2 连接 T_1 的端口的收发包速率具有相同的变化规律。据此我们可以推测在 53~102 s 之间，T_1-A_2 链路或者链路两端的端口出现了问题。

5 结束语

本文提出了一种在可编程网络环境下的基于 P4 语言的主动网络遥测平台 NetVision。NetVision 采用段路由机制支持运行时动态改变探测路径。本文也设计了双栈探针数据分组格式支持按需灵活地获取遥测数据；并且同时提出了一套简洁易

用对网络管理员友好的遥测原语，屏蔽了底层遥测操作的复杂性；最后在保证全网覆盖的前提下，采用了 Hierholzer 算法来计算探测环路达到减低探针开销的目的，减少对正常流量的影响。NetVision 相较于被动的 INT 技术具有覆盖范围更广和可扩展性更好的优势。

参考文献：

[1] GUO C, YUAN L, XIANG D, et al. Pingmesh: a large-scale system for data center network latency measurement and analysis[C]//ACM SIGCOMM Computer Communication Review. ACM, 2015: 139-152.

[2] VASUDEVAN V, PHANISHAYEE A, SHAH H, et al. Safe and effective fine-grained TCP retransmissions for datacenter communication[C]//ACM SIGCOMM Computer Communication Review. ACM, 2009: 303-314.

[3] ALIZADEH M, EDSALL T, DHARMAPURIKAR S, et al. CONGA: Distributed congestion-aware load balancing for datacenters[C]//ACM SIGCOMM Computer Communication Review. ACM, 2014: 503-514.

[4] BOSSHART P, DALY D, GIBB G. P4: Programming protocol- independent-packet processors[J]. ACM SIGCOMM Computer Communication Review, 2014, 44(3): 87-95.

[5] KIM C, SIVARAMAN A, KATTA N, et al. In-band network telemetry via programmable dataplanes[C]//ACM SIGCOMM. 2015.

[6] FILSFILS C, NAINAR N K, PIGNATARO C, et al. The segment routing architecture[C]//Global Communications Conference (GLOBECOM), 2015 IEEE. IEEE, 2015: 1-6.

[7] HE C H, CHANG B Y, CHAKRABORTY S, et al. A zero flow entry expiration timeout P4 switch[C]//Proceedings of the Symposium on SDN Research. ACM, 2018: 19.

[8] SHARMA N K, LIU M, ATREYA K, et al. Approximating fair queueing on reconfigurable switches[C]//USENIX Symposium on Networked Systems Design and Implementation. 2018.

[9] DA SILVA J S, BOYER F R, CHIQUETTE L O, et al. Extern Objects in P4: an ROHC header compression scheme case study[C]//2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft). IEEE, 2018: 517-522.

[10] P4 Language Consortium. 2017. P4 Language Specification, Version 1.0.4[EB/OL]. Available at https://p4.org/spces/.

[11] LI Y, MIAO R, KIM C, et al. Lossradar: Fast detection of lost packets in data center networks[C]//Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies. ACM, 2016: 481-495.

[12] SIVARAMAN V, NARAYANA S, ROTTENSTREICH O, et al. Heavy-hitter detection entirely in the data plane[C]//Proceedings of the Symposium on SDN Research. ACM, 2017: 164-176.

[13] METWALLY A, AGRAWAL D, EL ABBADI A. Efficient computation of frequent and top-k elements in data streams[C]//International Conference on Database Theory. Springer, Berlin, Heidelberg, 2005: 398-412.

[14] NARAYANA S, SIVARAMAN A, NATHAN V, et al. Language-directed hardware design for network performance monitoring[C]//Proceedings of the Conference of the ACM Special Interest Group on Data Communication.ACM, 2017: 85-98.

[15] FLEISCHNER H X. 1 Algorithms for eulerian trails[J]. Eulerian Graphs and Related Topics: Part 1 (Annals of Discrete Mathematics), 1991, 2(50): 1-13.

[作者简介]

刘争争（1994–），男，安徽蚌埠人，清华大学硕士生，主要研究方向为 SDN 控制平面与数据平面可扩展性、域间 SDN 互连机制、可编程数据平面。

毕军（1972–），男，辽宁大连人，中国教育部长江学者特聘教授、清华大学长聘教授、博士生导师，主要研究方向为网络空间安全、软件定义网络、网络体系结构、源地址验证等。

周禹（1994–），男，河北衡水人，清华大学博士生，主要研究方向为可编程数据平面。

王旸旸（1984–），男，安徽淮北人，清华大学助理研究员，主要研究方向为域间 SDN 互连机制。

林耘森箫（1995–），男，重庆人，清华大学博士生，主要研究方向为可编程数据平面。