计算机科学
COMPUTER SCIENCE

EAGLE:一种内核态及用户态中基于遥测数据图的网络遥测方案

肖肇斌, 崔允贺, 陈意, 申国伟, 郭春, 钱清

引用本文

肖肇斌, 崔允贺, 陈意, 申国伟, 郭春, 钱清. EAGLE:一种内核态及用户态中基于遥测数据图的网络遥测方案[J]. 计算机科学, 2024, 51(2): 311-321.

XIAO Zhaobin, CUI Yunhe, CHEN Yi, SHEN Guowei, GUO Chun, QIAN Qing. EAGLE:A Network Telemetry Mechanism Based on Telemetry Data Graph in Kernel and UserMode [J]. Computer Science, 2024, 51(2): 311-321.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.221100196

EAGLE:一种内核态及用户态中基于遥测数据图的网络遥测方案

肖肇斌[1,2,3]　崔允贺[1,2,3]　陈　意[1,2,3]　申国伟[1,2,3]　郭　春[1,2,3]　钱　清[4]
1 贵州大学计算机科学与技术学院　贵阳 550025
2 省部共建公共大数据国家重点实验室　贵阳 550025
3 文本计算与认知智能教育部工程研究中心　贵阳 550025
4 贵州财经大学信息学院　贵阳 550025
(2873121871@qq.com)

摘　要　网络遥测是一种新型的网络测量技术,具有实时性强、准确性高、开销低的特点。现有网络遥测技术存在无法收集多粒度网络数据、无法有效存储大量原始网络数据、无法快速提取及生成网络遥测信息、无法利用内核态及用户态特性设计网络遥测方案等问题。为此,提出了一种融合内核态及用户态的、基于遥测数据图和同步控制块的多粒度、可扩展、覆盖全网的网络遥测机制(a nEtwork telemetry mechAnism based on telemetry data Graph in kerneL and usEr mode,EAGLE)。EAGLE设计了一种能够收集多粒度数据且数据平面上灵活可控的网络遥测数据包结构,用于获取上层应用所需的数据。此外,为快速存储、查询、统计、聚合网络状态数据,实现网络遥测数据包所需遥测数据的快速提取与生成,EAGLE提出了一种基于遥测数据图及同步控制块的网络遥测信息生成方法。在此基础上,为了最大化网络遥测机制中网络遥测数据包的处理效率,EAGLE提出了融合内核态及用户态特性的网络遥测信息嵌入架构。在Open vSwitch上实现了EAGLE方案并进行了测试,测试结果表明,EAGLE能够收集多粒度数据并快速提取与生成遥测数据,且仅增加极少量的处理时延及资源占用率。

关键词:网络遥测;遥测效率;可编程数据平面;遥测数据图;内核空间
中图分类号　TP393

EAGLE:A Network Telemetry Mechanism Based on Telemetry Data Graph in Kernel and User Mode

XIAO Zhaobin[1,2,3],CUI Yunhe[1,2,3],CHEN Yi[1,2,3],SHEN Guowei[1,2,3],GUO Chun[1,2,3] and QIAN Qing[4]
1 School of Computer Science and Technology,Guizhou University,Guiyang 550025,China
2 State Key Laboratory of Public Big Data,Guiyang 550025,China
3 Engineering Research Center for Text Computing and Cognitive Intelligence,Ministry of Education,Guiyang 550025,China
4 School of Information,Guizhou University of Finance and Economics,Guiyang 550025,China

Abstract　Network telemetry is a new type of network measurement technology,which has the characteristics of strong real-time performance,high accuracy and low overhead. Existing network telemetry technologies have problems such as being unable to collect multi-granularity network data,unable to effectively store a large amount of original network data,unable to quickly extract and generate network telemetry information,and unable to design network telemetry solutions using kernel-mode and user-mode features. In order to solve the above problems,this paper proposes a multi-granularity,scalable,and network-wide network telemetry mechanism EAGLE,which integrates kernel mode and user mode,and is based on telemetry data graphs and synchronization control blocks. EAGLE has designed a flexible and controllable network telemetry packet structure on the data plane that can collect multi-granularity data,and is used to obtain the data required by upper-layer applications. In addition,in order to quickly store,query,count,and aggregate network status data,and realize the rapid extraction and generation of telemetry data required by network telemetry packets,EAGLE proposes a network telemetry information generation method based on telemetry data graphs and synchronization control blocks. On this basis,in order to maximize the processing efficiency of network telemetry packets in the network telemetry mecha-nism,EAGLE proposes a network telemetry information embedding architecture that in-

到稿日期:2022-11-24　返修日期:2023-06-22
基金项目:国家自然科学基金青年科学基金(62102111);贵州省科技计划项目([2020]1Y267);贵州大学引进人才项目([2019]52)
This work was supported by the Young Scientists Fund of the National Natural Science Foundation of China(62102111),Science and Technology Project of Guizhou Province([2020]1Y267) and Talent Introduction Project of Guizhou University([2019]52).
通信作者:崔允贺(yhcui@gzu.edu.cn)

tegrates the characteristics of kernel state and user state. Finally, this paper implements and tests the EAGLE scheme on Open vSwitch. The test results show that EAGLE can collect multi-granularity data and quickly extract and generate telemetry data with only a little increase in processing time and resource usage.

Keywords Network telemetry, Telemetry efficiency, Programmable data plane, Telemetry data graph, Kernel space

1 引言

随着计算机网络和信息技术的飞速发展，传统数据中心、企业、互联网服务提供商(Internet Service Provider，ISP)对网络的构建和管理面临着越来越大的挑战，同时传统的网络测量也越来越难以满足网络运维的市场需求。软件定义网络(Software-Defined Networking，SDN)[1]和其通信接口标准(OpenFlow 协议)[2]的提出为此提供了一种新的网络架构和网络测量思路。

与传统网络体系结构不同，SDN 将传统网络的控制功能和转发功能分离[3]，并且具有能够集中控制、网络可编程和接口开放的特点[4]。在 SDN 环境下，通过传统采样技术或流表项获取技术可以对数据包进行深入的分析[5]。随着数据平面可编程技术(Programmable Data Plane，PDP)的出现，SDN 的控制平面和转发平面变得更加灵活、开放，传统网络测量技术得到进一步发展。SDN 的转发分离架构使其具备了网络可编程性，网络设备的可编程性进一步催生了网络遥测技术。

网络遥测是一种网络测量技术，其通过自动收集网络中不同来源和类型的状态信息，并将这些信息嵌入数据包中进行传输，以便远程监控、分析和故障排除网络的性能和行为。相较于 sFlow[6]，IPFIX[7]，Netflow[8]，PSAMP[9]等传统网络测量技术，网络遥测可以按需收集细粒度网络状态信息。目前网络遥测技术已被应用于拥塞控制、流量工程、故障定位等领域。作为新型网络测量技术，目前关于网络遥测的研究主要集中在网络遥测机制、遥测数据的编码与传输、遥测任务编排等方面，相关研究成果处于“能测就行”的起步阶段[10]，缺少对大数据量的网络遥测、多粒度网络遥测等方面的研究。

带内网络遥测(In-Bind Network Telemetry，INT)[11]是随着软件定义网络和网络功能虚拟化(Network Function Virtualization，NFV)的发展而提出的一种由数据平面收集和报告网络状态的网络测量技术，其作为一种目前常用的被动网络遥测技术，存在诸多局限。首先，INT 的网络探测范围受到探测点位置的限制，难以获取全局网络视图。这使得 INT 只能探测存在探测点的网络链路出现的网络故障问题，即探测范围受限。其次，INT 在实现上是将网络遥测指令和网络数据封装到用户数据包，这会降低用户数据包的有效载荷比，降低用户体验，同时会限制 INT 遥测的最大数据量。最后，INT 需要在网络遥测的发送端和接收端进行用户数据包的网络遥测指令嵌入、遥测数据提取、网络遥测数据包构造等工作。这会导致用户数据包的传输再次受到影响，同时会额外占用网络设备的资源。

针对上述 INT 存在的网络遥测探测范围有限、可扩展性不足等问题，Liu 等提出并设计了主动网络遥测平台-NetVision[12]。其通过主动发送适当数量和格式的探针数据包实现网络遥测，在一定程度上扩大了网络遥测覆盖范围并提高了其可扩展性，但仍无法解决当前网络遥测技术普遍存在的网络遥测粒度较粗、网络遥测统计数据量局限性较大、遥测数据聚合等问题。

综上所述，现有网络遥测方法存在无法收集多粒度网络数据、无法有效存储大量原始网络数据、无法快速提取及生成网络遥测信息、无法利用内核态及用户态特性设计网络遥测方案的问题。

为解决上述问题，本文提出了一种融合内核态及用户态特性的、基于遥测数据图和同步控制块的多粒度、可扩展、覆盖全网的网络遥测机制(EAGLE)，主要贡献如下：

1)设计了一种能够收集多粒度数据且数据平面上灵活可控的网络遥测数据包结构，用于获取上层应用所需的数据。

2)提出了一种基于遥测数据图的网络遥测信息提取及生成方法。EAGLE 设计了包含同步控制块、遥测数据环形存储块、遥测数据块的遥测数据图存储网络遥测统计数据。EAGLE 是第一个使用环形存储来减少网络遥测统计数据内存占用，也是第一个在网络遥测中使用同步控制块管理内存和遥测数据存储的同步，保证遥测数据的一致性的方法。

3)设计了一种融合内核态及用户态特性的网络遥测信息嵌入架构。该架构针对用户数据包和网络遥测数据包设计了不同的处理流程，即内核态处理流程和用户态处理流程，使得用户数据包无需等待网络遥测数据包处理完成，即可实现网络遥测机制中网络遥测数据包的处理效率最大化和用户数据包吞吐率影响最小化。

4)本文在 OVS 上对 EAGLE 进行了实现及测试，测试结果表明，EAGLE 能够收集多粒度数据并快速提取与生成遥测数据，且仅增加极少量的处理时延及资源占用率。

2 相关工作

网络测量是获得网络行为指标和参数，从而进行高效可靠管控的基本手段[13]。目前有学者对网络遥测机制进行了研究。

阿里巴巴的研究者设计冗余流量事件去除，提出了用于获取交换机上流量事件的网络遥测架构[14]。北京大学研究者使用拆分-合并的方式进行网络遥测，并将其用于网络异常检测[15]。北京邮电大学课题组设计了基于数据分组的网络遥测数据包结构，提出了一种适用于数据中心的网络遥测系统[16]。

明尼苏达大学的研究者设计了包含网络遥测数据包处理、特征选择、在线检测的 BGP 异常检测方案[17]。美国哈佛大学的研究者将元数据嵌入到多个网络遥测数据包中，提出了一种概率网络遥测方法[18]。韩国浦项科技大学课题组通过网络遥测获取网络状态信息，设计了一种基于网络遥测的网络入侵检测方案[19]。

综上，虽然许多学者对网络遥测机制进行了研究，但现有

网络遥测方法收集的数据较为单一，无法收集多粒度网络数据，同时统计过多原始数据会导致数据量偏大，无法有效存储大量的网络原始数据，使其无法快速提取及生成网络遥测信息。此外，现有网络遥测方法不能有效地利用内核态及用户态特性设计网络遥测方案。

为解决以上问题，本文提出了一种融合内核态及用户态特性的、基于遥测数据图和同步控制块的多粒度、可扩展、覆盖全网的网络遥测机制（EAGLE）。

3 EAGLE-内核态及用户态中基于遥测数据图的网络遥测方案

3.1 EAGLE 整体架构

EAGLE 是基于 Open vSwitch（OVS）[20] 进行网络遥测机制的研究与实现。OVS 是一种基于软件编程的多层虚拟软件交换机，主要用于实现网络自动化部署，特别适合部署在虚拟机管理平台上。其能够高效地管理大规模网络，并且提供灵活的配置和控制方式，有助于提高网络管理的效率和可靠性。此外，本文通过 SDN 控制器部署流表的方式灵活控制网络遥测数据包的转发路径。

EAGLE 中的网络遥测路径部署采用了一个环形路径，使网络遥测数据包可以沿着这个路径从探测点出发，经过 SDN 网络设备的转发，最终回到原始探测点。这种路径部署方式可以有效地降低探测点的部署和维护成本，因为其避免了在网络中重复遥测设备，减少了设备数量和相关的维护开销。同时，这种路径还可以确保网络遥测数据包能够在最短时间内到达目的地，提高了遥测数据传输的效率和可靠性。基于 OVS 实现的 EAGLE 架构网络遥测技术路线和数据流向图如图 1 所示。

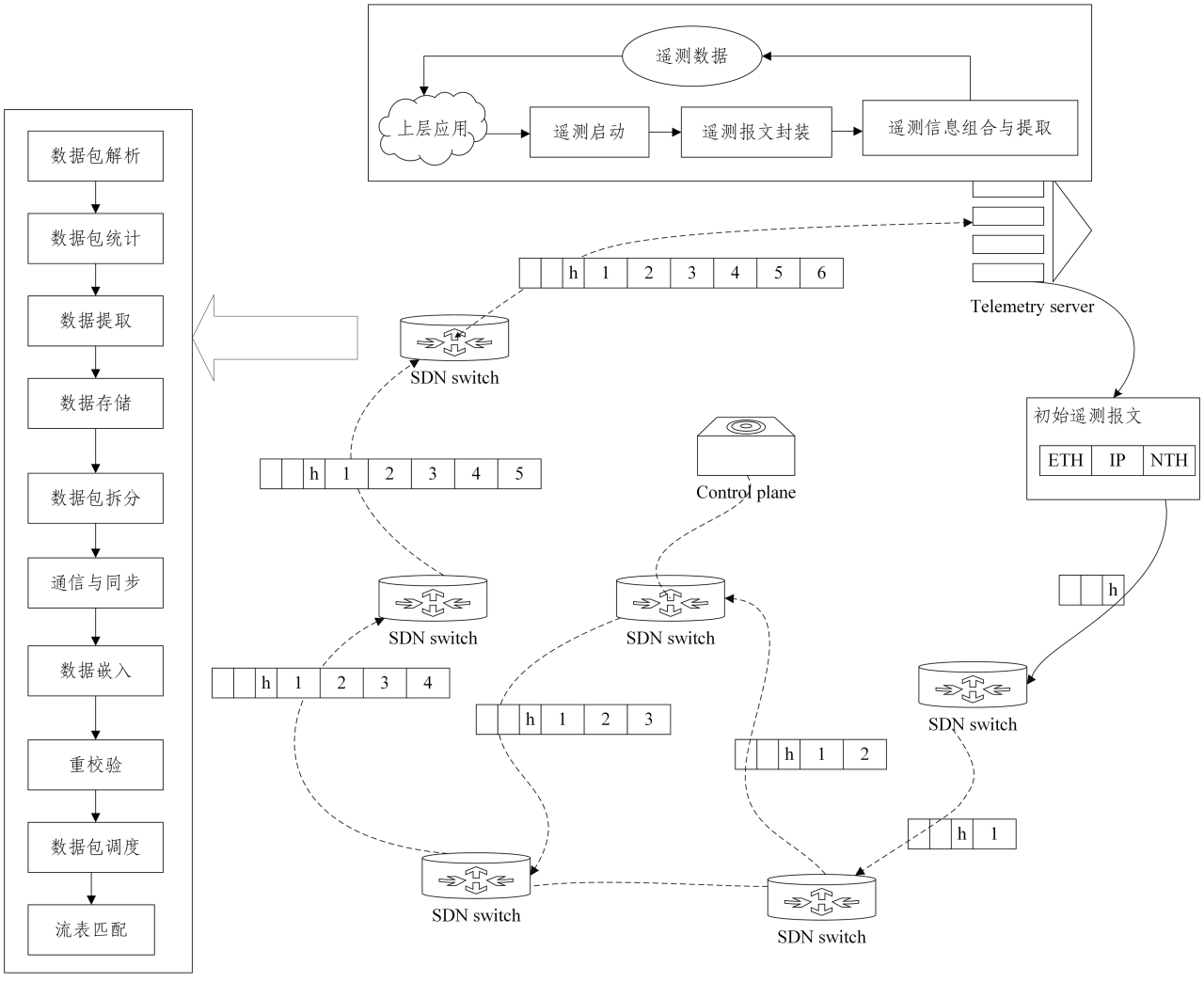


图 1 网络遥测技术路线和数据流向图

Fig. 1 Diagram of elemetry technology and data flow

为了简化系统设计和部署，本文将上层应用和探测点部署在同一台主机（简称遥测服务器）。当启动主动网络遥测时，本文设计的网络遥测数据包从遥测服务器出发，根据控制器部署的网络遥测路径进行转发，同时在每个经过的网络设备上进行数据收集、网络遥测信息嵌入工作，再回到遥测服务器进行网络遥测数据包整合、数据提取等操作，最终将数据发送给上层应用。

3.2 网络遥测数据包结构

EAGLE 设计了一种能够收集多粒度数据，且在数据平面上灵活可控的网络遥测数据包结构，用于获取上层应用所需的遥测数据。所设计的网络遥测数据包由以太网头部、IP 头部、遥测头部及遥测数据部组成。以太网头部和 IP 头部

主要用于确定网络遥测数据包的目的地址、源地址以及网络遥测数据包的识别信息。本方案将 IP 头部协议保留字段(254)作为网络遥测数据包的协议标识,便于网络设备识别网络遥测数据包,并对其进行专门的处理和调度。

同时,为了提高网络遥测数据包分片后的遥测效率,设计了 IP 头部协议保留字段(253)作为执行分片操作后的网络遥测数据包协议标识,网络设备对此数据包可以直接转发,以此降低网络设备的性能开销。

EAGLE 设计的网络遥测数据包结构如图 2 所示。其中,检测号和序列号组成网络遥测过程中的会话 ID,目的是保证网络遥测机制的健壮性。在网络遥测过程中,其一方面能够避免遥测服务器混淆不同上层应用所需的网络遥测数据包,另一方面还能够尽可能地避免网络遥测数据包的丢失。在实现上,当上层应用请求网络遥测时,遥测服务器向待检测网络发送多个具有相同检测号、不同序列号的网络遥测数据包。当遥测服务器收到多个检测号相同的网络遥测数据包时,只将第一个网络遥测数据包发送给上层应用,以此保证网络遥测数据包的时间段准确性。消息长度用于指示当前网络遥测数据包的数据集总长度,同时便于确定遥测数据完整性。

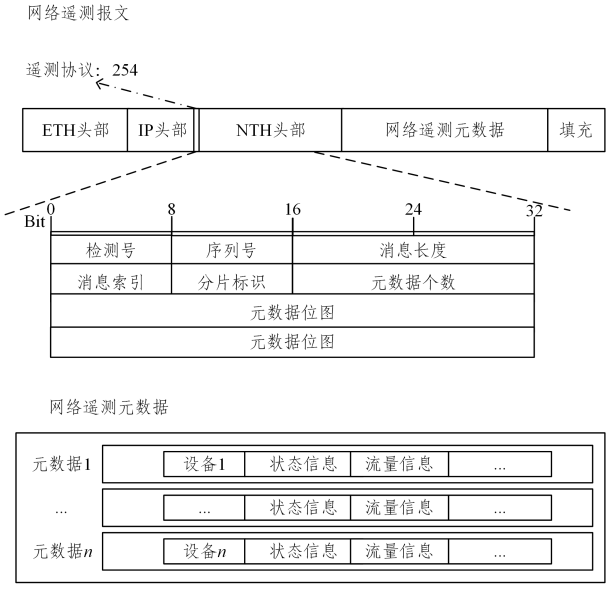


图 2 网络遥测数据包结构

Fig. 2 Network telemetry packet structure

消息索引和分片标识组成网络遥测数据包的分片 ID,主要用于网络遥测数据包传输过程中超过最大传输单元(Maximum Transmission Unit,MTU)时执行分片操作及后续遥测服务器的组合提取操作。在具体的实现上,消息索引主要用于标识分片操作后包的序号。分片标识主要确定后续是否还存在分片的网络遥测数据包。本方案为保证网络遥测数据包的处理效率和处理逻辑的简化,设计 MTU 的值为网络设备的最大传输单元。

元数据位图用于确定本网络遥测数据包需要收集的元数据类型,主要分为设备 ID、设备状态信息、设备级、流级、全网级等多个不同粒度的数据,交换机据此写入指定类型的数据。元数据位图标识关系如式(1)所示。$[\alpha_1\ \ \alpha_2\ \ \cdots\ \ \alpha_n]$ 标识元数据位;β_x 标识网络遥测分类;μ_{xy} 标识 β_x 中的具体分类数据;"ϕ"为自定义符号,表示位串拼接,即 $len_\beta + y = n$。元数据个数用于指明当前网络遥测数据包数据集部分包含的交换机网络信息个数。

$$[\alpha_1\ \ \alpha_2\ \ \cdots\ \ \alpha_n]=\begin{bmatrix}\beta_1\\ \vdots\\ \beta_x\end{bmatrix}\emptyset\begin{bmatrix}\mu_{11} & \cdots & \mu_{1y}\\ \vdots & \ddots & \vdots\\ \mu_{x1} & \cdots & \mu_{xy}\end{bmatrix} \tag{1}$$

网络遥测数据包数据部分由多个独立的元数据组成,每个元数据通过设备 ID 标识,同时由设备状态和需要收集的网络状态信息组成。

由于 SDN 环境下的网络设备 Datapath-ID 可以唯一标识一台 SDN 交换机,因此,使用 S_{id} 表征交换机,如式(2)所示:

$$S_{id}=\{datapath_{id}\} \tag{2}$$

设备状态信息包括流表项的增长速率、流表项空间占用率、设备内存占用率、CPU 占用率等信息。设备状态信息如式(3)所示:

$$S_{info}=\{rate_1, rate_2, \cdots, rate_n\} \tag{3}$$

可选的网络状态信息分为设备级别、流级别、全网级别等多种分类,其中各个分类又细分为多个特征分类数据。如式(4)所示,$type$ 为网络状态信息分类,共 n 种信息分类;$type_{ny}$ 表示第 n 分类中包含了 y 种数据分类,如设备级分类包含有流表状态信息、端口状态信息、链路状态信息等共 y 种数据分类,每种分类分别包含 $\gamma, \cdots, \delta$ 种具体数据。

$$S_{net}=\begin{bmatrix}\begin{pmatrix}type_{11} & data_{11} & \cdots & data_{1\alpha}\\ \vdots & \vdots & & \vdots\\ type_{1x} & data_{x1} & \cdots & data_{x\beta}\end{pmatrix}\\ \vdots\\ \begin{pmatrix}type_{n1} & data_{11} & \cdots & data_{1\gamma}\\ \vdots & \vdots & & \vdots\\ type_{ny} & data_{y1} & \cdots & data_{y\delta}\end{pmatrix}\end{bmatrix} \tag{4}$$

3.3 网络遥测数据包拆分-合并

网络遥测数据包沿着网络遥测路径根据元数据位图收集网络信息的过程中可能会因网络数据总量过大而出现 IP 分片,从而导致网络遥测处理逻辑更加复杂,降低了遥测处理效率。为此,本文设计了算法 1 所示的网络遥测数据包拆分方法和算法 2 所示的网络遥测数据包合并及信息提取方法来实现网络遥测数据包的拆分与合并。

算法 1 网络遥测数据包拆分方法

输入:原始网络遥测数据包 pkt_{old},网络遥测数据长度 len

输出:待嵌入遥测数据的网络遥测数据包 pkt_{new}

1. 获取 pkt_{old} 总长度 len_{old}
2. if len_{old} + len < MTU then
3. 根据 len 对 pkt_{old} 缓存空间扩容
4. return pkt_{old}
5. else 镜像 pkt_{old} 头部数据→pkt_{new}
6. 修改 pkt_{old} 遥测头部信息
7. 重新计算 pkt_{old} 校验值
8. 转发 pkt_{old}
9. 根据 len 对 pkt_{new} 按需扩容

10. 修改 pkt_{new} 遥测头部信息
11. return pkt_{new}
12. endif

当数据包到达 SDN 交换机时，本方案首先根据网络协议逐层解析头部数据，同时根据 IP 协议字段判断某个数据包是否为网络遥测数据包。如果为网络遥测数据包，首先读取镜像网络遥测数据包的元数据位图，根据元数据位图收集、统计指定的网络状态信息，获取待嵌入的遥测数据长度 *len*，然后执行网络遥测数据包拆分方法。如算法 1 所示，本方案根据 *len* 和原始网络遥测数据包 pkt_{old} 的长度 len_{old}，判断嵌入遥测数据后 pkt_{old} 的总长度是否达到 MTU。若没有达到 MTU，即 pkt_{old} 无需执行分片操作，则本方案根据 *len* 对 pkt_{old} 的存储空间进行扩容，然后返回扩容后的 pkt_{old}。若达到 MTU，本方案首先镜像 pkt_{old} 的头部信息得到新的网络遥测数据包 pkt_{new}；然后将 pkt_{old} 遥测头部的分片标识字段置为 1，再对 pkt_{old} 重新计算数据包校验值并转发；最后根据 *len* 对 pkt_{new} 进行按需扩容，同时修改 pkt_{new} 遥测头部信息，主要包括消息索引字段加 1、分片标识字段置为 0，并返回操作后的 pkt_{new}。

算法 2 网络遥测数据包合并及信息提取方法

输入：接收的网络遥测数据包 pkt，数据缓冲区指示域 ptr

输出：执行结果 result

1. 解析 pkt 遥测头部数据 $head_{pkt}$
2. /* 初始化处理记录，x_y 为最后处理的索引号，$result_{等待}$ 为最后处理的结果 */
3. status=$\{head_{pkt}, result_{等待}, x_y\}$
4. /* 相同检测号、不同序列号的网络遥测数据包是否已经有处理成功的数据 */
5. if 当前检测号存在 $result_{成功}$ then
6. return $result_{成功}$
7. /* 接收到的网络遥测数据包索引字段是否是期望的索引值 */
8. elif $head_{pkt} \cdot x_{pkt} != status.x_y + 1$ then
9. /* 更新处理记录 status */
10. status=$\{head_{pkt}, result_{错误}, x_y\}$
11. return $result_{错误}$
12. endif
13. 读取 pkt 遥测数据部 $data_{pkt}$
14. 数据缓冲区 ptr 按需扩容
15. 写入遥测数据 $data_{pkt} \rightarrow ptr$
16. /* 判断当前会话 ID 是否还存在分片 */
17. if $head_{pkt}$ 的分片标识字段==1 then
18. return $result_{等待}$
19. /* 后续不存在分片，读取遥测头部元数据位图字段，解析类型及具体数据 */
20. else 根据 $head_{pkt}$ 获取元数据位图
21. 解析数据缓冲区 ptr 的遥测数据
22. /* 更新处理记录 status */
23. status=$\{head_{pkt}, result_{成功}, x_y\}$
24. return $result_{成功}$
25. endif

在网络遥测过程中，检测号用于标识不同的网络遥测需求，序列号用于降低丢包带来的影响。遥测服务器首先解析接收的网络遥测数据包 *pkt* 遥测头部，若消息索引和分片标识均为 0，则说明该网络遥测过程不存在分片操作，无需进行数据包的组合，可以根据元数据位图字段解析遥测数据。否则，遥测服务器需要执行如算法 2 所示的网络遥测数据包合并及信息提取方法。遥测服务器首先获取到遥测头部数据 $head_{pkt}$，同时初始化处理记录 *status*。若当前检测号所需的遥测数据已经有处理成功的完整遥测数据，则不再处理相同检测号的遥测数据；反之，遥测服务器根据 $head_{pkt}$ 的消息索引和分片标识以及 *status* 对网络遥测数据包进行整合。当本次网络遥测过程中的分片数据全部写入数据缓冲区 *ptr*，遥测服务器根据数据包的元数据位图字段获取 SDN 交换机写入的源数据类型，然后根据设备 ID 字段、设备状态字段以及相应的源数据类型进行信息提取并发送给上层应用。

3.4 基于遥测数据图及同步控制块的网络遥测信息生成方法

3.4.1 遥测数据图存储结构

在使用网络遥测机制收集多粒度的遥测数据时，网络设备在一个周期内需要收集大量的原始网络状态数据，如网络设备 CPU 占用情况、内存占用情况、流表项增长数、网络设备所有端口收发数据情况等。因此，为实现细粒度的网络遥测信息收集及数据聚合，需要在 SDN 交换机上设计合理、高效的数据结构，以实现网络状态数据的快速存储、查询、统计和聚合。此外，需要在收集网络遥测信息时快速提取和生成遥测数据。

为解决上述问题，EAGLE 设计了名为遥测数据图的存储结构。遥测数据图存储结构由同步控制块、遥测数据环形存储块、遥测数据块组成。数据存储结构设计图如图 3 所示。

1）同步控制块包含统计控制块和遥测控制块。统计控制块包括遥测统计数据头指示域（*data_head_ptr*）和尾指示域（*data_tail_ptr*）、遥测统计读写同步标志（*data_rcu_sign*）和遥测统计数据存储扩展信息（*data_extern_msg*）。*data_head_ptr* 和 *data_tail_ptr* 用于控制遥测数据环形存储块的周期环形轮转及读写；*data_rcu_sign* 用于保证遥测数据存储块在周期轮转时的数据强一致性；*data_extern_msg* 保存有环形空间动态扩容控制信息以及上一周期数据读写指示域等，用于必要时对环形存储块进行扩容。遥测控制块包括遥测数据读指示域（*message_read_ptr*）、遥测数据存储空闲空间（*message_free_size*）、遥测数据读写同步标志（*message_rcu_sign*）、遥测数据扩展控制信息（*message_extern_msg*）。遥测控制块各个字段的作用与统计控制块类似，此处不再赘述。

2）遥测数据环形存储块存储网络遥测统计数据，包括设备 ID、设备状态数据、设备端口信息、IP 统计信息、流统计信息、网络遥测数据包时间戳信息、流表统计信息、流表项统计信息等。遥测数据环形存储块对每类统计信息采用不同的数据结构进行存储，对于较小数据量的统计数据类型（如流表类数据）采用 Hash 桶（Hash_Bucket）的方式进行存储；对于大数据量（如 IP 类数据）则采用 Hash 桶和平衡二叉树存储结构（Hash_AVL）来保证效率，同时，不同类别数据之间存储为有向图的形式，便于数据的检索和聚合。

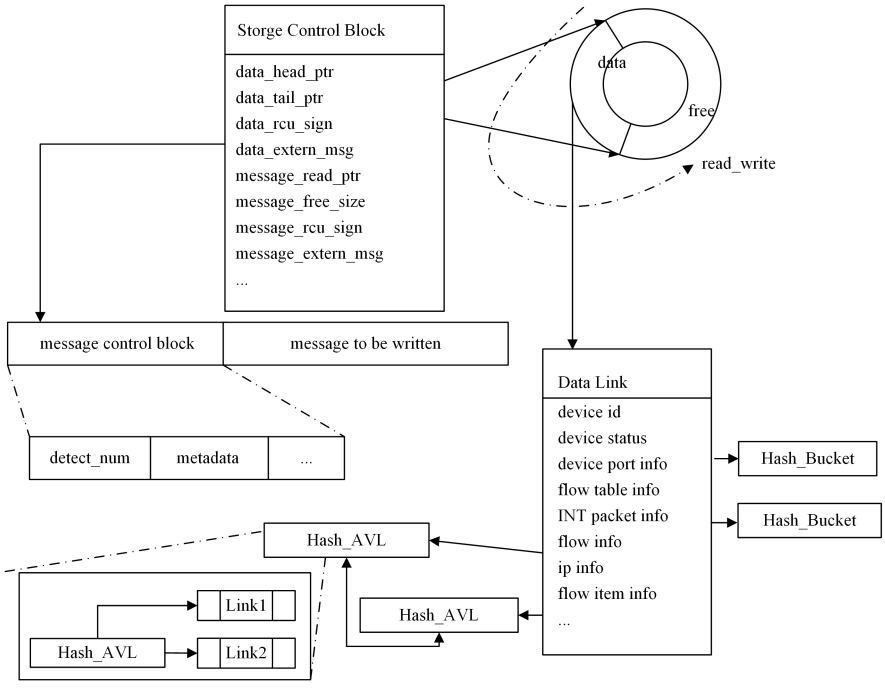


图 3 数据存储结构设计图

Fig. 3 Diagram of data storage structure design

3)遥测数据块由检测号、元数据位图、遥测数据组成。为避免网络遥测机制对同类网络遥测数据包进行重复的提取和生成操作,EAGLE 使用检测号和元数据标明是否执行过相应的网络遥测信息提取和生成操作,对于同类网络遥测数据包不再进行数据提取和生成操作。遥测数据按照元数据位图顺序存储在一片连续的物理空间中,便于后续过程直接嵌入网络遥测数据包。

3.4.2 遥测数据图读写方法

基于遥测数据图及同步控制块的网络遥测信息提取及生成方法需要使用遥测数据存储结构的读写方法进行数据交互并保证遥测数据的强一致性。EAGLE 设计了如算法 3 和算法 4 所示的遥测数据结构的读写方法,其中同步控制块实现所有操作的同步。

算法 3 数据结构读方法

输入:数据指针的指示域 ptr,数据类型 type,数据标识 key,结构类型 offset

输出:执行结果 result

1. /* 同步控制块(contro l_{block})存在写请求,即控制块位置记录信息需要周期更新进行环形存储空间轮转 */
2. if 周期重置标志==1then
3. 等待周期重置完成
4. return $result_{等待}$
5. endif
6. /* 通过 contro l_{block} 和 offset 确定结构类型,即统计控制块或遥测控制块 */
7. contro l_{block}+offset→$block_{type}$
8. /* 操作相应类型读写同步标志,添加数据到相应的读队列 */
9. $block_{type}$的$data_rcu_{sign}$. read+1
10. /* 通过$block_{type}$获取数据存储指示域ptr_{head}和扩展控制信息msg_{extern},保证读写正确性 */
11. 解析$block_{type}$→ptr_{head},msg_{extern}
12. /* 跟据ptr_{head}和msg_{extern}获取$block_{type}$数据类型 type 实际数据存储位置 ptr_tmp */
13. if msg_{extern}为空 then
14. ptr_tmp=ptr_{head}→type
15. else 解析msg_{extern}→ptr_{msg}
16. ptr_tmp=ptr_{head}→type+ptr_{msg}
17. endif
18. /* 判断 ptr_tmp 是否存在数据 */
19. if ptr_tmp 为空 then
20. $block_{type}$的$data_rcu_{sign}$. read-1
21. return $result_{NULL}$
22. endif
23. if ptr_tmp 存在同步结构rcu_{data} then
24. rcu_{data}的读队列添加数据
25. endif
26. /* 通过数据标识 key 和实际数据存储位置 ptr_tmp 检索出最终读取位置,并暂存在临时变量 ptr_final */
27. ptr_final=ptr_tmp+key(检索)
28. /* key 对应的数据不存在 */
29. if ptr_final 为空 then
30. ptr_tmp.rcu_{data}读队列销毁数据
31. $block_{type}$的$data_rcu_{sign}$. read-1
32. return $result_{NULL}$
33. endif
34. 根据 ptr_final 读取相应的数据
35. ptr 保存读取位置:ptr=ptr_final
36. ptr_tmp.rcu_{data}读队列销毁数据
37. $block_{type}$的$data_rcu_{sign}$. read-1
38. return $result_{成功}$

算法 4 数据结构写方法

输入:待写入数据指示域 ptr,数据类型 type,数据标识 key,结构类型 offset

输出:执行结果 result

1. /* 同步控制块(contro l_{block})存在写请求,即控制块位置记录信息需要周期更新进行环形存储空间轮转 */
2. if 周期重置标志==1 then
3. 等待周期重置完成
4. return $result_{等待}$
5. endif
6. /* 通过 contro l_{block} 和 offset 确定结构类型,即统计控制块或遥测控制块 */
7. contro l_{block}+offset→$block_{type}$
8. /* 操作相应类型读写同步标志,添加数据到相应的写队列 */
9. $block_{type}$的$data_rcu_{sign}$. write+1
10. /* 通过$block_{type}$获取数据存储指示域ptr_{head}和扩展控制信息msg_{extern},保证读写正确性 */
11. 解析$block_{type}$→ptr_{head},msg_{extern}
12. /* 跟据ptr_{head}和msg_{extern}获取$block_{type}$数据类型 type 实际数据存储位置并用临时变量 ptr_tmp 暂存 */
13. if msg_{extern}为空 then
14. ptr_tmp=ptr_{head}→type
15. else 解析msg_{extern}→ptr_{msg}
16. ptr_tmp=ptr_{head}→type+ptr_{msg}
17. endif
18. /* 判断待写入的位置是否分配空间 */
19. if ptr_tmp 为空 then
20. ptr_tmp→添加 type 类型链表节点
21. endif
22. if ptr_tmp 存在同步结构rcu_{data}then
23. rcu_{data}的写队列添加数据
24. endif
25. /* 通过数据标识 key 和 ptr_tmp 获取最终存储位置 ptr_final */
26. ptr_final=ptr_tmp+key(检索)
27. 将 ptr 的数据写入 ptr_final
28. ptr_tmp.rcu_{data}写队列销毁数据
29. $block_{type}$的$data_rcu_{sign}$. write−1
30. return $result_{成功}$

由于内核编程和用户程序各种锁的结构和接口均存在差异,为了保证遥测效率,上述读写同步使用原子变量实现。以下是基于遥测数据图及同步控制块的网络遥测信息生成方法的处理机制及相应的同步机制。

1)网络数据存储:当用户数据包到达时,EAGLE 首先对数据包进行头部解析操作获取数据标识(*key*)并提取各类型数据组织成数据节点(*node*,*type*);再通过算法 4、*key* 值和(*node*,*type*)异步将数据更新到遥测数据图;最后将用户数据包进行调度转发。

2)遥测数据处理:当网络遥测数据包到达时,EAGLE 首先对数据包进行头部解析,若 IP 协议为 253,则直接进行转发操作。若 IP 协议为 254,则获取遥测数据头部信息(*nt_hdr*);然后通过元数据位图(*nt_hdr->bitmap*)执行遥测数据读请求操作并阻塞等待;通过算法 3 读取遥测数据图并聚合遥测数据,同时通过算法 4 将数据写入遥测数据块;最后根据遥测数据块和网络遥测数据包拆分方法执行遥测数据嵌入及转发操作。

3)同步机制:算法 3 和算法 4 的读写同步即为遥测数据环形存储块和遥测数据块的同步机制。对于同步控制块自身数据的同步,先异步发送同步控制块的写请求,再将同步控制块的数据进行更新,然后等待原同步控制块对应算法 3 和算法 4 数据同步操作结束,最后释放原同步控制块。

3.5 融合内核态及用户态特性的网络遥测信息嵌入架构

根据元数据位图提取和生成网络遥测数据后,EAGLE 需要将上述信息嵌入网络遥测数据包中。在实现网络遥测数据嵌入时,EAGLE 需要在网络设备操作系统中编程实现网络遥测机制。考虑到 SDN 网络设备的转发效率和安全问题,需要针对用户态和内核态特性,设计内核态及用户态特性的网络遥测信息嵌入架构。

在 OVS 中,内核态进程能够实现快速的数据包封装、解封装等数据包的快速处理操作、流表项匹配及数据包转发等操作,但内核态默认不执行浮点运算,而用户态则默认进行数据浮点运算操作,因此,EAGLE 将用户数据包封装、解封装、流表匹配、转发处理等操作部署在内核空间执行,将网络设备状态计算、网络数据计算、遥测数据缓存等操作部署在用户空间执行。同时,为了遥测的安全,流表等数据在用户空间进行收集。内核态模块和用户态模块的具体处理方法如算法 5、算法 6 所示。网络遥测信息嵌入架构图如图 4 所示。

算法 5 内核态处理过程

输入:数据包信息 pkt

输出:执行结果 result

1. /* 解析 pkt 网络协议头部,确定数据包类型 type 并获取匹配域 key */
2. 解析 pkt→type,key
3. if type 不是网络遥测协议 then
4. /* 按照所需网络数据类型提取数据包信息组织成多个(node,type)并和 key 异步上传到用户态数据统计模块 */
5. 解析 pkt→(node,type)
6. (node,type),key→数据统计模块
7. 根据 key 匹配流表并执行动作集
8. 发送数据包 pkt
9. return $result_{成功}$
10. endif
11. /* 解析网络遥测数据包 pkt 获取遥测头部数据 nt_hdr */
12. 解析 pkt→nt_hdr
13. /* 网络遥测数据包元数据位图上传到用户态数据提取模块 */
14. nt_hdr.bitmap→数据提取模块
15. 异步等待用户态信号
16. 算法 1 对 pkt 进行拆分处理
17. 嵌入遥测数据到 pkt
18. 根据 key 匹配流表并执行动作集
19. 遥测调度模块转发 ptr
20. return $result_{成功}$

算法 6 用户态处理过程

输入:元数据位图 bitmap,统计数据 data

输出:执行结果 result

1. /* 同步控制块存在写请求,即控制块位置记录信息需要周期更新进行环形存储空间轮转 */
2. if 周期重置标志==1then

3. 等待周期重置完成
4. return result等待
5. endif
6. 交换机数据统计模块异步收集数据
7. 系统数据统计模块异步收集数据
8. /* 数据统计模块将统计数据 data 中的(node,type),key 异步存储到遥测数据图 */
9. data→遥测数据图
10. /* 数据提取模块根据 bitmap 从数据统计模块提取和聚合相应数据得到遥测数据 metadata */
11. bitmap→数据提取模块→metadata
12. /* 数据存储模块将 metadata 写入线性映射缓存空间 */
13. metadata→数据存储模块→缓存
14. 用户态发送处理完成信号给内核态
15. /* 内核态网络遥测数据包拆分模块读取缓存数据,再嵌入到网络遥测数据包 */
16. 缓存→遥测拆分模块→遥测数据包
17. 记录系统运行日志
18. return result成功

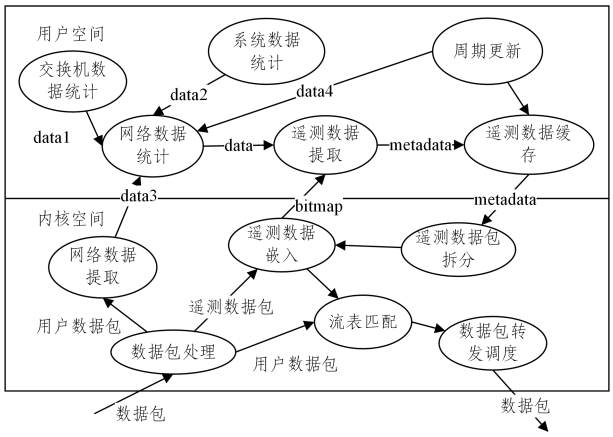


图 4 网络遥测信息嵌入架构图

Fig. 4 Architecture diagram of EAGLE

网络遥测信息嵌入与网络数据的存储以及网络遥测信息的生成有紧密联系,因此本架构和 3.4 节所述方法有较为紧密的耦合性。上述算法和架构将用户数据包和网络遥测数据包分开处理,下文对不同数据包的处理步骤进行阐述。

对于用户数据包,EAGLE 首先使用内核协议栈解析用户数据包,随后提取数据包数据并将上述数据异步写入遥测数据图。然后,遥测数据图在用户空间执行异步更新并根据数量判断是否需要执行动态扩容。最后在内核空间匹配流表并执行所匹配流表项规定的动作集。

对于网络遥测数据包,EAGLE 同样首先使用内核协议栈解析网络遥测数据包,获取遥测头部元数据位图。随后 EAGLE 向用户态模块发送遥测数据请求,然后,内核态模块异步等待用户态同步信号。用户态模块根据算法 3 提取并生成网络遥测数据。用户态成功提取并生成网络遥测信息之后将遥测数据传至内核态,同时发送用户态处理成功信息。内核态模块收到来自用户态的信息后,根据算法 1 判断是否需要执行分片操作。随后将用户态发送的遥测数据嵌入网络遥测数据包中。最后在内核态执行网络遥测数据包的流表项的匹配操作,并根据与之匹配的流表项的动作集处理并转发数据包。

4 实验测试

4.1 实验环境

本文基于 OVS 2.15.1 实现了 EAGLE 方案。本实验中,EAGLE_N 表示在 OVS 中添加了 EAGLE 方案,但网络中不存在网络遥测数据包,即此时并未执行网络遥测过程;EAGLE_E 表示在 OVS 中添加了 EAGLE 方案,同时执行网络遥测;RAW_OVS 表示原生的 OVS 系统。

实验中,所实现的 EAGLE 运行在具有 i7-10400K CPU、8GB 内存以及 240GB 机械硬盘的计算机上。此外,在这台物理机上,本文基于 VirtualBox 虚拟机创建了数据中心 jellyfish 型网络拓扑[21],并生成了遵循正态分布的背景流量[22]。该网络拓扑连通度为 4,由 15 台 OVS 交换机、一台 SDN 控制器、30 台边缘主机和一台遥测服务器组成,分别运行在 8GB 内存、Ubuntu20.04 操作系统的虚拟机中,并使用 Mininet 虚拟网络技术组网。

为保证实验结果的真实有效性,沿着相同环形网络遥测路径进行了 10 次实验,并对结果取平均值以消除实验误差。实验中,上层应用对网络拓扑图通过深度优先算法(Depth First Search,DFS)获取遍历全网的网络遥测路径。为了避免误差,实验中网络遥测路径保持不变。

4.2 时延测试

时延测试分为网络设备单跳处理时延测试和端到端时延测试。为了便于对比原生 OVS 交换机(RAW_OVS)和添加本网络遥测机制后的 OVS 交换机(EAGLE_N,EAGLE_E)的各项实验指标,同时为了解决 RAW_OVS 没有提供获取网络设备单跳处理时延的问题,也为了消除多台交换机之间时间同步等复杂操作,实验中为 RAW_OVS,EAGLE_N 和 EAGLE_E 添加了相同的时间戳记录模块。同时为了保证时间戳数据的有效性,使用未分配的 IP 协议字段 252 作为实验测试协议。

4.2.1 单跳处理时延

本实验通过 Scapy 每隔 5s 发送 5 个 IP 协议为 252 的测试包来获取实验数据,计算 EAGLE_N 和 RAW_OVS 网络设备的平均单跳处理时延。本实验通过 DFS 算法和生成树协议(Spanning Tree Protocol,STP)[27]算法生成不同的网络遥测路径。DFS 算法下网络设备单跳处理时延随时间的变化图如图 5 所示,图中展示了交换机 s7 和 s1 的实验结果。

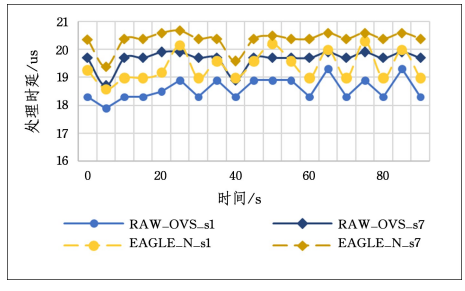


图 5 DFS 网络设备单跳处理时延变化图

Fig. 5 DFS single-hop processing delay

图 5 中，EAGLE_N 的单跳处理时延在分布上和 RAW_OVS 基本一致，均分布在 20us 左右。交换机 s7 相较于交换机 s1 的单跳处理时延偏大，这和 RAW_OVS 的单跳处理时延结果是一致的。EAGLE_N 在平均单跳处理时延上要略大于 RAW_OVS，平均增长约 0.742 us。

STP 算法下网络设备单跳处理时延随时间的变化图如图 6 所示，图中展示了与 DFS 网络遥测路径同一跳数的交换机 s5 和 s11 的实验结果。

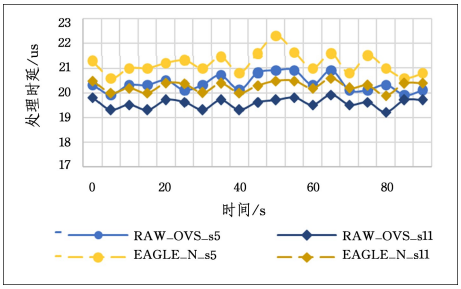


图 6 STP 网络设备单跳处理时延变化图

Fig. 6 STP single-hop processing delay

图 6 中，EAGLE_N 的单跳处理时延在分布上和 RAW_OVS 基本一致，均分布在 20us 左右。交换机 s5 相较于交换机 s11 的单跳处理时延偏大，这和 RAW_OVS 的单跳处理时延结果是一致的。对比图 5 和图 6 可知，图 6 中 RAW_OVS 的单跳处理处理时延比图 5 中 RAW_OVS 的偏大，这是因为网络拓扑中不同交换机流量负载存在差异，这对于 EAGLE_N 和 RAW_OVS 的对比实验没有影响，EAGLE_N 在平均单跳处理时延上要略大于 RAW_OVS，平均增长约0.749 μs，在结果上和 DFS 算法下网络设备单跳处理时延基本一致。

综上，EAGLE_N 单跳处理时延测试上基本和 RAW_OVS 一致。在不同网络遥测路径下 EAGLE_N 相比 RAW_OVS 平均增长的单跳处理时延也基本一致。实验结果表明 EAGLE 对 OVS 性能的影响较小。

4.2.2 端到端时延

端到端时延随时间的变化图如图 7 所示。端到端时延通过 Scapy 每隔 5s 发送 5 个 IP 协议为 252 的测试包，计算发送和接收测试包的时间间隔获取网络遥测路径的端到端时延。其中，EAGLE_E 所示的结果由每隔 5s 发送 5 个具有相同检测号、不同序列号的网络遥测数据包的平均端到端时延计算得出。

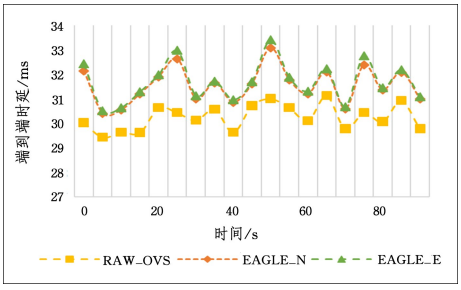


图 7 端到端时延变化图

Fig. 7 End-to-end delay variation graph

由图 7 可知，EAGLE_N 和 EAGLE_E 的端到端时延基本一致。与 RAW_OVS 相比，其端到端时延平均增加约 1.38 ms。具体地，EAGLE_N 和 EAGLE_E 仅在 0 s，25 s，50 s，75 s 的端到端时延相比 RAW_OVS 明显增加，然后马上恢复到和 RAW_OVS 基本一致。这是由于遥测数据图在这些时间点需要执行周期更新操作和遥测数据图的重构，除此之外端到端时延基本一致。由于每次实验都采用相同的网络遥测路径，且背景流量带宽不变，因此 RAW_OVS 的端到端时延理论上是一条接近平行的曲线。考虑实验中控制器和交换机之间的 OpenFlow 消息的影响，RAW_OVS 的实际端到端时延会略有波动，和理论基本一致。

4.3 性能测试

本小节内容包括多粒度数据收集 CPU 占用率测试、系统内存占用测试和链路带宽占用测试。实验过程中，遥测服务器每隔 5s 发送 5 个测试包，网络中仅存在 OpenFlow 消息、测试包和背景流量，并通过 Linux top 命令每隔 5s 获取 RAW_OVS 和 EAGLE_N 的 CPU 占用率、内存占用率、链路带宽占用率。然后遥测服务器根据探测路径每隔 5s 发送 5 个相同检测号、不同序列号的网络遥测数据包，并由此获得 EAGLE_E 的 CPU 占用率、内存占用率和链路带宽占用率。

CPU 占用率对比图、内存占用率对比图和链路带宽平均占用率对比图如图 8—图 10 所示。

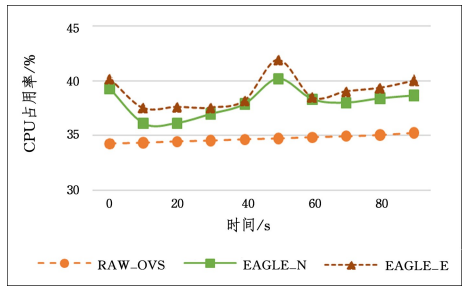


图 8 CPU 占用率变化图

Fig. 8 CPU usage change graph

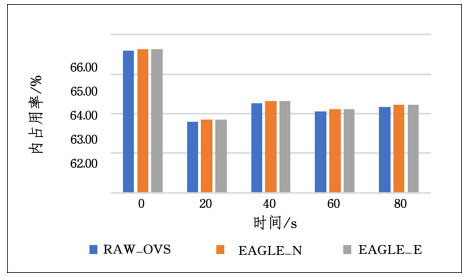


图 9 内存占用率变化图

Fig. 9 Memory usage change graph

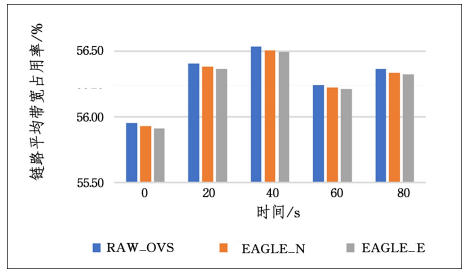


图 10 链路带宽平均占用率变化图

Fig. 10 Link bandwidth average occupancy rate

图 8 表明，EAGLE_E 的 CPU 占用率比 EAGLE_N 平均增加约 2.78%，这是由于网络遥测数据包需要解析头部数据并嵌入数据，上述过程需要消耗更高的 CPU 算力[19]，这是网络遥测无法避免的问题。同时，EAGLE_N 比 RAW_OVS 的 CPU 占用率平均增长 9.46%。

上述 CPU 利用率的差异是由网络遥测数据包解析头部数据、嵌入数据和周期更新时聚合统计数据导致的。其中周期更新时聚合数据最消耗 CPU 算力。在实际的部署中，可以考虑基于网络遥测数据包和系统负载设计更新策略，避免每个周期更新时进行数据聚合。

图 9 表明，EAGLE_N，EAGLE_E 和 RAW_OVS 的内存占用率基本一致。其中，EAGLE_N 和 EAGLE_E 的内存占用率几乎持平，均略高于 RAW_OVS，原因是本网络遥测机制在数据平面编程中设置了内存预分配，这可以避免网络设备频繁申请、释放空间导致性能降低，但会导致添加本网络遥测机制的 OVS 比原生 OVS 多消耗内存。

目前网络遥测机制的研究普遍将网络遥测数据包添加到正常用户数据包的处理队列中，这会使得网络吞吐率因网络遥测数据包的数据提取、嵌入等操作而降低，进而影响网络实际链路带宽。图 10 所示的结果表明，EAGLE_N 的链路带宽平均占用率略低于 RAW_OVS，这是因为 Mininet 虚拟网络设备使用相同的内核模块，EAGLE 从用户数据包中获取网络状态数据的同步操作会影响 SDN 交换机的端口对用户数据包的处理，进而影响链路带宽占用率。EAGLE_E 和 EAGLE_N 的链路带宽占用率基本保持一致，这是因为本网络遥测机制的数据平面处理架构进行了异步分离，用户数据包和网络遥测数据包可以异步执行，缓解了网络遥测数据包对用户数据包吞吐率的影响。

4.4 功能测试

功能测试分为多粒度数据收集测试和网络遥测数据包分片测试。实验中使用标识多种粒度的元数据位图来获取遥测数据，并通过 Wireshark 抓取网络遥测路径中任一网络设备端口获取网络遥测数据包。网络遥测数据包 Wireshark 抓包图如图 11 所示。

对于网络遥测数据包的分片测试，本实验使用的元数据位图足以触发分片操作，因此在网络遥测路径最后一个网络设备端口进行网络遥测数据包抓取。网络遥测数据包分片的 Wireshark 抓包图如图 12 所示。从图 12 中可以看出，已经分片的网络遥测数据包负载单元数据大小为 1 040 bytes，若再添加图 11 中展示的遥测负载数据 528 bytes，则网络遥测数据包总长度将大于 MTU，此时 EAGLE 将进行网络遥测数据包的分片操作。

实验结果表明，EAGLE 可以根据元数据位图实现不同粒度的遥测数据获取，有效地弥补了目前网络遥测机制统计数据量不足、遥测数据粒度较粗的问题。同时，当遥测数据量偏大时，EAGLE 可以有效执行分片操作，这有效地弥补了目前网络遥测机制存在的网络遥测数据量固定、网络遥测数据量有限的问题。

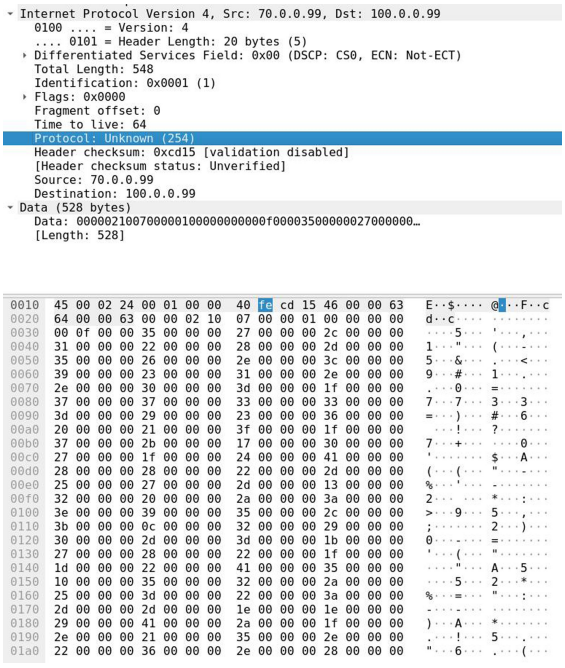


图 11 网络遥测数据包 Wireshark 抓包图

Fig. 11 Wireshark capture diagram of telemetry packet

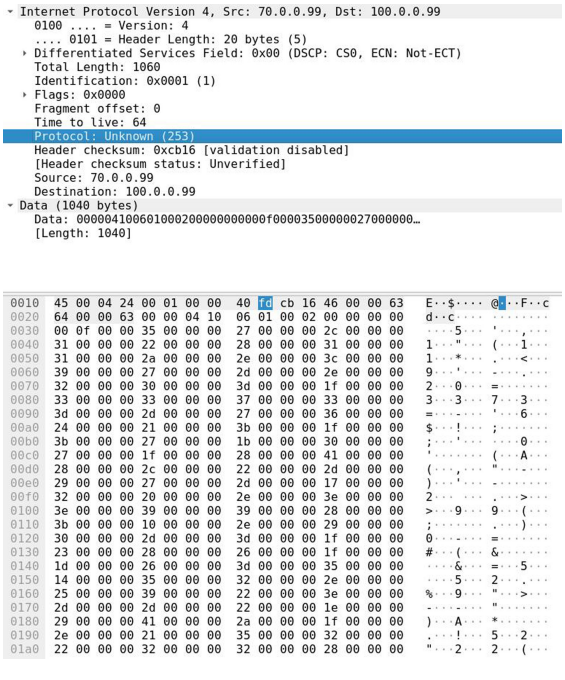


图 12 网络遥测数据包分片的 Wireshark 抓包图

Fig. 12 Wireshark capture diagram of packet fragmentation

结束语 本文针对现有网络遥测技术无法收集多粒度网络数据、无法有效存储大量原始网络数据、无法快速提取及生成网络遥测信息、无法利用内核态及用户态特性设计网络遥测方案等问题，设计了一种融合内核态及用户态的、基于遥测数据图和同步控制块的多粒度、可扩展、覆盖全网的网络遥测机制 EAGLE。EAGLE 设计了一种能够收集多粒度数据、数据平面上灵活可控的网络遥测数据包结构，用于获取上层应用所需的数据；提出了一种基于遥测数据图及同步控制块的网络遥测信息生成方法；还提出了一种融合内核态及用户态特性的网络遥测信息嵌入架构。最后本文在 OVS 2.15.1 中

实现了 EAGLE 方案并进行了测试。测试结果表明,EAGLE 能够收集多粒度数据并快速提取与生成遥测数据,同时仅增加极少量的处理时延及资源占用率。在未来的研究工作中,本方案将通过网络遥测数据包时序来动态调整遥测数据聚合时间,以进一步降低遥测资源占用率;本方案也将通过子网检测方法确定边缘 SDN 交换机实现遥测数据的网络边缘聚合,以此降低网络遥测过程中网络遥测数据包对网络的影响。

参 考 文 献

[1] GULENKO A,WALLSCHLÄGER M,KAO O. A practical implementation of in-band network telemetry in open vswitch [C]// 2018 IEEE 7th International Conference on Cloud Networking(CloudNet). IEEE,2018.

[2] MCKEOWN N,ANDERSON T,BALAKRISHNAN H,et al. OpenFlow:enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review, 2008,38(2):69-74.

[3] ZHANG H,CAI Z,LIU Q,et al. A survey on security-aware measurement in SDN[J/OL]. https://www. hindawi. com/journals/scn/2018/2459154/.

[4] PENG G B,CHEN M,BAI Y. Analysis of SDN Attack and Defense Technology [J]. Information Security Research, 2019,5(4):333.

[5] CAI Z,WANG Z,ZHENG K,et al. A distributed TCAM coprocessor architecture for integrated longest prefix matching,policy filtering,and content filtering[J]. IEEE Transactions on Computers,2011,62(3):417-427.

[6] PHAAL P,PANCHEN S,MCKEE N. InMon corporation's sFlow:A method formonitoring traffic in switched and routed networks[EB/OL]. https://www. rfc-editor. org/info/rfc3176.

[7] QUITTEK J,ZSEBY T,CLAISE B,et al. Requirements for IP flow information export(IPFIX)[EB/OL]. https://www. rfc-editor. org/info/rfc3917.

[8] SOMMER R,FELDMANN A. NetFlow:Information loss or win? [C]// Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment. 2002:173-174.

[9] CLAISE B,JOHNSON A,QUITTEK J. Packet sampling (PSAMP) protocol specifications[EB/OL]. https://www. rfc-editor. org/info/rfc5476.

[10] TAN L,SU W,ZHANG W,et al. In-band network telemetry:A survey[J]. Computer Networks,2021,186:107763.

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

[12] LIU ZZ,BI J,ZHOU Y,et al. Active network telemetry mechanism based on P4 [J]. Journal of Communications, 2018,39(A1):162-169.

[13] RAMANATHAN S,KANZA Y,KRISHNAMURTHY B. SDProber:A software defined prober for SDN[C]// Proceedings of the Symposium on SDN Research. 2018.

[14] ZHOU Y,SUN C,LIU H H,et al. Flow event telemetry on programmable 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. 2020:76-89.

[15] HUANG Q,SUN H,LEE P P C,et al. Omnimon:Re-architecting network telemetry with resource efficiency and full accuracy [C]// Proceedings of the 2020 Annual Conference of the ACM Special Interest Group on Data Communication on the Applications,Technologies,Architectures,and Protocols for Computer Communication. 2020:404-421.

[16] PAN T,LIN X C,ZHANG J,et al. In-band network telemetry system based on high-performance packet processing architecture VPP[J]. Journal of Communications,2021,42(3):75-90.

[17] FEZEU R A K,ZHANG Z L. Anomalous Model-Driven-Telemetry Network-Stream BGP Detection[C]// 2020 IEEE 28th International Conference on Network Protocols(ICNP). IEEE, 2020.

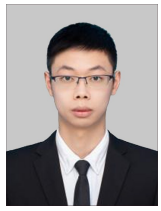
[18] BEN BASAT R,RAMANATHAN S,LI Y,et al. PINT:Probabilistic in-band network telemetry[C]// Proceedings of the 2020 Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures,and Protocols for Computer Communication. 2020:662-680.

[19] NAM S,LIM J,YOO J H,et al. Network anomaly detection based on in-band network telemetry with RNN[C]// 2020 IEEE International Conference on Consumer Electronics-Asia(ICCE-Asia). IEEE,2020.

[20] PFAFF B,PETTIT J,KOPONEN T,et al. The Design and Implementation of Open vSwitch[C]// 12th USENIX Symposium on Networked Systems Design and Implementation(NSDI 15). 2015:117-130.

[21] YUAN X,MAHAPATRA S,NIENABER W,et al. A new routing scheme for Jellyfish and its performance with HPC workloads[C]// Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. 2013.

[22] CUI Y,YAN L,LI S,et al. SD-Anti-DDoS:Fast and efficient DDoS defense in software-defined networks[J]. Journal of Network and Computer Applications,2016,68:65-79.



XIAO Zhaobin, born in 1997, master, is a member of CCF(No. N1797G). His main research interests include SDN, network telemetry, efficient data plane programming, network and information security.

CUI Yunhe, born in 1987, Ph. D, associate professor, is a member of CCF(No. F3600M). His main research interests include edge computing, network security, software-defined networks and data center networks, and network telemetry.

(责任编辑:何杨)