计算机科学
COMPUTER SCIENCE

# 基于概要数据结构的网络微突发流量检测方法

王佳宇, 于俊清, 李冬, 赵君杨

引用本文

王佳宇, 于俊清, 李冬, 赵君杨. 基于概要数据结构的网络微突发流量检测方法[J]. 计算机科学, 2025, 52(1): 374-382.

WANG Jiayu, YU Junqing, LI Dong, ZHAO Junyang. Network Microburst Traffic Measurement Method Based on Sketch Data Structure [J]. Computer Science, 2025, 52(1): 374-382.

---

**相似文章推荐（请使用火狐或 IE 浏览器查看文章）**

**Similar articles recommended (Please use Firefox or IE to view the article)**

基于邻居采样和图注意力机制的产业链风险评估模型

Risk Assessment Model for Industrial Chain Based on Neighbor Sampling and GraphAttention Mechanism

计算机科学, 2024, 51(10): 218-226. https://doi.org/10.11896/jsjkx.230900145

结合元学习的去中心化联邦增量学习方法

Decentralized Federated Continual Learning Method Combined with Meta-learning

计算机科学, 2024, 51(3): 271-279. https://doi.org/10.11896/jsjkx.230100125

基于规则推理的足球视频任意球射门事件检测

Shooting Event Detection of Free Kick in Soccer Video Based on Rule Reasoning

计算机科学, 2023, 50(3): 181-190. https://doi.org/10.11896/jsjkx.220300062

基于空间和多层级联合编码的图像描述算法

Spatial Encoding and Multi-layer Joint Encoding Enhanced Transformer for Image Captioning

计算机科学, 2022, 49(10): 151-158. https://doi.org/10.11896/jsjkx.210900159

面向金融活动的复合区块链关联事件溯源方法

Composite Blockchain Associated Event Tracing Method for Financial Activities

计算机科学, 2022, 49(3): 346-353. https://doi.org/10.11896/jsjkx.210700068

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.231200080

# 基于概要数据结构的网络微突发流量检测方法

王佳宇[1] 于俊清[1,2] 李 冬[2] 赵君杨[1]
1 华中科技大学网络空间安全学院 武汉 430074
2 华中科技大学网络与信息化办公室 武汉 430074
(15387261971@163.com)

**摘 要** 网络微突发流量是数据中心网络中常见的流量类型,其在极短的时间内迅速增长,对网络性能造成严重影响,且难以检测。目前的测量方法无法兼顾细粒度检测和低资源开销传输,文中基于概要数据结构(sketch)设计了一种轻量级细粒度的网络微突发流量测量方法。首先基于可编程交换机的架构特性,实时测量数据报文的排队时延,设计检测算法,监测微突发流量,实现基于数据报文的细粒度检测;然后根据检测结果采集微突发流,采用 sketch 存储微突发流信息,利用镜像传输方式在时间片或微突发流结束后向控制器传送,实现轻量级传输。测量方法基于可编程协议无关报文处理语言,在 P4 可编程交换机上进行了相应的系统实现,能够实时检测和展示网络微突发流量。实验结果表明该方法能够实时细粒度检测网络微突发流量,显著降低传输微突发信息的带宽开销。

**关键词**:可编程协议无关报文处理语言;可编程交换机;微突发流量;概要数据结构
**中图分类号** TP393

## Network Microburst Traffic Measurement Method Based on Sketch Data Structure

WANG Jiayu[1], YU Junqing[1,2], LI Dong[2] and ZHAO Junyang[1]
1 School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
2 Network and Information Office, Huazhong University of Science and Technology, Wuhan 430074, China

**Abstract** Microburst traffic is a common type of traffic in data center network, which grows rapidly in a very short period of time, and has serious effect on network performance and is difficult to detect. Existing microburst traffic detection methods cannot take into account both fine-grained detection and low-resource transmission. This paper proposes a lightweight fine-grained microburst detection method based on sketch data structure. Firstly, the architectural characteristics of the programmable switch is used to measure the queuing delay for each packet, microburst detection algorithm is put forward to process network traffic and the microburst traffic is filtered out to achieve the purpose of fine-grained detection. Then sketch is used to save microburst traffic information, which is sent to controller at the end of the time slice or the end of the microburst stream by mirroring transmission, so as to achieve lightweight transmission. Finally, the microburst traffic detection system is implemented on P4 programmable switch in real-world network environment. Experiments show that this method has good microburst measurement accuracy, and greatly reduces the bandwidth overhead required for microburst information transmission.

**Keywords** Programming protocol-independent packet processors language, Programmable switch, Microburst traffic, Sketch data structure

## 1 引言

过去十年中,数据中心的网络规模显著扩大,在数据中心网络带宽大幅增加和应用程序愈加复杂的环境下,确保网络高可用性和满足不同应用的服务级别协议(Service Level Agreement,SLA)具有高挑战性[1-2]。

高速网络中端到端时延极低[3],导致持续时间极短的网络拥塞也会产生高网络延迟,延长流完成时间(Flow Completed Time,FCT),同时可能会导致丢包,引发 TCP 重传,对网络性能产生严重影响。先前的工作将间歇发生的时间尺度极短的网络拥塞称为微突发(MicroBurst)[4],如果能够成功提取微突发的流信息,针对微突发信息重新设计交换机缓冲区和主动队列控制算法,就能够有效缓解微突发造成的网络延迟。

然而,微突发极小的时间尺度使得传统网络中粗粒度网管系统无法检测到微突发事件的发生。一些商业解决方案[5]

到稿日期:2023-12-12 返修日期:2024-05-08
基金项目:国家重点研发计划(2022YFB2901202)
This work was supported by the National Key R&D Program of China(2022YFB2901202).
通信作者:于俊清(yjqing@hust.edu.cn)

能够检测到微突发，但无法提供微突发流信息。在基于OpenFlow的软件定义网络(Software Define Network，SDN)环境中，控制器以最快速度轮询OpenFlow交换机内各类计数器进行检测，但只能统计微突发发生的轮询时间段。斯坦福大学教授Nick McKeown于2014年提出了协议无关报文处理语言(Programming Protocol-independent Packet Processors，P4)[6]，工业界跟进研制了一系列可编程交换机，P4可编程交换机可向数据包暴露设备内部性能数据，为细粒度检测微突发提供了技术基础。

因此，本文提出了一种轻量级细粒度的网络微突发流量检测方法，主要贡献如下：

1)采用sketch数据结构存储微突发流信息并进行传输，大幅减少了传输微突发流信息所占用的带宽开销，避免了交换机内部拥塞；同时，采用的sketch能够提取出完整的微突发流信息，便于分析网络中是否遭受了pulse-wave DDos攻击[7]并对其进行防御。

2)采用定时主动推送的方式将微突发流信息发送到控制器，降低了控制器主动获取微突发流的交互时延，使其能够完整地运行在交换机数据平面中。

3)采用P4语言结合可编程交换机数据平面的硬件资源限制实现了原型系统，并在真实的可编程交换机上进行了验证，便于实际部署，同时使用单个交换机进行微突发事件的测量，降低了检测成本。

## 2 研究现状

当网络设备出口端口在极短时间段(毫秒或亚毫秒级别)内收到大量突发数据包时，涌入流量的瞬时速率远高于平均速率，该端口产生瞬时排队拥塞，这种网络异常现象称为微突发。据思科报告，微突发事件的最短持续时间为100 $\mu$s[8]，介于100 $\mu$s～1.68 s之间，华为报告中造成丢包的微突发持续时间在1～100 ms[9]。在微突发发生时间段内，流量的瞬时速率高至平均速率的数十到数百倍，甚至超过端口最大带宽，轻则导致网络流的往返时延(Round-Trip Time，RTT)增大，重则拥塞网络节点造成丢包，引发TCP重传，造成严重的性能影响。

传统网络下基于简单网络管理协议(Simple Network Management Protocol，SNMP)的网管系统轮询交换机管理信息库(Management Information Base，MIB)，周期性获得交换机端口计数器信息，但SNMP在秒级的轮询粒度下无法检测微突发现象。

可编程网络中，利用P4交换机中可编程数据平面的高可见性，国内外学者针对微突发的测量，做出了许多卓有成效的工作。

2015年，文献[10]提出带内遥测(In-band Network Telemetry，INT)技术，奠定了P4交换机检测微突发的基础。INT利用PISA架构的特点，直接查询交换机内部性能数据进行细粒度的测量。INT架构如图1所示，INT源、中转设备、目的地均为P4交换机，向每个普通报文嵌入指定遥测数据，并在最后一跳提取遥测数据发送至服务器解析。INT检测每个数据包的排队时延，可筛选出微突发流，但INT传输所有产生微突发流的数据包从而获取微突发流信息，所需要的带宽开销极大，有可能影响正常数据包的转发。

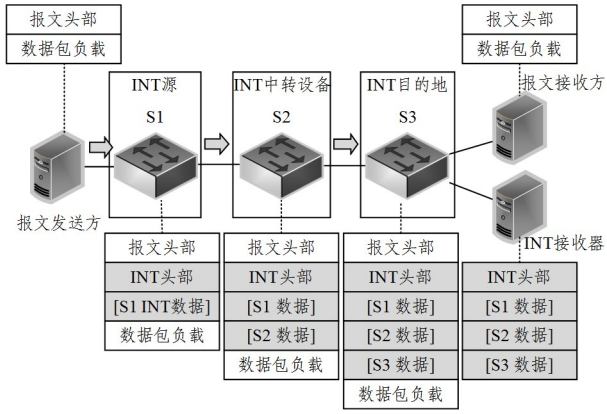


图1 INT示意图

Fig. 1 Schematic diagram of INT

2018年，文献[11]提出了一种实时测量微突发流的方法(BurstRadar)。BurstRadar基于INT，通过数据包排队时延筛选微突发流，并剔除报文载荷，只镜像传输微突发包头，以减少带宽开销。但其仍需传输所有微突发包，带宽开销仍然较大。

2019年，文献[12]提出了一种轻量级采样INT测量方法。系统基于主动测量的INT方案，发送一系列遥测报文穿过网络，同时在P4交换机部署水库采样算法，使遥测路径上每个交换机等概率地抽取数据包嵌入性能数据，发送较少的遥测报文即可测量整条路径的拥塞情况。但其无法完整获取微突发流信息，同时发送大量的遥测报文也会造成极大的带宽开销。

2019年，文献[13]提出了一种细粒度测量传统交换机缓冲区大小的方案。方案将INT与传统网络相结合，使用分光器(Test Access Point，TAP)完整镜像关键位置的流量到P4交换机，在P4交换机模拟传统交换机的排队时延，实时监控队列深度。但其仍使用INT的方法进行微突发的测量，并没有减少INT测量微突发所占用的带宽开销。

2019年，文献[14]提出了一种完全在数据平面运行的细粒度微突发流检测方法(ConQuest)。ConQuest在P4交换机部署一系列sketch快照组成的循环队列，持续更新存储前几毫秒的历史记录。当当前报文经历高排队延迟时，ConQuest计算排队时间所覆盖的快照范围，并汇总快照内该流计数获得微突发流信息，同时利用流信息对大流标记ECN位，尝试缓解微突发现象。ConQuest使用快照来获取微突发流个数，存在四舍五入的误差，同时无法获取到完整的微突发流信息。

2020年，文献[15]提出了一种在数据平面检测多种网络异常事件的测量系统(NetSeer)。NetSeer在数据中心网络全面部署P4交换机，利用完全可编程的数据平面细粒度检测各种异常事件。针对微突发，NetSeer基于INT，镜像全部高排队时延的流量到控制器进行存储分析，但其使用INT获取微突发流信息，所占用的传输带宽开销大。

2022年，文献[16]提出了一种使用Delta Sketch在数据平面进行微突发流检测的方法(MIDST)。MIDST在P4

交换机的入口阶段和出口阶段分别部署 Delta Sketch 用于保存经过交换机的每流的数据包个数，通过排队时延和计算每流经过入口和出口阶段时数据包个数的差值来识别突发流。但是 MIDST 仍需要通过传输所有的微突发流进行完整微突发流信息的获取，占用的带宽开销大。

2023 年，文献[17]提出了一种动态的 INT 算法用于实现细粒度网络指标监控的方案(DINT)。DINT 根据交换机中吞吐量的变化更改遥测信息插入遥测数据包的频率，同时剔除遥测数据包负载转发至解析服务器，从而实现减少 INT 传输微突发流信息时所占用的带宽开销的目的。但是 DINT 需要传输所有遥测数据包来进行微突发流的检测，所需要的带宽开销仍然较大。

2023 年，文献[18]提出了一种通过 TAP 设备将核心交换机的流量镜像至可编程交换机中进行微突发流检测的方法。该方法将核心交换机的所有入口和出口流量镜像至可编程交换机中，利用 Count-min sketch(CM sketch)[19]进行长流的识别，同时通过检测排队时延突增的数据包进行微突发流的检测。但其需要使用镜像传输的方式获取微突发流信息，带宽开销较大。

由此可见，现有的针对微突发的测量且能够提取完整微突发流信息的方法无法兼顾高准确性(细粒度)和低带宽开销(轻量级)两种特性。

因此，本文研究了一种轻量级细粒度的微突发测量方法，并实现了相应的系统原型，在准确检测并提取微突发流信息的同时能够显著降低传输微突发流信息的带宽开销，解决了目前现有微突发测量方法无法兼顾细粒度检测微突发和轻量化传输微突发流信息的问题。

## 3 检测原理和方法

### 3.1 检测系统总体架构

网络微突发流检测系统部署在基于 Tofino 芯片的 P4 交换机上，Tofino 架构(Tofino Native Architecture，TNA)如图 2 所示，分为入口流水线、流量管理器(Traffic Manager，TM)、出口流水线 3 部分。

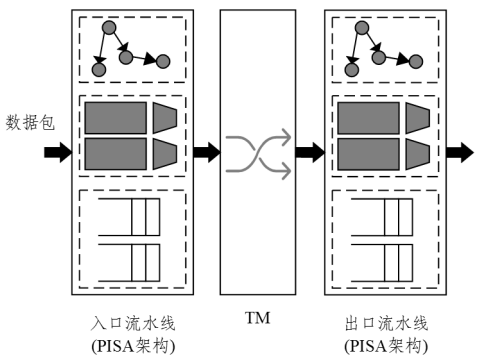


图 2 TNA 架构示意图

Fig. 2 Schematic diagram of TNA architecture

入口和出口流水线规格相同，均遵循 PISA 转发模型。TM 负责转发不同入口发出的报文至指定出口，交换机中队列也堆积于此，测量系统部署于出口流水线位置。

微突发测量系统整体框架如图 3 所示，控制平面通过 bfrt_grpc API 与数据平面 P4 交换机远程通信，进行流表增删改查，同时在交换机数据平面使用定时主动推送的方式主动镜像数据包携带 sketch 数据定时发送到控制器，从而在控制器中读取并解码 sketch 数据，获得微突发流信息。

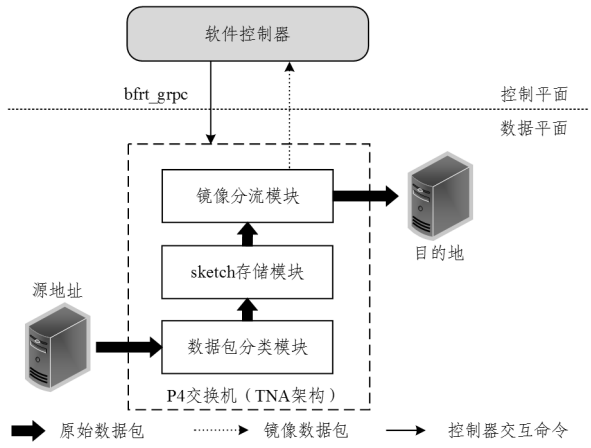


图 3 测量系统架构图

Fig. 3 Diagram of measurement system architecture

P4 交换机的数据平面能够正常转发报文，同时测量微突发流。测量系统分为微突发流检测、微突发流存储和微突发流传输 3 个模块。微突发流检测模块预处理进入出口流水线的包，检测并分类微突发流；微突发流存储模块部署 sketch，实现紧凑的微突发流信息存储功能；微突发流传输模块筛选少部分报文，将 sketch 数据拆分嵌入报文后镜像至控制器。

### 3.2 微突发流检测

测量微突发流，需要先检测微突发异常事件。系统将经历高排队时延的数据包定义为微突发流，利用 PISA 架构能够直接获取 P4 交换机内部性能信息的特性，在 P4 交换机数据平面内细粒度逐包检测排队时延和队列深度，同时系统分析交换机性能，设置了相应的阈值，将数据包分类为不同拥塞类型的数据包，从而达到细粒度检测微突发流的目的。

在 PISA 架构下，进入交换机的数据包会依次通过入口流水线、流量管理器和出口流水线 3 部分。当数据包转发到出口流水线时，P4 交换机在数据包头部向量(Packet Header Vector，PHV)内附加出口固有元数据，TNA 架构下包含出端口 id，以及出队列时队列深度和排队时延等拥塞相关性能数据。

系统读取每个数据包出队列时的队列深度，作为拥塞指标 $\tau$，通过阈值区分不同程度的微突发流。拥塞阈值 $\tau$ 需要分析交换机最大排队时延的具体数值。经过测试，Tofino 芯片下单个 10G 端口缓冲区最大为 15.5 Mb 左右，最大排队时延为 1 550 $\mu$s 左右，直连 P4 交换机的终端平均 RTT 在 80 $\mu$s 左右。

数据包分类如表 1 所列。设 $\tau_{max}$ 为最大队列深度，则队列深度 0～0.05$\tau_{max}$ 分类为无拥塞报文，不处理；中拥塞阈值为 0.05$\tau_{max}$，即分类为时延至少增加一倍 RTT，性能受到影响的报文；高拥塞阈值为 0.2$\tau_{max}$，表明此类报文经历严重拥塞，交换机甚至产生丢包。系统根据所划分的不同阈值进行微突发流的检测，将所检测到的不同程度的拥塞数据包放入微突发流

存储模块中进行存储。

表 1 数据包分类

Table 1 Packet classification

| 出队列时队列深度 | 数据包类型 |
|---|---|
| $0 \sim 0.05\tau_{max}$ | 无拥塞报文 |
| $0.05 \sim 0.2\tau_{max}$ | 中拥塞报文 |
| $0.2 \sim 1.0\tau_{max}$ | 高拥塞报文 |

### 3.3 微突发流存储

微突发发生时瞬时流量速率大于出端口带宽，产生大量堆积报文，系统使用两个 sketch 分别存储中、高拥塞报文，避免使用 INT 镜像每个拥塞报文，大幅减少带宽开销。微突发流分类存储过程如图 4 所示。

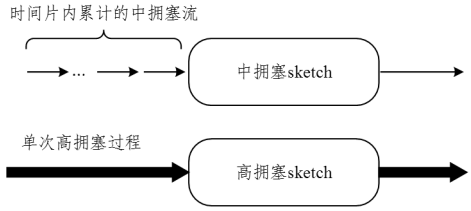


图 4 微突发流存储示意图

Fig. 4 Schematic diagram of microburst traffic storage

由于中拥塞过程报文数量少，且间隔时间极短，系统将单个时间片内所有中拥塞流信息累计存储至中拥塞 sketch 中；由于高拥塞报文数量多，同时高拥塞报文的产生表明产生了间歇性的严重微突发，需要及时进行拥塞流信息的展示，因此高拥塞 sketch 只存储单次高拥塞过程的流信息。

现有 sketch 如 Conservative-Update sketch[20]，Count-min sketch，Count sketch[21] 等只在哈希表内设置计数器，未记录流特征(id)，控制器需提前获得全部网络流 id 或镜像传输部分报文直接获得流 id 才可查询 sketch 数据。为了避免这部分带宽开销，同时为了能够在常数时间内存储完整的流信息，减少传输微突发流信息的带宽开销，系统首次在微突发流检测中选用可逆布隆查找表(Invertible Bloom filter Look-up Table，IBLT)[22] 存储流信息。

IBLT 的核心思想是固定哈希表大小，不处理哈希碰撞，而是主动“拥抱”碰撞，将待插入元素与槽位内已有数据使用异或操作相结合，实现常数级时间复杂度的数据插入过程。

IBLT 数据结构如图 5 所示，分为布隆过滤器(Bloom Filter，BF)和计数表两部分。在计数表内每个槽中存储以下 3 种数据：

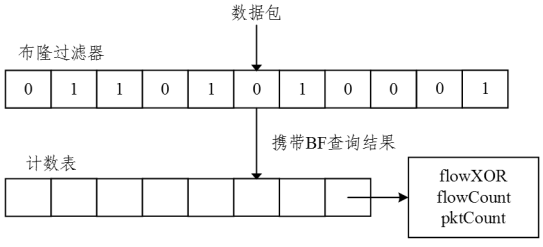


图 5 IBLT 结构图

Fig. 5 Diagram of IBLT structure

1)flowXOR：流 id 异或值，代表映射存储到该槽内所有网络流 id 的异或值。流特征可自定义，系统使用五元组(源/目的 IP，源/目的端口，传输层协议号)作为流 id。

2)flowCount：流计数，代表存储到该槽内所有不同网络流的流个数。

3)pktCount：包计数，代表存储到该槽内所有数据包的个数。

当数据包插入 IBLT 时，BF 先判断其是否为新的网络流(即流 id 不在原有的集合内)，之后根据 BF 的查询结果插入计数表。IBLT 数据插入过程如图 6 所示。流 F1，F2 首个报文通过 BF 判断为新流，则哈希后对应槽内 3 个值全部更新：flowXOR 与新流 id 异或存储，flowCount 加 1，pktCount 加 1。如果 BF 判断不为新流，表明该流 id 已存储，pktCount 加 1 即可。

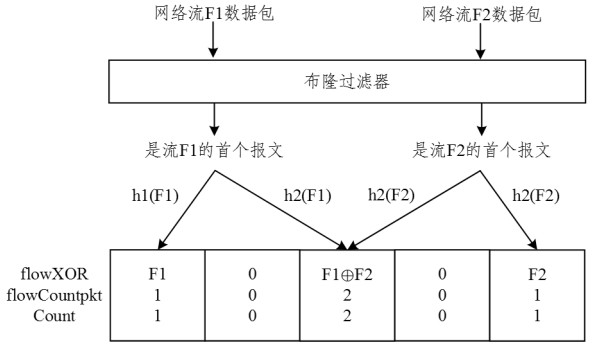


图 6 IBLT 数据插入过程

Fig. 6 IBLT data insertion process

IBLT 数据解码过程如图 7 所示，先遍历计数表查找 flowCount 值为 1 的槽，此类槽表示没有发生哈希碰撞，flowXOR，pktCount 即为流 id 和流大小。提取流信息后再次散列流 id 获得对应槽位，将槽内 flowXOR 再次异或流 id 删除该流特征，槽内包数量 pktCount 减去流大小，流计数 flowCount 减 1，即可完成单个流信息的解码和消去。循环解码单个流的操作，直至 IBLT 全部解码或剩余槽位无法解码。

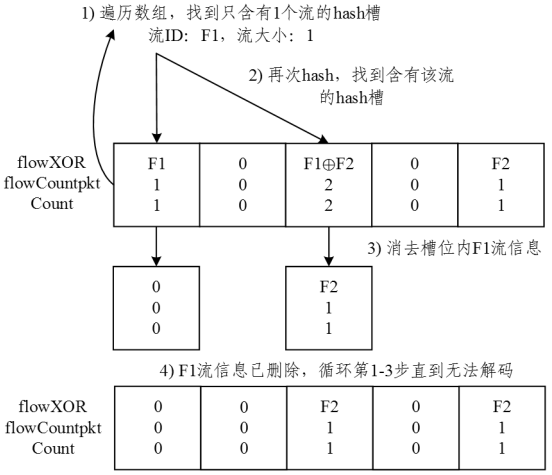


图 7 IBLT 数据解码过程

Fig. 7 IBLT data decoding process

IBLT 存储算法属于流算法的一种，设 BF 和计数表大小为 $m$，使用 $k$ 个哈希函数。其算法性能分析如下：

1)时间复杂度：网络流元素插入过程中，BF 和计数表只对每个哈希函数分别进行一次操作，IBLT 单次数据插入的时间复杂度为 $O(2k)$。IBLT 单次解码需遍历计数表找到

"纯净槽"并消去流信息,时间复杂度 O($mk$)。完整解码需要循环单次解码过程直至无法解码,假设插入 $n$ 个不同的网络流元素,则完整解码时间复杂度为 O($nmk$)。

2)误判率:IBLT 基于 BF 查询结果筛选新流,假设插入 $n$ 个不同的网络流元素,$BF$ 误判率如式(1)所示:

$$\left(1-\left[1-\frac{1}{m}\right]^{kn}\right)^{k}\approx(1-\mathrm{e}^{-kn/m})^{k} \tag{1}$$

对于计数表,已被证明如果表大小 $m>c_k n$($c_k$ 是与哈希函数个数 $k$ 正相关的常数系数),则计数表完全解码的成功率大于 O($1-n^{-k+2}$)。

3)空间复杂度:对于 BF,给定误判概率 $p$,最优的位数组大小 $m$ 的计算式如式(2)所示,$m$ 与插入元素个数成线性正比关系时最佳。

$$m=\frac{n\ln p}{(\ln 2)^2} \tag{2}$$

对于计数表,已被证明表大小 $m>c_k n$ 时成功解码率较高,则空间复杂度为 O($n$)。

在 TNA 架构的 P4 交换机中,存在的资源限制主要有以下两点:

1)流水线每个阶段只能访问各自附带的内存块,当一个数据包通过整个流水线时,流水线每一级的内存块只能被访问一次。这也是无法简单部署一个缓冲哈希表,当数据包入队列时插入,出队列时删除来存储队列信息的原因。

2)TNA 的内存以寄存器数组的形式进行分配,数组内单个元素只能是 8,16,32 位整型值,或是上述 3 种元素的一对(pairs 变量),即单个数据包在单个流水线阶段最多只能修改单个数组中的连续 64 位。

由于以上两种资源访问的限制,因此需要拓展原始 IBLT 数据结构以便部署于 P4 交换机内。IBLT 拓展后如图 9 所示,哈希表与哈希函数个数相同,不同哈希函数只对各自的哈希表进行散列;计数表被拆分为 4 份,每份内部元素长度均满足 TNA 限制。

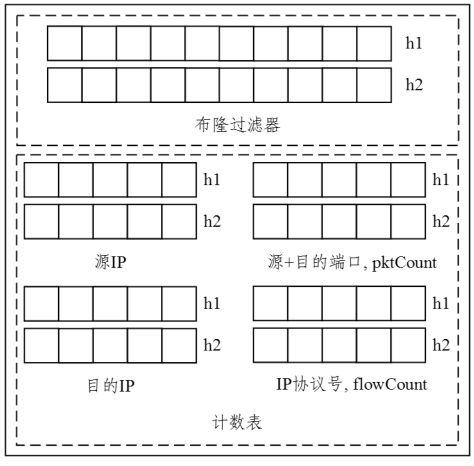


图 8 IBLT 实际部署示意图

Fig. 8 Schematic diagram of the actual deployment of IBLT

### 3.4 微突发流传输

微突发结束或者时间片结束后,系统控制器需要从交换机数据平面的寄存器中获取微突发流信息。现有方法使用 bfrt_grpc API 与交换机数据平面建立远程通信以读取/重置寄存器数组。经过测试,使用 API 读取长度为 1 024 的单个寄存器数组时间开销均在 200 ms 以上,性能较差。200 ms 内过多报文插入 sketch 可能产生大量哈希碰撞,导致 IBLT 解码成功率大幅下降,并且读取和重置 sketch 两条指令之间插入的微突发流信息会丢失。

为了降低系统从数据平面获取微突发流信息的时延,同时使用较小的带宽开销从而更轻量化地传输微突发流信息,系统采用镜像部分数据包并在数据包头部嵌入非零 sketch 数据将其定时主动推送到控制器,从而能够在控制器中读取交换机内的 sketch 数据并解码获取微突发流信息。交换机数据平面中的镜像数据包在嵌入微突发流信息的同时清空 sketch 数据,同时,镜像数据包的包头需携带其自身队列深度等信息,从而保证控制器收到的 sketch 数据和数据平面中所存储的 sketch 数据的一致性。

如图 9 所示,系统将 sketch 数据拆分嵌入到多个数据包中发送至控制器。当 1 ms 时间片结束时,系统通过到达出口流水线的报文遍历 IBLT 内计数表每一槽位,只将非零数据嵌入包头,同时记录 IBLT 序号和计数表列号后镜像传输至控制器,并将交换机内该槽位清零。IBLT 最后一列的数据无论是否为零都需镜像至控制器,这标志着传输过程结束。

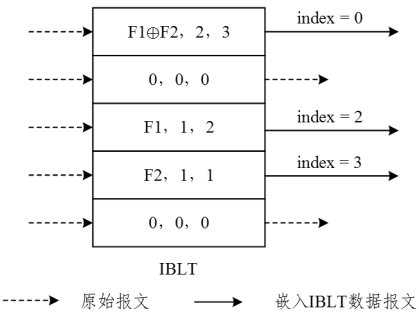


图 9 拆分嵌入 sketch 数据

Fig. 9 Splitting and embedding of sketch data

镜像数据包的报文格式如图 10 所示,系统剔除报文载荷,只保留数据包包头并嵌入 IBLT 数据信息,控制器解析位图后解码获得流信息。即使镜像报文也经历微突发,控制器也可根据包头和队列深度获得最新微突发流信息。

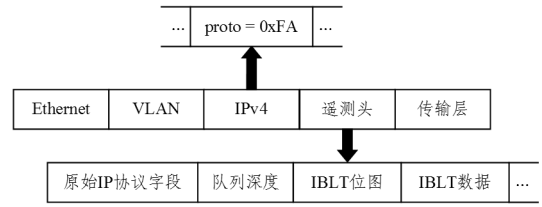


图 10 镜像报文格式

Fig. 10 Mirror packet format

## 4 实验测试与分析

### 4.1 实验平台

实验环境如图 11 所示,3 台服务器与 P4 交换机相连,右侧服务器接收所有流量,链路带宽 10 Gbps 为瓶颈链路,左侧服务器一台发送 9 Gbps 的背景流量,另一台间歇性以 10 Gbps 的速率发送短时突发流量,在 P4 交换机制造微突发。P4

交换机自身的管理 CPU 作为控制器，接收镜像报文并解码获得微突发流信息。设备信息如表 2 所列。

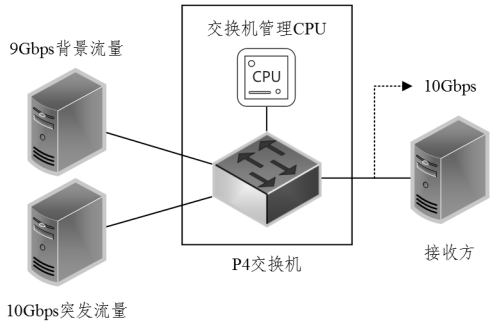


图 11 实验环境

Fig. 11 Experimental environment

表 2 实验设备配置信息

Table 2 Configuration information of experimental equipment

| 项目 | 配置信息 |
|---|---|
| P4 交换机芯片 | TF10-032D(Tofino 架构) |
| 交换机吞吐量 | 2.45 Tbps |
| 交换机管理扣卡 CPU | Inter(R) Atom(TM) C3558 |
| 管理扣卡内存 | 8 GB |
| 网络拓扑链路带宽 | 10 Gbps |
| 服务器网卡 | Intel 82599ES |

注：交换机管理扣卡即交换机控制器。

### 4.2 微突发流分析实验

为了测试算法的准确率，需要寻找突发性较强的开放数据集，收集微突发流进行软件拟真实验。实验选择公开的威斯康星大学数据中心流量[23]，对其中 UNI1(数据中心 1)数据集中的所有流量进行重放，依次统计带宽进行分析，结果如表 3 所列。

表 3 交换机节点流量统计结果

Table 3 Statistical results of switch node traffic

| 节点类型 | 节点编号 | 强突发性节点 |
|---|---|---|
| TOR 交换机 | pt1－pt14 | pt1，pt4，pt5，pt6，pt7 |
| 核心交换机 | pt15－pt20 | 无 |

TOR 交换机节点中 pt1，pt4－pt7 突发性较强(其中 TOR 交换机 pt1 带宽统计如图 12 所示)，而核心交换机无明显突发现象，故采用突发性较强的节点作为微突发流的原始数据集。

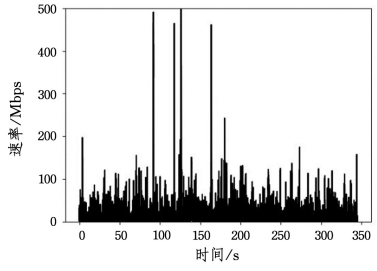


图 12 TOR 交换机 pt1 带宽统计(10 ms 粒度)

Fig. 12 Pt1 bandwidth statistics of TOR switch(10 ms granularity)

数据集描述中原始链路带宽为 100 Mbps 和 1 Gbps，整体平均吞吐量只有 25.3 Mbps，故 50 倍速率重放数据集，入口带宽不限制，峰值速率达到 25 Gbps，出口带宽 10 Gbps 作为瓶颈带宽，使用先进先出队列模拟排队时延。

部分结果如图 13 所示，纵轴是微秒单位的排队时延，横轴是原时间戳的 1/50，可看出 1.8～3.3 s 内发生了 4 次微突发，排队时延均大于 2ms，符合思科和华为定义的微突发持续时间。

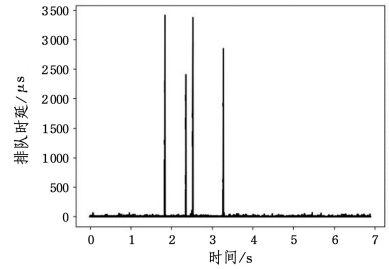


图 13 排队时延模拟结果

Fig. 13 Queuing delay simulation results

根据 3.2 节确定的中拥塞阈值为 $0.05\tau_{max}$ 即排队时延从 80 μs 起始；高拥塞阈值为 $0.2\tau_{max}$ 即排队时延从 310 μs 起始，通过阈值从原始数据集内筛选出拥塞流量。其中高拥塞过程中包数量为 5000～13000 个，平均含有 300 个流，中拥塞过程平均小于 500 个包。

实验统计每次拥塞过程中每个网络流的数据包个数作为流大小，进行微突发流特征的分析，并绘制了微突发流大小的累积分布函数图，如图 14 所示。纵轴为累计分布比例，横轴为网络流的流大小，以对数形式缩放。

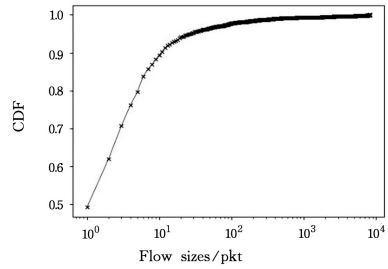


图 14 微突发流大小 CDF

Fig. 14 Microburst traffic size CDF

由 CDF 曲线可得，突发流的大小呈现出严重的长尾分布效应，90％的网络流只有不到 12 个数据包，而极少部分大流则飙升至 8000 个数据包。由此可见，微突发流大多以间接性突发集体产生，主要特点为持续时间短，拥塞流种类少，数据包个数多。

其中 CDF 图的最左侧点代表只有 1 个数据包的网络流，比例接近 50％，通过分析可知，其中含有大量的端口扫描报文。同时，发现了多对一的 TCP 流量，即 TCP Incast 现象。DDoS 同样可引发该现象，这表明检测微突发流可截获潜在的部分端口扫描流量和 DDoS 流量，具有一定安全功能。

### 4.3 IBLT 准确性验证实验

如图 8 所示，如果 IBLT 使用过多哈希函数，交换机内 SRAM，ALU 等资源的占用会成倍上升，空闲流水线级数也会被挤压。哈希函数个数设为 $k$，实验插入高拥塞流后解码，测试所有拥塞流数据集在不同哈希函数个数和不同哈希表大小下的流召回率和流大小查询准确率的平均值，

结果如图 15 和图 16 所示。

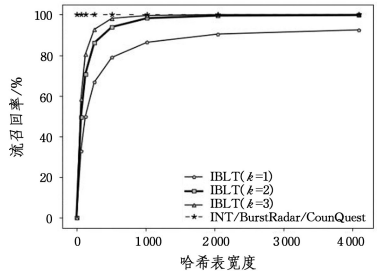


图 15 流召回率实验结果

Fig. 15 Flow recall experiment results

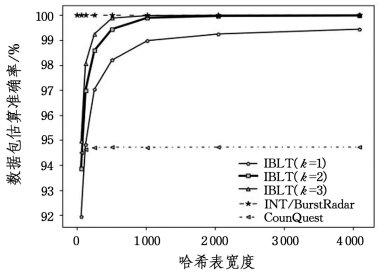


图 16 流大小查询准确率实验结果

Fig. 16 Experiment results of flow size query accuracy

由第 2 章研究现状可知，INT/BurstRadar 方法完整镜像所有拥塞报文，流召回率和流大小查询准确率为 100%，将其作为基准值。ConQuest 在数据平面检测高排队延迟的流，其流召回率为 100%，但其从快照中获取流大小时存在四舍五入的查询误差(例如报文经历 3.4 ms 排队时延，四舍五入只读取 3 个快照)，哈希表足够大时准确率只接近 95%。

从图 15 和图 16 可以看出，IBLT 的流召回率和流大小查询准确率随着哈希表个数和哈希表宽度的增加逐渐上升，但是在哈希函数个数 $k \geqslant 2$、哈希表宽度大于等于 1024 时，IBLT 的流召回率和流大小查全率接近于 100%，且增长缓慢。因此，为了节省更多交换机数据平面稀少的内存资源，同时也为了保证 IBLT 的准确性，我们选用哈希函数个数为 $k=2$，哈希表宽度为 1024，此时 IBLT 的流召回率高达 98.25%，流大小查询准确率高达 99.99%。

此外，为了进一步验证 IBLT 的准确性，我们从数据集中提取了 0～600 个流存入 IBLT 中，在哈希表宽度为 1024 时设置不同的 $k$ 值，在 $k=2$ 时设置不同的哈希表宽度进行解码，测试 IBLT 的流召回率和流大小查询准确率，其结果如图 17－图 20 所示。

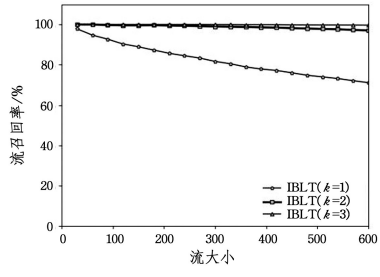


图 17 流召回率实验结果(不同 $k$ 值)

Fig. 17 Flow recall experiment results(different $k$-values)

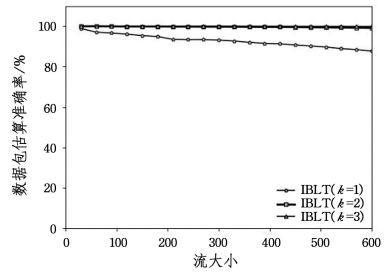


图 18 流大小查询准确率实验结果(不同 $k$ 值)

Fig. 18 Experiment results of flow size query accuracy (different $k$-values)

从图 17 和图 18 可以看出，随着流个数的增加，IBLT 在哈希函数个数 $k=1$ 时由于 IBLT 中纯净槽减少致使解码率下降，从而导致流召回率和流大小查询准确率不断下降，而哈希函数个数 $k=2$ 和 $k=3$ 的流召回率和流大小查询准确率基本不变，接近于 100%。

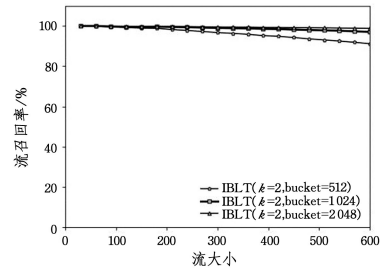


图 19 流召回率实验结果(不同哈希表宽度)

Fig. 19 Flow recall experiment results(different hash table widths)

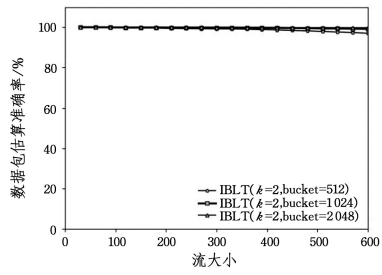


图 20 流大小查询准确率实验结果(不同哈希表宽度)

Fig. 20 Experiment results of flow size query accuracy(different hash table widths)

从图 19 和图 20 可以看出，随着流个数的增加，IBLT 在哈希表宽度为 512 时由于 IBLT 中纯净槽减少致使解码率下降，从而导致流召回率和流大小查询准确率不断下降，而当哈希表宽度为 1024 和 2048 时流召回率和流大小查询准确率基本不变，接近于 100%。

综合以上拟真实验结果可知，当 IBLT 哈希函数个数 $k=2$、哈希表宽度为 1024 时，算法具有高准确率，与原始 INT 统计结果几乎无差异，实现了细粒度检测微突发的目标。

### 4.4 系统带宽开销实验

通过仿真实验，确定 P4 程序内参数如下。

1)中拥塞流使用 IBLT 存储，IBLT 使用 2 个哈希函数，BF 和拆分后计数表各个部分的数组大小均为 1024，数据包每 10 ms 遍历 IBLT 数据并传输。

2）高拥塞流使用另一个 IBLT 存储，参数与中拥塞相同。当高拥塞开始后连续 1 ms 内未检测到高拥塞报文时，系统判断该次高拥塞过程结束，数据包遍历 IBLT 后镜像至控制器。

系统环境如图 11 所示，接收方持续收到 9 Gbps 速率的背景流量，突发服务器以 10 Gbps 的速率间歇性重放筛选的拥塞报文，制造微突发。交换机的管理 CPU 收到镜像报文，统计报文数量并与 INT/BurstRadar 方法进行对比，结果如图 21 所示。INT 镜像所有拥塞报文，BurstRadar 通过队列深度阈值筛选镜像的数据包，在拥塞期间不会减少镜像数据包数量，结果基本与 INT 一致。ConQuest 方法不传输拥塞报文，无法获取微突发流信息，故不作对比。本系统只镜像 IBLT 的非零列，单次拥塞过程中镜像包不超过 1 024 个，平均镜像报文数量为 INT/BurstRadar 的 15.35%。

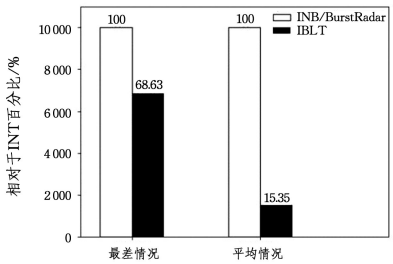


图 21 镜像数据包数量对比

Fig. 21 Comparison of the number of mirror packets

镜像报文带宽占用结果如图 22 所示，INT 和 BurstRadar 均使用数据包截断技术，即剔除镜像包载荷后只传输嵌入遥测数据的包头，带宽占用结果一致。本系统镜像报文中嵌入了两个 IBLT 的 sketch 数据，也含有少数遥测信息，数据包长度相较于 INT 遥测报文稍大，但是平均镜像带宽仅为 INT/BurstRadar 的 16.38%。

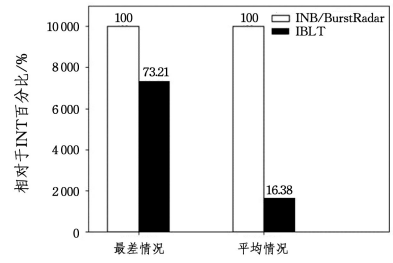


图 22 镜像带宽开销对比

Fig. 22 Comparison of mirror bandwidth overhead

综合实验结果，系统利用 IBLT 存储大量微突发流数据，避免镜像每个拥塞报文，大幅降低了传统 INT 方法的镜像带宽开销，实现了轻量化带宽开销的目标。

**结束语** 检测并提取网络微突发流信息对诊断网络故障、维护网络可用性具有重要意义。一种基于 sketch 的细粒度轻量级微突发流量方法实现了逐包细粒度检测微突发事件，并利用 sketch 分类聚合微突发流，将流信息压缩编码至紧凑的 IBLT 数据结构中，牺牲了少量准确率的同时大幅减少了传输流信息的镜像带宽开销，并且在数据平面采用定时主动推送的方式将微突发流信息传输到控制器，使微突发流检测能够完全运行在数据平面。本文方法在搭载 Tofino 芯片的 P4 交换机上进行了部署，实验使用开源数据中心流量数据集对系统进行了相应的实验。实验结果表明，该系统能够实时检测网络微突发流量，召回率和准确率分别超过 98%和 99.9%，同时显著降低了微突发信息传输带宽。

## 参 考 文 献

[1] ZHANG Q，LIU V，ZENG H，et al. High-resolution measurement of data center microbursts[C]// Proceedings of the 2017 Internet Measurement Conference. New York：Association for Computing Machinery，2017：78-85.

[2] DUTT D G. Cloud Native Data Center Networking：Architecture，Protocols，and Tools[M]. O'Reilly Media，2019：23-32.

[3] FIRESTONE D，PUTNAM A，MUNDKUR S，et al. Azure Accelerated Networking：SmartNICs in the Public Cloud[C]// 15th USENIX Sym-posium on Networked Systems Design and Implementation（NSDI 18）. BERKELEY：USENIX Association，2018：51-66.

[4] SHAN D F，REN F Y，CHENG P，et al. Micro-burst in data centers：Observations，analysis，and mitigations[C]// 2018 IEEE 26th International Conference on Network Protocols（ICNP）. Los Alamitos：IEEE Computer Society，2018：88-98.

[5] YASEEN N，SONCHACK J，LIU V. Synchronized network snap-shots[C]// Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication. New York：Association for Computing Machinery，2018：402-416.

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

[7] ALCOZ A G，STROHMEIER M，LENDERS V，et al. Aggregate-based congestion control for pulse-wave DDoS defense[C]// Proceedings of the ACM SIGCOMM 2022 Conference. New York：Association for Computing Machinery，2022：693-706.

[8] Monitor Microbursts on Cisco Nexus 5600 Platform and Cisco Nexus 6000 Series Switches[DB/OL]. https：//goo. gl/5Xxhpm，2022-5.

[9] What Is a Microburst? How to Detect a Microburst?. Huawei[DB/OL]. https：//www. support. huawei. com/enterprise/en/doc/EDOC1100086962，2020-11.

[10] KIM C，SIVARAMAN A，KATTA N，et al. In-band network telemetry via programmable dataplanes[C]// ACM SIGCOMM. New York：Association for Computing Machinery，2015.

[11] JOSHI R，QU T，CHAN M C，et al. BurstRadar：Practical real-time microburst monitoring for datacenter networks[C]// Proceedings of the 9th Asia-Pacific Workshop on Systems. New York：Association for Computing Machinery，2018：1-8.

[12] TAFFET P，MELLOR-CRUMMEY J. Lightweight，Packet-Centric Monitoring of Network Traffic and Congestion Implemented in P4[C]// 2019 IEEE Symposium on High-Performance Interconnects（HOTI）. LOS ALAMITOS：IEEE Computer Society，IEEE，2019：54-58.

[13] BUCCAPATNAM S，CHEN X Q，DUELL K，et al. Fine-grained

P4 measurement toolkit for buffer sizing in carrier grade networks[C]// BS'19: 2019 Workshop on Buffer Sizing. New York: Association for Computing Machinery, 2019.

[14] CHEN X Q, FEIBISH S L, KORAL Y, et al. Fine-grained queue measurement in the data plane[C]// Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies. New York: Association for Computing Machinery, 2019: 15-29.

[15] ZHOU Y, SUN C, LIU H H, et al. Flow event telemetry on program-mable data plane[C]// Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication. New York: Association for Computing Machinery, 2020: 76-89.

[16] FEIBISH S L, LIU Z, IVKIN N, et al. Flow-level loss detection with Δ-sketches[C]// Proceedings of the Symposium on SDN Research. New York: Association for Computing Machinery, 2022: 25-32.

[17] BRUM H B, DOS SANTOS C R P, FERRETO T C. Providing Fine-grained Network Metrics for Monitoring Applications using In-band Telemetry[C]// 2023 IEEE 9th International Conference on Network Softwarization (NetSoft). New York: IEEE, 2023: 116-124.

[18] MAZLOUM A, GOMEZ J, KFOURY E, et al. Enhancing perfSONAR Measurement Capabilities using P4 Programmable Data Planes[C]// Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis. New York: Association for Computing Machinery, 2023: 819-829.

[19] CORMODE G, MUTHUKRISHNAN S. An improved data stream summary: the count-min sketch and its applications[J]. Journal of Algorithms, 2005, 55(1): 58-75.

[20] ESTAN C, VARGHESE G. New directions in traffic measurement and accounting[C]// Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. New York: Association for Computing Machinery, 2002: 323-336.

[21] CHARIKAR M, CHEN K, FARACH-COLTON M. Finding frequent items in data streams[C]// International Colloquium on Automata, Languages, and Programming. AMSTERDAM: ELSEVIER, 2002: 693-703.

[22] GOODRICH M T, MITZENMACHER M. Invertible bloom lookup tables[C]// 2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton). New York: IEEE, 2011: 792-799.

[23] BENSON T, AKELLA A, MALTZ D A. Network traffic characteristics of data centers in the wild[C]// Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement. New York: Association for Computing Machinery, 2010: 267-280.

**WANG Jiayu**, born in 1999, postgraduate. His main research interests include programmable data plane and network security.

**YU Junqing**, born in 1975, Ph.D, professor, Ph.D supervisor, is a member of CCF (No. 05665S). His main research interests include digital media processing and retrieval, network security, multi-core computing and stream compilation.

(责任编辑:何杨)