

协议无感知转发环境下的转发规则冲突检测

吕 威, 王 雷

(中国科学技术大学 自动化系, 安徽 合肥 230027)

摘 要: 协议无感知转发(POF)是对基于 OpenFlow 协议的 SDN 架构的一种扩展性改进, POF 交换机无需了解转发协议即可完成转发操作, 控制器因此获得了最大程度的可编程能力, 可以更加灵活地控制网络, 但这同时也带来了新的转发规则定义冲突问题, 以往的 OpenFlow 控制器的冲突检测方法不再适用. 在对 POF 中规则冲突问题进行分析的基础上, 提出一种以流为单位的基于交换机流表匹配的转发规则冲突检测方案, 并在 POF 控制器中实现. 实验结果证明此规则冲突检测方案不仅实现简单而且快速准确.

关键词: 软件定义网络; 协议无感知转发; 规则冲突检测; 流匹配

中图分类号: TP31

文献标识码: A

文章编号: 1000-7180(2015)11-0078-04

Rule Conflict Detection in Protocol-Oblivious Forwarding

LV Wei, WANG Lei

(Department of Automation, University of Science and Technology of China, Hefei 230027, China)

Abstract: Protocol-Oblivious Forwarding(POF) is an extension for OpenFlow based SDN architecture. POF switch can handle flows without knowing the forwarding protocols, which provides controller the highest programmability to control the network more flexibly. Because of POF's new mechanism, traditional approach for rule conflict detection in other OpenFlow controller is no more applicable. This paper gives analysis of rule conflict problem in POF and introduce a flow level rule conflict detection method, which based on flow table matching, and implements it in current POF controller. The evaluation shows it fast and accurate.

Key words: software-defined networking; POF; rule conflict detection; flow matching

1 引言

当今互联网已成为人们生活的重要组成部分, 各种丰富多彩的网络应用丰富了这个庞大的虚拟世界. 然而, 人们对于网络的控制能力仍然十分有限. 底层网络功能愈加复杂, 但网络设备的静态性却严重限制了网络操作效率. 如何通过简单的接口控制庞大的网络一直是工程师及研究员们的探索方向. SDN^[1]正是一个有效的解决方案. SDN 通过将网络的控制与转发相分离, 实现了对底层设备的抽象, 使得网络开发人员能够通过控制器对网络进行控制, 极大地提高了网络的可编程性. 作为 SDN 最成功的实现协议, OpenFlow^[2] 以及以 NOX^[3]、Flood-

light^[4] 代表的 OpenFlow 控制器的相继诞生与发展, 正推动着 SDN 技术的不断进步. 协议无感知转发(POF)^[5-6] 是对基于 OpenFlow 的 SDN 架构的一个扩展性改进. POF 在 OpenFlow 的基础上, 对转发层使用的协议及交换机的动作进行进一步抽象, 交换机能通过简单的指令集识别和操作流, 而无需预先了解转发协议即可完成转发操作. 这使得 POF 网络在引入新的协议时, 只需通过控制器定义新协议, 不用对网络设备进行更新, 节省了网络升级的成本, 进一步提高了网络编程的灵活性. 尤其是对于如信息中心网络^[7] 等目前各种脱离 IP 的未来网络架构的探索, POF 具有着天然的适应性. 由于 SDN 允许上层的不同应用通过控制器对网络进行控制, 如

收稿日期: 2014-12-14; 修回日期: 2015-01-27

基金项目: 中科院先导专项课题(XDA06011202)

何避免不同应用生成的转发规则之间发生冲突成为一个重要问题. 基于 OpenFlow 的控制器对此进行了一些探索, 如 FortNOX^[8] 使用 Alias Set Rule Reduction 算法进行冲突判断. 然而 POF 不使用固定的传输协议, 没有源 IP 等固定的数据域, FortNOX 的方法并不适用. 当前的 POF 控制器亦尚无相关的冲突检测功能. 本文对此问题进行探讨.

2 POF

基于 OpenFlow 的 SDN 可以通过对数据包中不同的数据域进行匹配以识别不同的数据流, 进而应用相应的操作对这些数据流进行控制. 然而在识别流时只能依照已有的协议进行数据域匹配, 若要支持新的协议则需要对网络设备进行升级, 更新其程序代码.

POF 对网络协议进行了抽象. 在 POF 中, 一个数据域由一个 (Type, Offset, Length) 三元组表示. 其中 Type 表示该数据域的类型, 值为 0 表明是数据包中的域; 值为 1 则表明是 metadata 中的域. Offset 指出该域的起始位置, 而 Length 则指出了该域的长度.

POF 的流表对数据包中特定的域进行匹配, 按匹配结果执行不同的操作. 一个流表可以表示为

$$\text{Table} = \{\text{Fields}, \{\text{Entry}\}\}$$

$$\text{Entry} = (\text{Value}, \text{Action}, \text{Priority})$$

其中 Fields 是该表匹配的数据域, 各表项中 Value 为匹配的值, Action 为执行的操作, Priority 则是该表项的优先级.

POF 交换机在识别和处理流的过程中, 使用 Goto-Table 指令在不同的流表间切换. 由于数据在网络中传输时会使用多种协议逐层封装, POF 交换机在处理每个数据包时亦是逐层解析. 每一层都可定义若干流表对其中的数据域进行匹配、读取和修改数据等操作, 以及解析下一层协议类型并转到相应的流表. 此外, 每个 POF 交换机均有一个起始流表, 作为处理流程的入口. 单个交换机的处理逻辑可以用一个判别树表示, 如图 1 所示.

目前的 POF 控制器是基于 Floodlight 开发而成的. 如图 2 所示, POF 控制器使用 Floodlight 作为主要通信模块, 通过 OpenFlow 协议与交换机进行通信. 在此基础上, 加入了 POF 模块以及 GUI 等. BypassManager 模块是 POF 模块与 Floodlight 之间的桥梁, 用于检测控制器接收到的 OpenFlow 协议的类型并调用相关 PofManager 功能. PofManager 模块负责维护 PMDatabase, 并为上层应用提供 API. PMDatabase 是控制器在内存中维护的一个数据库 (由控制器

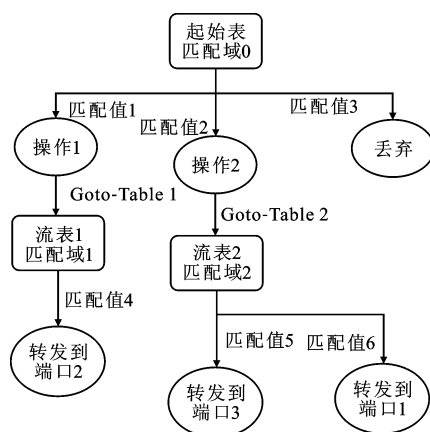


图 1 POF 流处理判别树

定义的一个数据结构, 而非引用的外部数据库程序), 用以存储 POF 环境相关的状态数据, 包括定义的传输协议以及各交换机的流表数据等. 该数据库可以文件的形式导入和导出. GUI 是控制器的用户交互界面, 通过 GUI 可以查看当前网络中接入的交换机的信息, 定义转发协议 (POF 没有预定义的转发协议), 制定各交换机使用的流表等.

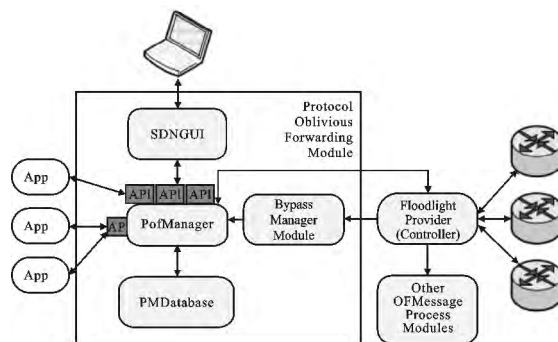


图 2 POF 控制器模块

3 冲突检测

由于 POF 控制器允许控制面上的多个应用同时对网络进行操作, 为了保证网络的正确运行, 应避免不同应用对网络的控制产生冲突. 冲突的最直接体现即是不同的应用对同一个流施加了不同的规则. 对数据流的控制规则的最终表现形式是交换机的流表, 因此需要对流表的表项进行冲突检测.

当某流表中的两条表项具有相同的匹配值, 但操作不同, 表明二者对同一种匹配的流进行了不同的操作. 若同时二者具有相同的优先级, 则交换机无法进行选择, 进而无法决定对满足该匹配的流进行何种操作. 对于这种情况, 判定这两条表项相冲突. 表项冲突判断的伪代码如下:

```
check_entry_conflict
Input:
    entry1 = (value1, action1, priority1)
    entry2 = (value2, action2, priority2)
if value1 = value2 and priority1 = priority2 then
    if action1 ≠ action2 then
        return CONFLICT
    else
        return EXIST
else
    return ACCEPT
```

假设已有的流表中不存在冲突,那么仅需要对新加入的流表项进行检测,查看其是否与已有的某表项冲突即可.显然,当且仅当新添加的表项与表中其他某项具有相同匹配值时才可能产生冲突.由于初始状态下的流表为空,检测添加项的方法可以保证流表无冲突.

一个完整的流处理过程往往涉及多个流表.例如“阻止从 IP 为 192.168.1.3 的主机到 IP 为 192.168.1.4 的主机的 SSH 通信”,即需要涉及处理 IP 和处理 TCP 的两个流表.首先在处理 IP 的流表 1 中,交换机匹配数据包的源 IP 地址,当为 192.168.1.3 时,将目的 IP 写入 metadata 中,并转到处理 TCP 的流表 2;在流表 2 中交换机匹配数据包的端口号和 metadata 中的目的 IP,当二者分别为 22 和 192.168.1.4 时,丢弃该包.

为了保证流处理规则的完整性,流表的冲突检测需要以流为单位进行.定义交换机的一条处理流程中涉及的所有匹配域及相应的匹配值构成的集合为匹配集 M ,如 $\{(Source\ IP, 192.168.1.3), (Port, 22)\}$,则 M 即可作为该流的标识.同时定义该流程中对流的全部操作构成的集合为操作集 A ,相应的优先级构成的序列为 P .则该交换机对该流的完整处理规则 R 可表示为 $R=(M, A, P)$.

当一个交换机中存在 $R_1=(M_1, A_1, P_1)$ 和 $R_2=(M_2, A_2, P_2)$,使得

$$M_1 = M_2, A_1 \neq A_2, P_1 = P_2,$$

则判定 R_1 与 R_2 相冲突.

然而,从流表的层面来看,无法判断不同流表的表项是否属于同一规则.为此需要建立一个暂存层,暂存待写入的表项数据.如图 3 所示,暂存层接收上层应用生成的流表项,同时要求上层给出各流表项所属的规则 ID,并依此对表项进行划分,将属于同一规则的表项加入一个集合中.当上层应用完成规则生成后提交某规则时,调用冲突检测模块对该规

则包含的表项进行检测.只有当集合中的全部表项均无冲突时,该规则才通过检测,之后调用 PofManager 的表项写入函数将其写入 PMDatabase;否则向上层反馈错误.

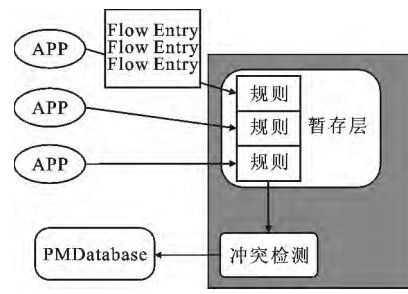


图 3 包含暂存层的冲突检测模块

4 实验与结果分析

由于冲突检测功能完全运行在控制层,与转发层无关,因此相关实验无需 POF 网络环境亦可完成.实验平台使用一台 Intel Core i3-2120 3.30 GHz 的 CPU 和 4GB DDR3 内存的 PC, Linux 系统环境,内核版本 3.12.21,以及 IcedTea JDK 6.1.13.5(POF 控制器使用 Java 开发).虽然无需 POF 网络环境,但控制器仍需要交换机已有的流表数据作为比对.

实验 1:本实验使用包含 4 个流表,每个流表 100 条表项的交换机数据,并分别使用了 6 组不同的待检测样本进行检测.结果如表 1 所示.检测用时主要受待检测样本规模影响.此外,同样本规模下冲突表项数越多,检测用时越少,这是由于无冲突的表项在检测时会遍历全表,而冲突表项在检测到冲突时即停止遍历.

表 1 冲突检测结果

待检测表项数	预期冲突数	检测结果	检测用时/ms
10	3	3	6.037
10	8	8	4.519
50	15	15	25.574
50	26	26	23.285
100	13	13	36.835
100	36	36	34.841

实验 2:实验 1 中的交换机各流表包含的表项数相同,无法判断流表规模对检测用时的影响.本实验使用了实验 1 中的交换机 1 数据,以及另一个包含 8 个流表,各表 50 条表项的交换机 2 数据,分别对 4 组样本进行检测,结果如图 4 所示.虽然两个交换机的总表项数相同,但由于单流表中的表项数不同,导致检测用时不同.显然,当流表规模较小时,遍历该表的时间较短,冲突检测的时间也因此较短.

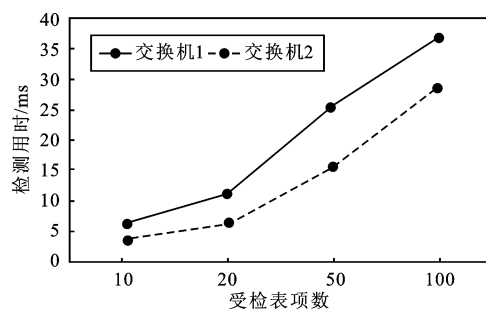


图4 不同交换机数据的检测用时

5 结束语

本文基于协议无感知转发的工作原理,探讨了POF环境下流表冲突的问题,设计出一种冲突检测方案,并在POF控制器上实现。通过实验证明检测的效率与受检规则包含的表项数及流表的规模呈负相关。

参考文献:

- [1] 左青云, 陈鸣, 赵广松, 等. 基于 Open Flow 的 SDN 技术研究[J]. 软件学报, 2013, 24(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] Gude N, Koponen T, Pettit J, et al. NOX: towards an operating system for networks [J]. ACM SIGCOMM Computer Communication Review, 2008, 38

(3): 105-110.

- [4] Big Switch Networks. Floodlight open flow controller [EB/OL]. [2014-12-13]. <http://www.projectfloodlight.org/floodlight/>
- [5] Song H. Protocol-oblivious forwarding: unleash the power of SDN through a future-proof forwarding plane [C] // Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. China, Hongkong, ACM, 2013: 127-132.
- [6] Huawei Technologies. Protocol oblivious forwarding [EB/OL]. [2014-12-10]. <http://www.poforwarding.org/>
- [7] Ghodsi A, Shenker S, Koponen T, et al. Information-centric networking: seeing the forest for the trees [C] // Proceedings of the 10th ACM Workshop on Hot Topics in Networks. Cambridge, ACM, 2011: 1.
- [8] Porras P, Shin S, Yegneswaran V, et al. A security enforcement kernel for open flow networks [C] // Proceedings of the first workshop on Hot topics in software defined networks. Finland, Helsinki, ACM, 2012: 121-126.

作者简介:

吕威男, (1990-), 硕士研究生. 研究方向为基于协议无感知的 SDN 技术.

王雷(通讯作者)男, (1972-), 博士, 副教授. 研究方向为未来网络、云计算以及分布式系统优化.

E-mail: wangl@ustc.edu.cn.

(上接第 77 页)

参考文献:

- [1] 毛光灿, 何大可. 3G 核心加密算法 KASUMI 算法 [J]. 通信技术, 2002, 35(11): 92-94.
- [2] 马炯. 分组密码算法 KASUMI 及其在 WCDMA 中的应用 [J]. 信息安全与通信保密, 2005, 26(2): 123-126.
- [3] Satoh A, Morioka S. Small and high-speed hardware architectures for the 3GPP standard cipher KASUMI [C] // 5th International Conference on Information Security. Sao Paulo, 2002: 48-62.
- [4] Balderas Contreras T, Cumplido Parra R A. An efficient reuse-based approach to implement the 3GPP KASUMI block cipher [C] // 1st International Conference on Electrical and Electronics Engineering. China, Shenzhen, 2004: 113-118.
- [5] 3GPP TS 35.202. Specification of the 3GPP Confidentiality and integrity algorithms; Document 2: KASUMI specifications [S]. USA: 3GPP, 2009.
- [6] 3GPP TS 35.201. Specification of the 3GPP confiden-

tiality and integrity algorithms; Document 1: f8 and f9 specifications [S]. USA: 3GPP, 2009.

- [7] Dai Yamamoto, Kouichi Itoh, Jun Yajima. A very compact hardware implementation of the KASUMI block cipher [C] // Information Security Theory and Practices. Berlin, Heidelberg, Springer, 2010 (6033): 293-307.
- [8] Marinis K, Moshopoulos N K, Karoubalis F, et al. On the hardware implementation of the 3GPP confidentiality and integrity algorithms [C] // 4th International Conference on Information Security. Milaga, Spain, 2001: 248-265.

作者简介:

许静雯女, (1990-), 硕士. 研究方向为信息安全算法的数字大规模集成电路设计与实现.

E-mail: xujingwen_career@163.com.

李树国男, (1963-), 教授, 博士生导师. 研究方向为信息安全算法的数字大规模集成电路设计与实现.