



一种大象流两级识别方法

严军荣, 叶景畅, 潘鹏

(杭州电子科技大学通信工程学院, 浙江 杭州 310018)

摘要: 基于大象流的识别准确度高且开销低, 对于解决 SDN 流量管理过程中控制器单点故障问题具有重要意义。针对现有大象流识别方法识别开销大的问题, 提出一种大象流两级识别方法。该方法在第一阶段提出基于 TCP 发送队列的可疑大象流识别算法, 在第二阶段提出基于流持续时间的真实大象流识别算法; 第一阶段是在端系统中识别可疑大象流, 用于降低第二阶段真实大象流识别过程中 SDN 控制器所需监测的网络流数量。实验分析表明, 在保证大象流识别的高准确度前提下, 大象流两级识别方法较基于采样的大象流识别方法可以降低约 85% 的控制器识别开销。

关键词: 大象流识别; TCP 发送队列; 软件定义网络

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2017076

A two-level method for elephant flow identification

YAN Junrong, YE Jingchang, PAN Peng

School of Communication Engineering, Hangzhou Dianzi University, Hangzhou 310018, China

Abstract: The high accuracy and low overhead of elephant flows identification have a great meaning on solving the controller's single point of failure problem in SDN traffic management. Aiming at the problem of high overhead of the existing elephant flows identification method, a two-level method for elephant flows identification was proposed which included a suspicious elephant detection algorithm based on TCP write-queue in the first stage and areal elephant detection algorithm based on flow duration in the second stage. During the first stage, the suspicious elephant flows were identified in the end systems to reduce the amount of flows monitored by the SDN controller at the second stage. Analysis and simulation prove that, under the premise of ensuring the accuracy of elephant flow identification, the two-level method for elephant flows identification reduces about 85% overhead of identification compared with sampling identification method.

Key words: elephant flow identification, TCP sending queue, SDN

1 引言

软件定义网络 (software defined networking, SDN) 是一种新型网络架构^[1,2], 其通过剥离网络设备的控制平面与数

据转发平面并将控制平面抽象为集中式控制器, 为网络提供了逻辑统一的转发控制平台, 基于 SDN 特有的全网视图可快速实现流量调度策略的制定以及交换机流表的配置。流量管理过程中, 作为网络大脑的 SDN 控制器需接

收稿日期: 2016-11-28; 修回日期: 2017-02-21

基金项目: 浙江省公益技术应用研究计划基金资助项目 (No.2016C31G2041123); 第 56 批中国博士后科学基金面上资助项目 (No.2014M561693); 国家自然科学基金青年科学基金资助项目 (No.61401130)

Foundation Items: Zhejiang Provincial Public Technology Application Research Program (No.2016C31G2041123), The 56th China Postdoctoral Science Foundation (No.2014M561693), The National Natural Science Foundation of Youth Science Foundation (No.61401130)

收、处理海量数据以保证网络的正常运行,这对其性能提出巨大挑战。为SDN控制器“减负”是避免控制器单点故障的有效方法,通过采用“抓大放小”的流量管理方式^[3],即精细管控大象流(传输数据量大、持续时间长)、粗放管控老鼠流(传输数据量小、持续时间短)^[4],可有效减少控制器在大象流识别过程中的处理负载。因此,对SDN中的高准确、低开销的大象流识别方法进行研究是很有必要的。

针对现有大象流识别算法识别开销大的问题,本文考虑在端系统中预先识别并筛选出可疑大象流,以减少SDN控制器监测网络中大象流的开销,从而提出一种大象流两级识别方法。首先通过分析主机端数据流的尺寸特征实现第一阶段可疑大象流的识别,然后通过SDN控制器监测网络中可疑大象流的持续时间特征实现第二阶段真实大象流的识别,最后通过Mininet实验仿真验证大象流两级识别方法性能。该方法包括在第一阶段提出的基于TCP发送队列的可疑大象流识别算法和在第二阶段提出的基于流持续时间的真实大象流识别算法。其中,第一阶段是在端系统中识别可疑大象流,用于降低第二阶段真实大象流识别过程中SDN控制器所需监测的网络流数量。

2 相关工作

传统大象流识别方法主要包括基于采样的大象流识别方法、基于计数的大象流识别方法以及基于LRU(least recently used)的大象流识别方法^[5]3种。基于采样的大象流识别算法通过提取、分析网络流样本数据分组的特征推导出网络整体流量的特征。NetFlow技术和sFlow技术是目前常用的两种流采样方法,Bi C H等人^[6]提出一种适用于数据中心的大象流两级识别系统,通过sFlow筛选出可疑流并通过控制器从可疑流中分析、识别大象流;Mori等人^[7]提出一种基于Bayes原理计算大象流阈值并利用分组采样方法采样、识别大象流的方法,具有较高的识别准确率但其识别开销较大。

Robert等人^[8]最早提出基于计数的大象流识别方法,基于“匹配—计数”的原理,通过统计监测设备中流数据分组的匹配结果以实现网络大象流的识别。Bai L等人^[9]提出一种基于超时计数的大象流识别方法,利用多级布隆过滤器(bloom filter, BF)对数据进行计数,以实现老鼠流的过滤和大象流的识别,流识别可靠性较高但BF实现较复杂。赵小欢等人^[10]提出一种基于CBF-SS(counting bloom filter-space

saving)的大象流识别方法,利用CBF剔除老鼠流并依据SS算法识别大象流,时间复杂度较低但识别实时性较差。

Smitha等人^[11]提出了基于LRU的大象流识别算法,通过在一个大小固定的流记录缓存置换流记录,实现了高频大象流的识别。参考文献[12,13]通过在LRU算法前增加过滤步骤以剔除网络老鼠流,提高了LRU算法的大象流识别准确性。谢冬青等人^[14]提出一种基于LRU和SCBF(space code bloom filter)的大象流识别方法,将新入网数据流记录于SCBF中并将满足条件的流记录提升至LRU顶部,通过LRU与SCBF间的淘汰循环机制实现大象流的快速汇聚,识别准确性较高但其较复杂且实时性欠缺。

伴随SDN技术的发展,基于SDN的大象流识别技术应运而生,利用SDN特有的全网视图及强大的流量监测能力可快速实现大象流的识别。在SDN中的流量信息采集方法主要包括主动式和被动式^[15,16]两种,其中被动式又包括基于事件触发和基于时序触发两种。Lin C Y等人^[17]提出一种基于区域锁定的大象流识别方法,首先依据分层统计原理追溯并锁定大象流产生区域,然后利用SDN控制器针主动查询该区域内设备记录的流信息以实现大象流的识别,该方法通过锁定大象流产生区域有效降低控制器的主动查询开销但其实现较复杂。Curtis等人^[18]提出一种基于套接字缓存监测的大象流识别方法,其识别实时性较高但未对端系统识别结果进行修正。Chao等人^[19]提出一种基于霍夫丁决策树的大象流识别算法,控制器首先依据数据流持续时间筛选潜在大象流,然后分析潜在大象流数据分组的大小特征识别大象流,有效提高了大象流识别的准确性,但该方法的控制器识别开销很大。

3 研究内容

本文提出的大象流两级识别方法如图1所示。第一阶段基于端系统的TCP发送队列监测实现可疑大象流识别,认定发送队列中数据量超过可疑流阈值的流为可疑大象流并对其标记;第二阶段基于SDN控制器的流量监测功能实现真实大象流的识别,认定网络中持续时间超过特定值的可疑大象流为真实大象流。

3.1 第一阶段识别

该阶段通过监测端系统TCP发送队列中数据流的尺寸特征实现可疑大象流的识别,为此提出一种基于TCP发送队列的可疑大象流识别(suspicious elephant detection based on write queue, SED-WQ)算法。在数据发送初始阶段

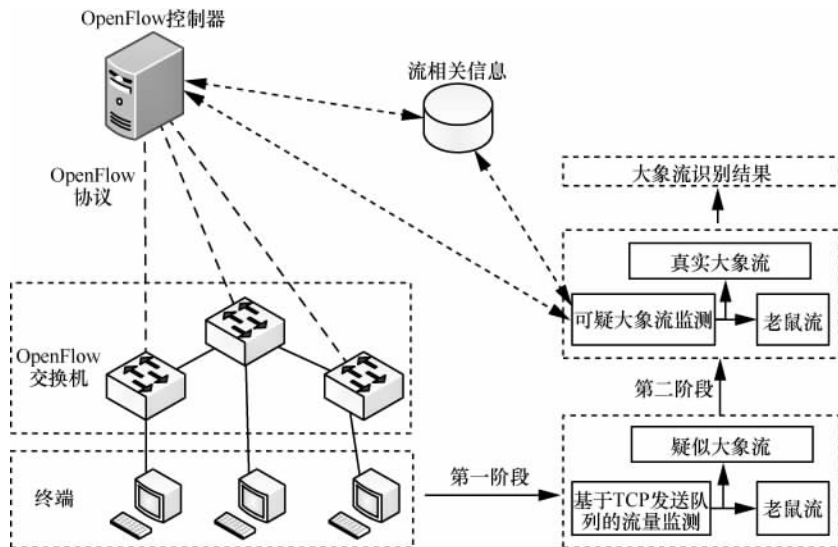


图 1 大象流两级识别算法示意

(数据仍在主机缓存中), SED-WQ 算法通过传输控制块 (transport control block, TCB)^[20] 启用 TCP_CORK 选项暂时阻塞 TCP 发送队列, T_0 时间 (T_0 值可设置) 后遍历 TCP 发送队列并统计其保存的总数据量, 据此判断该流是否为可疑大象流, 标记并发送识别出的可疑大象流。SED-WQ 算法流程如图 2 所示, 具体步骤如下。

步骤 1 本地主机与对端主机通过 3 次握手建立 TCP 连接并读取 *elephant_detection* 值 (该值初始值为 0, 表示未处理该流), 如其值等于 0, 则调用 *sys_setsockopt()* 函数开启 TCB 的 TCP_CORK 选项, 然后将 *elephant_detection* 值加 1, 进入步骤 2; 如大于 0, 则表示该流类型已确定, 直接进入步骤 8。

步骤 2 定时调用 *skb_queue_empty()* 函数以监测发送队列是否为空, 如函数返回 true, 表明发送队列为空, 继续等待数据存入; 如返回 false, 表明应用程序开始产生数据并存入套接字缓存 (SKB)^[20] 中, 进入步骤 3。

步骤 3 启动计时器 1 记录时间 t_1 , 数据持续存入发送队列 SKB 中直到 $t_1 > T_0$, 终止计时, 进入步骤 5。

步骤 4 读取发送队列链表头 *sk_buff_head* 保存的 *qlen* 值, 从头开始遍历发送队列中的 *qlen* 个 SKB, 依次读取第 i 个 SKB 保存的 *len* 值并累加得到发送队列中保存的总数据量 *queue_data*, 其中 $0 < i \leq qlen$ 。

步骤 5 将 *queue_data* 与可疑流阈值 M 进行比较, 如果 $queue_data \geq M$, 则认定该流为可疑大象流, 进入步骤 6; 如果 $queue_data < M$, 则认定该流为老鼠流, 进入步骤 7。

步骤 6 通过 *ipv4_change_dsfield()* 函数设置该流

IP 分组头 DSCP 位为可疑流标识符, 本文定义可疑流标识符为 48 (0x30), 与之对应的 ToS 值为 192 (0xC0)。

步骤 7 调用 *sys_setsockopt()* 函数关闭 TCB 的 TCP_CORK 选项, 解除 TCP 发送队列的阻塞状态。

步骤 8 正常发送数据。

其中, 步骤 1 中提到的 *setsockopt()* 函数用于获取或设置相关联的套接口选项 (通过设置套接口选项可实现数据处理过程相关参数或行为的配置), 函数参数包括 *socket*、*level*、*option_name*、*option_value* 以及 *option_len* 这 5 个, *socket* 表示数据流的套接字描述符, *level* 指表示调用的具体函数类型 (包括 SOL_Socket 类、SOL_TCP、Ipproto_IP 类等), *option_name* 表示待配置套接口选项名称, *option_value* 表示下发至内核的数据, *option_len* 表示下发的数据长度。

3.2 第二阶段识别

该阶段通过对网络中可疑大象流持续时间特征的分析实现真实大象流的识别, 提出一种基于流持续时间的大象流识别 (real elephant detection based on duration time, RED-DT) 算法。RED-DT 算法基于流持续时间特征在第一阶段识别的基础上对网络中的可疑大象流进行二次筛选, 认定持续时间大于 T_1 (T_1 可设置) 的可疑大象流为真实大象流, 快速剔除了第一阶段的误识别大象流, 保证了大象流识别算法的准确性和实时性。该阶段可通过控制器编写识别应用实现, 利用 SDN 主动向交换机请求可疑大象流的统计信息如命中流表项的累计字节数、命中该流表项的累计分组数, 如两次请求所得统计量发生改变则可认定该可疑大象流仍存在, 可认定其为真实大象流。RED-DT 算法流

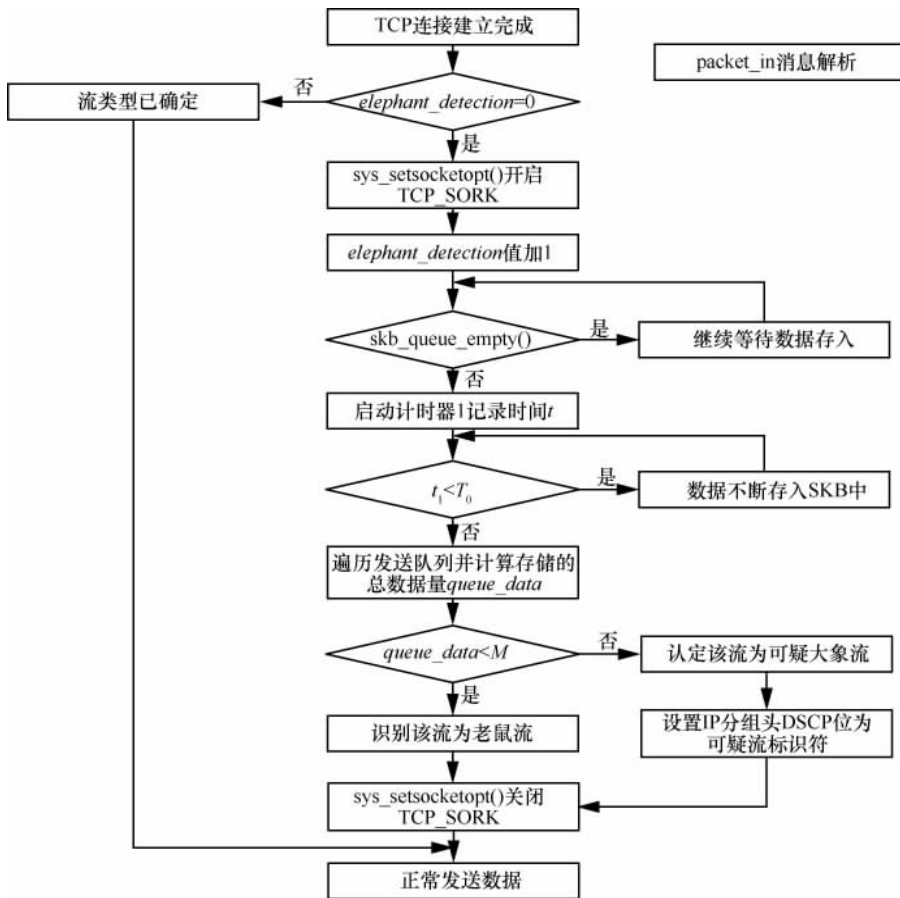


图2 第一阶段 SED-WQ 算法流程

程如图3所示,具体步骤如下。

步骤1 通过 `packet_in_handler()` 函数解析 `packet_in` 消息^[2],然后利用 `get_protocol()` 方法获取该流数据 IP 数据分组头 `TOS` 值。

步骤2 将所得 `TOS` 值与 192 进行比较(可疑大象流标识符为 192),如 $TOS=192$,则认定该流为第一阶段识别出的可疑流并将该流相关信息包括 `datapath`、`table_id`、`match` 以及 `status`(默认值为 `suspicious`)等记录在大象流列表中,进入步骤3;如 $TOS \neq 192$,则认定该流为老鼠流。

步骤3 启动计时器2用于记录时间 t ,在 $t=T_1$ 与 $t=T_1+T$ 这两个时刻,Ryu 控制器通过 `ofp_flow_stats_request` 流请求消息向交换机查询可疑流相关信息。

步骤4 Ryu 解析交换机返回的两个 `ofp_flow_stats_reply` 消息,获取 T_1 与 T_1+T 时刻计数器统计的累积命中字节数并记为 b_1 、 b_2 。如 $b_1=b_2$,则认定该可疑大象流为老鼠流,将其从大象流列表中删除;如 $b_1 \neq b_2$,则认定该可疑大象流为真实大象流,修改大象流列表中该流 `status` 属性为 `real`,表明该流为大象流。其中大象流列表见表1。

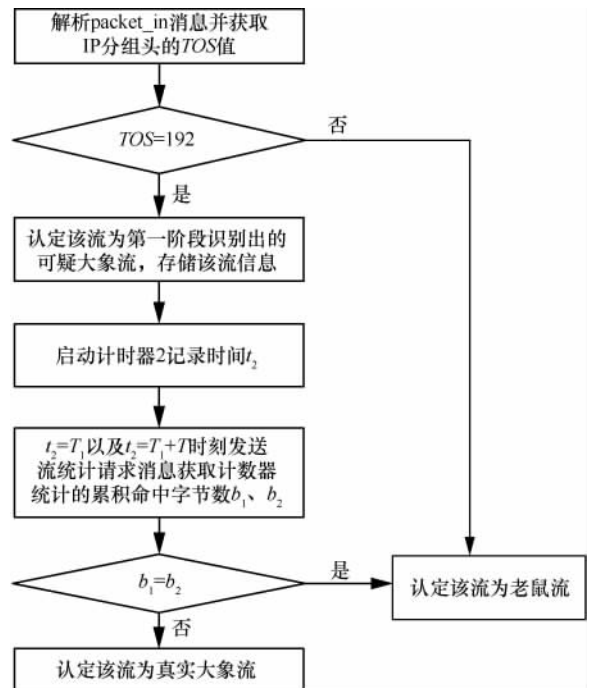


图3 第二阶段识别 RED-DT 算法流程



表 1 大象流列表

交换机编号	流表号	五元组	状态
*	0	源/目的 IP 地址;源/目的端口;四层协议	suspicious/real

3.3 性能分析

首先对大象流两级识别算法两个阶段的识别过程进行分析,然后通过仿真将大象流两级识别算法与基于采样的大象流识别算法进行对比和分析。

3.3.1 第一阶段识别验证

第一阶段的可疑大象流识别利用 SED-WQ 算法实现了可疑大象流的识别与标记。为简化实验,利用文件传输协议(file transfer protocol,FTP)服务发送数据并使用 wireshark 抓取网卡上的数据分组,分析分组头 DSCP 位验证第一阶段可疑大象流识别结果。FTP 是基于 TCP 的应用层协议,数据发送较快且传输数据量可设置(通过改变传输文件的大小实现),分析该阶段识别结果与传输文件的大小可快速实现 SED-WQ 算法识别结果的分析与验证。

数据中心网络中 90% 左右的数据流其承载的数据量不超过 1 MB^[22],因此本文设置可疑大象流阈值为 1 MB。为模拟实际网络环境中的流类型比例,本次实验生成 1~100 KB 大小的文件 75 个,100 KB~1 MB 大小的文件 16 个,1 MB 以上文件 9 个。在数据发送前,需对 Linux 主机发送缓存区的默认值和最大值进行修改,利用 `sysctl-wnet.core.wmem_default=1048576` 指令修改主机默认发送缓存大小为 1 MB,利用 `sysctl-wnet.core.wmem_max=10485760` 指令修改主机最大发送缓存大小为 10 MB。

发送上述文件并抓取数据分组,图 4 和图 5 是

wireshark 抓取的两个 FTP 数据分组,分析其 DSCP 值可知,图 4 被识别为可疑大象流,对照传输记录发现本次传输文件大小为 115 MB,识别结果符合实际;图 5 被识别为老鼠流,对照传输记录发现本次传输文件大小为 3.5 KB,识别结果符合实际。

3.3.2 第二阶段识别验证

第二阶段的真实大象流识别利用 RED-DT 算法实现了对第一阶段误检可疑大象流的快速剔除,该阶段验证的网络拓扑如图 6 所示。

Host1 发出的 FTP 流通过 OVS 时会被发送至控制器, Ryu 读取数据分组 TOS 位识别出第一阶段的可疑大象流并将其信息记录在大象流列表,依据可疑大象流的持续时间特征实现真实大象流的识别,部分代码如下。

```
@set_ev_cls ( ofp_event. EventOFPPacketIn, MAIN_
DISPATCHER)

def_packet_in_handler(self, ev):
    msg = ev.msg
    datapath = msg.datapath
    dpid = datapath.id, /* 获得交换机设备号 */
    parser = datapath.pfproto_parser
    table_id = msg, /* 获得流表号 */
    in_port = msg.match['in_port']
    /* 解析 packet_in 消息携带的数据分组或报头 */
    pkt = packet.Packet(msg.data)
    ip_tos = pkt.get_protocol(ipv4.ipv4.tos), /* 获得 TOS 值 */
    ip_src = pkt.get_protocol(ipv4.ipv4.src), /* 获得源 IP
地址 */
```

```
▶Frame 264: 1442 bytes on wire (11536 bits), 1442 bytes captured (11536 bits) on interface 0
▶Ethernet II, Src: Vmware_98:03:42 (00:0c:29:98:03:42), Dst: Vmware_f3:06:aa (00:50:56:f3:06:aa)
▼Internet Protocol Version 4, Src: 192.168.170.130 (192.168.170.130), Dst: 192.160.170.131 (192.160.170.131)
  Version: 4
  Header Length: 20 bytes
  ▼Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    1100 00.. = Differentiated Services Codepoint: Class Selector 6 (0x30)
    ....00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
  Total Length: 1428
  Identification: 0x8543 (34115)
```

图 4 抓取的疑似大象流数据分组

```
▶Frame 495: 522 bytes on wire (4176 bits), 522 bytes captured (4176 bits) on interface 0
▶Ethernet II, Src: Vmware_98:03:42 (00:0c:29:98:03:42), Dst: Vmware_f3:06:aa (00:50:56:f3:06:aa)
▼Internet Protocol Version 4, Src: 192.168.170.130 (192.168.170.130), Dst: 192.160.170.131 (192.160.170.131)
  Version: 4
  Header Length: 20 bytes
  ▼Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    0000 00.. = Differentiated Services Codepoint: Default (0x00)
    ....00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
  Total Length: 508
  Identification: 0x4514 (17684)
```

图 5 抓取的老鼠流数据分组

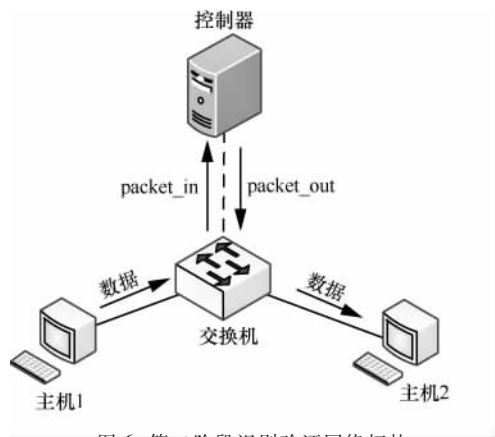


图 6 第二阶段识别验证网络拓扑

ip_dst = pkt.get_protocol(ipv4.ipv4.dst),/* 获得目的 IP 地址 */

tcp_src_port = pkt.get_protocol(tcp.tcp.src_port),/* 获得源端口 */

tcp_dst_port = pkt.get_protocol(tcp.tcp.dst_port),/* 获得目的端口 */

利用 Mininet 进行仿真并查看 Ryu 维护的大象流列表,其内容见表 2。

分析表 2 中数据可以发现,第二阶段识别通过 RED-DT 算法能够实现真实大象流的识别,两次识别过程剔除了不满足条件的大象流,并将真实大象流状态标记为 real。

3.3.3 大象流两级识别算法性能分析

首先分析大象流两级识别算法的识别准确性,然后在 大象流识别开销方面与基于采样的大象流识别算法进行 对比和分析。

(1) 两级识别算法的识别准确性分析

改变 M 值进行重复试验,统计分析网络大象流的比例、大象流的识别准确性以及大象流误识率,结果如图 7 所示。由图 7 可知,大象流两级识别算法的识别结果与测试数据文件基本吻合,具有较高的识别准确性。

(2) 控制器识别开销分析

在保证大象流两级识别方法识别准确性的同时下,从 算法识别开销角度将大象流两级识别算法与基于采样的 大象流识别方法进行比较分析,假定网络环境见表 3。

表 2 大象流列表记录信息

交换机编号	流表号	五元组	状态
5e3e089e01616580	0	10.0.0.1,1120,10.0.0.2,21,tcp	real
5e3e089e01616580	0	10.0.0.1,1123,10.0.0.2,21,tcp	real
5e3e089e01616580	0	10.0.0.1,1124,10.0.0.2,21,tcp	suspicious

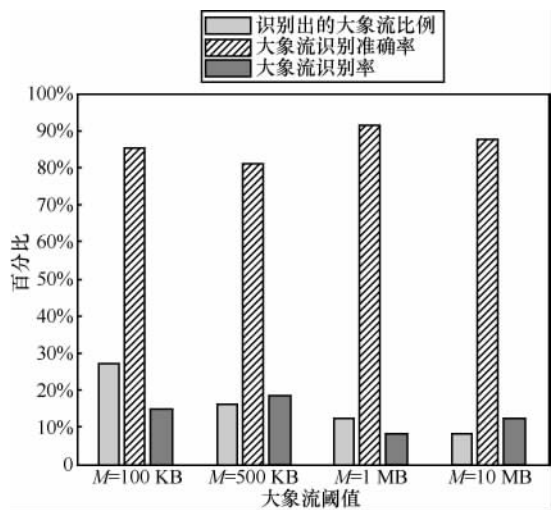


图 7 不同阈值下大象流的识别性能

表 3 模拟环境配置

参数	功能	数值
$H_{sum}/台$	主机总数	128
$H_p/台$	每台交换机连接的主机数	8
$S/台$	交换机总数	16
$Flow/条$	每个终端每秒平均产生的新流数量	10
P_s	采样频率	1/50
$D_s/byte$	每次采样的数据量(采集分组头)	54
T/s	流表项存在时间	60

① 基于采样的大象流识别

该方法利用分组采样的方式采样数据流并统计,依据 样本特征推导采样流的特征以此实现网络大象流的识别。 在该识别算法中,识别开销可定义为所有交换机上每秒采 样的数据总和,该值可由式(1)计算得到。

$$sum = \frac{H_{sum} \cdot D_s}{P_s} \quad (1)$$

采样方法仅需发送数据分组头至控制器即可,故 D_s 取值为 54 byte(MAC 分组头、IP 分组头及 TCP 分组头之和),则 可计算出基于采样的大象流识别方法理论上每秒需处理 337.5 KB 的数据。

② 大象流两级识别方法

大象流两级识别方法的控制器识别开销为第二阶段



的可疑大象流监测开销。针对特定可疑大象流控制器需处理 1 条 packet-in 消息、2 条流统计请求消息以及 2 条应答消息,则该方法的识别开销可由式(2)计算得到。

$$sum = E \cdot H_{sum} \cdot Flow \cdot (2B_{request} + 2B_{reply} + B_{packet-in}) \quad (2)$$

其中, E 为网络中可疑大象流的比例,可疑大象流阈值为 1 MB 时 E 值为 12%, packet-in 消息 $B_{packet-in}$ 为 128 byte, ofp_flow_stats_request 请求消息 $B_{request}$ 为 48 byte(分组头 8 byte, 消息主体 40 byte), ofp_flow_stats_reply 消息 B_{reply} 为 64 byte(分组头 8 byte, 消息主体 56 byte),则可计算大象流两级识别方法控制器每秒需处理 52.8 KB 的数据。

为更直观地比较上述两种大象流识别方法的开销,现将式(1)与式(2)的右边相除得到两种识别算法开销比值 r 的式(3)。其中, E 与可疑大象流阈值有关。由此可以得出, r 值与大象流阈值的关系,如图 8 所示。

$$r = \frac{E \cdot P_s \cdot Flow \cdot (2B_{request} + 2B_{reply} + B_{packet-in})}{D_s} \quad (3)$$

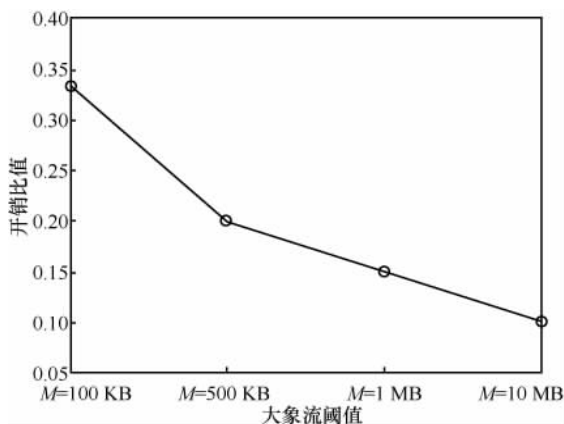


图 8 大象流两级识别算与采样识别算法开销比较

从图 8 可看出,可疑大象流阈值 M 越大,大象流两级识别算法相较于采样识别算法开销越小,表明大象流两级识别算法通过两阶段的大象流可有效识别 SDN 控制器的识别开销,保证系统运行的可靠、稳定,但在 M 值选取过程中也考虑端系统内存利用率情况。

综合来说,本文提出的 SDN 中的一种大象流两级识别方法,在保证大象流识别准确性的同时可有效降低 SDN 控制器的识别开销,在避免控制器单点故障问题上较基于采样的大象流识别方法具有明显优势。当可疑大象流阈值 $M=1$ MB 时,大象流两级识别算法的大象流识别准确性为 92%且其控制器识别开销较基于采样的大象流识别算法降低了约 85%。

4 结束语

本文针对现有基于 SDN 的大象流识别方法存在的控制器开销较大的问题,提出一种大象流两级识别方法,在第一阶段采用基于 TCP 发送队列的识别算法检测可疑大象流,在第二阶段采用基于流持续时间的识别算法检测真实大象流。端系统中的第一阶段可疑大象流识别剔除了大部分老鼠流,可有效降低第二阶段 SDN 控制器所需监测的网络流数量,以实现控制器识别开销的“减负”,仿真结果表明本文提出的大象流两级识别算法在保证较高大象流识别准确性的同时,可有效降低 SDN 控制器的识别开销。

参考文献:

- [1] ZUO Q Y, CHEN M, ZHAO G S, et al. OpenFlow-based SDN technologies [J]. Journal of Software (Chinese), 2013 (5): 1078-1097.
- [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] CRISTIAN E, VARGHESE G. New directions in traffic measurement and accounting: focusing on the elephants, ignoring the mice [J]. ACM Transactions on Computer Systems (TOCS), 2003, 21(3): 270-313.
- [4] BENSON T, AKELLA A, MALTZ D A. Network traffic characteristics of data centers in the wild [C]// The 10th ACM SIGCOMM Conference on Internet Measurement, November 1-30, 2010, Melbourne, Australia. New York: ACM Press, 2010: 267-280.
- [5] XIA J B, REN G M. Survey on elephant flow identifying methods [J]. Control and Decision, 2013, 28(6): 801-807.
- [6] BI C H, LUO X, YE T, et al. On precision and scalability of elephant flow detection in data center with SDN [C] // 2013 IEEE Globecom Workshops (GC Wkshps), December 9-13, 2013, Atlanta, Georgia, USA. New Jersey: IEEE Press, 2013: 1227-1232.
- [7] MORI T, UCHIDA M, GOTO S. Identifying elephant flows through periodically sampled packets [C] // The 4th ACM SIGCOMM Conference on Internet Measurement, Oct 25-27, 2004, Taormina, Sicily, Italy. New York: ACM Press, 2004: 115-120.
- [8] ROBERT B, STROTHER R M. MJRTY—a fast majority vote algorithm [M]. Amsterdam: Springer Netherlands, 1991.
- [9] BAI L, LIU W J. Algorithm based on multiple filters for elephant flows identification [C] // 2011 International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE), December 16-18, 2011, Changchun, China. New Jersey: IEEE Press, 2011: 1084-1087.

- [10] ZHAO X H, LI M H. Large flow identification based on counting bloom filter and space saving [J]. Journal of University of Chinese Academy of Sciences, 2015, 32(3): 391-397.
- [11] SMITHA, KIM I, NARASIMHAREDDY A L. Identifying long-term high-bandwidth flows at a router [C] // The 8th International Conference on High Performance Computing, December 17-20, 2001, Hyderabad, India. Heidelberg: Springer, 2001: 361-371.
- [12] LIU X L, LIU Y, WANG C L. An identification algorithm of network elephant flow based on FEFS and CBF [J]. Computer Engineering, 2015, 41(9): 68-73.
- [13] BAI L, TIAN L Q, CHEN C. Elephant flow detection algorithm for high speed networks based on flow sampling and LRU [J]. Computer Application and Software, 2016, 33(4): 111-115.
- [14] XIE D Q, ZHOU Z H, LUO J W. An algorithm based on LRU and SCBF for elephant flows identification and its application in DDoS defense[J]. Journal of Computer Research and Development, 2011, 48(8): 1517-1523.
- [15] TOOTOONCHIAN A, GHOBADI M, GANJALI Y. OpenTM: traffic matrix estimator for OpenFlow networks [C] // 2010 International Conference on Passive and Active Network Measurement, April 7-9, 2010, Zurich, Switzerland. Heidelberg: Springer, 2010: 201-210.
- [16] BENSON T, ANAND A, AKELLA A, et al. MicroTE: fine grained traffic engineering for data centers[C] // The Seventh Conference on Emerging Networking Experiments and Technologies, December 6-9, Tokyo, Japan. New York: ACM Press, 2011: 8.
- [17] LIN C Y, CHIEN C, CHANG J W, et al. Elephant flow detection in datacenters using openflow-based hierarchical statistics pulling [C] // 2014 IEEE Global Communications Conference, December 8-12, 2014, Austin, Texas, USA. New Jersey: IEEE Press, 2014: 2264-2269.
- [18] CURTIS A R, KIM W, YALAGANDULA P. Mahout: low-overhead datacenter traffic management using end-host-based elephant detection[C] // 2011 IEEE INFOCOM, April 10-15, Shanghai, China. New Jersey: IEEE Press, 2011: 1629-1637.
- [19] CHAO S C, LIN K C J, CHEN M S. Flow classification for software-defined data centers using stream mining [J]. IEEE Transactions on Services Computing. doi:10.1109/TSC.2016.2597846.
- [20] FAN D D, MO L. Analysis of Linux kernel [M]. Beijing: China Machine Press, 2010.
- [21] OpenFlowSwitch Specification. Open networking foundation, version 1.3.0(wire protocol 0x04)[S]. 2012.
- [22] KANDULA S, SENGUPTA S, GREENBERG A, et al. The nature of data center traffic: measurements & analysis[C]//The 9th ACM SIGCOMM Conference on Internet Measurement Conference, November 4-6, 2009, Chicago, Illinois, USA. New York: ACM Press, 2009: 202-208.

[作者简介]



严军荣(1974-),男,博士,杭州电子科技大学讲师,主要研究方向为无线通信与软件定义网络。



叶景畅(1992-),男,杭州电子科技大学硕士生,主要研究方向为软件定义网络。

潘鹏(1983-),男,博士,杭州电子科技大学副教授,主要研究方向为多用户检测技术、协作通信理论与技术。