SDN跨层回环攻击的检测与防御

张 云[1,3],江 勇[2],郑 靖[1,3],庞春辉[1,3],李 琦[1,2]
(1. 清华大学网络科学与网络空间研究院,北京 100084;2. 清华大学深圳研究生院,广东深圳 518055;
3. 清华大学计算机科学与技术系,北京 100084)

摘 要: 软件定义网络(Software Define Network,SDN)将控制层和数据层进行分离,给网络带来灵活性、开放性以及可编程性.然而,分离引入了新的网络安全问题.我们发现通过构造特定规则可以构造跨层回环攻击,使得数据包在控制器和交换机之间不断循环转发.跨层回环会造成控制器拥塞,并导致控制器无法正常工作.现有的策略一致性检测方案并不能检测跨层回环攻击.为此,本文提出了一种实时检测和防御跨层回环的方法.通过构造基于Packet-out的转发图分析规则路径,从而快速检测和防御回环.我们在开源控制器Floodlight上实现了我们提出的回环检测和防御方案,并在Mininet仿真器上对其性能进行了评估,结果表明本方案能够实时检测并有效防御跨层回环攻击.

关键词: 软件定义网络;控制层;数据层;跨层回环检测;策略一致性检测
中图分类号: TN915.8 **文献标识码:** A **文章编号:** 0372-2112 (2019)05-1146-06
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.05.023

Detecting and Defending Against Controller-to-Switch Loop Attacks in SDN

ZHANG Yun[1,3],JIANG Yong[2],ZHENG Jing[1,3],PANG Chun-hui[1,3],LI Qi[1,2]
(1. *Institute for Network Sciences and Cyberspace*,*Tsinghua University*,*Beijing* 100084,*China*;
2. *Graduate School at Shenzhen*,*Tsinghua University*,*Shenzhen*,*Guangdong* 518055,*China*;
3. *Department of Computer Science and Techonlogy*,*Tsinghua University*,*Beijing* 100084,*China*)

Abstract: Software-Defined Networking (SDN) separates data plane from control plane,which makes it more flexible,opening and programmable,compared with traditional IP networks. However,the separation incurs many security problems. In this paper,we find that we can construct controller-to-switch loop (CSL) attacks by leveraging dedicated rules and well constructed packets. The attacks can effectively exhaust controller resource,which leads to denial of service (DoS). The existing OpenFlow policy verification schemes only focus on detecting data plane loop,and cannot detect such controller-to-switch loops. In order to detect CSL attacks,we proposed a novel policy verification scheme. The scheme constructs a packet forwarding graph by analyzing network update events and packet-out messages,and efficiently identifies the forwarding loops by traversing the graph. In order to evaluate our defense,we implement it in the Floodlight controller,and perform experiments with Mininet. The experimental results show that our defense can precisely detect the loop attacks and effectively throttle them.

Key words: software-defined networking;control plane;data plane;controller-to-switch loop detection;policy consistency check

1 引言

SDN诞生于斯坦福大学的Clean Slate课题,它的核心思想是将控制和转发进行分离.这使得其较传统网络而言,网络更加开放,部署管理更为方便,所以SDN在近来得到了广泛的关注.但随着SDN的发展,它不仅仅带来便利,暴露出来的安全问题也越来越多,例如,数据平面的可达性问题、转发回环问题以及转发异常问题等[1,14].

本文研究一个新的SDN安全缺陷,即控制层-数据层回环漏洞.这个漏洞的本质原因在于SDN中控制层与数据层的分离,具体来说,控制和转发的分离导致控

收稿日期:2017-04-17;修回日期:2017-12-18;责任编辑:郭游
基金项目:国家重点研发计划(No.2016YFB0800102);国家自然科学基金(No.61572278,No.U1736209);深圳市基础研究基金(No.JCYJ20170307153259323)

制层和数据层之间需要按照南向接口 OpenFlow 协议完成网络通信,数据层交换机通过 Packet-in 消息向控制器报告数据包,控制器通过下发 Flow-mod 消息,使得转发路径上的交换机获得新的转发流表规则. 同时,回复 Packet-out 消息以指示当前数据包的转发操作. 但是,当网络中存在错误配置、SDN 应用间相互干扰、控制器出现 Bug 或者当恶意应用下发的高优先级规则主动触发 Packet-in 消息时,交换机向控制器报告的数据包返回到交换机后,又会重新报告给控制器. 控制器收到后,又再次将该数据包发送到交换机,如此往复,数据包一直在控制器和交换机之间来回发送,从而造成控制层和数据层之间的回环,即跨层回环. 跨层回环会导致数据层面无法转发数据包,并不断向控制器请求转发规则. 控制器的处理能力遭遇瓶颈时,所有新流都无法及时获得处理,导致网络流量不能正常转发,进而造成网络瘫痪.

虽然 SDN 中有许多检测回环的机制,例如 Veriflow[3] 和 Netplumber[1] 等,但是这些机制往往关注于数据层面流表项的静态或动态的分析,因此都只能检测数据层面的回环,跨层回环并不在它们的检测范围内. 针对上述安全问题,本文创新性地提出一种实时的跨层回环检测机制,该机制的基本原理是将数据平面的网络数据转发消息和控制平面的控制消息流进行统一建模,形成一个统一的转发图. 通过实时监测转发图中是否形成转发回环来确定攻击的发生. 我们通过实时监听网络状态更新事件,提取更新事件中的流表规则,并依此建立数据平面的转发图来描述任意网络流的转发. 同时,当收到出端口为 TABLE 的 Packet-out 消息时,我们将待处理的数据包在控制平面的消息流(Packet-out 消息)作为转发规则加入到转发图. 通过计算当前流的规则处理路径,我们就能快速地检测出跨层回环. 此外,我们提出了一种跨层回环防御机制,它通过概率性地丢弃 Packet-in 消息并丢弃异常的 Packet-out 消息来打破当前回环,这种机制的优点在于,它既能减少控制器资源的浪费又不影响其他正常数据流的转发.

2　CSL 攻击

本节我们提出一个跨控制层和数据层的回环攻击(CSL). 我们发现通过安装特定的流表规则,可以使得匹配这些规则的数据包不断在控制器和交换机之间的回环中转发. 根据 SDN 路由机制,当出现新流,控制器下发的 Flow-mod 成功安装到交换机后,该流的后续数据包发送到交换机时,会匹配这些流表项并执行对应的操作进行转发. 然而,我们发现当控制器应用下发一条特定规则,例如流表跳转规则,它与交换机中其它规则配合起来会导致后续数据包不断触发 table-miss 机制,向控制器发送 Packet-in 消息. 需要注意,如果控制器的应用需要交换机处理一些复杂功能,会在交换机中利用 OpenFlow 协议的多流表机制,建立多个流表完成不同的功能,并将 Packet-out 消息从 TABLE 端口发出. 此时,数据包会回到当前交换机头部从第一个流表开始匹配,重新触发 Packet-in 消息. 因而,数据包就会一直在控制器和交换机之间来回发送,形成跨层回环攻击. 我们将产生回环的流对应的 Packet-in 消息定义为可疑 Packet-in 消息.

如图 1,我们从一个使用多流表的场景来展现跨层回环攻击的过程. 图中有 3 个交换机,其中 S2,S3 是 S1 的下一跳,数据包发送到 S1 时,通过选择不同的下一跳来实现复杂均衡. 当源地址为 a,目的地址为 b 的数据包到达该交换机,首先执行 table0 中流表项的对应操作,设置完 VLAN 后将其发送至 table1,而 table1 中并没有可以匹配的流表项,会触发 table1 中的 table-miss 操作,将数据包发送给控制器,控制器将数据包从 TABLE 端口发出,交换机又重新匹配 table0 中的规则,触发 table1 的 table-miss 操作向控制器报告,这样,我们成功构建了跨层回环攻击. 需要注意,table0、table1 中的表项本身可能并不是恶意的,但通过关联就可以构造攻击. 因此,检测这一类攻击非常困难. 此外,当控制器没有实现应用隔离,攻击者可以利用丢弃或者篡改其他应用的 Flow-mod 消息来实现跨层回环攻击.

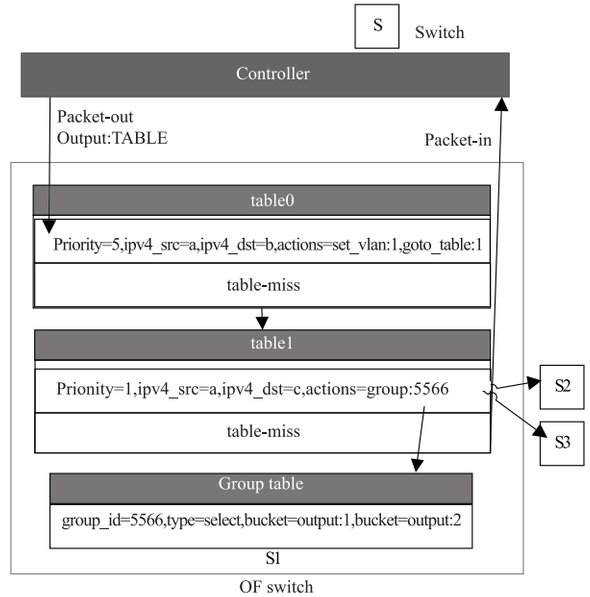

图1　跨层攻击实例

为了防御攻击,我们无法直接丢弃可疑的 Packet-in 消息,因为后续正常流可能会产生同样的 Packet-in 消息,直接丢弃会导致正常流的数据包被错误丢弃.

配置错误和恶意表项等导致的跨层回环不仅会

出现在多流表场景中,也会出现在单流表场景中,本文重点研究单流表导致的回环. 我们的方案可以直接扩展到多流表场景,所以本文不重点讨论多流表场景.

3 方案概述

本文提出的跨层回环攻击检测与防御方案(DDL)实现在 SDN 控制器操作系统中,从而可以捕获控制器和交换机之间的 Packet-out 消息和 Flow-mod 消息. DDL 采用基于 Packet-out 消息的转发图检测回环,并使用基于概率的方式来缓解回环. 方案架构如图 2 所示,主要分为三个部分:基于 Packet-out 消息构建转发图、基于规则路径检测回环以及基于概率缓解回环. 当 DDL 截获到出端口为 TABLE 的 Packet-out 消息时,它根据这个 Packet-out 消息构建转发图;遍历转发图为当前流寻找规则路径,并基于规则路径检测回环. 若检测到回环,通过概率性丢弃 Packet-in 消息和丢弃异常的 Packet-out 消息来缓解回环.

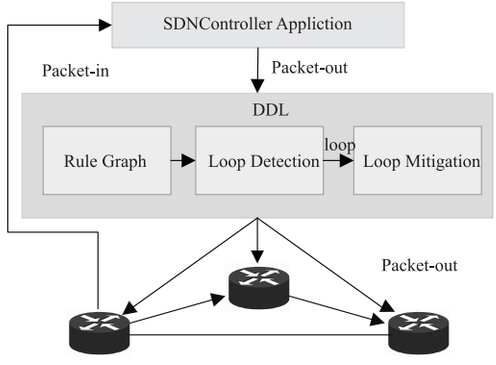

图2 方案概要图

然而,实现本方案还要面临以下几点挑战. (1)转发图中节点表示的是流表规则,如何扩展转发图以支持 Packet-out 消息的遍历? (2)转发图中包含全网的规则节点,如何在遍历转发图的同时实时检测转发异常? (3)如何实现概率丢弃从而有效缓解和抑制跨层回环攻击?

4 设计与实现

DDL 方案通过概率丢包来抑制回环攻击. 我们为每个可疑的 Packet-in 设定一个丢弃概率. 当收到 Packet-in 消息时,查找丢弃概率 p,并生成一个[0,1]的随机数∂,若$\partial < p$,将 Packet-in 消息丢弃;若$\partial > p$,将 Packet-in 消息发送到控制器内核模块. 当截获到它对应的出端口为 TABLE 的 Packet-out 消息时,执行基于 Packet-out 构建转发图和基于转发图检测回环部分,并根据检测结果计算和更新丢弃概率. 此外,若检测到回环,将当前 Packet-out 消息丢弃.

4.1 基于 Packet-out 的转发图构建

转发图是用来存储全网的流表规则及规则之间的依赖关系的数据结构,其中节点表示规则,节点之间的单向边表示规则之间的依赖关系. 依赖关系根据 HSA[5] 算法计算获得. 我们把 Packet-out 消息转换成规则加入到转发图中,成为控制层节点,从而支持基于转发图的控制流遍历. SDN 中每个 Packet-in 消息都有它对应的 Packet-out 消息,通过分析 Packet-out 消息我们可以提取规则信息,转发图中从流表规则构成的数据层节点[4]到控制层节点的边表示 Packet-in 消息. 因而,我们可以在转发图中完整地表示规则处理路径.

4.2 基于转发图的回环检测

为了实现高效的回环检测以减少检测延迟,我们采用基于规则路径的回环检测算法. 我们关联转发图中每个规则节点和它的前驱、后继节点,以待分析的 Packet-out 消息对应的控制层节点为起点,可以有效查找转发图中的规则处理路径,从而,判断当前流会不会产生跨层回环. 一般我们通过以下两个准则检测回环.

(1)若规则处理路径中出现重复的控制层节点,则表示存在跨层回环. 这是由于攻击者篡改应用下发的规则使其主动触发 Packet-in 消息从而导致的回环.

(2)若规则处理路径中只有 Packet-out 节点,也表示存在跨层回环. 这是由于攻击直接丢弃应用下发的规则使得正常的 Flow-mod 消息无法下发,从而导致的回环. 如果是多流表场景下攻击,则规则路径会以跳转规则节点为终点,这是由于多流表中规则相互关联导致的回环(如图 2 的例子).

一般来说,Flow-mod 消息优先于 Packet-out 消息下发,当检测模块截获到 Packet-out 消息时,检测模块已经截获过当前流对应的 Flow-mod 消息. 因此,在转发图中加入 Packet-out 对应的控制节点之前,Flow-mod 消息对应的规则节点已经加入完毕,例如节点 A. 当在转发图加入控制节点 P 后,若存在跨层回环,在转发图中必有环 P→A→P. 此外,若当前流在转发图中没有规则节点,则存在环 P→table-miss→P. 因此,我们的方案可以检测所有的回环.

4.3 基于概率缓解回环

产生回环的关键是 Packet-in、Packet-out 消息的转发,回环可能仅存在一段时间. 因此,回环消失后不需要丢弃 Packet-in 消息,如果直接丢弃所有产生回环的流的 Packet-in 消息,会使得后续正常流的数据包被错误丢弃. 如果只丢弃产生回环的 Packet-out 消息的话,每个 Packet-out 消息都需要进行检测,控制器处理产生回环可能性极大的流时,又会造成资源的浪费,所以,我们采取了基于概率的方式缓解回环:当控制器收到 Packet-in 消息时,以 p 的概率丢弃 Packet-in 消息,以 1-p 的

概率检测 Packet-out 消息(即检测未被丢弃的 Packet-in 消息对应的 Packet-out 消息). 检测完成后,根据检测结果更新丢弃概率,并丢弃产生回环的 Packet-out 消息. 这样,不仅能节省控制器的资源,还能有效地缓解回环.

为了实现概率性地丢弃 Packet-in 消息,我们为每个产生回环的流计算和维护一个丢弃概率,计算公式如下:

$$p = 1 - \frac{1}{1+n} \tag{1}$$

其中,n 表示检测到回环的次数,p 表示丢弃概率. ,我们根据检测结果来更新概率表,检测到回环时,将 n 加 1,并使用式(1)计算并更新丢弃概率. 当某条流的数据包通过检测时,就将 n 和 p 都重置为零.

5　实现与评估

5.1　实现和实验设置

我们在开源控制器 Floodlight-1.2 版本中实现了一个支持攻击和防御的应用模块,在 Ubuntu14.04 LTS 中使用 Mininet 模拟网络拓扑来评估方案的性能. 为了避免不必要的干扰,在实验过程中,我们将控制层和数据层分开部署. Floodlight 控制器部署在 Intel Core i5 2450 2.5 GHz 的 CPU,8GB 的 RAM 的笔记本上,Mininet 部署在服务器上.

我们模拟四个真实拓扑:Internet2、BandCon、BICS、China Telecom,并分别从攻击检测和防御两个方面评价 DDL 的性能.

5.2　检测性能

我们根据不同的网络拓扑和转发图中节点数量评价检测延迟.

实验 1　不同拓扑下的检测延迟. 本实验分别在上述 4 个拓扑下进行了 1000 次回环检测,根据这 1000 次实验结果画成 CDF 图,如图 3 所示,从结果可以看出,所有的检测耗时基本都在 100μs 内,90% 的检测耗时能在 30μs 之内.

实验 2　不同节点数量的检测延迟. 我们选用了 BandCon 拓扑在转发图中包含 10、100、1000 个规则节点时分别进行 1000 次检测. 图 4 为检测延迟的 CDF 图. 根据结果我们观察到,90% 左右的回环检测都能在 30 微秒内完成,并且转发图中的节点数量对检测延迟的影响不大. 因此,我们提供了高效的跨层回环攻击检测.

实验 3　不同拓扑下的更新延迟. 我们测量了规则插入转发图的延迟,并分别在 Internet2、BandCon、BICS、China Telecom 这 4 个拓扑下进行了 1000 次检测. 实验结果如图 5 所示,80% 左右的更新能在 1ms 内完成. 由于我们的在线实验考虑了规则插入事件的并发性,引入了加锁和解锁的延迟. 总体来说,本方案的规则插入延时是可以接受的.

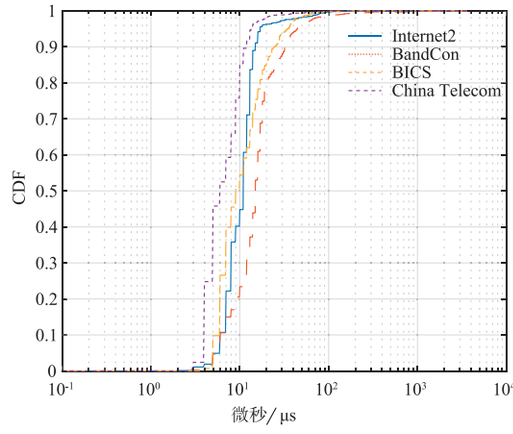

图3　不同拓扑的检测时间

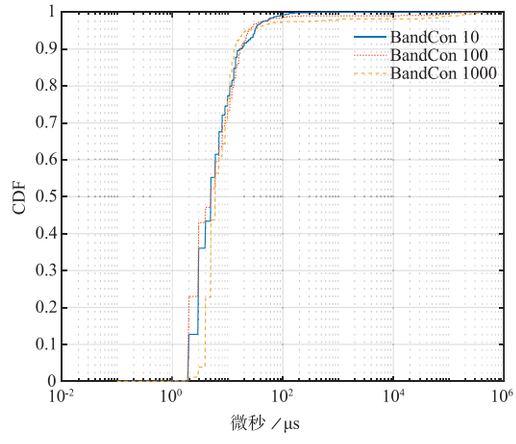

图4　BandCon包含不同数量的规则时的检测时间

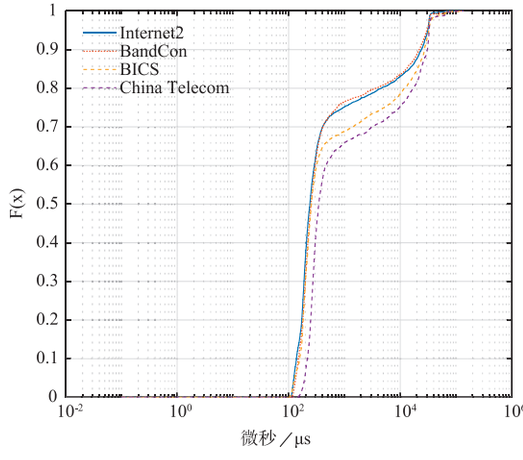

图5　不同拓扑下的更新时间

检测机制的开销主要来自于规则插入和检测两个操作,根据检测的实验结果可以看出,整个检测过程基本能在 1ms 左右完成. 因此开销是可以接受的.

5.3 防御性能

防御部分的性能主要考虑回环缓解率和缓解前后 Packet-in 消息的处理时间的对比,在 Internet2 拓扑下进行下面两个实验.

实验 4 Packet-in 消息数量. 我们在存在回环防御机制和无回环防御机制时,在网络中发送分别 1、10、100、1000 个会产生回环的数据包,统计 1 分钟内交换机产生的 Packet-in 消息的数量,并分别进行了 10 组实验. 通过表 1 可以看出,我们的防御机制使得每个数据包仅产生了一个 Packet-in 消息,因而消除了所有回环产生的 Packet-in 消息. 所以,我们的缓解机制是非常有效的.

表 1 Packet-in 数量比较

	1	10	100	1000
无防御	49605	78549	107669	125704
有防御	1	10	100	1000

实验 5 处理正常流的 Packet-in 消息的时间. 我们在存在防御机制和无防御机制时,在网络中发送 1、10、100、1000 个会产生回环的数据包,在产生回环的包发送完后,发送 1000 个正常流的数据包,并记录每个包的 Packet-in 消息的处理延时(即数据包在控制器中的处理时间),我们将执行 1000 组实验的结果取平均数. 从表 2 可以看出,当存在攻击时,存在防御机制时的消息处理时间分别加快了 63.7%、68.6%、77.3%、89.0%. 另外,随着产生回环的数据包的增多,我们的防御机制的 Packet-in 消息处理延迟比较稳定. 因此,我们的防御机制有效减少了控制器的负荷并确保了控制器的性能.

表 2 处理 Packet-in 的延迟(单位:μs)

	1	10	100	1000
无防御	146.705	170.825	238.902	491.848
有防御	53.205	53.628	54.177	53.965

实验总结:根据实验结果我们可以看出中构建跨层回环攻击给交换机和控制器带来极大负载. 当存在多个回环时,控制层和数据层甚至会瘫痪. 我们的回环检测和防御方案可以有效解决这个问题.

6 相关工作

目前已经有很多 SDN 转发策略一致性验证的工作[1~13]. FlowChecker[4] 通过二元决策树技术对流表的配置信息进行重新编码,并提出 FlowChecker 检测系统,通过形式化的方法对流规则的一致性进行检测; Anteater[2] 通过将需要检测的策略属性转换成 SAT 问题,从而进行策略一致性检测. 但这两种方法都不是实时检测.

VeriFlow[3] 采用多维 trie 树来将网络划分为不同的等价类,通过等价类来实时验证全网范围内的不变量; NetPlumber[1] 利用 HSA[5] 实现了一个独立于协议的实时检测策略一致性的框架,能验证数据层的策略违背; CCG[6] 通过扩展 VeriFlow[4] 来实现网络配置时策略一致性检测; Atomic Predicates Verifier[7] 计算了一组很小的、独一无二的判断式来过滤规则,在时间和空间上提供了更有效的验证; PGA[8] 利用策略图来确定访问控制和检测服务链策略冲突. 现有这些实时检测方案[1,3,5~8] 没有考虑由于控制层和数据层的交互带来的安全问题,故不能检测跨层回环.

此外,还有一些其他安全策略验证工作[9~13] 也无法检测跨层回环攻击. FortNox[9] 是通过基于角色的授权和安全限制的下发来保证策略一致性,该方法通过将修改的 IP 地址与防火墙策略中的地址相比较,来检测策略违背,但是这个方案针对的是现有防火墙容易被绕过的问题; Pyretic[10] 提出了一种应用于程序策略合成的高级语言,它可以检测防火墙策略和流表策略的直接违背; FLOVER[11] 是一个可以部署在 SDN 网络中的新型的策略检测系统,解决动态插入的流规则不会违反潜在的安全策略问题; FlowGuard[12] 通过在运行时转发和过滤策略来检测和解决策略违背; SE-Floodlight[13] 通过将控制器中加入基于角色的访问控制保证了正确规则的实施.

与本文工作最相似的方案是 NetPlumber[1]. 我们都使用了 HSA 算法来构建转发图,但是 NetPlumber 只关注数据层的内容,只能检测数据层的安全问题. 而本方案关注的是控制层和数据层间的安全问题,转发图不仅能表示数据层的内容,还能表示控制器的内容,并实现了实时在线检测. 另外,NetPlumber 无法提供实时的防御,本方案可以实时防御和缓解跨层回环攻击.

7 结论

本文利用控制层和数据层交互产生的漏洞提出跨层回环攻击,并采用基于 Packet-out 的转发图实时检测和防御攻击. 我们在开源控制器 Floodlight 上实现了 CSL 攻击和攻击检测及防御方案,并在 Mininet 中对本方案进行了性能评估. 实验数据表明,DDL 可以实时检测并抑制跨层回环攻击.

参考文献

[1] Kazemian P, Chang M, Zeng H, et al. Real time network policy checking using header space analysis[A]. NSDI'13:Usenix Conference on Networked Systems Design and Implementation[C]. Lombard, IL:ACM, 2013. 99 – 112.

[2] Mai H, Khurshid A, Agarwal R, et al. Debugging the data

plane with anteater[J]. ACM Sigcomm Computer Communication Review,2011,41(4):290 – 301.

[3] Khurshid A, Zhou W, Caesar M, et al. Veriflow: verifying network-wide invariants in real time[J]. ACM Sigcomm Computer Communication Review, 2012, 42 (4): 467 – 472.

[4] Al-Shaer E, Al-Haj S. FlowChecker: configuration analysis and verification of federated openflow infrastructures[A]. SafeConfig '10: ACM Workshop on Assurable and Usable Security Configuration[C]. Chicago: ACM, 2010. 37 – 44.

[5] Kazemian P, Varghese G, Mckeown N. Header Space Analysis: Static Checking For Networks[A]. NSDI'12: Usenix Conference on Networked Systems Design and Implementation[C]. San Jose, CA: ACM, 2012. 9 – 9.

[6] Zhou W, Jin D, Croft J, et al. Enforcing customizable consistency properties in software-defined networks[A]. NSDI'15: Usenix Conference on Networked Systems Design and Implementation [C]. Oakland, CA: ACM, 2015. 73 – 85.

[7] Yang H, Lam S S. Real-time verification of network properties using Atomic Predicates[A]. ICNP'13: IEEE International Conference on Network Protocols [C]. Germany: IEEE, 2013. 1 – 11.

[8] Prakash C, Turner Y, Turner Y, et al. PGA: Using Graphs to Express and Automatically Reconcile Network Policies [A]. SIGCOMM'15: ACM Conference on Special Interest Group on Data Communication[C]. London: ACM, 2015. 29 – 42.

[9] Porras P, Shin S, Yegneswaran V, et al. A Security Enforcement Kernel for OpenFlow Networks[A]. HotSDN'12: Hot Topics in Software Defined Networking (HotSDN) [C]. Helsinki: ACM. 2012. 121 – 126.

[10] Monsanto C, Reich J, Foster N, et al. Composing software-defined networks[A]. NSDI'13: Networked Systems Design and Implementation[C]. Lombard, IL: ACM, 2013. 1 – 13

[11] Son S, Shin S, Yegneswaran V, et al. Model checking invariant security properties in OpenFlow[A]. ICC'13: IEEE International Conference on Communications[C]. Hungary: IEEE, 2013. 1974 – 1979.

[12] Hu H, Han W, Ahn G J, et al. FLOWGUARD: building robust firewalls for software-defined networks[A]. SIGCOMM'14: ACM Special Interest Group on Data Communication[C]. Chicago: ACM, 2014. 1 – 3.

[13] Porras P, Cheung S, Fong M, et al. Securing the software defined network control layer[A]. NSDI'15: Network and Distributed System Security Symposium [C]. Oakland: ACM, 2015.

[14] 刘艺,张红旗,杨英杰. 基于启发式调度的 OpenFlow 网络规则一致更新方案[J]. 电子学报,2017,45(7):1637 – 1645.
LIU Yi, ZHANG Hong-Qi, YANG Ying-Jie. Consistent rule update scheme based on heuristic scheduling for openflow networks [J]. Acta Electronica Sinica, 2017, 45((7):1637 – 1645. (in Chinese)

作者简介

张　云　女,1992 年生,清华大学硕士研究生,主要研究领域为 SDN 安全等.
E-mail:zhangyun15@ mails. tsinghua. edu. cn

江　勇　男,1975 年生,教授,博士生导师,主要研究领域为计算机网络体系结构.
E-mail:jiangy@ sz. tsinghua. edu. cn

郑　靖　男,1985 年生,清华大学硕士研究生,主要研究领域为网络安全.
E-mail:zhengj14@ mails. tsinghua. edu. cn

庞春辉　男,1993 年生,清华大学硕士研究生,主要研究领域为网络安全、网络调试和网络测量.
Email:chunhui. pang@ outlook. com

李　琦(通讯作者)　男,1979 年生,博士,副研究员,计算机学会高级会员(E200020487S),主要研究领域为互联网安全、云安全和隐私保护等.
E-mails:qi. li@ sz. tsinghua. edu. cn