计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200300119

软件定义网络安全问题研究综述

董　仕
周口师范学院计算机科学与技术学院　河南 周口 466001

摘　要　软件定义网络是一种新型的网络体系结构,其通过 OpenFlow 技术来实现网络控制面与数据面的分离,从而达到对网络流量的灵活控制,目前已成为下一代互联网的研究热点。随着 SDN 的发展及广泛应用,其安全问题已成为亟待解决的重要研究内容。近年来,国内外学者在 SDN 安全研究领域取得了一定的成果,文中针对 SDN 的 3 层架构分别对各层所面临的安全问题及其解决方案进行了系统总结。首先给出了 SDN 的定义和 3 层框架;接着依次总结了数据层、控制层和应用层的安全问题以及相应的解决方案;然后分析并讨论了传统网络安全与 SDN 安全的异同;最后对软件定义网络安全问题未来研究可能面临的挑战进行了展望。

关键词:软件定义网络;数据层;控制层;应用层;OpenFlow
中图法分类号　TP391

Survey on Software Defined Networks Security

DONG Shi
School of Computer Science and Technology,Zhoukou Normal University,Zhoukou,Henan 466001,China

Abstract　Software-defined networks(SDN) is a new network architecture,which enables separate network control plane from data planes through OpenFlow technology,thus the network traffic can be flexible controlled. SDN has become a hot topic in the next generation of Internet. With the development and wide application of SDN,its security problem has become an important research topic and some achievements have been made by the domestic and foreign scholars in recent years. Based on three-layer architecture of SDN,the security problems and solutions of each layer are summarized. Firstly,the definition and three frameworks of SDN are presented;then security issues and corresponding solutions are outlined under the data layer,the control layer and application layer;in next,the security of similarities and differences between traditional network and SDN are discussed;and finally,the research challenges in future are prposed.

Keywords　Software defined networks,Data plane,Control plane,Application plane,OpenFlow

软件定义网络(Software Defined Networks,SDN)是一种新型网络创新架构,最初由斯坦福大学研究团队发起,其通过 OpenFlow[1] 技术将网络设备控制面与数据面分离,并利用网络控制与转发解耦合构建开放可编程的网络体系结构。随着开放性 SDN 的发展及其广泛的部署和应用,SDN 将面临诸多新的安全问题。与传统的网络架构不同,SDN 作为一种创新架构,其特殊的分层特征将衍生出不同的攻击方法及安全问题,相应地也会产生很多安全防范措施。

本文着重对已有的 SDN 安全问题及对策进行综述研究。SDN 的安全问题是当前 SDN 研究领域中的一个重要课题。随着新的 SDN 应用的不断涌现以及研究人员对 SDN 架构研究的日益深入,近年来有关 SDN 安全问题的研究取得了大量的成果,其安全架构设计以及攻击防御等成为了判断一个 SDN 系统是否可以交付使用的重要依据。因此,针对该问题的深入研究对提高和保障 SDN 软件产品的质量具有重要意义。

本文第 1 节介绍了 SDN 的概念及系统架构;第 2 节总结了已有的 SDN 安全问题;第 3 节对已有 SDN 安全问题的解决方法进行了总结;第 4 节比较并总结了传统网络和 SDN 的安全问题;第 5 节对未来值得关注的研究方向进行了初步探讨;最后总结全文。

1　软件定义网络的概念及系统架构

狭义的 SDN 特指基于 OpenFlow 南向接口的网络,广义的 SDN 则指具备这种理念的所有网络。SDN 与传统网络的最大差别在于网络控制模式,其将底层网络分成控制层与数据层(转发层)。控制层采用集中式控制器管控不同的网络设备,让比特流在数据层顺利传输,这样网络更容易被控制与管理。控制器通过安全通道与 OpenFlow 交换机进行通信,下发流表与控制规则来决定流量的流向,以实现路由机制、封包分析、网络虚拟化等功能。可编程性作为 SDN 的一个重要特

到稿日期:2020-03-20　返修日期:2020-08-30
基金项目:河南省科技攻关项目(192102210125)
This work was supported by the Key Science and Technology Program of Henan Province,China(192102210125).
通信作者:董仕(njbsok@163.com)

征，为开发者提供了诸多便利条件。然而，其作用范围一直局限于控制层，直到 P4 编程语言[2-3]出现，才打破了硬件设备对数据转发平面的限制，使“软件定义”延伸到硬件层面，同时也标志着 SDN 进入了 SDN2.0 时代。P2 可以通过编程控制数据包的解析和转发流程，使 SDN 快速实现可编程化。SDN 针对不同的使用需求，建立了服务层级协议，为使用者构建应有的存取服务保障。SDN 的三层架构如图 1 所示。

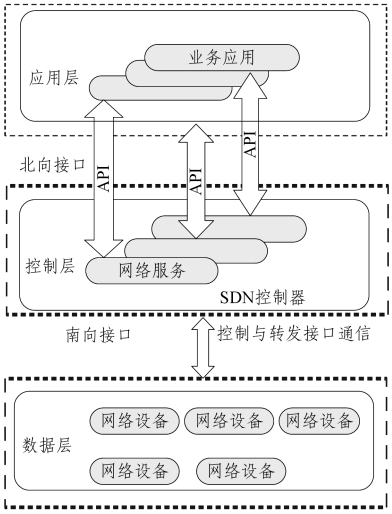

图 1 软件定义网络 3 层框架图

Fig. 1 Software defined networks architecture

尽管 OpenFlow 是 SDN 控制平面和数据平面之间的多种通信协议之一，但实际上，OpenFlow 因其良好的灵活性、规范性已被看作 SDN 通信协议事实上的标准。该标准类似于 TCP/IP 协议，是互联网的通信标准[4]，目前大部分安全系统的研究都是基于 OpenFlow 的。OpenFlow 协议是控制器与 SDN 交换机进行通信的桥梁，传统网络中由交换机或路由器完成的报文转发功能，现在由控制器和交换机共同完成。协议规定网络设备中需要维护一个流表，且所有的流量需要通过流表进行报文处理。而流表自身的生成、维护及规则的下发完全由控制器来实现。OpenFlow 协议主要通过对不同类型的信息进行处理来实现控制器和交换机之间的路由控制和交换。目前 OpenFlow 协议主要支持 3 种消息类型，分别是 controller-to-switch，symmetric（对称型消息）以及 asynchronous（异步消息类型）。如图 1[5]所示，基于 OpenFlow 的 3 层架构分别由应用层、控制层和数据层构成。下文将详细介绍基于 OpenFlow 的 3 层架构。

(1)SDN 应用层。该层包含提供管理及云端虚拟化等服务，主要为用户提供 SLA、QoE、监控、负载均衡、拓扑发现、安全与防火墙等网络服务功能，这些服务最终都以应用程序的方式表现，通过北向接口[6]与 SDN 控制层进行数据交互。

(2)SDN 控制层。该层以远端控制器为主，并搭配 SDN 控制软件，处于 SDN 中间层，能通过北向接口和南向接口分别与上层的应用层和下层的数据层进行互通。采用集中式管理架构，依靠编程方式实现对 SDN 交换机的集中控制。一个 SDN 中可以有多个控制器，控制器之间可以是主从关系也可以是对等关系。通过 SDN 交换机下发流规则使每个控制器可以控制多台设备，每台设备也可以被多个控制器控制。

(3)SDN 数据层。交换机、路由器及网络芯片是数据层的核心元素，数据层主要负责网络数据包的转发功能，交换机和路由器中的端口和流表主要完成与上层控制器的数据通道的建立以及向控制层下达转发规则。当有新报文发送到 SDN 交换机后，首先匹配流表中的流表项，若匹配失败，则需要上报给上层控制层，等待控制层下达转发规则给数据层，规则一旦下发，报文将匹配流表项完成报文转发功能。

2 SDN 安全问题

本节将依据 SDN 分层结构，详细介绍每层所面临的安全问题。

2.1 应用层安全问题

SDN 采用可编程的方式可有效地管理网络设备资源，集中体现了以控制器为中心的管理模式，但这种模式会给各种恶意应用软件带来相应的攻击机会。尽管 OpenFlow 可以提供一些基于流的安全检测算法[7]，但其潜在的假设是在 SDN 中应用软件未受到恶意攻击或北向接口未受到破坏的前提条件下，这会对 SDN 起到一定的安全保护作用，但对 DDoS 攻击却束手无策。一旦 SDN 应用层的应用软件或北向接口受到一定的攻击或破坏，后果将不堪设想。这也是应用层非常关注的问题。针对 SDN 应用层安全问题，我们将从以下两个方面进行分析。

2.1.1 身份认证

SDN 应用层与控制层之间需要使用应用服务来进行数据交互，而保证应用服务的安全是整个应用层安全问题的关键。鉴于这些应用服务能任意使用网络资源，而应用软件自身没有安全防护措施，一旦软件受到安全威胁，就会导致应用层和控制层的安全受到威胁。文献[8]明确指出，目前在应用层和控制层之间还没有建立起有效的安全机制。如何构建一种有效的安全机制是目前面临的重要问题。身份认证作为一种计算机安全防护技术已经得到广泛使用，它能确定该应用软件对网络资源是否具有访问和使用的权限，进而利用访问策略防止假冒或有害的应用程序访问或使用网络资源，在保障应用软件合法利益的同时，还能保护整个网络不受破坏，因此应用软件的身份认证是构建应用层安全机制的一项重要解决方案。

2.1.2 应用程序安全性

SDN 中存在诸多应用服务，其中以控制器服务类的应用程序最为常见。SDN 控制器通过这些应用程序制定下发到数据层交换机的流规则，一旦这些应用程序隐含恶意代码或代码被恶意篡改，就会给控制器及整个 SDN 带来很大的破坏。Hartman 等[9]认为有 3 种应用程序可能会影响 SDN 的安全。第一种是为网络中的一些敏感数据提供特殊传输路径的应用程序；第二种是用于提供网络服务的应用程序，如访问控制或防火墙、内容检查或入侵检测服务等；第三种是将第一和第二种网络服务打包的应用程序，或一个应用程序请求另一个应用程序的实例作为虚拟的网络单元。这样，恶意应用程序就可以利用第二种应用程序的一个实例而绕过访问控

制。除此之外,嵌套访问控制的应用(如使用另一个应用实例的应用程序)将成为 SDN 的一个真正挑战。文献[10]提到,应用中的部分内容对 SDN 来说是已知的,而有些则是未知的。SDN 应用程序能直接与 SDN 控制器通信,但是对于 SDN 来说,应用程序的间接通信所采用数据报的特定格式是未知的。这样 SDN 应用就成为了一个未经授权的访问控制层的网关。

2.2　控制层安全问题

控制层作为决策中心很容易成为攻击目标,主要有以下几种安全危险。

2.2.1　来自应用的攻击

SDN 控制层的应用配置对控制层有着较大的安全威胁,控制层的安全一般受制于对应用配置的认证,被确认安全的应用将有权获取相应的应用访问权限[6,9],在应用程序获取有效资源之前,有必要对其进行分类和限制。由于不同的应用有不同的功能,因此必须对每个应用的安全需求进行分类,如一些负载均衡的应用需要统计报文字节数或报文数,而一些入侵检测应用则主要考察报文头的信息。不同的应用在访问同一资源时也应该有不同的优先级。目前,对于基于应用类型安全需求的分类还没有研究人员得出相应的研究成果。

2.2.2　DoS 和 DDoS 攻击

DoS 和 DDoS 攻击是 SDN 控制器最常见也是最危险的攻击方式,在 OpenFlow 中复杂的任务一般都放到控制层进行决策[11]。若对每个新流都配置相应的流策略,那么整个控制层将陷入瓶颈。文献[12]指出,在 10 Gbps 的高速网络环境下控制层不能处理大规模的新流量,因此控制层很容易被 DoS 和 DDoS 攻击。另外,目前控制层中的控制器多为单个,当网络流急增时,将因无法得到及时处理而造成一定的时延,时延的长短严重依赖于控制器的处理器的速度。控制器本身的局限性往往会导致单点失效。文献[13]指出,简单地利用多控制器无法处理单点失效问题。文献[14]指出,DoS 对 SDN 控制器的攻击是通过攻击者不断地发送带有随机报文头的 IP 报文,使控制器处于无响应的状态。Fonseca 等试图采用另一个控制器来解决该控制器目前的不良状态,但无论是 DoS 还是 DDoS 检测都会将另一个控制器看作 DoS 或者 DDoS 的攻击对象,因此采用多控制器机制并不能有效防止 DDoS 攻击,已经有相关文献证实了这一结论[10]。

2.2.3　分布式控制层的挑战

为了管理更多的网络设备,仅靠单个控制器是无法完成的,因此多控制器被提出并将网络分成了多个子域。但是,如果网络被分成多个基于软件定义的子网络,信息的汇聚以及每个子网将会有自己的私有规则,并将面临很大的挑战。文献[15]提出了应用层流量优化(ALTO)机制并将其配置在 SDN 应用层。ALTO 应用需要网络层信息(如拓扑和链路信息),这些来自控制层的信息主要用于对特殊应用进行优化。文献[16]提出了 MSPS 模型,该模型能为应用提供网络服务。然而,当应用经过多网络域时可能会碰到与认证相关的安全威胁。若没有 SLAS 想要获得第三方网络信息和资源,就需要进行身份认证和隐私保护。

2.2.4　南北向接口的安全

南向接口的攻击有 4 种类型,分别为交互攻击、窃听攻击、可用性攻击和 TCP 攻击。其中,窃听攻击通过窥探数据层和控制层之间的交换信息来获取关键信息并为其企图进行的破坏行动做准备;交互攻击则利用窃听攻击提供的关键信息对两个层之间的交换信息进行修改,以达到攻击的目的。文献[17]提出一种利用 ARP 中毒而导致客户端和 SDN 控制器流量被拦截的中间人攻击策略。可用性攻击也称为拒绝服务式攻击(DoS),是因大量的请求报文不能得到及时处理而引起的洪泛,从而导致网络策略无法实现。文献[18]中,攻击者通过计算探测包的延迟时间来推断 SDN 中的流规则并进行分类。一旦获取其反馈规则,攻击者就会发送大量的 Packet-in 报文,这样会造成控制器因负担过重而失效。

北向接口连接应用层与控制层,负责应用程序与控制器之间的信息交互。由于应用程序种类繁多且更新较快,因此北向接口针对应用程序的认证机制还不够健全和完善。相对于南向接口而言,北向接口在控制器和应用程序之间建立的信任关系还较脆弱,攻击者可利用其可编程性和开放性对某些重要资源进行访问。对于攻击者而言,其攻击门槛较低。目前,北向接口所面临的主要攻击包括数据泄露、消息篡改、非法访问、身份假冒、应用程序自身的漏洞以及不同应用程序在合作时引入的新漏洞等。

2.3　数据层安全问题

文献[19]指出,在 SDN 中数据层的安全与控制层密切相关。如果控制层被攻击,那么数据层也会受到很大的影响。若一台交换机不能从控制器获取转发信息,或者因控制层失效而不能连接,那么数据层也表现为离线状态[15,20],控制层和数据层之间的连接常被作为攻击对象。

攻击者一般选择网络中的某个节点作为攻击目标(如 OpenFlow 交换机及相关的主机)。通过探测安全漏洞和脆弱节点来尝试攻击这些目标,通常采用拒绝服务式攻击(DoS)来完成攻击任务。另外,攻击者会利用特定的协议服务来完成欺骗的拦截,阻止正常的流量传输,并引入被攻击者操控的新流量,同时在 OpenFlow 交换机中添加新的流表项。原始的 OpenFlow 通过定义传输层安全[21]和数据传输层安全[22]来保障控制层和路由器之间的安全通信,并使用 TLS 协议建立认证加密机制,然而 TLS 协议的复杂性又往往使其更易成为攻击的对象。文献[23]描述了中间人攻击在 Openflow 中发生的概率大于传统网络。中间人攻击经常利用拦截和篡改发布给交换机的转发规则,来控制数据包的转发,这已成为攻击 SDN 的常用手段。文献[24]指出,TLS 的使用不仅不能对 TCP 层进行保护,而且还会带来更多的攻击。SDN 能通过防火墙管理网络流量来保障数据层的安全,但当管理交换机和控制器之间的信息时,需要花费很多的时间来处理因 DDoS 攻击所造成的错误流信息。Dimitri 等[25]指出,要在 50ms 之内恢复 OpenFlow 网络是难以实现的。建立流规则而造成的时延可能会导致引入授权和认证、检查内容和流规则应用程序比较困难。文献[26]指出,对 SDN 数据层的攻击可分为 3

种类型:设备攻击、协议攻击和侧信道攻击。设备攻击指利用SDN的软件或硬件漏洞来危害SDN的数据平面的攻击。例如,攻击者可能将软件错误或转发设备的硬件特性(如TCAM存储器)等漏洞作为攻击目标。文献[27]提出了一种能推断出网络参数的基于有限流的推理攻击策略,实验结果表明其具有较高的准确率。协议攻击指利用转发设备中的网络协议漏洞(如BGP攻击)攻击数据层。侧信道攻击指攻击者通过分析转发设备的性能指标来推断网络的转发策略。例如,输入缓冲区可用于识别规则,通过分析数据包的处理时间,攻击者可以识别转发策略[28]。图2给出了数据层中OpenFlow交换机及其流表的结构。流表是由匹配字段、优先级、计数器、指令、超时时间、附属属性组成。SDN各层的主要安全问题如表1所列。

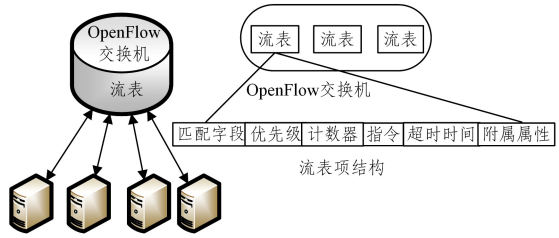

图2 SDN数据层的OpenFlow交换机及流表结构

Fig. 2 OpenFlow switch and flow table structure in SDN data plane

表1 SDN各层的主要安全问题

Table 1 Major security problems in SDN planes

安全问题	主要描述	SDN层		
		应用层	控制层	数据层
身份认证	应用软件本身没有实施安全防护措施,一旦软件受到安全威胁,就会导致应用层和控制层的安全受到危害	√		
应用程序的安全	应用程序的安全问题最常见的为控制器服务的应用程序。SDN控制器通过这些应用程序制定下发到数据层交换机的流规则,一旦这些应用程序隐含恶意代码或代码被恶意篡改,就会给控制器及整个SDN带来很大的破坏	√		
来自应用的攻击	控制层上的应用配置需要被认证,否则会被恶意应用程序所利用而成为攻击的对象		√	
规模带来的威胁	控制层在处理高速的数据流量时,不能及时得到处理。特别对于单控制器而言更是这样,很容易受到DDoS攻击		√	
DoS和DDoS攻击	由于控制层具有逻辑中心化的特点,容易造成单点失效,而DDoS攻击很擅长这样的攻击		√	
分布式控制层的挑战	分布式控制层将利用多控制器的思想并行处理流量,但这样会导致各个子网拥有自己的私有流规则,因而不便于管理,需要引入一定的授权和认证机制来确保相互网域之间的信息互信		√	
与控制层密切相关的问题	控制层与数据层之间的连接以及数据层的数据传输的安全性			√

3 SDN安全对策

SDN控制层和数据层分离后,在逻辑上体现了集中式的管理模式,并采用全局视图方式提供集中决策支持。这样一来,SDN凭借自身的结构特点更有利于网络管理、行为分析及快速敏捷的响应,并能利用可编程性的特征方便安全策略的订制[29]。SDN可以通过定期分析采集到的网络信息来识别威胁源并动态更新安全策略,也可以通过编程的方式修改应用服务软件来降低因错误的网络配置或网络攻击而带来的安全威胁。由于全局网络的可见性,防火墙或入侵检测系统(IDS)可根据安全策略部署在指定的通信节点。本节将分别对SDN不同层的安全对策进行讨论。

3.1 应用层安全对策

SDN应用层位于最上层,通过北向接口与控制器进行数据通信,里面包含大量的应用程序及服务。SDN控制器通过上述的集中式管理方式能快速便捷地部署新的安全服务及应用。为简化SDN应用开发,各种网络编程语言,如Frentic[30],Procera[31],Netcore[32]相继被提出。此外,为开发安全应用软件,设计出了FRESCO[33]脚本语言。该编程语言使开发人员能够在控制器和OpenFlow交换机上开发新应用。除上述编程语言外,还有众多的安全策略被提出,具体如下。

(1)访问控制。应用程序只有通过相应的访问控制权限才能使用或获取网络资源。PermOF[34]是一种利用OpenFlow控制器来控制应用层应用程序访问的权限系统。该系统通过一组权限和隔离机制来执行权限控制。该权限分为读、写和系统权限,该权限集被进一步划分成各个子类。读权限用于管理敏感的可用性信息;写权限使应用程序能对控制器或交换机上的状态进行修改;系统权限让应用管理程序具有本地系统资源访问的权利。传统的信任机制已无法满足SDN环境下的安全需求,为此文献[35]提出一种不同于PKI的分布式信任管理机制TRUFL,其利用并行计算方式在SDN环境下快速建立信任与验证机制。

(2)应用程序的安全防范。SDN应用程序必须能够不间断地观测网络的变化情况。文献[36]提出一种通过不断调试方法来适应网络条件不断变化的SDN应用程序。基于断言的调试和验证语言,开发者可通过高级的程序语句验证控制器中应用程序的动态属性。该调试方法有助于应用程序在部署之前找出Bug。由于VeriFlow[37]能在运行时提供流规则

检查机制,因此文献[36]采用带有增量数据结构的 VeriFlow 算法有效地验证了具有动态变化条件的属性。文献[38]提出 Flover 检测系统,其采用 OpenFlow 验证该流规则是否违背网络安全策略。Flover 是运行在控制器上的 OpenFlow 应用程序,主要用于检查控制器所产生的新的流规则与指定的属性是否保持一致。文献[39]提出了一种识别 OpenFlow 应用程序 Bug 的自动测试程序。另外,文献[40]和文献[41]分别提出了 NDB 和 OFRewind 两种框架。其中,NDB 框架是一种调试工具,它能为网络程序员找出错误根源,OFRewind 框架能对网络流量的异常情况进行记录和重放,这两种框架都能够定位到造成网络安全威胁的应用程序。

3.2　控制层安全对策

控制层的安全对策将从以下 4 个方面分别讨论:1)应用程序的攻击防范;2)DoS 或 DDoS 的攻击防范;3)可靠控制器放置;4)控制层和数据层之间的智能权衡。

3.2.1　应用程序的攻击防范

由于应用程序通过控制层来访问网络资源信息,若想确保控制层的安全,可以根据程序的功能对其进行权限划分以实现对恶意或错误应用程序的控制。文献[42]提出了一种针对应用程序攻击的防范方法,其中指出安全增强(SE)Floodlight 控制器是 Floodlight 控制器的扩展版本[43],是一种理想的、安全的 SDN 控制器。该控制器通过添加一个可编程的安全北向 API 接口中间件来完成特权分配。程序运行过程中引入了一种 OpenFlow 应用验证机制来对产生的流规则类模块进行完整性验证。为解决基于角色冲突的问题,SE-Floodlight 通过比较相互冲突的规则角色的权威性,来指定授权角色给 OpenFlow 应用程序。类似地,可以通过各种应用和安全的流规则来限制 PACKET_OUT 消息的产生。此外,SE-Floodlight 还引入了新的 OpenFlow 审计子系统,该系统能追踪所有发生的与安全相关的事件。

3.2.2　DoS 或 DDoS 攻击防范

分析流行为和 OpenFlow 交换机流量统计信息可以减少 DoS 或 DDoS 攻击的发生。由于交换机的统计信息在 OpenFlow 控制器中很容易获取,与其他控制器相比,基于 OpenFlow 的统计过程的开销较小,成本较低。在以 OpenFlow 为标准协议的 SDN 中,控制器与每个客户端单独连接的规则称为“微流”,这将导致路由器的数据流很大,控制器负荷过重。因此,很多研究都建议通过实现分布式控制层功能、提升处理能力和扩充控制器的内存来减小控制器的负载。此外,OpenFlow 支持使用通配符,能将客户端向服务器的请求集合保存为副本。通配符机制利用交换机支持通配符规则这一特征,除了能实现可扩展性控制器上的负载均衡,还能够利用通配符规则算法来完成目标流量分布,并根据流量分布情况自动调整由负载均衡策略所引起的变化。文献[44]对各种主动和被动 OpenFlow 控制器的可扩展性进行了比较分析。其中,主动控制器完成从接收流的首个报文开始到填充整个流表为止的整个过程,而被动控制器在流到达之前就已将流规则设置完成。文献[44]证明,主动控制器与被动控制器相比更具可扩展性。一个纯主动控制器需要事先知道所有流量,但实际上这是不可能的。因此,该文提出了一种混合控制器体系结构(HybridCtrl),其中所述的控制器被动配置路由,并积极预定路径、观测流量行为、增加控制器处理能力,在一定程度上使 SDN 控制器得到了扩展,使其能承担起更多的任务,同时减小了 DDoS 攻击的概率。McNettle[45] 是由多个 CPU 内核构成并支持控制算法的可扩展 SDN 控制器,对流到达率的状态变化有全局可见性。McNettle 可以用高级函数式编程语言进行扩展,与 NOX[37,46] 相比,McNettle 更具可扩展性,因为它可以扩展到 46 核,而 NOX 只能扩展到 10 核。文献[45]和文献[47]都采用多核并行处理器来提高控制器的处理性能,使其具有更高的可扩展性和可用性。文献[48]和文献[49]提出了分布式 SDN 控制层(DISCO),能为分布式、异构和覆盖网络提供控制层功能。DISCO 部署在 Floodlight 系统 OpenFlow 控制器的顶端[43],且使用了高级消息队列协议 (AMPQ)[50],它由两部分组成,即域内和域间。域内模块实现网络监控和管理,采用控制器来计算流的优先路径的优先级。这些模块在对重定向、停止网络通信问题的动态反应方面可起到关键性的作用。域间模块则管理由消息和代理构成的控制器间的通信。消息模块的功能是发现周边控制器并为这些临域提供控制通道。代理则使用这些通道与其他控制器交换网络信息。HyperFlow[51] 是一种物理上分布和逻辑上集中的基于事件可扩展的控制平台。它能让网络操作者部署多个控制器,从而最大限度地发挥控制器的可扩展性并缩短流量设置时间。Heller 等[52] 对控制器的可用性和状态分布进行了权衡,并建议把控制器放到最小化延迟点上,然后使用负载均衡算法来平衡控制器之间的负载。文献[53]对负载均衡技术进行了描述,它可以增加 SDN 控制层的可扩展性[54]。文献[55]提出了一种基于自组织映射(SOM)[56]的轻量级 DDoS 攻击检测方法。SOM 是一种将某个给定的人工神经网络的 n 维数据映射成二维映射图的方法。在执行过程中进行拓扑排序,采集有相似统计特征的数据并对其进行进一步处理。在此采用 SOM 机制寻找网络流量中的隐藏关系。该方法使用 3 个模块,即流量采集、特征提取和分类器。其中流量采集模块收集来自 OpenFlow 交换机在预定的时间间隔内所有流表的流量。特征提取模块针对 DDoS 攻击按照六元组的形式进行重要特征提取,并将其传递给分类器。所提取的特征包括流平均报文数、流平均字节数、流平均持续时间、双向流百分比、流长及不同的端口号。SOM 训练足够大的六元组流量样本(包含攻击流量和正常流量),不同的区域呈现不同的流量类型。这样,通过训练产生的分类器就能够准确地分析出正常流量和攻击流量。文献[57]提出了 DDoS 攻击度的概念,并采用改进的 K 近邻算法对 DDoS 攻击进行检测,该方法具有较高的识别正确率。文献[58]提出一种基于统计的防御机制,该机制包括名义阶段、准备阶段和主动阶段 3 个部分。其中,名义阶段和准备阶段都是为主动阶段提供数据采集、分析和处理服务,而主动阶段则是利用前两个阶段采集的数据进行决策判定,并对 DDoS 攻击进行相应的检测。文献[59]认为,大多数控制层的 DDoS 攻击比较容易检测,而对于数据层的 DDoS 攻击因其具有较强的隐蔽性而不易进行检测,为此该文提出基于因子分解机的多属性 DDoS 识别算法,该算法能有效检测 SDN 数据层中的 DDoS 攻击,

检测总体精度能达到 95.80%。

3.2.3 可靠控制器放置

文献[52]对控制器放置问题进行了讨论，指出控制器数量和拓扑位置是影响 SDN 可扩展性的两个主要因素。因此，控制器的最佳位置选取问题获得了学术界的广泛关注，一些最佳位置选取算法也相继被提出。其中，模拟退火(Simulated annealing，SA)算法作为一种通用的概率算法一直被认为是控制器的最佳位置选取算法[60-62]。为改善网络适应能力，并获取有效的控制器位置，文献[20]提出了基于图划分最小割边算法。文献[60]将控制器位置选取问题作为 NP 问题，以最大化 SDN 控制操作的可靠性，同时满足响应时间的要求。该文指出，控制器之间的状态同步和控制协调是必要的，并且可以实现如控制器分层等技术。为了获得最佳放置，控制路径损耗的预期百分比被用作可靠性的衡量标准，其中所述的控制路径损耗指由于网络故障而导致的控制路径破坏数。文献[61]提出的机制依靠最小化控制路径损耗的预期比例来优化网络。一些控制器位置选取算法采用真正的拓扑结构进行实验并讨论了算法的可靠性和延迟之间的平衡。文献[62]讨论了动态控制器配置问题(Dynamic Controller Provisioning Problem，DCPP)，并提出了一种动态部署多个控制器的框架，其中控制器的编号和位置根据网络动态进行调整。DCPP 是使用流量模式来尽量减小控制器间同步成本的线性规划(IGP)程序。此外，文献[62]还提出了控制器部署策略，其在 SDN 流量设置时间和通信开销之间增加了一个公平的权衡。文献[63]提出了一种基于帕累托最优的 SDN 控制器(Pareto_based Optimal Controller-placements，POCO)框架，并表明单个控制器可能足以满足时延和约束，但是为了满足网络弹性的要求，需要更多的控制器(至少需要所有节点的 20%是控制器)加入。文献[63]提出的框架对 SDN 控制器之间的时延和负载平衡进行了优化，并在延迟和故障恢复能力之间进行了权衡考量。

3.2.4 控制层和数据层之间的智能权衡

控制器和交换机之间的智能权衡可以增强控制器的可用性，为此很多模型相继被提出。其中，DevoFlow[64]是为了最小化控制层和数据层之间的相互作用而对 OpenFlow 进行修改后的模型。该结构尽管对模型进行了一些修改，将一部分控制权交还给了交换机，但是控制器的中央控制角色并没有改变。文献[64]认为，中央控制器对所有流的可见性会加重其可扩展性的负担。为解决该问题，DevoFlow 使用 OpenFlow 规则，且交换机可以根据自身情况自行做出路由决策，无需由控制器对每个流进行审查。在 DevoFlow 中，大多数微流都在数据层进行数据处理，但是操作者出于管理的目的可以管理或审查与之相关的任何流量。在 OpenFlow 中，一个控制器通过链路层发现协议来获取包括交换机端口连接信息及 OpenFlow 交换机的连接状态[65]。对于故障管理，该控制器可使用 LLDP 监测网络中的链接。在这种情况下，需要控制器参与所有的 LLDP 信息监测，这会导致控制器的可扩展性存在局限性。为解决这一问题，文献[66]提出了一种体系结构，为了降低从控制器到交换机的链路监视操作管理和维护能力，其在交换机上配备了通用消息的产生和处理功能，扩展 OpenFlow1.1 协议以支持监控功能。网络核心编程语言(NetCore)[32]是一种能表达 SDN 报文转发策略的高级解释性语言。它把报文处理的任务分给控制器和交换机，让部分交换机代替控制器对报文进行处理，以调解由控制器中的所有数据包处理决策所导致的延迟，从而有助于编译算法和运行系统的配合。

3.3 数据层安全对策

首先，对于数据层的安全对策，必须考虑到数据层所安装软件的安全性，因为恶意软件的安装会修改数据路径上的流规则，造成错误流规则的产生，导致数据层的安全性受到威胁。安全机制工作需要更加细致，特别是对于可能改变流规则的应用程序的身份验证和授权。FortNox[67]是一个使用 NOX OpenFlow 控制器实时检查流规则的系统平台，在改变流规则之前给这些 OpenFlow 应用程序授权。FortNOX 通过数字签名的方式提供基于角色的授权，并通过扩展的 NOX 的 OpenFlow 控制器来实现安全限制。使用一个规则冲突检测引擎，FortNOX 调整所有的 OpenFlow 规则冲突插入请求，从而有助于规则插入算法的分析。一旦安全应用程序插入流规则，在同一个 OpenFlow 网络中，FortNOX 就会限制其他与之产生冲突的流规则应用程序的插入。FlowChecker[68]是跨越不同平台的交换机和控制器一致性配置的验证工具，可以验证新协议设置的正确性，也可用作 OpenFlow 应用程序或主控制器的分析和验证，执行 OpenFlow 在运行时端到端的配置。VeriFlow[37]是一个用于查找 SDN 应用程序插入错误流规则的网络调试工具。

控制器与交换机之间的连通性是非常重要的，在两者之间进行通信时，冗余连接或快速链路恢复机制有助于保障控制器与交换机之间的连通性。实际上，OpenFlow 协议本身也能使用连接检测技术来检查交换机到控制器之间的连通情况，如果连接失败，则可以使交换机及时得到恢复，例如定期给控制器发送活动探测消息，一旦控制器发生故障，OpenFlow 协议能灵活地配置备用控制器以代替失效的控制器。文献[14]提出一种控制器备份或复制策略，当主控制器失效或出现错误时，交换机仍能正常工作。该策略主要是通过交换机周期性地给控制器发送探测消息，若控制器在一定时间间隔内没有响应，交换机则会假设控制器失效或出现故障。然后，OpenFlow 交换机会尝试通过握手的方式与备份控制器建立连接[14]，这样就能确保控制器正常工作。

SDN 数据层也存在一些常见类型的攻击，如扫描攻击、OpenFlow 泛洪攻击、交换机泄露攻击和 arp 攻击。根据以上攻击类型的特征，文献[69]提出一种新型的隐马尔可夫模型(HMM)来对网络状态进行量化，得到 SDN 安全态势评估值，依据观测特征值，该方法即可检测这 4 种攻击，并基于 HMM 对网络状态进行预测。

正确的网络规划也有助于增强 OpenFlow 交换机的弹性并最大化其与控制器的连接。一个经常与控制器连接的 OpenFlow 交换机不容易受到攻击，主要原因是其不会存储持续时间较长的异常流。文献[20]证明了交换机和控制器之间的路径长度与连接丢失率成正比。因此，文献[20]建议，在进行网络规划时，除了考虑控制器下的最佳的交换机数量，控制

器和交换机之间的路径长度还必须为最短。这不仅将提高延迟约束条件下系统的性能,而且能实现系统的快速恢复,提高安全应用程序内容的可用性,从而实现快速安全分析。Resonance[70]是一种基于流信息和实时警报的安全系统,其采用动态访问控制策略来保证 SDN 的安全。表 2 列出了 SDN 各层的主要安全解决方案。

表 2　SDN 各层的主要安全解决方案

Table 2　Major security solution in SDN planes

安全方案	描述	SDN 层		
		应用层	控制层	数据层
FRESCO[33]	采用可编程的脚本语言编写安全代码	√		
PermOF[34]	采用 OpenFlow 控制器来控制应用层应用程序访问的权限系统	√		
Assertion[36]	基于断言的调试和验证语言能使应用程序开发者通过高级的程序语句验证控制器中应用程序的动态属性	√		
Flover[38]	用于检查控制器所产生的新的流规则与指定的属性是否一致	√		
OFTesting[39]	识别 OpenFlow 应用程序 Bug 的自动测试程序	√	√	
SE-Floodlight[42]	引入了一个 OpenFlow 应用验证机制对产生流量规则类模块进行完整性验证		√	
HybridCtrl[44]	一种混合控制器架构,可提升控制器处理能力并减小 DDoS 攻击的可能性		√	
DISCO[48-49]	分布式 SDN 控制层(Distributed SDN Control Plane,DISCO),能给分布式、异构和覆盖网络提供控制层功能		√	
Ctl-Placement[52,60-61]	指出控制器数量和控制器的拓扑位置是影响 SDN 可扩展性的两个主要因素		√	
HyperFlow[51]	一种物理上分布和逻辑上集中的、基于事件的可扩展控制平台		√	
DDosDetction[55]	采用 SOM 的机制对 DDoS 进行攻击检测		√	
FortNOX[67]	使用 NOX OpenFlow 控制器实时检查流规则的系统平台			√
FlowChecker[68]	跨越不同平台的交换机和控制器一致性配置的验证工具			√
VeriFlow[37]	用于查找 SDN 应用程序插入错误流规则的网络调试工具			√
Resonance[70]	采用访问控制策略			√
CPRecovery[14]	采用控制器备份或复制策略			√

4　传统网络安全和 SDN 安全的比较

与传统网络相比,尽管 SDN 具有控制层和数据层解耦、可编程性和可扩展性等优势,但是也带来了新的安全挑战,如由于集中式管理模式而带来的单点失效问题,以及扁平化转发平面的数据处理逻辑使得数据层更容易受到配置错误的影响。另外,还有针对各层的 DoS 和 DDoS 攻击以及非法访问等。与传统网络相比,很多安全问题和策略确有不同之处,现将从以下几个方面进行讨论。

4.1　不同的网络架构

SDN 通过 OpenFlow 将网络设备控制面与数据面分离,从而实现对网络流量的灵活控制。但与传统的网络相比,也带来了新的安全问题。交换机与控制器的频繁通信容易遭受 DoS 攻击,这将极大地降低网络性能。不同网络之间的"分片"可能产生如规则冲突以及在 OpenFlow 交换机与控制器进行通信时数据加密对 CPU 资源消耗所产生的安全威胁等问题。在 OpenFlow 协议方面,包括 TLS 会话建立过程中的安全威胁以及控制器和 OpenFlow 交换机之间的授权协议的安全威胁等。而 SDN 控制器作为 SDN 的核心部分,将遭受控制器 OS 的安全威胁、控制器上运行的 APP/API 的安全威胁等。

4.2　安全设备部署点及交换机功能

(1)安全设备部署点。如图 3 所示,传统的网络安全模型和设备部署在网络边界,形成了一道安全的屏障来管控交换机及终端节点。安全保障主要来源于软硬件设备(防火墙等)。而 SDN 是基于流规则驱动的网络,控制器通过向 SDN 交换机下发流规则来完成数据包的转发,因此安全设备部署在交换机之间可以最大限度地保障在正常流规则下报文转发的安全性。SDN 安全设备配置及部署较为简单,可实现自动化,其实质是网络安全设备与控制器之间所形成的多对一关系及其实现的过程。

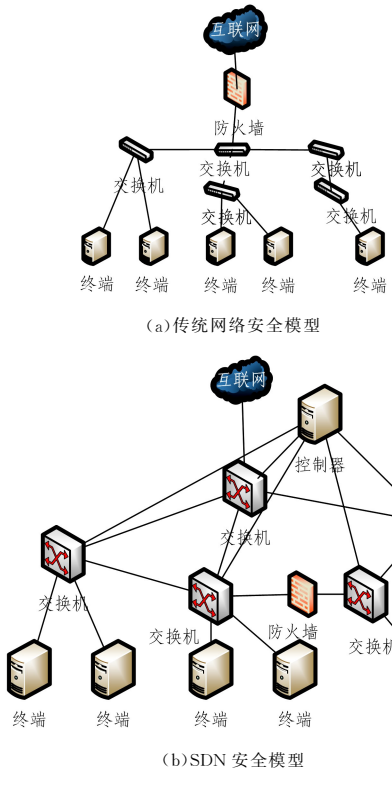

图 3　传统网络安全模型与 SDN 安全模型

Fig. 3　Traditional network security model and SDN security model

(2)交换机功能。传统网络的交换机除了完成转发任务外还负责制定转发规则,而 SDN 是通过控制器来实现转发规则的制定,交换机只负责转发。仅从交换机功能的角度来分析,传统网络的安全问题更集中在交换机自身,而对于 SDN

来说，控制器和 SDN 交换机都可能成为攻击的目标，且一个受到攻击后会对另一个产生一定的影响。

4.3 转发技术

传统网络转发受到各种网络协议的控制，属于纯分布式控制方式，控制层和转发层在同一个设备中紧密耦合，管理人员无法直接操控转发行为，只能通过配置网络协议来控制转发行为。不同的网络协议对转发行为有不同的影响模式，如路由协议只能靠目的 IP 地址进行转发，MPLS 协议则靠 MPLS 标签进行转发，缺乏统一的控制。而 SDN 则采用集中式控制方式，通过控制层和数据层的分离，控制器全局控制和管理交换机，管理人员能够直接操控转发行为，便于网络管理，但集中式控制方式很容易遭受 DDoS 攻击。从转发流程和所用的配置来看，Openflow 交换机的转发流程就是接收的报文被解析后再到流表中进行匹配，如果匹配成功则执行或者记录相应的动作，然后继续下一个表项，直到最终走完所有表项或者无后续表项需要匹配为止。传统交换机中功能配置类型有寄存器、基于 HASH 查找的 RAM 与少量 ACL 表项所用的 TCAM 等，而 OpenFlow 交换机所用的基础单元与传统交换机不太一样，需要少量寄存器和大量的 TCAM 表项，这必然会使交换芯片的硬件成本大大增加，而且会导致表项匹配字段和表项条数增多。实际上，每条表项可能只匹配几个字段，甚至一两个字段，而动作往往是一两个，表项可能也只用到很少的一部分，这样势必造成了一定程度的浪费。因此，从资源使用来说，传统交换机在结合业务转发上存在一些不足之处，但是相比 OpenFlow 交换机确实比较节省资源和成本。从转发技术的角度分析，对 SDN 的攻击主要是为了破坏控制层的控制器，从而造成流规则不能正常下发，导致位于数据层的 SDN 交换机无法正常进行报文转发，特别是 DDoS 攻击针对单控制器的攻击问题，一旦控制器因受到攻击而失效，就会造成整个 SDN 工作陷入瘫痪。而传统网络的转发主要由各交换机本身进行，因此不会轻易发生单点失效，即使一台交换机发生故障也不会对其他交换机造成影响。

总之，传统网络的一些安全问题可能在 SDN 中依然存在，但不同的网络结构使得安全问题的解决方法有所不同。

5 未来的方向

SDN 具有全局可视化、集中式控制、可编程性等特点，这些都有益于增强整个网络的安全性。然而，SDN 特殊的分层结构给每一层都提出了新的安全要求，另外集中式控制和可编程的特点将对 SDN 下的网络安全提出较大的挑战。因此，许多有关 SDN 的安全解决方案和平台已经被提出(如前文所述)。但是，SDN 在商用之前仍有一些问题需要研究和解决。下文将根据目前研究所面临的问题和挑战归纳出未来研究的几个发展方向。

5.1 SDN 中程序和开发模式的规范化问题

开发人员利用 SDN 既能够开发出新的网络协议、架构及应用，同时也便于测试或使用业务网络。尽管新开发的应用程序在一定程度上具有创新性，但其在运行时可能会引入一些不安全因素，从而使整个 SDN 面临新的安全挑战。各种独立和松散的互联开发环境及任意控制平台的部署都会给相对独立的应用程序的功能带来严重的安全威胁。若在同一控制层下使用不同的应用开发环境，除所开发的应用程序功能在相同的数据层下可能会产生安全漏洞之外，还存在其他挑战，如安全策略的碰撞。此外，在互联网应用中配置问题是导致安全漏洞的关键因素[71]。因此，为减小产生安全漏洞和冲突模块的可能性，编程模型和开发环境必须标准化。在分布式系统中冲突模块能创建安全漏洞，如泄露敏感的网络信息或 API，产生相互矛盾的流规则。在 SDN 中，位于控制层顶端的应用程序能实现大部分网络功能。然而，不同的应用程序有各自所需的特殊功能需求，如一些应用程序可能需要考虑网络负载均衡，一些可能需要统计报文的样本等。应该考虑给各个不同需求的应用程序提供差异化服务。与此同时，某些应用程序可能需要让流量按照某种特征的路径进行传输，例如，在文献[9]中，为避开一些链路，让流量保持在一定的管辖范围之内，并将所使用路径的成本控制在限制范围内。此外，在 SDN 中，安全应用程序通过控制层访问网络数据样本和资源。工作在数据层上的应用程序有时需要直接观测网络报文行为。然而，控制层自己观测的结果往往与应用程序直接观测的网络行为不一致。因此，一些应用程序需要直接访问网络信息和资源。另一方面，就应用程序开发的当前趋势而言，选择直接访问各种应用程序可能会带来安全方面的挑战。因此，有必要根据应用程序功能和对底层网络资源访问的要求先对其进行分类。其次，对每个类的安全策略进行定义(例如，在 OpenFlow 交换机流表中，定义用于访问网络资源的应用程序的请求或网络统计信息)。在 IETF 草案[9]中，对影响 SDN 安全的 3 种类型的应用程序进行了讨论，这 3 种程序分别称为网络敏感的应用、网络服务和打包式网络服务。由于大量用于访问网络资源或用于获取网络统计信息的应用程序增加，对每个应用程序进行验证和授权将成为控制层的瓶颈。应用程序的分类减少了控制器对应用程序安全策略进行验证和授权的开销。

5.2 SDN 的可扩展性问题

鉴于 SDN 逻辑上集中式的管理方式，其可扩展性已经成为自身所面临的重要问题和挑战。在 SDN 中，随着网络拓扑和规模的不断增大，流向控制器的流量也在不断增加，这将导致流建立时间变长[51]，OpenFlow 控制器的控制和操作能力将受限。文献[72]表明，OpenFlow 交换机能耗尽一个网络资源。因此，SDN 的可扩展性对于因控制器和交换机之间的通信过量而导致的控制器饱和等攻击尤为重要[19]。对于 SDN 的可扩展性的研究能降低控制器单点失效所导致的安全威胁。为降低该威胁，研究人员提出了多控制器概念。但是，仅仅简单地添加多个控制器可能会导致级联控制器故障[13]。因此，有必要从 SDN 安全性和可扩展性两个方面来考虑设计安全的 SDN 架构，从而保证控制层的高可用性。在 SDN 中，单控制器或由多控制器组成的组控制器为很多转发设备提供控制层服务时，往往会导致节点到节点或节点至控制器网络信息交换的延迟，这对 SDN 的可用性带来了一定的挑战[29]。由单控制器管理的活跃 OpenFlow 交换机数量的不断增加导致控制器用于设置流规则的响应时间变长，主要原因在于建立流请求数的增加[73]。控制层和数据层之间的一个折衷解

决方案是减少交换机对控制器的过分依赖。在 SDN 中,智能权衡将有利于解决两个方面的主要挑战。首先,通过增强的 OpenFlow 可扩展性结构和本地决策可缩短时延,其次增强交换机的可用性和快速恢复机制。从网络安全的角度,智能化分享将使网络不易出现单点故障,并且不容易受到 DoS 攻击。然而,智能权衡作为一种尝试,在提高网络安全性方面尚未得到证实。对于数据层,负载均衡和连接迁移技术可用来分担 OpenFlow 交换机的负载,这不仅有助于满足服务质量要求,而且能最大限度地减少洪泛和数据平面饱和度攻击的发生。

5.3　网络安全及网络流量的同步和自动化

网络安全是网络管理的一个重要组成部分。其中,稳定而强大的安全策略部署需要对所有的因素进行分析,包括网络元素策略配置等,以避免安全性程序冲突和不一致的发生,从而减少出现严重的安全漏洞[6]。在网络管理中,策略协助和部署同步是网络安全的重要挑战,其要求网络处于畅通状态,即使出现安全漏洞或有其他情况发生,也要保证网络安全策略与流量行为变化保持紧密协助。尽管这种策略协助在一些项目中已被提出,如 4D[74],SANE[75] 和 Ethane[76] 等,但都未真正实现。其主要原因是缺乏安全和流量管理之间的协同工作。因此,Greenberg 等[74] 指出,传统的 IP 网络具有不稳定性和复杂性,其中路由协议的错误配置可能导致严重的级联影响。在 SDN 中,由于控制器负责控制和管理整个网络,安全失误可能会造成控制器损坏,进而影响数据层的流规则。例如,在 OpenFlow 网络中,DoS 对控制器的攻击将增加交换机建立流规则的时延。与其他网络结构相似的是,SDN 的流量特征也可以用来检测分布式 DoS 攻击[55,77-78]。然而,基于中心集中式 SDN 控制层和数据层(转发层)的可编程性可以让独立策略和部署相互依存。因此,若要充分利用 SDN 技术,则必须利用相互依存的安全性和流量转发策略(即安全流量的转发机制)。在传统网络中,为应对上下文变化以及控制系统不断循环学习和未来自身的行为更新,已经出现了相应的交错部署自动化技术[79]。Hamed 等[80] 指出,采用手动配置网络安全技术(如防火墙设备和 IPSec 技术的扩展集等)时容易产生错误,策略冲突容易造成严重的安全漏洞和威胁。防火墙配置错误研究[81] 表明,复杂的配置是企业网络中存在安全隐患的一个主要的原因。SDN 从底层的网络设备中抽象出来,能使用编程设计语言和网络控制器自动感知网络变化并做出相应的反应[82]。开放网络基金会(ONF)在文献[83]中称,SDN 提供了灵活的网络自动化和管理框架,这使得其可能开发出能形成自动化任务管理(即手动完成)的工具,以减小运营开销,降低因操作人员的错误或新兴的自助服务配置模型所带来的网络不稳定性。虽然面向服务质量 QoS[84] 的自动化框架 Procera[31] 和响应自动控制平台 OMNI[85] 等已被提出,但目前仍没有经证实的、可行的 SDN 安全自动化机制。集中式网络管理和控制需要网络管理员承担较大的责任,因此管理员应有较强的素质和相关的背景知识,一旦管理员不具备这些知识和素质,就可能成为 SDN 管理的一个瓶颈。因此,为了有效保障 SDN 的安全,建立仅需较少管理员或者不需要管理员干预的自动安全机制和自动恢复机制非常必要。

5.4　身份识别问题

身份识别一直是网络安全领域的一个重要研究课题。Juniper 等公司指出,目前传统的网络安全问题主要来源于较弱的网络身份识别[86]。同样,SDN 也存在这样的问题。由于控制器通过北向接口与应用层的应用相连接,这些应用可以得到不同的网络权限,控制器可以获取一个抽象的网络状态,而第三方应用则可以通过北向接口读取这些网络视图。一旦攻击者伪装成控制器或某个应用接入到 SDN 中,那么它就有可能读取该网络的状态,甚至改写整个网络状态以达到控制整个网络的目的。由此可知,SDN 架构缺乏对用户身份的认证。尽管 OpenFlow 提供了基于分组首部字段的流识别或数据包机制,但目前对于用户标识的绑定办法还未给出。虽然 IP 地址能作为主机的身份标识,但这会导致灵活性网络架构下的语义冲突[87]。在 OpenFlow 下使用 IP 地址时,因流动性而造成地址发生变更会破坏整个流处理过程,需要快速和定期更新流表信息,但这样将会造成额外的开销。可以考虑将一些新技术(如 HIP 技术[88])应用到 SDN 中,这不仅能提供永久的标识(如主机标识标签或主机标识符),而且终端到终端的安全是通过一系列相关的安全协议辅助完成的,具有较高的安全性和可靠性。也可以通过密码标识[89] 的方式对进出网络的数据内容进行签名,根据用户身份、文件属性或业务内容等特征信息生成密码标识,以保证数据的真实性。但是,通过加密算法对数据内容进行加密会增加识别本身的计算复杂度,如何降低由此带来的时间开销是以后需要解决的问题。

5.5　数据泄露和恶意数据修改问题

在 OpenFlow 交换机中,规定每一个输入的数据包必须与流表项中的记录进行匹配,若匹配失败,则将其丢弃或封装成 packet_in 消息发送给控制器。由此,攻击者可以通过对某数据包进行时间分析,来判断每个动作大约所需的时间。攻击者将一个数据包从进入交换机的某个接口到离开交换机所经历的时间与一个数据包被重定向到控制器所需要的时间进行对比,就能够推测出该交换机的配置是主动的还是被动的。另外,攻击者也可以通过精心伪造数据包来判断出更多的交换机的配置。通过这些信息,攻击者可以伪造同类型的数据包发动洪泛攻击。另一个数据泄露表现在 SDN 中存在的一些敏感信息(如密码和权限等),这些信息一旦泄露也将导致 SDN 的安全性遭受威胁。例如,若一个 SDN 中的某 OpenFlow 交换机和与之相关联的证书或密码没有进行分开处理,则可能导致敏感数据泄露,从而使交换机遭到入侵和破坏。控制器作为 SDN 的重要部分,一旦被攻击者劫持,攻击者则可以通过控制器来操纵 SDN 中的流向,并将其转发到被攻击者控制的网络设备。如何减少或避免 SDN 中因数据泄露和恶意数据修改所带来的安全隐患是未来急需解决的问题。

结束语　尽管 SDN 的分层结构和可编程性特点给网络管理带来了很多便利,但控制器的中心性集中式的控制方式和由可编程性引起的新应用程序的出现给 SDN 带来了安全方面的新挑战。本文通过对 SDN 3 层结构(应用层、控制层和数据层)特点的分析,分别总结出了 SDN 每层所面临的安全问题。结合当前已有的研究成果,归纳出各层在面临安全威胁和恶意攻击时所采取的一些对策,并分别从模型和技术

两个层面与传统网络安全进行了对比和分析。尽管 SDN 安全中的一些问题已经得到解决，但是仍然有一些研究内容和研究问题值得我们进行进一步的分析并提出解决方案。本文通过对已有的 SDN 安全问题和解决方案的分析研究，总结归纳出了几个未来的研究方向，以供研究人员参考。

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] BOSSHART P, DALY D P, GIBB G, et al. P4: programming protocol-independent packet processors[J]. ACM Special Interest Group on Data Communication, 2014, 44(3): 87-95.

[3] WANG H, SOULE R, DANG H T, et al. P4FPGA: A Rapid Prototyping Framework for P4[C]// symposium on Sdn Research. 2017: 122-135.

[4] ZUO Q Y, CHEN M, ZHAO G S, et al. Research on OpenFlow-based SDN technologies[J]. Journal of Software, 2013, 24(5): 1078-1097.

[5] DONG S, ABBAS K, JAIN R. A Survey on Distributed Denial of Service (DDoS) Attacks in SDN and Cloud Computing Environments[J]. IEEE Access, 2019, 7: 80813-80828.

[6] YU Y, WANG Z L, BI J, et al. A survey on the languages in the northbound interface of the software defined neworking[J/OL]. Journal of Software, 2016. http://www.jos.org.cn/1000-9825/5028.htm.

[7] SHIN S, PORRAS P, YEGNESWARAN V, et al. A Framework For Integrating Security Services into Software-Defined Networks[C]// Proceedings of the 2013 Open Networking Summit (Research Track poster paper). 2013.

[8] KREUTZ D, RAMOS F, VERISSIMO P. Towards secure and dependable software-defined networks[C]// Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. ACM, 2013: 55-60.

[9] HARTMAN S, WASSERMAN M, ZHANG D. Software driven networks problem statement[J/OL]. Network Working Group Internet-Draft, 2013. https://tools.ietf.org/html/drafthartman-sdnsec-requirements-00.

[10] XIE H, TSOU T, LOPEZ D, et al. Use cases for ALTO with software defined networks[J/OL]. Working Draft, IETF Secretariat, Internet-Draft, 2012. https://tools.ietf.org/html/draft-xie-alto-sdn-use-cases-01.

[11] NAOUS J, ERICKSON D, COVINGTON G A, et al. Implementing an OpenFlow switch on the NetFPGA platform[C]// Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS '08). 2008: 1-9.

[12] JARSCHEL M, OECHSNER S, SCHLOSSER D, et al. Modeling and performance evaluation of an OpenFlow architecture[C]// 23rd International Teletraffic Congress (ITC 2011). IEEE, 2011.

[13] YAO G, BI J, GUO L. On the cascading failures of multi-controllers in software defined networks[C]// 21st IEEE International Conference on Network Protocols (ICNP). IEEE, 2014.

[14] FONSECA, BENNESBY R, MOTA E, et al. A replication component for resilient OpenFlow-based networking[C]// IEEE Network Operations and Management Symposium. 2012: 933-939.

[15] SEEDORF J, BURGER E. Application-layer traffic optimization (ALTO) problem statement[OL]. http://www.rfc-editor.org/rfc/rfc5693.txt.

[16] NADEAU T, PAN P. Software driven networks problem statement[J/OL]. Network Working Group Internet-Draft, 2011. https://tools.ietf.org/html/draft-nadeau-sdn-problem-statement-00.

[17] BROOKS M, YANG B. A man-in-the-middle attack against opendaylight sdn controller[C]// Proceedings of the 4th Annual ACM Conference on Research in Information Technology. ACM, 2015: 45-49.

[18] LIN P C, LI P C, NGUYEN V L. Inferring openflow rules by active probing in software-defined networks[C]// 2017 19th International Conference Advanced Communication Technology (ICACT). IEEE, 2017: 415-420.

[19] SHIN S, YEGNESWARAN V, PORRAS P, et al. Avant-guard: Scalable and vigilant switch flow management in software-defined networks[C]// ACM Sigsac Conference on Computer & Communications Security. 2013: 413-424.

[20] ZHANG Y, BEHESHTI N, TATIPAMULA M. On resilience of splitarchitecture networks[C]// Proceedings of the Global Communications Conference. 2011: 1-6.

[21] DIERKS T. The Transport Layer Security (TLS) protocol version 1.2 [EB/OL]. http://tools.ietf.org/html/rfc5246.

[22] RESCORLA E, MODADUGU N. Datagram Transport Layer Security Version 1.2[EB/OL]. http://tools.ietf.org/html/rfc6347.

[23] BENTON K, CAMP L J, SMALL C. OpenFlow vulnerability assessment[C]// Acm Sigcomm Workshop on Hot Topics in Software Defined Networking. 2013: 151-152.

[24] LIYANAGE M, GURTOV A. Secured VPN models for LTE backhaul networks[C]// IEEE Vehicular Technology Conference (VTC Fall). 2012: 1-5.

[25] STAESSENS D, SHARMA S, COLLE D, et al. Software defined networking: Meeting carrier grade requirements[C]// 18th IEEE Workshop on Local & Metropolitan Area Networks (LANMAN). 2011: 1-6.

[26] SHAGHAGHI A, KAAFAR M A, BUYYA R, et al. Software-Defined Network (SDN) Data Plane Security: Issues, Solutions and Future Directions[J]. arXiv: 1804.00262, 2018.

[27] ZHOU Y D, CHEN K Y, ZHANG J J, et al. Exploiting the Vulnerability of Flow Table Overflow in Software-Defined Network: Attack Model, Evaluation, and Defense[J]. Security and Communication Networks, 2018, 2018: 1-15.

[28] SCOTT-HAYWARD S, NATARAJAN S, SEZER S. A Survey of Security in Software Defined Networks[J]. IEEE Communications Surveys & Tutorials, 2016, 18(1): 623-654.

[29] SEZER S. Are we ready for SDN? Implementation challenges

for software-defined networks[J]. IEEE Communication Magazine,2013,51(7):36-43.

[30] FOSTER N. Frenetic:A network programming language [J]. SIGPLAN Notices,2011,46(9):279-291.

[31] VOELLMY A,KIM H,FEAMSTER N. Procera:A language for high-level reactive network control[C]// First Workshop on Hot Topics in Software Defined Networks. 2012:43-48.

[32] MONSANTO C,FOSTER N,HARRISON R,et al. A compiler and run-time system for network programming languages[J]. SIGPLAN Notices,2012,47(1):217-230.

[33] SHIN S. FRESCO:Modular composable security services for software-defined networks[C]// Proceedings of Network and Distributed Security Symposium. 2013:1-16.

[34] WEN X,CHEN Y,HU C,et al. Towards a secure controller platform for OpenFlow applications[C]// Proceedings of the second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. 2013:171-172.

[35] CHOWDHARY A,HUANG D,ALSHAMRANI A,et al. Trufl:Distributed trust management framework in sdn[C]// ICC 2019-2019 IEEE International Conference on Communications (ICC). IEEE,2019:1-6.

[36] BECKETT R. An assertion language for debugging SDN applications[C]// Proc3rd ACM Workshop Hot Topics Software. Defined Network,2014:91-96.

[37] KHURSHID A,ZOU W,ZHOU W X,et al. Veriflow:Verifying network-wide invariants in real time[C]// Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation. 2013:15-28.

[38] SON S,SHIN S,YEGNESWARAN V,et al. Model checking invariant security properties in openflow[C]// IEEE International Conference on Communications. 2013:1974-1979.

[39] CANINI M,KOSTIC D,REXFORD J,et al. Automating the testing of OpenFlow applications[C]// Proceedings of the 1st International Workshop on Rigorous Protocol Engineering (WRiPE). 2011:1-6.

[40] HANDIGOL N,HELLER B,JEYAKUMAR V,et al. Where is the debugger for my software-defined network? [C]// Workshop Hot Topics Software. Defined Network,2012:55-60.

[41] WUNDSAM A,LEVIN D,SEETHARAMAN S,et al. OFRewind:Enabling record and replay troubleshooting for networks [C]// Usenix Conference on Usenix Technical Conference. 2011:29.

[42] Security-enhanced floodlight. SDx Central,Sunnyvale,CA,USA [EB/OL]. http://www. sdncentral. com/education/ towardsecure-sdn-control-layer/2013/10/.

[43] SWITCH B. Developing floodlight modules. Floodlight Openlow controller[EB/OL]. http://www. projectfloodlight. org/floodlight/.

[44] FERNANDEZ M. Comparing openflow controller paradigms scalability:reactive and proactive[C]// IEEE International Conference on Advanced Information Networking & Applications. 2013:1009-1016.

[45] VOELLMY A,WANG J. Scalable software defined network controllers[J]. Acm Sigcomm Computer Communication Review,2012,42(4):289-290.

[46] 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.

[47] CAI Z,COX A L,EUGENE N G. Maestro:A system for scalable OpenFlow control[J/OL]. Cs. rice. edu, https://scholarship. rice. edu/bitstream/handle/1911/96391/TR10-11. pdf?sequence=1&isAllowed=y.

[48] PHEMIUS K,BOUET M,LEGUAY J. DISCO:Distributed multidomain SDN controllers[C]// IEEE Network Operations and Management Symposium (NOMS). 2014:1-4.

[49] PHEMIUS K,BOUET M,LEGUAY J. DISCO:Distributed SDN controllers in a multi-domain environment[C]// IEEE Network Operations and Management Symposium (NOMS). 2014:1-2.

[50] Advanced Message Queuing Protocol[EB/OL]. http://www.amqp. org.

[51] TOOTOONCHIAN A,GANJALI Y. HyperFlow:A distributed control plane for OpenFlow[C]// Internet Network Management Conference on Research on Enterprise Networking. USENIX Association,2010:3.

[52] HELLER B,SHERWOOD R,MCKEOWN N. The controller placement problem[C]// Acm Sigcomm Workshop on Hot Topics in Software Defined Networking. 2012:7-12.

[53] AHMAD I,KARUNARATHNA S N,YLIANTTILA M,et al. Load balancing in software defined mobile networks[C]// Software Defined Mobile Networks (SDMN):Beyond LTE Network Architecture. Hoboken,NJ,USA:Wiley,2015:225-245.

[54] NAMAL S,AHMAD I,GURTOV A,et al. SDN based inter-technology load balancing leveraged by flow admission control [C]// IEEE SDN for Future Networks and Services(SDN4FNS). 2013:1-5.

[55] BRAGA R,MOTA E,PASSITO A. Lightweight DDoS flooding attack detection using NOX/OpenFlow[C]// The 35th Annual IEEE Conference on Local Computer Networks. 2010:408-415.

[56] KOHONEN T. The self-organizing map [J]. Neurocomputing,1998,21(1):1-6.

[57] DONG S,SAREM M. DDoS Attack Detection Method Based on Improved KNN With the Degree of DDoS Attack in Software-Defined Networks[J]. IEEE Access,2020:5039-5048.

[58] KALKAN K,ALTAY L,GÜR G,et al. JESS:Joint Entropy-Based DDoS Defense Scheme in SDN[J]. IEEE Journal on Selected Areas in Communications,2018,36(10):2358-2372.

[59] WU Z J,XU Q,WANG J J,et al. Low-Rate DDoS Attack Detection Based on Factorization Machine in Software Defined Network[J]. IEEE Access,2020,8:17404-17418.

[60] HU Y,WANG W,GONG X,et al. On reliability optimized controller placement for software-defined networks[J]. China Communication,2014,11(2):38-54.

[61] HU Y N,WANG W D,GONG X Y,et al. Reliability aware controller placement for software-defined networks[C]// FIP/IEEE International Symposium on Integrated Network Management. 2013:672-675.

[62] BARI M. Dynamic controller provisioning in software defined

networks[C]// International Conference on Network & Service Management. 2013:18-25.
[63] HOCK D. Pareto-optimal resilient controller placement in SDN-based core networks[C]// Proceedings of the 2013 25th International Teletraffic Congress (ITC). 2013:1-9.
[64] MOGUL J C. DevoFlow:Cost-effective flow management for high performance enterprise networks[C]// Acm Sigcomm Workshop on Hot Topics in Networks. 2010:1-6.
[65] GE J G, SHEN H J, PENG Y E, et al. An OpenFlow-based dynamic path adjustment algorithm for multicast spanning trees [C]// 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications. 2013: 1478-1483.
[66] KEMPF J. Scalable fault management for OpenFlow[C]// IEEE International Conference on Communications (ICC). 2012:6606-6610.
[67] Porras P. A security enforcement kernel for OpenFlow networks [C]// ACM SIGCOMM Workshop on Hot Topics in Software Defined Networks. 2012:121-126.
[68] AL-SHAER E, AL-HAJ S. FlowChecker:Configuration analysis and verification of federated openflow infrastructures[C]// Proceedings of the 3rd ACM Workshop on Assurable and Usable Security Configuration. 2010:37-44.
[69] FAN Z, XIAO Y, NAYAK A. et al. An improved network security situation assessment approach in software defined networks [J]. Peer-to-Peer Networking and Applications, 2019, 12(2): 295-309.
[70] NAYAK A K, REIMERS A, FEAMSTER N, et al. Resonance: dynamic access control for enterprise networks[C]// Proc 1st ACM Workshop Res. Enterprise Network, 2009:11-18.
[71] KEROMYTIS A. Voice-over-IP security:Research and practice [J]. IEEE Security Privacy, 2010, 8(2):76-78.
[72] SHIN S, GU G. Attacking software-defined networks:a first feasibility study[C]// Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. 2013:165-166.
[73] TOOTOONCHIAN A, GORBUNOV S, GANJALI Y, et al. On controller performance in software-defined networks[C]// Proc. USENIX Workshop Hot-ICE, 2012:10.
[74] GREENBERG A. A clean slate 4D approach to network control and management[J]. Computer communication review, 2005, 35(5):43-54.
[75] CASADO M, GARFINKEL T, AKELLA A, et al. SANE:A Protection Architecture for Enterprise Networks[C]// Conference on Usenix Security Symposium. USENIX Association, 2006.
[76] CASADO M, FREEDMAN M J, PETTIT J, et al. ETHANE: Taking Control of the Enterprise[C]// Proceedings of the ACM SIGCOMM 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. Kyoto, Japan, 2007:27-31.
[77] WANG L, LI Q, JIANG Y, et al. Woodpecker:Detecting and mitigating link-flooding attacks via SDN[J]. Computer Networks, 2018, 147:1-13.
[78] LI C, WU Y, YUAN X, et al. Detection and defense of DDoS attack-based on deep learning in OpenFlow-based SDN[J]. International Journal of Communication Systems, 2018, 31(5): e3497.
[79] JENNINGS B, MEER S V D, BALASUBRAMANIAM S, et al. Towards autonomic management of communications networks [J]. IEEE Communication Magazine, 2007, 45(10):112-121.
[80] HAMED H, AL-SHAER E. Taxonomy of conflicts in network security policies[J]. Communications Magazine, IEEE, 2006, 44(3):134-141.
[81] WOOL A. A quantitative study of firewall configuration errors [J]. Computer, 2004, 37(6):62-67.
[82] KIM H, FEAMSTER N. Improving network management with software defined networking[J]. IEEE Communication Magazine, 2013, 51(2):114-119.
[83] Software-defined networking:The new norm for networks[EB/OL]. https://www.opennetworking.org/sdn-resources/sdn-library/whitepapers/benefits-of-OFB-SDN.
[84] KIM W, SHARMA P, LEE J, et al. Automated and Scalable QoS Control for Network Convergence[C]// Proc. Internet Network Management Workshop/Workshop on Research on Enterprise Networking (INM/WREN). 2010.
[85] MATTOS D M. OMNI:OpenFlow management infrastructure [C]// International Conference on the Network of the Future. 2011:52-56.
[86] REXFORD J, DOVROLIS C. Future internet architecture:clean-slate versus evolutionary research[J]. Communications of the ACM, 2010, 53(9):36-40.
[87] LI T. Design goals for scalable internet routing[OL]. https://www.rfc-editor.org/rfc/pdfrfc/rfc6227.txt.pdf.
[88] GURTOV A. Host Identity Protocol (HIP):Towards the Secure Mobile Internet[M/OL]. https://onlinelibrary.wiley.com/doi/book/10.1002/9780470772898.
[89] QIN X, TANG G D, CHANG C W. SDN security control and forwarding method based on cipher identification[J]. Journal on Communications, 2018, 39(2):31-42.

DONG Shi, born in 1980, Ph.D, professor, is a member of China Computer Federation. His main research interests include distributed computing and network management.