

基于 SDN 的云安全应用研究综述

施江勇, 杨岳湘, 李文华, 王森

(国防科技大学计算机学院, 湖南 长沙 410072)

摘要: 软件定义网络(SDN, software defined network)技术将传统网络的控制层和数据层进行解耦合, 使网络具有可编程性, 极大地方便了网络应用的开发。对基于 SDN 的云安全应用进行了全方位的分析, 在分类的基础上, 对每一类安全应用顺着其发展脉络分析了相关工作的优缺点, 并进行了总结。另外, 针对当前云安全服务模型不够完善、不能跟计算和网络服务有机整合的问题, 提出了一种基于 SDN 的云安全服务模型 SDCSec。最后, 对基于 SDN 的云安全应用的未来发展方向做了展望。

关键词: 软件定义网络; 云计算; 云安全; 应用开发

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2017.00165

Research on SDN-based cloud security application

SHI Jiang-yong, YANG Yue-xiang, LI Wen-hua, WANG Sen

(School of Computer Science, National University of Defense Technology, Changsha 410072, China)

Abstract: SDN decoupled the control plane and data plane of traditional network, making the network programmable and simplifying the development of network applications. A comprehensive survey was made to current studies on SDN-based cloud security applications. Based on classification of the applications, the advantages and disadvantages of each application were studied and each category was summarized. What's more, a service model of cloud security based on SDN was proposed, to overcome the inefficient of current works and integrate security with network service. Lastly, some suggestions about the future development of SDN-based cloud security applications were made.

Key words: SDN, cloud computing, cloud security, application development

1 引言

传统安全解决方案在云数据中心部署时面临网络边界模糊、流量监控困难、策略无法跟随虚拟机迁移、安全无法按需交付以及安全管理边界难以定义等问题。软件定义网络的出现为这些问题的解决带来了契机。软件定义网络的概念始于2006年斯坦福大学 Clean State 项目组, 并于2009年得到了系统的阐述, 即基于分层的思想将数据与

控制分离, 通过软件实现对网络控制的编程。OpenFlow^[1]作为 SDN 的典型实现, 以其简明的结构、清晰的 API 定义得到了广泛的研究, 成为 SDN 领域最知名的协议。

SDN 主要由基础设施层、控制层和应用层组成, 如图 1 所示。通过分离传统网络的数据面板和控制面板, 实现灵活的网络管理和敏捷的应用开发。SDN 的主要特性包括集中控制、开放接口、网络虚拟化。集中控制的特点方便网络管理员或

收稿日期: 2017-03-05; 修回日期: 2017-04-24。通信作者: 施江勇, shijiangyong@nudt.edu.cn

基金项目: 国家自然科学基金资助项目 (No.61170286)

Foundation Item: The National Natural Science Foundation of China (No.61170286)

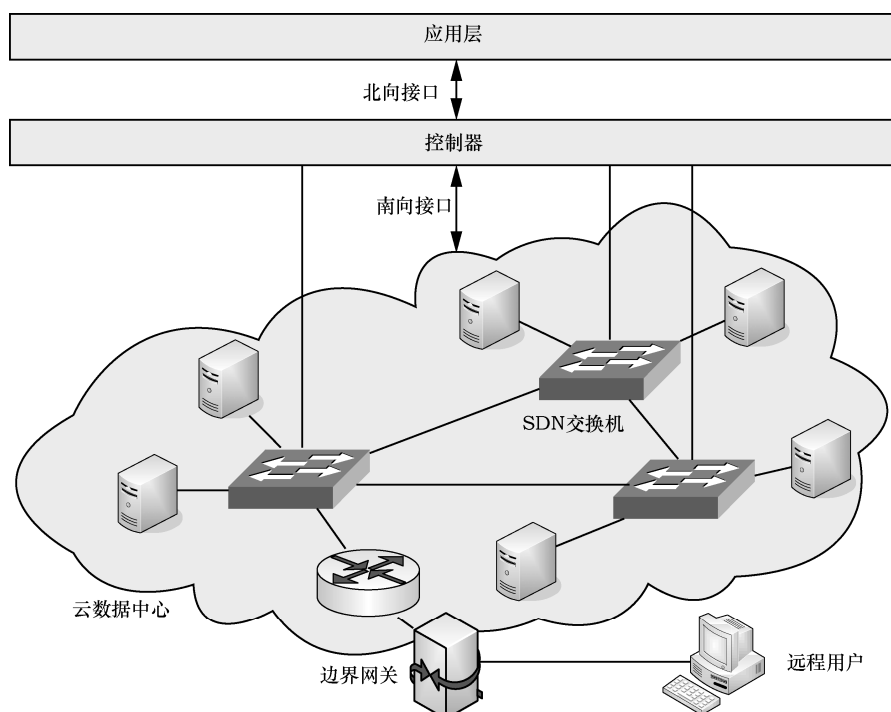


图1 SDN 架构及其在云数据中心的部署

安全管理员对数据中心、云网络进行统一的管理，在提高效率的同时避免因管理不当造成的单点失效问题。开放接口的特点极大地方便了网络安全应用的开发，以控制器为核心，通过南向接口连接各类网络设备，通过北向接口方便应用开发，最终实现应用和网络的无缝集成。

云计算的虚拟化、弹性、按需提供特性跟 SDN 的网络虚拟化、灵活性、开放性特点非常相似，两者的出发点都是为了提高资源的可管理性和利用率，只不过前者侧重于计算资源、后者侧重于网络资源。两者的结合势必进一步推动云计算的发展，尤其是在网络安全领域，基于 SDN 开发云安全应用成为云计算发展的重要方向。SDN 跟云计算的结合，使网络资源能够与计算资源、存储资源一同以服务的方式交付用户，同时 SDN 对于提高数据中心的能效、增强网络的可管理性以及提高网络安全性具有广泛的应用空间。本文研究 SDN 跟云安全结合的应用案例，在细粒度分类的基础上总结基于 SDN 云安全应用的一般性问题，并针对现有工作的不足对未来研究方向提供一定的建议。

2 SDN 云安全应用分类研究

基于 SDN 的云安全应用可以分为以下类别：

防火墙、DDoS 检测、IDS/IPS、异常检测、网络监控，以及其他安全应用。本文以具体的应用类型对现有工作进行梳理，并总结其优缺点，从中发现基于 SDN 安全应用的未来发展方向。

2.1 防火墙

云数据中心的防火墙既要提供严格的外网访问控制，又要在多租户的环境中对云内部进行有效的网络隔离，是云用户最广泛的安全需求。在 SDN 出现之前，云数据中心防火墙主要基于硬件防火墙进行实现。硬件防火墙成本高，且配置不灵活，而云数据中心网络规模巨大、网络拓扑随虚拟机迁移动态变化，使防火墙策略管理非常困难。SDN 的出现及其集中控制的特点，使软件防火墙非常容易实现。

Oftables^[2]基于 OpenFlow 实现了细粒度的防火墙，OpenFlow 将传统的 7 层网络扁平化，通过丰富的匹配字段可以实现细粒度的包过滤和访问控制，如基于 MAC 地址、IP 地址、端口、协议类型等。

DC-firewall^[3]考虑 P2P 网络的分布式特点构建了基于 SDN 的分布式防火墙。DC-Firewall 使用模块化设计提供扩展性，以适应不同的 P2P 网络拓扑结构。然而 DC-Firewall 并没有讨论如何实

现不同防火墙之间的协调以及如何避免防火墙之间策略的冲突,也没有具体实现细节。

Floodlight-Firewall^[4]是Floodlight控制器的一个模块,提供的是防火墙功能。Floodlight-Firewall通过监控 Packet-In 消息实现规则匹配并将规则转换为流表下发交换机,实现访问控制。Packet-In消息是指交换机在找不到数据分组对应匹配流的时候发送给控制器的消息,该消息携带了数据分组的内容,因此可供控制器分析并做出响应。通过给防火墙规则对应的流表分配优先级,实现不同防火墙规则按优先级进行匹配。Floodlight-Firewall 存在的问题是防火墙规则到流表的转换不够灵活,如传统防火墙支持的端口范围匹配在Floodlight-Firewall上就无法实现,此外,删除防火墙规则时,交换机中对应的流表并没有立即删除,而是等到其超时后才失效,这会留下安全隐患。

OpenFlow 协议是无状态的,即流表的匹配跟历史数据无关,这使攻击者能够通过改写交换机中的流表项绕过现有的 SDN 防火墙。FortNox^[5]针对该问题,提出了流表修改检测方法,该方法通过将流表中被修改过的源地址和目的地址加入IP地址集合中,然后与防火墙策略中的IP地址进行比较来检测策略冲突。该方法能够检测出部分策略冲突,但由于缺乏跟踪数据流路径的机制,从而导致存在误报。

此外,由于SDN交换机的流表频繁地发生变化,包括流表增添、修改和删除,导致防火墙的规则也可能发生冲突。FlowGuard^[6]和FlowVerifier^[7]针对这一问题进行了研究,并提出了一种动态策略冲突检测和消除方法。其冲突检测方法基于头部空间分析构建流路径空间和防火墙策略授权空间,当检测到流路径空间和防火墙授权空间发生重叠时判定为冲突,然后通过重新

选择流路径或给流增加标记来消除冲突。虽然FlowGuard和FlowVerifier考虑流表更新给防火墙规则带来的冲突问题,但没有解决虚拟机迁移时防火墙策略的迁移问题。

表1总结了基于SDN防火墙应用的优缺点,由表1可知,基于SDN的防火墙正由无状态向有状态演变,规则设置的粒度也越来越细,然而流表的转换规则、策略冲突问题和策略迁移问题依然是目前防火墙应用普遍存在的问题,有待未来研究进一步解决。

2.2 DDoS 检测

DDoS 攻击一直是互联网安全严重的威胁之一,由于其发送的数据分组本身可能是正常的业务数据分组,使其很难通过分组检测的方式进行检测。尤其是对于云内部的DDoS攻击,不仅会消耗目标虚拟机的资源,也会消耗主机的资源,影响云内其他租户。因此,基于SDN高效地检测DDoS攻击很有必要。

Lightweight-DDoS^[8]利用SDN控制器的集中特性提取DDoS攻击的特征,包括流平均报文数、流平均字节数、流平均持续时间、对流比例、单流增长率、端口增长率,然后使用自组织映射人工神经网络算法进行分类,实验结果表明,跟其他基于报文检测的方法相比,Lightweight-DDoS具有更高的效率和相似的效果,并且可以动态调整获取流特征的交换机以适应网络的变化以及攻击的来源,进一步提高效率。然而,Lightweight-DDoS使用固定周期查询流统计信息,适应性不够,若周期太长,则检测结果不准确;周期太短,则开销大。此外,流量过大、检测任务过重时需要多个检测器协同,如何最优化任务划分以及情报共享并没有论述,且Lightweight-DDoS只能检测到DDoS攻击的存在,无法精确定位攻击者。

表 1

基于 SDN 的防火墙应用

防火墙	类型	优点	缺点
Oftables	无状态	细粒度, 分布式	防火墙之间策略冲突
DC-firewall	无状态	分布式, 模块化设计	防火墙之间策略冲突
POX-Firewall	无状态	实现了不同防火墙规则按优先级进行匹配	规则到流表的转换不够灵活, 流表删除不及时
FortNox	有状态	能够检测出部分策略冲突	冲突检测结果存在误报
FlowGuard, FlowVerifier	有状态	动态策略冲突检测和消除	防火墙策略不能迁移

Behavior Threat Detection^[9]关注于大流检测问题,包括高速、持久化的DDoS攻击,在总结不同网络层检测大流的方法后,提出SDN/NFV在大流检测领域的优势,主要是可以动态调整流信息处理的参数阈值,如带宽阈值和时间阈值;调整的方式既可以根据不同流类型调整,也可以根据不同攻击类型调整,从而实现自适应的流信息统计和高效的大流检测,然后该文并没有实现具体的应用。

Early DDoS Detection^[10]基于SDN控制器,利用目的IP的熵变化检测DDoS攻击,其优点是速度快(前500个数据分组即可检测到DDoS攻击)、开销低,可以避免SDN控制器本身遭到DoS攻击。然而,单纯基于目的IP地址变化检测很容易被绕过。此外,没有解决针对SDN控制器本身DDoS攻击的问题,因为攻击者可以构造目的IP分布很均匀、源IP也很均匀的陌生数据分组,造成Packet-In消息拥堵。

DaMask^[11]分析了SDN和云环境下DDoS检测面临的新困难,包括云内部的网络攻击难以检测、防御机制必须根据网络拓扑动态更新而调整、SDN处理分组的效率较低等,并针对性地提出了DaMask解决方案,该方案使用基于Chow-Liu算法的自动特征选择,提出了一种快速低开销的DDoS攻击检测算法以减少通信开销,并针对数据集漂移问题,提出基于图模型的局部更新和全局更新这2种方案。但该方案以SDN交换机作为最小监控粒度,多租户共存的情况下无法区分不同用户的流量,存在误报率高的问题。

FlowGuard2^[12]是一种针对控制器饱和(DoS)攻击的解决方案,该方案不同于上述工作,其保护的对象并非SDN网络内的主机或虚

拟机节点,而是SDN控制器本身。通过主动流规则分析和分组迁移技术实现基于Packet-In的DoS攻击检测。

Bohatei^[13]是一种弹性可扩展的DDoS防御系统,为了避免控制器和交换机之间的性能瓶颈,Bohatei将防御虚拟机部署在关键网络位置,而非SDN控制器上。Bohatei可以根据攻击流量的大小和位置灵活部署虚拟机或增加防御虚拟机的资源,具有很强的扩展性,最大可以处理高达500 Gbit/s的DDoS攻击。但是Bohatei需要大量虚拟机资源作为支撑。

表2是对现有基于SDN的DDoS攻击检测研究的小结,由表2可知,基于SDN的DDoS攻击检测目前研究较为充分,主要利用SDN集中控制的特点,既方便攻击信息的收集(通过流表统计),也便于在检测到DDoS攻击后及时响应(安装静态流表),存在的主要问题有以下3个:1) 流信息的收集对SDN控制器的开销较高;2) 对DDoS攻击的检测精度和速度问题,由于要实现实时的检测和响应,需要及时快速的检测,同时保证一定的精度;3) 对控制器本身DoS攻击的防护问题,以避免控制器单点失效。

2.3 IDS/IPS

传统的入侵检测系统(IDS, intrusion detection system)和入侵防护系统(IPS, intrusion prevention system)无法适应云的动态特性,虚拟机迁移、弹性IP分配以及虚拟之间流量的检测都给传统IDS/IPS带来了挑战。SDN的出现为这些问题的解决带来了曙光,基于SDN集中控制的优点,IDS可以灵活地选择检测的网络范围,且在虚拟机迁移时及时探测并动态调整策略,此外,检测到入侵行为后可以通过安装静态流表快速对入侵行

表 2

基于SDN的DDoS检测应用

DDoS 检测方案	优点	缺点
Lightweight-DDoS	高效、自适应(根据网络变化调整收集流信息的交换机)	查询周期固定,缺乏协同,无法定位攻击者
Behavior Threat Detection	自适应(根据流类型和攻击类型调整带宽和时间阈值)	缺乏具体实现
Early DDoS Detection	速度快、开销低	特征简单,容易被绕过
DaMask	快速低开销	粒度粗,误报率高
FlowGuard2	防止控制器受DoS攻击	不能检测普通DDoS攻击
Bohatei	弹性可扩展	需要消耗大量计算资源

为响应,如阻止、隔离、或导流到蜜罐深入分析。

NICE^[14]基于 OpenFlow 可编程 API 实现了一个分布式的监控架构,以阻止包含漏洞的虚拟机被利用。首先通过攻击图模型和虚拟机画像对虚拟机的威胁性进行评估,然后基于 SDN 将可疑虚拟机的流量导流到 Snort 进行检测,在检测出攻击行为后通过 VLAN 隔离攻击虚拟机或者直接丢弃可疑流量,从而确保攻击行为不影响其他正常虚拟机。NICE 专注于解决云内部的入侵行为,对于外部攻击缺乏检测能力。

SnortFlow^[15]把 Snort 和 OpenFlow 结合起来做 IDS,然后并没有具体的分组分析方法,实验也只是测试了 Snort agent 在 dom0 和 domU 内的 CPU 开销以及分组检测率,也没有基于 OpenFlow 实现任何响应措施。相比之下,SDNIPS^[16]除了充分利用了 SDN 的集中控制特性实现了基于 SDN 的入侵防护系统,还在检测到入侵行为后通过控制器安装静态流表对恶意流量进行导流或阻拦。此外,通过模块化设计,在 SDN 控制器上实现了入侵检测、分析和防护的统一,且可以根据具体威胁类型增加新的模块。

SDNIPS、NICE 和 SnortFlow 的不足之处在于都使用 Snort 作为入侵检测引擎,而 Snort 主要是基于规则匹配的入侵检测引擎,其对于高级威胁,如 Web 漏洞利用、APT 攻击等缺乏检测能力。此外,对于多个 agent、多个 IDS 之间的协同检测问题也没有考虑。

Security with SDN^[17]提供了一个基于 SDN 的 NIDS 应用案例,通过数据层采集相关安全数据(报文头),然后转发给 NIDS 进一步分析以检测入侵行为,再通过控制器写入相关流策略到交换机,实现入侵检测和防护。实验分析得出以下 2 个结论:1) 控制命令的消耗(新流的处理)要大于报文负载(流规则匹配);2) 交换机硬件支持对性能的影响很大(软件流规则表效率低)。Security with SDN 没有考虑在高速网络中 IDS 的扩展能力。

ScalableIDS^[18]考虑 IDS 的扩展性问题,并提出了一种基于 SDN 的负载均衡方法,其主要思路是通过部署在 SDN 控制器上的中心 NIDS 节点周期性地收集各个 IDS 的负载情况,并在流量分类

的基础上将不同类型的流量导流到不同的 IDS 节点进行检测。类似的工作还有 Multiple IDS^[19], Multiple IDS 针对大规模网络(如云数据中心网络)多个 IDS 协同检测的问题进行了研究,并提出了基于 SDN 的多个 IDS 负载均衡的解决方案。Multiple IDS 基于相同的攻击往往具有类似的路由路径的观察结果将流分组,使用 SDN 将不同分组导流到不同的 IDS 检测,提高检测效率,同时结合分组的流速度调整分组的大小,以达到 IDS 的负载均衡。

考虑控制器性能瓶颈的问题, Sampling for IDS^[20]提出了一种 SDN 采样方法以减小 IDS 对整体网络性能的影响。Sampling for IDS 提出捕获失败率这个指标,并将恶意流量采样问题转化为使恶意流量捕获失败率最小的最优化问题,并且根据 IDS 检测的结果更新实时采样率,实现以最佳的速率采样到尽可能多的恶意流量。不足之处在于求解最佳采样率的过程中引入了一定的延时,对实时性有一定影响。

VMI-IDS^[21]通过结合 SDN 和 VMI 技术,实现了细粒度的网络流量检测。VMI-IDS 首先基于 SDN 控制虚拟化平台的网络流量,转发给 NIDS 进行检测;其次在 NIDS 检测到入侵行为后,使用 VMI 对产生异常流量的虚拟机进行进一步的检测,分析发出异常流量的具体进程。

类似的工作还有 SDNSec^[22],结合了基于 SDN 的 NIDS 和基于 VMI 的 HIDS,构建完整云环境下软件定义的 IDS,从而实现主机层和网络层安全的有效统一。SDNSec 总结了基于 VMI 和基于 SDN 的云安全方案各自的优点和不足,且只提出了框架结构,但并没有进行实现也没有完整的性能评估,各个模块之间的接口具体如何实现也没有清晰的描述。

表 3 是对当前基于 SDN 的 IDS/IPS 的小结,由表 3 可知,SDN 大大降低了云环境中 IDS/IPS 实现的成本,基于 SDN 的集中控制特性很容易实现对入侵的响应。不过,对于云内大规模流量的检测还需要考虑 IDS/IPS 的扩展能力和对高级威胁的检测能力,因此基于 SDN 的分布式 IDS、基于采样的 IDS 以及与深度分组检测技术、蜜罐技术的结合还有待进一步研究。

表 3 基于 SDN 的 IDS/IPS 应用

IDS/IPS 应用	类型	优点	缺点
NICE	IPS	检测防护一体	无法检测外部攻击
SnortFlow	IDS	较低的开销	无响应措施
SDNIPS	IPS	检测防护一体	无法检测高级威胁
Security with SDN	IPS	检测防护一体	无扩展能力
ScalableIDS	IDS	基于流量分类的负载均衡	无响应措施
Multiple IDS	IDS	基于流分组的负载均衡	无响应措施
Sampling for IDS	IDS	引入采样降低开销	采样率计算有一定的延时
VMI-IDS	IDS	细粒度，可检测到进程级	无响应措施
SDNSec	IDS	结合了 VMI 和 IDS 各自的优点	无具体实现

2.4 异常检测

异常检测的范围比较广，既包含 DDoS 攻击，也包含端口扫描、蠕虫、P2P 僵尸网络等，DDoS 攻击检测方面的研究在上文已经进行了讨论，这里主要讨论其他类型的异常检测。相比于 IDS，异常检测更倾向于检测未知威胁；相比于 IPS，异常检测更倾向于检测而非防御，在检测到异常后，有可能需要进一步的分析以判断具体的威胁类型。异常检测对于云内未知攻击的及时发现具有重要作用。

Revisting^[23]基于 OpenFlow 协议，将 4 种异常检测算法在 OpenFlow 交换机和 NOX 控制器上进行了实现，包括 TRW-CB、Rate-Limiting、最大熵检测算法和 NETAD。具体实现都是基于 Packet-In 消息和 OpenFlow 提取的流特征进行分析，检测到异常后再下发流表进行导流或过滤。实验结果表明，基于 OpenFlow 实现具有跟传统方法相似的检测精度，但更加灵活，而且可以以线速度进行异常检测，非常适合于在线部署。该文还通过实验验证了在终端（家庭 SOHO 路由）进行异常检测比在核心网络进行异常检测具有更高的准确性。不过，由于数据分组需要通过 Packet-In 消息交给控制器进行决策，会影响控制器的性能，也会受到控制器和交换机之间通信瓶颈的限制。

OpenFlow-sFlow^[24]使用 sFlow^[25]对网络流进行采样，并基于熵变化的网络异常检测方法分析采样的网络流计数，实现对端口扫描、DDoS 和蠕虫的检测。在检测到异常后基于 OpenFlow 实

现对异常的响应，通过安装流表丢弃异常数据分组，且安装流表的优先级高于普通数据转发流表，通过白名单排除误检的合法服务流。不足之处在于，OpenFlow-sFlow 流表统计字段过于简单（源 IP，目的 IP，源端口，目的端口四元组），无法检测高级威胁（如 P2P 僵尸主机）。

类似的工作还有 ADM^[26]，只是 ADM 的出发点是为了克服传统的 RTBH 异常处理方法的缺陷，更侧重于检测出异常之后的异常处理过程。传统的 RTBH 异常处理方法以目标 IP 为粒度，将受害主机的流量导到空端口，从而实现网络隔离，然而正常流量也会无法到达。ADM 则利用 OpenFlow 的流表匹配功能，将受害主机的流量导流到检测引擎检测后再安装细粒度（如特定的源 IP、端口、协议等）的流表，以阻止可疑流量，正常流量不受影响。

Security with SDN^[16]提出了一种通过周期性查询 OpenFlow 交换机的流表统计信息检测网络扫描和 DDoS 攻击行为的方法，实验结果表明，仅是基于流统计信息的异常检测方法的性能开销很小，可以用于真实场景中。存在的问题是查询流统计信息的时间间隔固定，容易被攻击者探测到，而且流数量增加的时候效率不够高。

OpenTAD^[27]首先在控制器上在线获取 OpenFlow 交换机的流表信息，并构造整个网络的流量矩阵与样本熵矩阵进行组合，然后采用主成分分析方法检测异常流量。相比传统网络中利用主成分分析方法分别单独处理离线的流量矩阵或样本熵矩阵的方法，在线流量异常检测方法实

现和处理方式简单、有效，异常流量能够得到快速隔离。

为了提高异常检测的效率，OpenWatch^[28]提出了一种预测算法，该算法比较当前流计数器跟历史数据的线性变化率，并根据结果动态调整流量监控的粒度，变化越大，监控粒度越小，反之则越大。监控粒度的调整通过控制分组转发流的匹配字段来实现。OpenWatch 能够通过对变化的感知来实现低开销的异常检测，然而，由于流统计字段的类型有限，对于应用层的异常流量（如 JavaScript 等）难以检测。

类似的工作还有 AdaptCount^[29]，也是通过当前流计数器跟历史数据的变化动态调整流量监控的粒度，不同之处在于 AdaptCount 的调整粒度是监控的子网大小，而 OpenWatch 则支持端口、子网、IP 等粒度，相对更加丰富，实现也更加复杂。

表 4 是对当前基于 SDN 的异常检测应用的小结，异常检测的范围主要是端口扫描、DDoS、蠕虫和突发流量，异常检测系统的性能开销、准确率和时间是基于 SDN 开发时要着重考虑的因素，当前的研究应用到云环境中还需要克服检测时间长、性能开销高以及自适应机制不够灵活等不足。

2.5 网络监控

不同于前几类安全应用，SDN 在网络监控领域的应用不仅可以用于安全相关的应用，也可以用于其他流量工程，如路径规划、QoS 等，这对提高云环境的运转效率、增强云计算的可用性和可靠性都有重要作用。

CloudWatcher^[30]在网络操作系统（SDN 控制器）的上层实现网络安全设备和安全策略的统一管理，并通过路由规则生成器将安全策略转换为流规则，并将流规则推送到网络设备上（SDN 交

换机）。此外，还定义了脚本语言 SLI，方便设备管理和规则生成，如添加安全设备或编写安全策略。CloudWatcher 重点放在基于 OpenFlow 的网络流路由策略查找上，即流满足流规则后如何将数据分组转发给相应的安全设备，以及如何在多个安全设备之间实现最佳路由，提出了 4 种算法，并通过实验测试了不同算法在不同安全设备路由选择上的性能。

在 CloudWatcher 的基础上，NetSecVisor^[31]进一步提出了网络安全虚拟化（NSV，network security virtualization）的概念，以实现网络安全设备的集中管理和虚拟化，其组成主要包括注册安全服务和策略的脚本语言、安全设备最佳路由决策算法、处理不同安全事件的安全响应策略。但是，CloudWatcher 和 NetSecVisor 都没有考虑多个安全设备同时工作时的路由选择和规划问题，也没有考虑不同路径的负载和延迟。

FlowSense^[32]利用 OpenFlow 提供的 Packet-In 和 FlowRemoved 消息进行网络链路利用率的评估。FlowRemoved 消息是指流表项在失效前发送给控制器的消息，携带有流的统计信息，之后流就会从交换机中移除。FlowSense 具有较低的开销，但准确性不够，只在存在大量短流时评估比较准确，无法准确评估突发性流量时的链路利用率。

Payless^[33]虽然也是基于 OpenFlow 的 Flow-Removed 消息实现对流统计信息的获取，但 Payless 使用了自适应的网络数据统计（采样）算法，通过监控临近 2 个周期内流数据的变化设定流的 timeout 时间，变化越大，timeout 越小（相当于采样率越高），变化越小，timeout 越大（相当于采样率越低）。开销相比由控制器使用固定间

表 4 基于 SDN 的异常检测应用

异常检测应用	检测类型	优点	缺点
Revisting	TCP Flood、UDP flood	灵活、快速、准确	存在性能瓶颈
OpenFlow-sFlow	端口扫描、DDoS 和蠕虫	优先级和白名单机制	无法检测高级威胁
ADM	端口扫描、僵尸主机	网络隔离、细粒度	缺乏动态调整功能
Security with SDN	网络扫描和 DDoS 攻击	实现简单	查询间隔固定、效率不够高
OpenTAD	阿尔法异常、闪拥异常、DDoS，入口/出口移动异常	简单、有效，异常流量得到快速隔离	提取的流量特征较为单一
OpenWatch, AdaptCount	闪拥异常，小波异常	动态调整流量监控的粒度，低开销	无法检测应用层的异常流量

隔查询流统计信息大大降低；准确性相比完全依赖于 FlowRemoved 消息统计的 FlowSense 更高，这得益于根据网络状态动态调整 timeout 的时间，可以更及时地收集变化的网络数据。不过，仅根据变化率调整 timeout 很可能遭到攻击者恶意构造的请求欺骗，导致流表 timeout 过小和流频繁的变化，甚至导致 SDN 控制器 DoS。

OrchSec^[34] 结合网络监控工具 sFlow 和 OpenFlow，在其上部部署一层 Orchestrator 作为协调者和应用开发层，以充分利用 sFlow 的流量采样功能（效率）和 SDN 的流量控制功能（深度分析、响应），同时保证应用开发的灵活性和监控的整体性。不过，OrchSec 将不同协议的流量转发给不同的过滤设备进行采样处理，需要额外的硬件投资。

OpenTM^[35] 通过周期性轮询流统计信息构造流量矩阵（TM，trcoffice matrix），周期的选择对结果影响较大，周期越小越准确，但开销较大；周期太大，则开销小但准确性降低。OpenTM 局限于生成流量矩阵供离线使用，没有统计分组丢失率和延迟的功能。

相比之下，OpenNetMon^[36] 则较全面，同样基于 OpenFlow 协议，OpenNetMon 可以测量网络的吞吐量、分组丢失率、延时等性能指标，为流量工程（QoS、网络安全监控等）提供数据。OpenNetMon 使用轮询交换机流计数的方式测量吞吐量，且使用自适应速率进行轮询，新流产生时轮询速率加快，静态环境下轮询速率减慢，在降低开销的同时保证准确性。此外，OpenNetMon

使用固定路径上的头尾交换机计数差测量分组丢失率，使用主动注入的方式测量延时，并使用 VLAN 传送延时探测分组，以避免交换机控制面板调度时的不稳定性。

OF-ping^[37] 通过构造带时间戳和端口信息的 PACKET_OUT 消息测量 2 个 SDN 交换机之间的延迟。PACKET_OUT 消息是指控制器主动发送给交换机的消息，可通过该消息将数据分组注入 SDN 网络中。整个探测消息只有 ICMP 分组的 19%，但延迟探测的效果跟 ping 结果几乎一致。该方法假设控制器上下行消耗的时间一致，实际中可能并非如此。此外，通过 STATISTICS_REQUEST（流查询请求）测试上下行时间时没有考虑交换机处理该请求的延迟，所以最后结果有微小的偏差。

SDM^[38] 提出了软件定义监控的概念，其主要目标是高速网络中快速的应用层协议分析，非常适用于云环境。SDM 结合了 NetFlow^[39] 的灵活性和硬件加速器的处理性能，可以处理高达 100 Gbit/s 的流量。但是 SDM 依赖于硬件加速器，无法实现分布式集中控制和响应，此外，硬件资源的限制导致模块化应用开发受限。

表 5 总结了现有基于 SDN 的网络监控应用的研究，从中可以发现网络监控应用主要包括测量网络吞吐量、分组丢失率、延时、利用率等流量工程，以及应用层协议分析、安全应用开发等安全类应用。当前研究面临的主要问题是如何提高性能的同时保证准确性，性能提高方面可以采用硬件加速和采样技术，准确性提高则依赖于具体

表 5

基于 SDN 的网络监控应用

网络监控应用	具体用途	优点	缺点
CloudWatcher, NetSecVisor	安全设备和安全策略的统一管理	丰富的路由选择算法，便捷的脚本语言	没有考虑不同路径的负载和延迟，以及多个设备同时工作时的路由选择和规划
FlowSense	网络链路利用率的评估	较低的开销	准确性不够
Payless	抽象监控层，向上提供 API 供应用开发	自适应的轮询、准确性高	采样调整算法容易被欺骗
OrchSec	抽象的监控层，供安全应用开发	结合 sFlow、高效	需要额外的硬件
OpenTM	生成流量矩阵供离线使用	周期性轮询流统计信息	没有统计分组丢失率和延迟的功能
OpenNetMon	测量网络的吞吐量、分组丢失率、延时	自适应的轮询、开销低、准确性高	侧重于性能监控，缺乏安全监控
OF-ping	测量延时	开销低	结果有偏差
SDM	应用层协议分析	结合硬件加速器可以处理高达 100 Gbit/s 的流量	缺乏集中控制，硬件资源限制了应用开发

的监控算法。另外,应该更加充分利用 SDN 的灵活性和集中控制特性实现尽可能多的监控应用。

2.6 其他安全应用

基于 SDN 的其他安全应用还有很多,本文只选择几个作为代表说明 SDN 在云安全领域应用的广泛前景。

现有蜜网面临指纹探测的问题以及粗粒度数据控制的问题, HoneyMix^[40]采用 SDN 解决这些问题。基于 SDN 的集中控制特性,将同一攻击通过 SDN 镜像或 NAT 交给不同的蜜罐同时处理。通过安装特定流表,拦截指纹探测数据分组,修改后再返回攻击者。此外, HoneyMix 还基于行为评分选择最具攻击效果的蜜罐连接返回给攻击者,从而方便研究其行为,也避免攻击影响蜜网内其他蜜罐,同时采用快照恢复被攻击蜜罐的初始状态。但 HoneyMix 的开销较大。一是指纹探测检测需要深度报文检测;二是同时多个蜜罐处理攻击时,要对所有连接进行选择,尤其是有状态的连接(TCP)还需要维持状态表,开销大。

MTD^[41]使用 SDN 控制器实现主机 IP 的动态变化,从而实现移动目标防御,通过 SDN 控制器为每次连接随机分配不一样的虚拟 IP, SDN 交换机实现主机虚拟 IP 和实际 IP 的转换,而主机使用的实际 IP 不变,增加攻击者持续攻击的难度。不过, MTD 无法应对基于 rDNS 和基于主机名字的扫描和攻击,此外, IP 转换需要创建大量流表,会增加交换机开销和网络延迟。

Ransomware Mitigation^[42]基于前期对勒索软件特征的研究,提出使用 SDN 匹配 DNS 黑名单的方式实现实时的勒索软件与控制者通信的拦截,通过 DNS 流量镜像的方式在实现检测的同时降低合法用户的 DNS 响应延迟,相比硬件防火墙,对网络的影响更小,规则更新也更灵活快速。

AMQ^[43]基于 SDN 的网络感知能力进行动态调整,以检测潜在的恶意软件数据流并进行隔离,从而克服传统的恶意软件隔离系统只能处理静态流的缺点。AMQ 包括 Bot Hunter 和 Threat Responder 这 2 个安全模块, Bot Hunter 实时监控网络状态以检测受感染的僵尸主机, Threat Responder 在控制器的辅助下根据 Bot Hunter 检测的结果实现受感染主机的隔离。

这里只列举 SDN 在安全领域应用的一部分,更多基于 SDN 的云安全应用有待研究人员和相关企业的进一步开发。

3 SDN 安全应用要解决的关键问题

综合上述调查和讨论,本节对 SDN 安全应用需要解决的 2 个基础问题进行总结:一是如何克服控制器多样性带来的 SDN 应用不兼容问题;二是如何在使用 SDN 提供安全服务时降低对性能的影响。针对这 2 个问题,本文对当前这方面的研究进行了归纳和探讨。

3.1 统一的开发环境

为了解决控制器多样性带来的 SDN 安全应用不兼容问题,急需一个统一的开发环境进行 SDN 应用的开发。统一的开发环境一方面需要屏蔽不同控制器底层实现的差异,另一方面需要一个开发框架,提供安全应用开发的基本组件,并在最上层提供高级编程语言以方便应用开发。

FlowVisor^[44]是一种网络 hypervisor,试图将网络资源虚拟化,包括带宽、拓扑、流量、设备 CPU、流表等, FlowVisor 是一个特殊的控制器,充当交换机和多个其他控制器之间的代理。通过抽象底层资源,利用切片灵活定义网络资源,将不同切片进行有效隔离,然后交给不同的控制器管理。然而, FlowVisor 没有提供给租户完全自主管理其网络资源的能力,且网络地址空间无法在不同租户间共享。

跟 FlowVisor 类似的还有 OpenVirtex^[45], OpenVirtex 将 SDN 网络本身进行虚拟化,在提供有效隔离的基础上,通过开放的北向 OpenFlow 接口允许租户使用自己的网络操作系统(NOS, network operation system)来控制其网络,非常适合跟云服务结合,向用户提供从计算、存储到网络的完整服务和自主管理。

前两者侧重于底层结构的差异,对控制器之上的应用层开发关注不够。由于不同控制器提供的接口各异, SDN 安全应用很难同时运行于不同的控制器之上。为此,需要统一的开发平台来实现应用的快速适配和统一开发。

FRESCO^[46]采用模块化设计,使用脚本语言定义模块,方便开发 SDN 应用,其主要组成包括:

1) FRESKO 资源管理器, 用于管理交换机的流, 调度流表空间; 2) 控制器中的 FRESKO SEK 模块, 用于协调流规则, 保证安全流的优先级, 实现流的认证; 3) 应用层的 FRESKO 开发环境, 包括脚本到模块转换、数据库管理和事件管理等。在 FRESKO 开发平台基础上开发了几个实例应用, 包括扫描检测、BOT 检测和 P2P 恶意软件检测, 其核心是将传统安全设备检测结果通知给 FRESKO 应用, 然后通过 SDN 控制器安装特定流表进行处理和响应。

NetsecVisor^[31]则在安全设备管理上更加具体, 针对安全设备的添减、安全策略的定义、安全服务的申请, 以及安全设备路由的规划进行了较详细的分析, 不过并没有解决多个安全设备的服务编排问题。比如, 用户同时申请串行的防火墙和并行的 IDS 服务, 流量先经过 IDS 再经过防火墙检, 还是先经过防火墙再经过 IDS 是一个优先级的问题, 因为经过防火墙过滤后的流量可能对 IDS 来说不够完整, 路径规划和导流时也需要考虑这类需求。

除了屏蔽 SDN 底层差异、统一管理安全设备, 基于 SDN 安全应用的开发还需要从效率和适应性角度通过编程语言来克服不同控制器的接口差异, 以加速应用的开发和部署。这方面的研究包括 Frenetic、Pyretic、NetCore、FML、Nettle、Procera 等^[47~52]。

Frenetic^[47]是一种类似于 SQL 的 SDN 编程语言, 将底层控制器控制指令进行抽象化 (runtime system), 开发者只需要关注顶层功能的设计, 流表操作由 runtime 进行查询及安装, runtime 具备

流表冲突检测 (一个模块的结果其他模块要用) 和更新功能 (同时更新所有交换机上某个分组对应的流表), 对于有重叠的规则, 通过为规则分配优先级克服冲突。

Pyretic^[48]类似于 JAVA 语言, 基于 Pyretic 编写的 SDN 应用可以无需修改并运行于不同控制器之上。NetCore^[49]提供了一种将分组转发策略用高级语言表示的方法, 可以用于安全策略到流表的转换。其他 SDN 相关的编程语言还包括 FML、Nettle、Procera^[50~52]等, 这类编程语言都是被动式的, 在事件触发后作为响应, 只适用于特定的应用场景。

表 6 是对当前 SDN 应用开发环境的一个小结, 由表 6 可知当前已经在各方面分别展开研究, 但还没有一个完整的解决方案出现。

3.2 流采样技术

SDN 集中控制的特点虽然便于网络集中管理和安全监控, 但也面临控制器的性能瓶颈问题, 针对该问题, 一是研究多控制器的协同, 包括层次化的控制器模型和分布式的控制器模型等; 二是对网络流进行高效的采样, 从而减少控制器需要处理的数据量, 加快安全分析和响应的速度, 同时保证采样能够充分代表原始数据, 或包含足够多感兴趣的数据, 以保证安全分析的准确性。

传统的网络数据分组采样技术 (如 sFlow) 对流量统计特性的估计不如流准确, 而且采样间隔固定, 缺乏灵活性。基于流的采样越来越得到重视, 由于 SDN 交换机维持流表的详细统计信息, 可以把流表统计信息的查询等同于流采样。

表 6

SDN 应用开发环境

开发环境	类型	特点
FlowVisor	网络 hypervisor	网络资源虚拟化, 灵活划分, 有效隔离, 网络地址空间无法在不同租户间共享
OpenVirtex	网络 hypervisor	租户可自主管理其网络资源
FRESKO	开发框架	使用脚本语言开发模块
NETSECVISOR	开发框架	高效的路径规划
Frenetic	SDN 编程语言	类似于 SQL, 通过 runtime 将底层控制器控制指令抽象化
Pyretic	SDN 编程语言	类似于 JAVA, 应用可以无需修改运行于不同的控制器上
NetCore	SDN 编程语言	实现了高级语言编写的安全策略到底层流表的转换
FML, Nettle, Procera	SDN 编程语言	被动式, 只适用于特定的应用场景

采样方式可分为基于时间的采样、基于报文数的采样以及随机采样。基于时间的采样可以转换为 OpenFlow 协议中流表查询的时间间隔, 或流的 timeout 大小。基于报文数的采样又可分为 one-in-k 和 m-in-k, 即离散采样和连续采样。随机采样是指对每个报文生成一个随机数, 当该数大于预设的阈值时对其采样, 否则不采样。

Flexam^[53]的做法是扩展 OpenFlow 的 action, 添加采样行为, 通过参数可以指定采用随机采样或连续采样, 采样既可发送给控制器也可转发给单独的机器处理。Flexam 新定义的采样行为有 6 个参数: destination 指定采样发往的位置, scheme 指定采样的方法, 当为随机采样时, p 表示采样的概率; 当为连续采样时, 用 q 、 m 、 k 参数表示跳过最开始的 q 个包, 每 m 个包采取前 k 个。由于无需增加新的计数器也无需新的硬件支持, 因此实现简单, 但对交换机本身的性能有一定影响。

通常流表的管理采用白名单和黑名单机制。白名单代表正常网络服务, 黑名单代表恶意数据流, 剩下的大部分流都是通配流, OpenFlow 提供通配流机制, 对于既不存在于白名单, 也不存在于黑名单中间的流进行统一操作, 默认行为为转发, 然而通配流中其实可能包含隐藏的恶意数据分组。但是通配流数量巨大, 每一个数据分组进行检测的代价太高。Wildcard Sampling^[54]针对通配流的采样问题, 修改了 OpenFlow 协议, 增加采样请求消息, 将采样作为一种 action 由交换机完成, 对交换机本身的性能有一定要求。实验表明, 只有在 $\frac{1}{100}$ 以下的采样速率, 才不会对整体网络性能有较大影响。

OpenSample^[55]使用 sFlow 上报的信息, 寻找 2 个相同的 TCP 流采样, 通过其序列号和时间戳信息估计分组速率, 从而为流量工程提供依据, 可用于路径规划和大流检测。相比其他方式 (轮询和最大似然估计) 速度快。OpenSample 的理论依据是在一定条件下 2 个相同采样出现的概率接近 1, 然而其适用范围主要是大流检测, 采样数据分组对高级威胁 (如 P2P、Web 入侵等) 缺乏代表性。

此外, OpenNetMon^[36]、OpenWatch^[28] 和 AdaptCount^[29]均从流查询的角度提出了自适应查询的算法, 通过比较当前流计数器跟历史数据的

变化动态调整流查询的间隔, 变化越大, 查询间隔越小, 反之则越大。一定程度上减少了开销, 然而仅根据变化率调整粒度较粗, 对流量工程较为有用, 对安全应用则有必要进一步根据安全数据的变化来设计更为精确的调整算法。

Adaptive Sampling^[56]为 ONOS 设计了一种自适应流采样机制, 将流分类为立即流、短流、中流和长流, 对不同类型的流设置不同的采样间隔, 流越长采样间隔越大, 从而保证总体采样数据的代表性。Adaptive Sampling 相比固定间隔的采样, 采样数减少约 50%, 但并没有对采样后的数据进行分析。

Sample&Pick^[57]设计了一种两步采样法以降低采样速率和交换机内存开销, 第一步通过简单的采样规则采样以发现可疑流, 第二步插入计数规则, 对可疑流进行分组计数。最后, 控制器轮询该计数, 交给入侵检测算法进行检测。存在的问题是第一步只能发现简单可疑流, 对复杂恶意流无检测能力。

考虑 IDS 处理能力的限制, Sampling for IDS^[20]提出捕获失败率这个指标, 并通过数据建模将恶意流量采样问题转化为在 IDS 处理能力的约束下使恶意流量捕获失败率最小的最优化问题。捕获失败率是指多个 SDN 交换机节点上采样得到的数据分组都不包含恶意流数据的概率。由于该最优化问题依赖于已知恶意流的速率, 因此需要根据 IDS 检测的结果实时更新该速率, 从而更新实时采样率。在 ODL 上的实验验证了该采样算法在入侵检测时比固定采样的方式更为快速。

表 7 总结了以上采样方法的主要实现途径和缺点, 当前研究主要存在以下 2 个问题: 一是部分流采样的实现依赖于硬件性能, 二是采样缺乏对于不同流量的自适应性。现有自适应流采样算法的动态调整策略粒度不够细, 采样率计算方法过于复杂, 有待于进一步改进。

4 SDCSec 云安全服务模型

云计算的安全服务是用户最关心的问题, 然而, 如何将网络安全作为一种服务交付给用户一直是云计算企业比较头疼的问题。过去数年出现了很多这方面的概念, 包括安全即服务 (SECaaS^[58])、

表 7

SDN 流采样方法

采样方法	实现途径	缺点
FleXam	扩展 OpenFlow 协议, 添加采样行为	对交换机性能有一定影响
Wildcard Sampling	增加采样请求消息和 action	对交换机本身的性能有一定要求
OpenSample	结合 sFlow 采样	限于分组速率估计
OpenNetMon, OpenWatch, AdaptCount	自适应查询, 根据变化率调整流表查询的粒度	对于流量工程有用, 对于安全应用缺乏适应性
Adaptive Sampling	基于流分类设置不同的采样间隔	缺乏对于采样后的数据分析
Sample&Pick	两步采样, 降低采样速率和交换机的内存开销	第一步采样过于简单, 可能漏掉很多攻击数据分组
Sampling for IDS	将恶意流量采样问题转化为使恶意流量捕获失败率最小的最优问题	依赖于已知恶意流的速率, 采样率更新有一定的延迟

软件定义安全 (SDSec^[59])、网络安全虚拟化^[31]等。这些服务模型大多关注某个特定的安全服务类型, 如 IDS、访问控制等。此外, 多数安全服务模型只关注安全本身, 而不注意安全服务模型跟业务模型的有机结合, 在实现安全的同时缺乏对性能和管理成本的综合考虑。鉴于此, 本文提出一种基于 SDN 的云安全服务模型 SDCSec (software defined cloud security)。

如图 2 所示, SDCSec 云安全服务模型将云安全服务作为一种资源供用户使用, 用户可在自己的客户端像管理计算资源和网络资源一样管理

自己的云安全资源。用户在申请云安全服务后, 云管理平台调用安全资源管理组件将用户的安全服务申请进行编排, 以克服不同安全应用的冲突。云安全应用基于 SDN 控制器对网络资源集中控制和编程实现, 包括防火墙、DDoS 检测、IDS/IPS 等。此外, SDCSec 还通过监控虚拟机的迁移行为实现安全策略的同步迁移。SDCSec 的核心是安全策略的管理, 包括为了克服安全应用之间冲突而设置的安全策略优先级划分和策略冲突检测模块, 为了保证虚拟机在迁移后安全服务依然可用的安全策略迁移模块, 以及为了将安全策略转

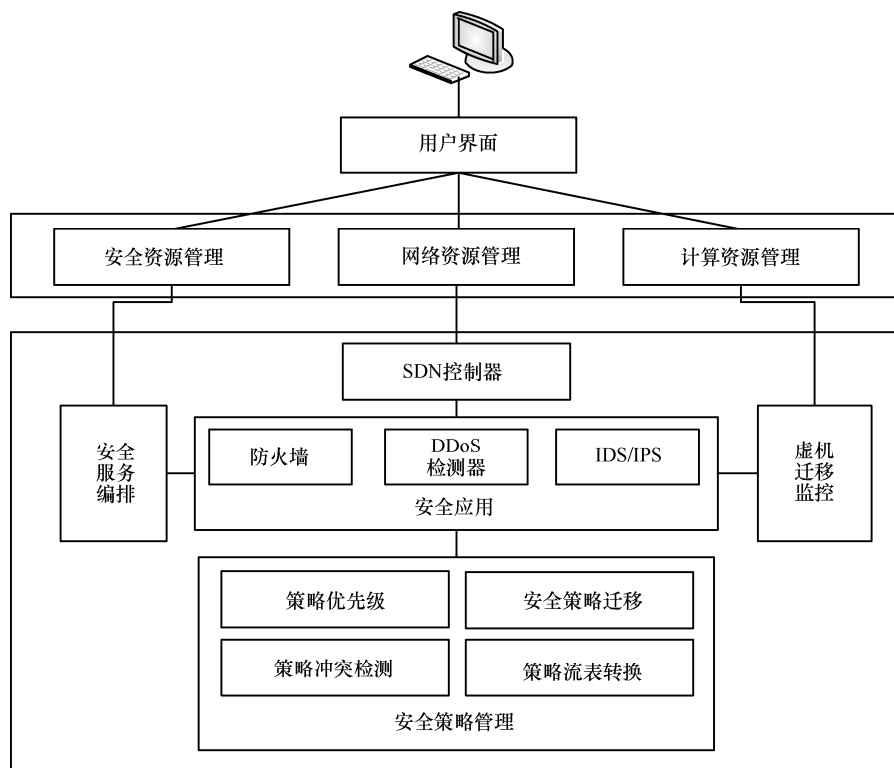


图 2 SDCSec 云安全服务模型

换为 SDN 交换机流表项,从而控制网络行为的策略流表转换模块。

需要指出的是,SDCSec 是一个初步的云安全服务模型,具体每一个模块的实现还需要深入研究,如 SDN 控制器与当前云平台网络管理模块的集成、虚拟迁移状态的监控,以及策略冲突检测等。最终目的是将安全资源跟计算资源、网络资源、存储资源一起以服务的方式交付给用户,实现完整的云服务解决方案。

5 未来展望

基于 SDN 的云安全应用受到了广泛关注,当前也有很多这方面的研究,但依然存在不少有待研究人员解决的问题。基于上文对现有安全应用的讨论,笔者提出一些有待改进和探索的研究点,供相关研究人员参考。

1) 基于 SDN 的新型云安全应用

SDN 技术跟云计算有很多共同点,如灵活调度、集中管理、可编程等,两者的结合在提高云数据中心的管理效率、发挥集中可控的优势方面具有重大应用前景。虽然目前已有许多此方面的研究,但多集中于将传统安全机制用 SDN 进行实现,以提高效率。针对云计算特有的安全问题,如多租户隔离问题、云存储访问控制问题、云内攻击问题等有待进一步深入研究。此外,如何跟现有的云安全机制进行结合以提供全面的安全即服务功能,是推广落实这类应用的必经之路。

2) SDN 安全策略的转换和迁移问题

基于 SDN 的云安全应用大多通过将安全策略转换成相应的流表来实现,而由于流表定义的限制,并非所有安全策略都能一对一地转换成流表项。以防火墙为例,传统防火墙可以通过端口范围来定义防火墙策略,而 OpenFlow 一条流表项只能匹配一个端口号,如果用多条流表项实现必然会大大增加交换机的流数目,不仅占用宝贵的流空间,而且由于匹配项增多,会降低数据分组处理的速度。

此外,考虑云计算环境的特殊性,虚拟机经常在不同主机之间进行调度和迁移,基于 SDN 的安全应用也需要考虑策略迁移的问题。一种可能的方法是当虚拟机迁移时 SDN 控制器将安全应

用对应的安全策略同步迁移到目标位置新的控制器上,不过这要求 SDN 控制器能够检测到虚拟机迁移事件,以及不同控制器之间安全应用的协同和通信机制。具体的实现有待研究人员的进一步探索。

3) SDN 安全设备管理和安全服务编排

基于 SDN/NFV 的云安全应用可能利用现有的硬件安全设备,因此需要加强安全设备的管理。安全设备的管理一方面需要统一的接口方便控制器和安全设备之间的通信,另一方面要有合理的安全通信和访问控制机制,防止未授权的访问。此外,多个安全服务的编排对实际应用来说也非常关键。一方面,需要根据安全设备的类型分配优先级,串联设备的优先级一般要高于并联设备;另一方面,要克服不同安全服务之间策略的冲突问题。虽然 NETSECVISOR 支持安全设备的注册和安全服务的申请,其并没有考虑控制器和安全设备之间的通信问题,也没有考虑存在多个安全设备时安全服务的编排问题。

4) SDN 自适应流采样算法与分组采样的结合

为了克服 SDN 控制器跟交换机之间的性能瓶颈,当前有很多工作基于流采样来提高基于 SDN 安全应用的效率,第4节已经进行了讨论。然而,当前的采样方法一类是基于轮询方式,无法应对云环境中动态变化的网络流量,尤其是突发式流量,且容易被攻击者探测和绕过。另一类采样方法是基于流计数变化的动态调整算法,将流计数(如报文数、IP 变化等)统计信息作为网络状态变化的象征,动态调整流查询间隔或采样率,虽然一定程度上能够在提高性能的同时检测网络状态的变化,但对于网络应用层的攻击缺适应性,如脚本攻击、应用层 DDoS 攻击等。此类攻击并不会在流表中产生显著的变化,需要通过深度分组检测报文内容。因此,有必要结合分组采样和流采样技术以达到最佳检测效果。

5) SDN 控制器本身安全问题

SDN 控制器是软件定义网络的核心,由于其集中控制特性,其本身的安全性对网络的整体安全有重要的影响,部署于其上的安全应用也依赖于控制器的安全和可靠性。当前已有许多这方面的研究,如控制器漏洞修复、控制器组件隔离、

分布式多控制器结构等, 文献[60]对其进行了很好的总结和展望, 这里不再赘述。

6 相关研究

虽然已有一些关于 SDN 安全方面的综述文章, 但多从基于 SDN 的网络安全和 SDN 本身的安全这 2 个方面展开, 没有具体到某一类安全问题进行详细的分析和论述。前者相关的综述有文献[61,62], 后者相关的综述有文献[63,64]。

文献[61]主要对 SDN 本身的安全问题和现有解决思路进行探讨, 包括 SDN 安全控制器的开发、控制器可组合安全模块库的开发和部署、控制器 DoS/DDoS 攻击防御方法、流规则的合法性和一致性检测、北向接口的安全性和应用程序安全性。类似的综述文章还有文献[62]。SDN 本身的安全问题不是本文论述的重点。

文献[63]对基于 SDN 的防火墙、网络监控和分析方面的工作进行了归纳总结, 其侧重于对安全模型的分类, 缺少对现有工作不足的分析。文献[64]则通过分析 SDN 带来的新特性及其在信息安全方面的应用展开论述, 是基于特征的分类总结, 相比之下, 本文的工作则是基于应用的分类总结。

对比可知, 本文的特色是面向 SDN 在云安全这一个具体领域的现有工作进行论述, 在对这些工作进行基于应用分类的基础上, 针对云的特点和安全需求对现有研究的优缺点进行总结, 并提出了未来研究的一些方向。

7 结束语

SDN 的出现极大地方便了网络安全应用的开发, 其动态、灵活的特点跟云计算能够很好地结合, 为云安全提供了新的解决思路。本文总结了目前基于 SDN 云安全应用的研究, 对其进行了分类, 对每一类安全应用顺着其发展脉络分析了相关工作的优缺点, 并进行了总结。另外, 针对当前云安全服务模型不够完善、不能跟计算和网络服务有机整合的问题, 提出了一种基于 SDN 的云安全服务模型 SDCSec。最后, 针对当前研究的不足, 对基于 SDN 云安全应用的未来发展方向做了一些展望, 供相关研究人员参考。

参考文献:

- [1] 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.
- [2] KOERNER M, KAO O. Oftables: a distributed packet filter[C]//2014 6th International Conference on Communication Systems and Networks (COMSNETS). 2014.
- [3] JIA X, WANG J K H. Distributed firewall for P2P network in data center[C]//ICCE-China Workshop (ICCE-China), IEEE. 2013: 15-19.
- [4] SUH M, PARK S H, LEE B, et al. Building firewall over the software-defined network controller[C]//The 16th International Conference on Advanced Communication Technology, IEEE. 2014: 744-748.
- [5] PORRAS P, SHIN S, YEGNESWARAN V, et al. A security enforcement kernel for OpenFlow networks[C]//The First Workshop on Hot Topics in Software Defined Networks, ACM. 2012: 121-126.
- [6] HU H, HAN W, AHN G J, et al. FlowGuard: building robust firewalls for software-defined networks[C]//The Third Workshop on Hot Topics in Software Defined Networking. 2014: 97-102.
- [7] 王鹏, 王江, 焦虹阳, 等. 一种基于 OpenFlow 的 SDN 访问控制策略实时冲突检测与解决方法[J]. 计算机学报, 2015, 38(4): 872-883.
WANG J, WANG J, JIAO H Y, et al. A method of openflow-based real-time conflict detection and resolution for SDN access control policies[J]. Chinese Journal of Computers, 2015, 38(4): 872-883.
- [8] BRAGA R, MOTA E, PASSITO A. Lightweight DDoS flooding attack detection using NOX/OpenFlow[C]//2010 IEEE 35th Conference on Local Computer Networks (LCN). 2010: 408-415.
- [9] KRISHNAN R, KRISHNASWAMY D, MCDYSAN D. Behavioral security threat detection strategies for data center switches and routers[C]//2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW). 2014: 82-87.
- [10] MOUSAVI S M, ST-HILAIRE M. Early detection of DDoS attacks against SDN controllers[C]//2015 International Conference on Computing, Networking and Communications (ICNC). 2015: 77-81.
- [11] WANG B, ZHENG Y, LOU W, et al. DDoS attack protection in the era of cloud computing and software-defined networking[J]. Computer Networks, 2015, 81: 308-319.
- [12] WANG H, XU L, GU G. FloodGuard: a DoS attack prevention extension in software-defined networks[C]//2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. 2015: 239-250.
- [13] FAYAZ S K, TOBIOKA Y, SEKAR V, et al. Bohatei: flexible and elastic DDoS defense[C]//24th Usenix Security Symposium (USENIX Security 15). 2015: 817-832.
- [14] CHUNG C J, KHATKAR P, XING T, et al. NICE: network intrusion detection and countermeasure selection in virtual network systems[J]. IEEE Transactions on Dependable and Secure Computing, 2013, 10(4): 198-211.
- [15] XING T, HUANG D, XU L, et al. Snortflow: a OpenFlow-based intrusion prevention system in cloud environment[C]//2013 Second GENI Research and Educational Experiment Workshop (GREE). 2013: 89-92.

- [16] XING T, XIONG Z, HUANG D, et al. SDNIPS: enabling soft-ware-defined networking based intrusion prevention system in clouds[C]//The 10th International Conference on Network and Service Management (CNSM) and Workshop. 2014: 308-311.
- [17] YOON C, PARK T, LEE S, et al. Enabling security functions with SDN: a feasibility study[J]. *Computer Networks*, 2015, 85: 19-35.
- [18] JEONG C, HA T, NARANTUYA J, et al. Scalable network intrusion detection on virtual SDN environment[C]//2014 IEEE 3rd International Conference on Cloud Networking (CloudNet). 2014: 264-265.
- [19] HA T, YOON S, RISDIANTO A C, et al. Suspicious flow for warding for multiple intrusion detection systems on soft-ware-defined networks[J]. *IEEE Network*, 2016, 30(6): 22-27.
- [20] HA T, KIM S, AN N, et al. Suspicious traffic sampling for intrusion detection in software-defined networks[J]. *Computer Networks*, 2016.
- [21] 崔竞松, 郭迟, 陈龙, 等. 创建软件定义网络中的进程级纵深防御体系结构[J]. *软件学报*, 2014, 25(10): 2251-2265.
- [22] CUI J S, GUO C, CHEN L, et al. Establishing process-level defense-in-depth framework for software defined networks[J]. *Journal of Software*, 2014, 25(10): 2251-2265.
- [22] SHIN Y, SON K, PARK H. SDN-based security in virtualized environments for cloud computing[J]. *Life Science Journal*, 2014, 11(7): 642-647.
- [23] MEHDI S A, KHALID J, KHAYAM S A. Revisiting traffic anomaly detection using software defined networking[C]//The International Workshop on Recent Advances in Intrusion Detection. 2011: 161-180.
- [24] GIOTIS K, ARGYROPOULOS C, ANDROULIDAKIS G, et al. Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments[J]. *Computer Networks*, 2014, 62: 122-136.
- [25] JASINSKA E. sFlow—I can feel your traffic[C]//23C3: 23rd Chaos Communication Congress. 2006.
- [26] GIOTIS K, ANDROULIDAKIS G, MAGLARIS V. Leveraging SDN for efficient anomaly detection and mitigation on legacy networks[C]//2014 Third European Workshop on Software Defined Networks. 2014: 85-90.
- [27] 左青云, 陈鸣, 王秀磊, 等. 一种基于SDN的在线流量异常检测方法[J]. *西安电子科技大学学报*, 2015, 42(1): 155-160.
- [28] ZUO Q, CHEN M, WANG X L, et al. Online traffic anomaly detection method for SDN[J]. *Journal of Xidian University*, 2015, 42(1): 155-160.
- [28] ZHANG Y. An adaptive flow counting method for anomaly detection in SDN[C]//The 9th ACM Conference on Emerging Networking Experiments and Technologies. 2013: 25-30.
- [29] GARG G, GARG R. Security of networks using efficient adaptive flow counting for anomaly detection in SDN[M]//Artificial Intelligence and Evolutionary Computations in Engineering Systems. 2016: 667-674.
- [30] SHIN S, GU G. CloudWatcher: network security monitoring using OpenFlow in dynamic cloud networks (or: how to provide security monitoring as a service in clouds?) [C]//2012 20th IEEE International Conference on Network Protocols (ICNP). 2012: 1-6.
- [31] SHIN S, WANG H, GU G. A first step toward network security virtualization: from concept to prototype[J]. *IEEE Transactions on Information Forensics and Security*, 2015, 10(10): 2236-2249.
- [32] ZHANG Y, SINGH V, WANG Y, et al. FlowSense: light-weight networking sensing with OpenFlow: U.S. Patent 8,918,502[P]. 2014-12-23.
- [33] CHOWDHURY S R, BARI M F, AHMED R, et al. Payless: a low cost network monitoring framework for software defined net-works[C]//2014 IEEE Network Operations and Management Symposium (NOMS). 2014: 1-9.
- [34] ZAALOUK A, KHONDOKER R, MARX R, et al. Orchsec: an orchestrator-based architecture for enhancing network-security using network monitoring and SDN control functions[C]//2014 IEEE Network Operations and Management Symposium (NOMS). 2014: 1-9.
- [35] TOOTOONCHIAN A, GHOBADI M, GANJALI Y. OpenTM: traffic matrix estimator for OpenFlow networks[C]//The International Conference on Passive and Active Network Measurement. 2010: 201-210.
- [36] VAN-ADRICHEM N L M, DOERR C, KUIPERS F A. Opennetmon: network monitoring in OpenFlow software-defined networks[C]//2014 IEEE Network Operations and Management Symposium (NOMS). 2014: 1-8.
- [37] PHEMIUS K, BOUET M. Monitoring latency with Open-Flow[C]//The 9th International Conference on Network and Service Management (CNSM 2013). 2013: 122-125.
- [38] KEKELY L, KUČERA J, PUŠ V, et al. Software defined monitoring of application protocols[J]. *IEEE Transactions on Computers*, 2016, 65(2): 615-626.
- [39] KAPOOR R D, CALABRESE A D, DUBEY R K, et al. Sample netflow for network traffic data collection: U.S. Patent 7,193, 968[P]. 2007-3-20.
- [40] HAN W, ZHAO Z, DOUPÉ A, et al. HoneyMix: toward SDN-based Intelligent Honeynet[C]//The 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization. 2016: 1-6.
- [41] JAFARIAN J H, AL-SHAER E, DUAN Q. OpenFlow random host mutation: transparent moving target defense using software defined networking[C]//The First Workshop on Hot Topics in Software Defined Networks. 2012: 127-132.
- [42] CABAJ K, MAZURCZYK W. Using software-defined networking for ransom ware mitigation: the case of cryptoWall[J]. *arXiv preprint arXiv:1608.06673*, 2016.
- [43] MCBRIDE M, COHN M, DESHPANDE S, et al. SDN security considerations in the data center[J]. *Open Networking Foundation ONF Solution Brief*, 2013.
- [44] SHERWOOD R, GIBB G, YAP K K, et al. Flowvisor: a network virtualization layer[J]. *OpenFlow Switch Consortium*, 2009: 1-13.
- [45] AL-SHABIBI A, DE-LEENHEER M, GEROLA M, et al. Open-VirteX: make your virtual SDNs programmable[C]//The 3rd Workshop on Hot Topics in Software Defined Networking. 2014: 25-30.
- [46] SHIN S, PORRAS P A, YEGNESWARAN V, et al. FRESKO: modular composable security services for software-defined networks[C]//NDSS. 2013.
- [47] FOSTER N, GUHA A, REITBLATT M, et al. Languages for software-defined networks[J]. *IEEE Communications Magazine*, 2013, 51(2): 128-134.
- [48] REICH J, MONSANTO C, FOSTER N, et al. Modular sdn programming with pyretic[J]. *Technical Reprint of USENIX*, 2013.

- [49] MONSANTO C, FOSTER N, HARRISON R, et al. A compiler and run-time system for network programming languages[C]//ACM SIGPLAN Notices. 2012: 217-230.
- [50] HINRICHS T L, GUDE N S, CASADO M, et al. Practical declarative network management[C]//The 1st ACM Workshop on Research on Enterprise Networking. 2009: 1-10.
- [51] VOELLMY A, HUDAK P. Nettle: Taking the sting out of programming network routers[C]//The International Symposium on Practical Aspects of Declarative Languages. 2011: 235-249.
- [52] VOELLMY A, KIM H, FEAMSTER N. Protera: a language for high-level reactive network control[C]//The first Workshop on Hot Topics in Software Defined Networks. 2012: 43-48.
- [53] SHIRALI-SHAHREZA S, GANJALI Y. FleXam: flexible sampling extension for monitoring and security applications in Open-Flow[C]//The second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. 2013: 167-168.
- [54] WETTE P, KARL H. Which flows are hiding behind my wildcard rule? adding packet sampling to OpenFlow[J]. ACM SIGCOMM Computer Communication Review, 2013, 43(4): 541-542.
- [55] SUH J, KWON T T, DIXON C, et al. OpenSample: a low-latency, sampling-based measurement platform for commodity SDN[C]//2014 IEEE 34th International Conference on Distributed Computing Systems (ICDCS). 2014: 228-237.
- [56] TAESANG C. Adaptive flow monitoring[C]//Open Networking Summit. 2016.
- [57] AFEK Y, BREMLER-BARR A, LANDAU F S, et al. Sampling and large flow detection in SDN[C]//ACM SIGCOMM Computer Communication Review. 2015: 345-346.
- [58] HUSSAIN M, ABDULSALAM H. SECaaS: security as a service for cloud-based applications[C]//The Second Kuwait Conference on e-Services and e-Systems. 2011: 8.
- [59] AL-AYYOUB M, JARARWEH Y, BENKHELIFA E, et al. SDSecurity: a software defined security experimental framework[C]//2015 IEEE International Conference on Communication Workshop (ICCW). 2015: 1871-1876.
- [60] AHMAD I, NAMAL S, YLIANTTILA M, et al. Security in software defined networks: a survey[J]. IEEE Communications Surveys & Tutorials, 2015, 17(4): 2317-2346.
- [61] 王蒙蒙, 刘建伟, 陈杰, 等. 软件定义网络: 安全模型、机制及研究进展[J]. 软件学报, 2016, 27(4): 969-992.
- WANG M M, LIU J W, CHEN J, et al. Software defined networking: security model, threats and mechanism[J]. Journal of Software, 2016, 27(4): 969-992.
- [62] COUGHLIN M. A survey of SDN security research[J]. University of Colorado Boulder, 2014.
- [63] FRANÇOIS J, DOLBERG L, FESTOR O, et al. Network security through software defined networking: a survey[C]//The Conference on Principles, Systems and Applications of IP Telecommunications. 2014: 6.
- [64] SHIN S, XU L, HONG S, et al. Enhancing network security through software defined networking (SDN)[C]//2016 25th International Conference on Computer Communication and Networks (ICCCN). 2016: 1-9.

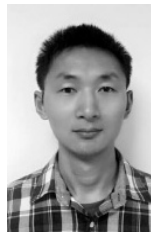
作者简介:



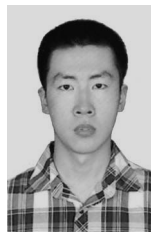
施江勇 (1990-), 男, 湖南岳阳人, 国防科技大学博士生, 主要研究方向为虚拟化和 SDN 安全。



杨岳湘 (1965-), 男, 湖南岳阳人, 国防科技大学研究员、博士生导师, 主要研究方向为网络安全、云安全和移动安全。



李文华 (1983-), 男, 河南周口人, 国防科技大学工程师, 主要研究方向为云计算和数据中心管理。



王森 (1995-), 男, 陕西咸阳人, 国防科技大学本科生, 主要研究方向为信息化建模与处理。