基于 OpenFlow 的 SDN 网络攻防方法综述

武泽慧 魏 强 王清贤
(解放军信息工程大学 郑州 450001) (数学工程与先进计算国家重点实验室 郑州 450001)

摘 要 软件定义网络(Software Defined Network,SDN)的控制与转发分离、统一配置管理的特性使其网络部署的灵活性、网络管理的动态性以及网络传输的高效性均有大幅提升,但是其安全性方面的问题却比较突出。综述了基于 OpenFlow 的 SDN 在安全方面的研究现状,首先根据 SDN 的三层架构分析了其脆弱性,介绍 SDN 不同平面面临的安全威胁,并根据网络攻击的流程来介绍当前主要的攻击手段,包括目标网络探测、伪造欺骗实现网络接入以及拒绝服务攻击和信息窃取;其次,针对不同攻击环节,分别从探测阻断、系统加固、攻击防护 3 个方面对当前主要的防御手段进行论述;最后,从 SDN 潜在的攻击手段和可能的防御方法两方面来探讨未来 SDN 安全的研究趋势。

关键词 网络安全,软件定义网络,虚拟化,动态防御
中图法分类号 TP393.0 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.06.021

Survey for Attack and Defense Approaches of OpenFlow-enabled Software Defined Network

WU Ze-hui WEI Qiang WANG Qing-xian
(PLA Information Engineering University,Zhengzhou 450001,China)
(State Key Laboratory of Mathematical Engineering and Advanced Computing,Zhengzhou 450001,China)

Abstract Software defined network (SDN) grants the network an omnipotent power to increase the flexibility of network deployment,the dynamic of network management and the efficiency of network transmission by centralizing the control plane and separating it with data plane. However,the security of SDN is still outstanding. In this paper,we aimed at analyzing and categorizing a number of relevant research works toward OpenFlow-enabled SDN security. We first provided an overview on threats of SDN with its three layers architecture,and further demonstrated their vulnerabilities within each layer. Thereafter,we presented existing SDN-related attacking approaches according to the procedures of network attacking,such as network probing,defraud inserting and remote controlling. And then we dedicated the next part of this paper to study and compared the current defense approaches underlying probe blocking,system strength,and attack defensing. Furthermore,we reviewed several potential attack and defensed methods as some foreseeable future research challenges.

Keywords Cyber security,Software defined network,Virtualization,Dynamic defense

1 引言

Internet 规模的迅速扩张、网络应用的急剧增加、网络中海量流量的处理以及云计算和大数据时代的到来,要求现有网络架构具有更高的可靠性、扩展性、开放性、灵活性和管控性。软件定义网络(Software Defined Network,SDN)应运而生,以满足上述未来互联网需求的新型网络架构[1]。斯坦福大学 Nick McKeown 教授于 2008 年在 SIGCOMM 上发表文章《OpenFlow:Enabling Innovation in Campus Networks》[2],第一次详细阐述了 OpenFlow/SDN 的概念,列举了校园实验性通信网络协议支持、网络隔离、网络管理和访问控制等 4 大应用。自此,OpenFlow/SDN 技术开始被工业界和学术界广泛关注,并且迅速发展起来。当前的数据中心、云计算平台[3]普遍开始采用 SDN 架构来完成数据转发,如 2012 年谷歌已经使用 OpenFlow 开源协议在数据中心之间转发数据[4],并在广域网中部署其与 Nicira 共同开发的网络控制器[5];2014 年,Facebook 推出基于 SDN 架构的"Wedge"网络交换机,用于连接数据中心的服务器,以创建高速数据中心网络[6];2014 年 3 月,华为推出 SDN 与云计算结合的产品解决方案[7],2015 年 5 月发布了全球首个基于 SDN 架构的敏捷物联解决方案[8]。国际研究机构 Gartner[9]甚至将 SDN 技术列为未来 5 年内 IT 领域十大关键技术之一。

SDN 的核心思想是通过数据与控制平面的分离,借助统一、厂商无关的控制与数据平面开放接口,使得无论是研究人员还是网络管理人员都可根据顶层服务或网络控制应用构建和呈现逻辑的全网视图,完成网络分片和底层网络的虚拟化,

到稿日期:2016-05-27 返修日期:2016-09-27
武泽慧(1988-),男,博士,主要研究方向为网络安全,E-mail:wuzehui2010@foxmail.com;魏 强(1979-),男,副教授,博士生导师,主要研究方向为工业控制系统安全;王清贤(1960-),男,教授,博士生导师,主要研究方向为网络与信息安全。

从而为形式各异的网络应用提供灵活的承载服务[10-11]。

SDN 的控制与转发分离、统一配置管理的特性使其网络部署的灵活性、网络管理的动态性，以及网络传输的高效性均有大幅提升。但是其安全性方面的问题却比较突出，主要有以下 4 个方面的原因[12]：

(1)集中控制的特性降低了攻击的门槛。SDN 采用 OpenFlow 协议作为控制器与转发设备的南向通信协议，但是由于 OpenFlow 协议允许所有未匹配到流表的数据包发送 PACKET-IN 消息至控制器，由控制器制定该类数据包转发所需的流表项，因此用户主机或者转发设备可以伪造 PACKET-IN 消息发送给控制器。虽然 OpenFlow 协议支持 TLS 协议(Transport Layer Security Protocol)对转发设备和控制器的身份进行认证，可以阻断上述身份伪造攻击，但是 TLS 无法阻止内部被攻击者俘获的转发设备发动攻击，并且当前许多控制器产品均未开启 TLS 服务。

(2)传统网络中的攻击方法在 SDN 网络中仍然适用，但是传统网络的安全防御手段在 SDN 网络中却无法胜任。因为传统网络安全的手段是假设转发设备(如路由器)具有一定的配置、学习、管理能力，而不像 SDN 网络中的转发设备仅仅负责查表转发。因此，直接将传统网络安全方法应用到 SDN 网络中需要对 OpenFlow 协议以及转发设备进行升级，这势必会影响 SDN 的设计初衷，降低网络的运行效率。

(3)在 SDN 网络中，转发设备通常使用虚拟化的方法实现(如 Open vSwitch)，而传统网络的转发设备使用硬件完成。硬件转发设备(如路由器)很难被攻击者俘获，而软件转发设备则相对容易，从而导致虚拟化使 SDN 网络的灵活性得到提升的同时也扩大了 SDN 网络的攻击面。

(4)传统网络中的攻击普遍由被攻击者控制的主机完成，但是由于在 SDN 中转发设备通过软件的方法实现，因此网络攻击可以通过主机或转发设备完成，并且在实现难度上差别不大，如主机可以伪装成转发设备发动针对控制器的 flood 攻击。

尽管学术界和工业界对 SDN 安全问题都做出了许多尝试，如 IETF 发布 SDN 安全架构[13]、SUCCESS Lab 实现 SDN 攻击检测原型系统[6]、商用 SDN 控制器 Floodlight 安全加固版本 SE Floodlight[14-15]的发布等，但是攻防的对抗始终存在，对 SDN 的攻击新方法也不断出现，如传统的中间人攻击[16-17]、欺骗攻击[18-19]，以及针对 SDN 网络的流表更新攻击[20]等。

当前关于 SDN 网络安全方面的综述普遍根据 SDN 的三层架构进行分类，但是在具体网络对抗中往往是跨层次进行的，因此按照网络攻击的步骤阐述 SDN 的安全性在实际网络对抗中更具有实际意义。对此，本文在前人研究的基础上从网络攻防对抗的角度综述 SDN 的安全性，对 SDN 面临的网络安全问题进行分析，讨论 SDN 攻防演变过程中产生的攻击和防御方法。首先对 SDN 的攻击面进行介绍，分析 SDN 不同组件以及不同网络层的脆弱性(见第 2 节)；接着根据网络攻击的 3 个步骤(探测、植入、攻击)阐述 SDN 网络的攻击方法(见第 3 节)；其次在攻击的基础上讨论不断发展的 SDN 安全防御手段(见第 4 节)；然后对 SDN 未来可能面临的安全挑战和可能采用的新型防御手段进行分析介绍(见第 5 节)；最后对全文进行总结。图 1 示出了本文的组织结构及每部分的主要内容提纲。

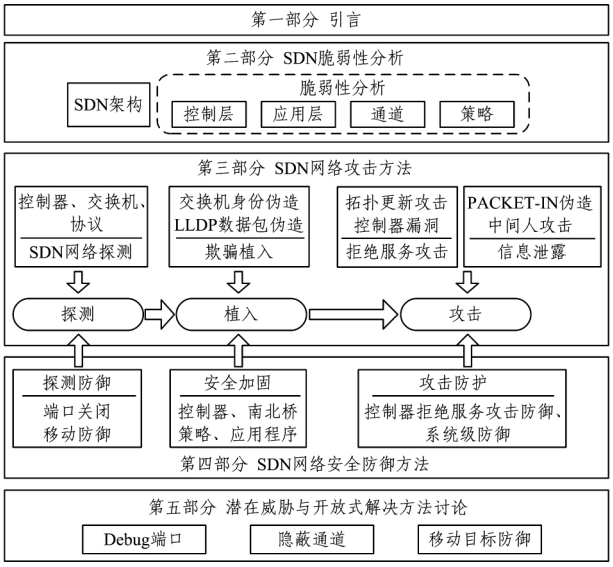

图 1 SDN 综述论文组织结构

2 SDN 脆弱性分析

2.1 SDN 架构

SDN 基本架构从上至下主要分为应用层、控制层和转发层 3 层[21-22]，如图 2 所示。

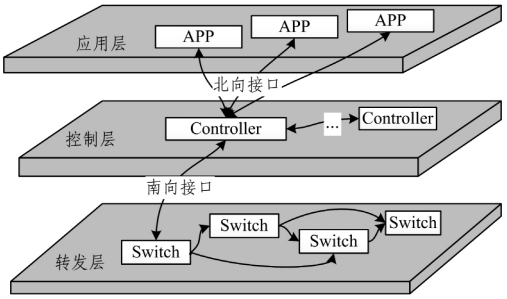

图 2 SDN 架构图

(1)应用层根据网络的不同应用需求，调用与控制层相接的应用编程接口 API，实现具有不同功能的应用程序。通过调用北向 API 接口，应用层可实现如路由、组播、安全、访问控制、带宽管理、流量工程、服务质量以及处理器和存储优化等网络服务，并为应用层 APP 量身定制这些服务以满足业务目标。

(2)控制层由控制软件实现，主要负责集中维护网络拓扑及网络状态信息，实现不同业务特性的适配。通过调用南向 API 接口，控制层可以对底层网络设备的资源进行抽象，并获取底层网络设备信息，生成全局的网络抽象视图，然后通过北向 API 提供给上层应用。

(3)转发层由网络的底层转发设备构成，包含了对特定转发平面的抽象。在 SDN 中，底层转发设备仅负责数据转发，降低了负载和对网络设备的硬件要求。

本文研究使用 OpenFlow 协议[2,23-24]作为南向接口的 SDN 网络，该网络使用 OpenFlow 协议完成控制器与 OpenFlow 交换机(以下简称交换机)的通信。基于 OpenFlow 的 SDN 可以使企业和用户具备更加灵活的网络管控能力、更高

效的资源利用能力和更弹性的资源调度能力。图 3 示出了基于 OpenFlow 的 SDN 控制器与交换机的认证流程[23]。OpenFlow 协议采用 TCP 建立连接。1）首先由交换机向控制器发起 TCP 连接请求，控制器在 6633 端口对交换机的连接请求进行监听；2）当监听到连接请求后，控制器向交换机发送 OF HELLO 响应包；3）交换机接收到该响应包后回复控制器 OF HELLO 包。至此，控制器与交换机的三次握手连接建立。然后，控制器需要查询交换机上的状态信息为后续数据包转发做准备。4）首先控制器向交换机发送 OF FEATURES_REQUEST 包，查询交换机的一些特征信息；5）交换机回复控制器 OF FEATURES_REPLY 包；6）控制器获得交换机的特征后需要再向交换机发送一些配置信息、状态查询等消息，通过 OF SET_CONFIG，OFGET_CONFIG_REQUEST，OF STATES_REQUEST 消息进行查询；7）交换机接收到上述查询包后，回复 OF GET_CONFIG_REPLY 和 OF STATES_REPLY 包；8）最后控制器再次向交换机发送 OF FLOW_MOD 包，完成查询过程。

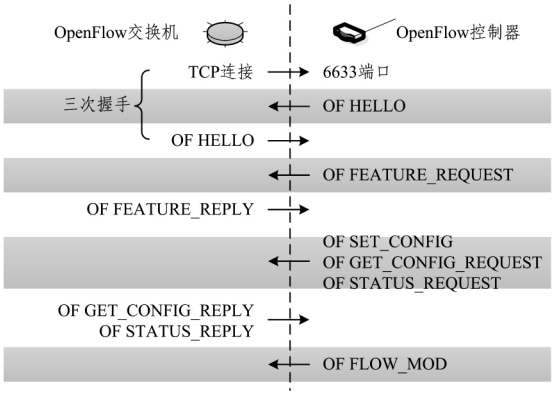

图 3　基于 OpenFlow 的 SDN 控制器与交换机的认证流程图

2.2　SDN 安全威胁

上文对 SDN 架构进行了描述，下面根据上述结果分析 SDN 面临的安全威胁[28]。

（1）控制层安全性分析

SDN 集中化的控制层承载着网络中的所有控制流，是网络服务的中枢，其安全性直接关系着网络服务的可用性、可靠性和数据安全性，是 SDN 安全首先要解决的问题[29]。在控制层中，面临的威胁包括以下几个方面。

1）单点失效。由于 SDN 采用集中控制的架构，因此控制器面临单点失效的威胁，容易成为攻击的目标，控制了控制器即控制了整个网络。

2）网络监听。攻击者可从 Internet 上获取控制器的网络切入点，对控制器上的控制信令进行伪造和修改，威胁网络资源的配置。

3）IP 地址欺骗。攻击者通过网络监听，伪造源地址为合法控制器 IP 地址的数据包，骗取交换机的信任，并向交换机下达流表修改命令，对网络进行破坏。

4）分布式拒绝服务攻击。攻击者向控制器发送多个伪造的服务请求，直至控制器因过载出现拒绝服务现象。

5）病毒、蠕虫及木马攻击。由于控制器通过软件实现，因此攻击者可通过控制器中存在的漏洞来获取控制器的控制权，执行恶意代码。

（2）应用层安全性分析

随着 SDN 的推广应用，应用层将通过应用程序提供各种复杂的网络服务，安全威胁也将随之而来[4]，主要包括以下方面。

1）恶意应用。在 SDN 中，网络操作系统之上可以运行各种应用程序，但当前缺乏保证各种应用协调一致的机制。SDN 开放的北向 REST 接口使得 SDN 应用的功能日益丰富，但同时也为病毒、蠕虫等恶意代码的生存提供了空间。攻击者可通过在应用层的应用中植入蠕虫、间谍程序等来达到窃取网络信息、更改网络配置、占用网络资源等目的，从而干扰控制面的正常工作进程，影响网络的可靠性和可用性等。

2）PC 或服务器开放环境的安全隐患。SDN 将网络的智能提升到 PC 或服务器中，以通用软件的形式运行，与传统路由器、交换机的封闭运行环境相比，PC 或服务器面临的安全风险更大。

（3）通道安全性分析

在 SDN 的架构中有南向和北向两个通道[30]。南向通道主要指 OpenFlow 协议，连接控制层和数据转发层。OpenFlow 协议的安全通道属于南向通道，协议规定需要采用 SSL/TLS 对数据进行加密，防止控制器与交换机的交互信息被外界窃取。南向通道的威胁包括以下两个方面：1）中间人攻击。由于多数常用的上层协议如 HTTPS、IMAP、SIP 等完全信任 SSL/TLS 协议，攻击者可利用上层协议实现中间人攻击。2）单一加密协议面临的安全威胁。使用单一的加密协议容易被攻击者破译，所以 OpenFlow 协议仅使用 SSL/TLS 作为安全加密协议仍存在较大的安全风险。北向通道主要指控制层向第三方应用程序开放的 API 接口，用户可以通过这些开放接口开发应用服务并在 SDN 上部署自己的应用程序，体现了 SDN 技术的开放性和可编程性。由于控制层对上层相对更加开放，因此北向通道在控制器与应用之间所建立的信赖连接关系也更加脆弱，对攻击者而言攻击门槛更低。

（4）策略安全性分析

SDN 策略运行在控制器上，由控制器将策略转变为流表项下发到底层的交换机上，交换机不具备策略分析能力，只是简单地按照策略对应的流表项匹配执行[31]。策略面临的安全威胁包括以下两个方面。

1）策略冲突。当 SDN 中有多个策略在同时执行或更新时，控制器难以对各个策略进行协调，可能发生策略冲突。

2）策略更新不一致。SDN 控制器将策略转变成流表项后，需要下发才能执行。在下发过程中，可能会由于交换机完成流表部署的时延不一致而导致逻辑不一致等问题。

3　SDN 网络攻击方法

当前厂商普遍将 SDN 与传统网络结合组成混合 SDN 网络，或者在传统网络之上构建 SDN 网络，如华为 SDN 解决方案。因此传统网络的攻击方法在 SDN 网络中仍然有效，但是 SDN 采用流表来完成数据包的匹配和控制数据包的转发，因此也存在一些只对 SDN 网络有效的攻击方法，如流表更新攻击[20]。通常网络攻击分为探测、植入、攻击 3 个步骤，本节针对上述 3 个步骤，在探测阶段讨论 SDN 网络探测的方法，在

植入阶段讨论身份伪造欺骗的手段，在攻击阶段讨论拒绝服务(Denial of Service，DoS)和信息窃取两种攻击方法。

3.1 网络探测

SDN 网络探测是指探测目标网络是否为 SDN 网络，目标网络使用的协议类型、版本，以及目标设备是控制器还是交换机。

(1)基于 OpenFlow 协议包格式的网络类型判断方法

控制器和转发节点间的通信一般采用 OpenFlow 协议，其所交互的数据包具有特定格式。目前 OpenFlow v1.3 规范针对控制器和网络交换机之间的通信通道，提供了一种使用 TLS 协议的可选安全机制，其支持基于证书的双向认证和通信加密，但这种安全机制还存在缺陷，即它允许控制通道不采取任何安全措施，也没有规定安全认证失败后所采取的应对操作[19]。因大多数厂商考虑到认证的代价，并没有采用 TLS 加密，而只是采用简单的 TCP 连接，所以攻击者通过抓包分析可以观察到所交互的数据包内容，利用该数据包格式的不同即可判断是否使用 OpenFlow 协议及版本等，进而判断 SDN 网络的存在性。

此外，控制器和交换机之间的消息共分为 3 种类型[32]：Controller-to-Switch 消息、异步(Asynchronous)消息(即 Switch-to-Controller 消息)和对称(Symmetric)消息。利用这些消息数据包格式的不同，建立不同的指纹，采集并形成包含 OpenFlow 及不同厂商独有特征的指纹库，基于此指纹库即可判断节点间是否在采用 OpenFlow 协议进行交互，进而也可实现对 SDN 网络的探测。

(2)基于包处理时延特性的网络类型判断方法

SDN 网络对不同类型的数据包的响应时间不同，因为根据不同的数据包是否在流表中存在处理该数据包的流规则，SDN 采取不同的处理措施，会导致不同的处理时间。而传统网络中则不会存在这种明显的不同[16]。利用该特性可以判断 SDN 网络的存在性。

具体而言，假设该数据包是第一次接收到的数据包(即流表中不存在处理该数据包的流规则)，则其响应时间为 $T1$，而如果该数据包是旧数据包(即流表中存在处理该数据包的流规则)，则其响应时间为 $T2$。通过度量 $T1$ 和 $T2$ 的值，可判断目标网络是否为 SDN 网络，具体流程如图 4 所示。

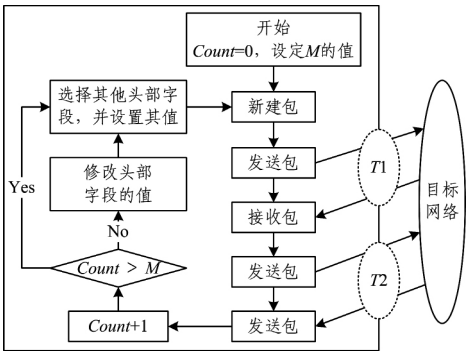

图 4 基于包处理时延的 SDN 网络存在性判断流程图

第一步 采集 $T1$ 和 $T2$ 的值。首先，发送两个或多个特殊构造的数据包到目标网络，并记录每个数据包的响应时间，将第一个数据包的响应时间记为 $T1$，将后继数据包的响应时间记为 $T2$；然后，修改数据包中的某个首部后，构建新数据包，重复上述过程。采集多个拥有不同首部值的数据包的响应时间 $T1$ 和 $T2$。

第二步 为降低噪声的影响，采集到多个不同的 $T1$ 和 $T2$ 值后，采用 SPSS 方差分析与卡方校验等统计测试方法或其他机器学习方法来区分 $T1$ 和 $T2$。

第三步 为了区分该行为模式可能与其他的安全防护措施所产生的行为相混淆，还需要将其与其他防护措施作对比，进而构建独有的行为模式，以提高精准度，降低误报率。

判断出 SDN 网络的存在性后，通过观察 SDN 网络中产生的频繁通信所体现的主从关系来判断不同组件的角色，继而再进一步探测获知 SDN 网络关键组件的部署、连接、配置等相关信息，以供后继攻击时参考使用。

表 1 SDN 网络探测方法

研究成果	探测误差 大	探测误差 小	说明
OpenFlow 协议包探测[2,9,32]		●	可探测目标网络类型及协议版本号
包处理时延探测[16]	●		网络噪声影响探测结果
控制器与交换机身份探测[16]	●		容易被攻击者误导

3.2 欺骗植入

(1)OpenFlow 交换机伪造欺骗

当两个交换机具有相同硬件地址(DPID 和 MAC 地址)和相同交换机名时，控制器无法正确区分二者。因此可以使用主机模拟出一个与目标交换机具有相同硬件特征和交换机名的虚拟交换机，利用该虚拟交换机向控制器发送 HELLO、FEATURES REPLY 等消息，模拟正常交换机与控制器的握手过程[34]。通过上述方法，如果使用其中一个交换机先与控制器连接，建立连接后使用第二个交换机再与该控制器连接，此时由于控制器中缓存的流表的生命周期还未结束，控制器会断掉与第一个交换机的连接，转而连接到第二个交换机。因此，利用上述特性，除了可以实现欺骗植入外，如果将第一个交换机视为合法交换机，第二个交换机使用主机模拟实现，视为攻击者，则攻击者可以通过上述交换机伪造的方法实现对特定链路的攻击，迫使被攻击的交换机与控制器之间的连接断开[33]。

(2)LLDP 数据包伪造欺骗

SDN 控制器使用链路发现服务(Link Discovery Service，LDS)构建全网拓扑。OpenFlow 协议调采用 LLDP(Link Layer Discovery Protocol)协议完成上述功能[35]。如控制器获取交换机 X 到交换机 Y 的单向链路发现过程如下：1)控制器首先发送一个 PACKET-OUT 消息给交换机 X，PACKET-OUT 中包含 LLDP 报文，LLDP 中又包含交换机 X 的 DPID 以及 X 的输出端口号；2)交换机 X 收到 LLDP 报文后，将该报文以广播的形式发送给 X 的所有端口，所有端口接收到该报文后再转发给与其相连接的下一跳交换机；3)当下一跳交换机 Y 收到该报文后，直接向控制器发送 PACKET-IN 消息，汇报 Y 中负责接收该报文的端口号以及 Y 的 DPID；4)若控制器接收到 Y 发来的 PACKET-IN 消息，则认为在 X 和 Y 之间存在一个单向链路(X→Y)。

攻击者可以通过截获交换机之间发送的 LLDP 数据包对其进行解码，修改其中的 DPID、PORT 等信息，实现数据包的

伪造，并将伪造后的数据包发送至目标控制器[20]，实现远程欺骗植入。

表 2 列出了 SDN 网络欺骗植入方法，并对欺骗植入的成功率进行了对比分析，由表 2 可知采用 LLDP 数据包欺骗植入的方法的成功率较高，难以被检测出。

表 2　SDN 网络欺骗植入方法

研究成果	植入成功率 高	植入成功率 低	说明
OpenFlow 交换机身份伪造植入[34]		●	无法对抗采用认证机制的控制器
LLDP 数据包欺骗植入[35]	●		通过修改底层协议数据包实现，难以检测

3.3　拒绝服务攻击

攻击者在完成网络探测和远程植入后，需要执行最终的网络攻击操作，通常包括迫使目标网络停止服务的拒绝服务攻击和从目标网络获取信息的信息窃取攻击。本节首先介绍当前 SDN 网络下实现拒绝服务攻击的手段。

(1)使用拓扑更新攻击实现 SDN 网络拒绝服务

SDN 网络使用支撑树算法来计算转发路由，当拓扑更新产生时，支撑树服务被触发，重新计算新的支撑树，删掉冗余的转发端口。利用上述特点，攻击者可以伪造 LLDP 数据包，改变交换机之间的链路，使得控制器计算出错误的支撑树，并且按照该支撑树进行转发时，数据包无法到达目的地，从而实现对 SDN 全网络的拒绝服务攻击[20]。假设攻击者俘获几个连接到接入层交换机的主机，则按照如下攻击步骤来实现对 SDN 网络的拒绝服务攻击：1)在所有俘获的主机上监听 LLDP 消息，当获得两个交换机的 DPID 值时，攻击者选择 DPID 值较小的交换机 A，使用连接到交换机 A 的主机 H 作为攻击平台；2)利用上述选择的主机 H 伪造一个 LLDP 消息，该 LLDP 消息中包含一个目的地址为目标交换机 B 的假链路；3)该 LLDP 消息发送给控制器 C[20,36]。

由于支撑算法具有如下特征：如果某条链路连接的两个交换机的 DPID 值为最大和次大，那么支撑树算法会自动删除该链路，因此上述攻击后会产生两个结果：1)处于汇聚层的交换机的 DPID 值小于前面的交换机 A，此时汇聚层交换机与目标交换机 B 之间的链路不断开，因为该链路连接的不是 DPID 值最大的两个交换机，但是由于伪造了一个指向目标交换机的链接，则等价于从目标交换机 B 上删掉一个端口，如图 5(a)所示；2)处于汇聚层的交换机的 DPID 值大于交换机 A，此时汇聚层交换机与目标交换机之间的链路断开，因为该链路连接的是 DPID 值最大的两个交换机，由于构造的链路是虚假的，因此此时支撑树算法得到的路由表中目标交换机是不可达的，如图 5(b)所示。

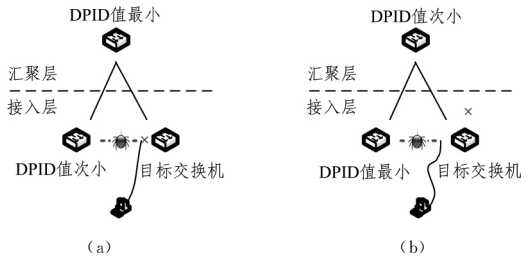

图 5　使用 LLDP 数据包伪造实现拒绝服务攻击示意图

(2)利用控制器漏洞实现拒绝服务攻击

Floodlight 控制器和交换机使用包含两种类型格式的数据结构完成握手和流表修改通知：FEATURES_REPLY 和 PORT_STATUS。前者负责应用层的握手，交换机在握手过程中使用该数据结构通知控制器自己的基本属性；后者是交换机用来通知控制器添加、修改、删除的端口情况。

由图 3 可知，FEATURES_REPLY 消息是交换机为响应控制器发出的 FEATURES_REQUEST 消息而产生的。而 Floodlight 控制器在上述握手之后若再次向控制器发送了 FEATURES_REPLY 消息，并在该消息中使用一个新的 DPID 字段，则此时控制器会在流表中增加新的表记录。上述发现说明通过修改 DPID 的方法可以使得控制器在流表中增加新的项，并且由于不是通过 PORT_STATUS 添加的，即便攻击者与控制器断开连接，上述添加的项在流表生命周期结束前也不会失效，因为上述添加的项是通过在握手之后再次发送 FEATURES_REPLY 实现的，对控制器而言，握手还没结束，因此其不会被丢弃。

利用上述漏洞，使用如下步骤可以实现对控制器的拒绝服务攻击：1)首先使用主机模拟大量的交换机；2)使用上述模拟的交换机与控制器建立连接，在每次握手完成的最后再额外使用新的 DPID 构造一个 FEATURES_REPLY 消息，并发送给控制器。上述攻击脚本执行后，每个交换机与控制器的连接都会在控制器中留下一个表记录。使用上述攻击方法，攻击一段时间后控制器的内存空间即被占满，CPU 也几乎被全负荷占用，控制器无法为正常交换机提供服务。除 Floodlight 控制器外，OpenDaylight 控制器也存在上述类似的漏洞。

表 3 列出了 SDN 拒绝服务攻击方法，由表 3 可知对 SDN 的拒绝服务攻击有两种类型：1)对 SDN 全网络的拒绝服务攻击，通过拓扑更新攻击使得 SDN 网络中充满大量无法抵达目的地的消息，从而迫使 SDN 全网络无法提供正常服务；2)利用控制器的软件漏洞实施针对控制器本身的拒绝服务，如果 SDN 网络采用分布式分域控制器部署方法，则该方法不会导致全网络的拒绝服务，但是依然能够导致控制器无法正常提供服务。

表 3　SDN 拒绝服务攻击方法

研究成果	植入成功率 高	植入成功率 低	说明
拓扑更新攻击实现 SDN 网络拒绝服务[20]	●		针对 SDN 全网络的拒绝服务
利用控制器漏洞实现拒绝服务[16,34]	●		针对 SDN 控制器的拒绝服务

3.4　信息窃取

(1)利用 PACKET-IN 消息伪造实现信息窃取

SDN 使用主机追踪服务(Host Tracking Service，HTS)[37]对 SDN 网络的节点实现更灵活的控制，如 HandOff 的某个节点。而 HTS 根据 PACKET-IN 消息更新控制器中存储的主机信息数据库。采取的步骤是：首先控制器被动监听 PACKET-IN 消息，当监听到 PACKET-IN 消息后，匹配控制器中已有的主机信息；若未匹配到对应的主机信息，则触发 JOIN 事件，将当前 PACKET-IN 中的主机信息添加到主机信

息数据库中;若匹配到对应的主机信息,但是某些选项有差异,则触发 MOVE 事件,修改对应的选项值,如 Location 值。

表 4 列出了当前主流控制器的主机信息数据库所采用的选项列表,主要使用 3 个方面的信息来标识主机:1)Mac 地址;2)IP 地址;3)位置信息,如 DPID 和端口号以及时间戳。通常仅使用 Mac 地址即可标记主机,但也有同时使用 MAC 地址和 VLAN ID 的情况,如 Floodlight。

表 4 主流控制器的主机信息数据库选项列表

控制器平台	主机信息数据库选项
Ryu	MAC,IP,Location
NOX	MAC,Location
POX	MAC,IP,Location
Floodlight	MAC,VLAN ID,IP,Location
OpenDaylight	MAC,VLAN ID,IP,Location
Beacon	MAC,VLAN ID,IP,Location

控制器会根据交换机发来的 PACKET-IN 消息触发 JION 和 MOVE 两个事件,但是并未对交换机的身份进行认证。因此可以通过 Scapy 工具伪造 PACKET-IN 消息包欺骗控制器,篡改主机信息数据库中存储的{MAC,VLAN ID,IP,Location}映射关系,实现网络钓鱼,从而达到信息窃取的目的。

(2)利用中间人攻击实现信息窃取

当控制器检测到一个新的链路建立时,会重新计算最短路径。因此攻击者可以利用 LLDP 协议伪造一个连接到两个目标交换机的最短路径[20]。按照此方法,通过这两个目标交换机的流量都将通过该链路通信,俘获该链路上的主机作为中间人攻击(Man In Middle Attacking,MIMA)主机,即可利用中间人攻击实现信息窃取,如图 6 所示。

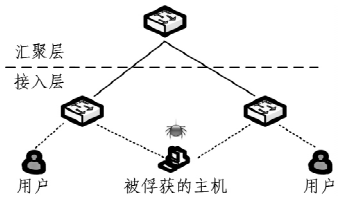

图 6 使用中间人攻击实现信息窃取示意图

表 5 列出了 SDN 信息窃取的方法,传统网络中利用中间人攻击实现信息窃取的方法在 SDN 网络中仍然适用。并且由于 SDN 采用 OpenFlow 协议完成网络管理,攻击者可以利用 OpenFlow 协议中定义的动作完成网络配置信息的篡改,实现信息窃取。

表 5 SDN 信息窃取方法

研究成果	窃取成功率		说明
	高	低	
PACKET-IN 消息伪造[37]	●		修改底层协议包,难以检测
中间人攻击[20]		●	对于采用加密机制的消息无法准确窃取

4 SDN 安全防御方法

与传统网络相比,SDN 采用集中控制、控制转发分离的模式来解决传统网络安全问题的同时也引入了新的安全威胁。本文第 3 节按照网络攻击的 3 个步骤,对攻击的各个步骤中涉及到的 SDN 网络的攻击方法进行了论述,本节在此基础上,针对上述 3 个攻击步骤分别讨论 SDN 安全防御方法。在网络探测阶段介绍当前的探测防御方法;在欺骗植入阶段介绍系统安全加固的方法,如通过系统安全加固阻断伪造欺骗类的攻击;在网络攻击阶段重点介绍当前网络攻击的防护方法,如入侵检测系统、防火墙、攻击检测等。

4.1 探测防御

与传统网络攻击类似,攻击 SDN 需要首先探测目标网络使用的协议,以及目标设备的类型等指纹信息。传统的防御手段如关闭 ping 服务等在 SDN 网络中无法应用,因为 SDN 南向接口普遍使用 OpenFlow 协议,北向接口的协议类型较多,且与 TCP/IP 协议相比,协议相对复杂,无法直接套用传统的禁止端口等方法。此外,SDN 的控制器和交换机基于虚拟化技术实现,如 Floodlight、Open vSwitch 使用软件完成硬件的功能,因此攻击者也可以通过在虚拟化软件中植入木马或者通过主机伪装成交换机进行网络探测。

针对上述问题,Jafarian 等[38]提出一种 IP 地址动态变化的方法来隐藏主机和系统资源,以对抗蠕虫的传播和攻击者的扫描行为,并且基于 OpenFlow 协议实现 OF-RHM(OpenFlow Random Host Mutation)防御框架。但是上述方法属于传统网络防御方法的改进,没有结合 SDN 的自身特性提出有针对性的防御手段。

对此,Kampanakis 等[39]引入移动目标防御思想从而提出 SDN 的抵抗拓扑发现、网络扫描、设备信息隐藏,以及主机和路由随机化的方法。如通过控制器配置对应的策略,对刺探包随机给出响应,使得攻击者无法获取准确信息,同时也能够达到迷惑攻击者的目的。也可以使用 OpenFlow 协议完成虚拟地址的映射与管理,实现控制器和交换机地址的动态变化,以增大攻击者的探测范围。

表 6 列出了 SDN 探测防御方法,由于 SDN 具备可编程和动态配置管理的特性,因此 SDN 探测防御方法普遍基于移动目标防御的思想,通过随机变化 IP 地址、端口号、网络拓扑等信息,增大攻击者的探测难度。

表 6 SDN 探测防御方法

研究成果	防御对象			
	IP 地址	端口号	主机 OS	网络拓扑
OF-RHM[38]	●	●	●	
MTD-SDN[39]	●	●	●	●

4.2 安全加固

通过安全加固可以对控制器与交换机之间的通信进行审查,从而有效对抗通过身份伪造实现的远程植入行为;也可以对 SDN 中的转发策略、控制器上的应用程序进行安全审查,从而提升 SDN 网络的稳定性和安全性。

(1)控制器安全

2015 年以前,商业控制器如 Floodlight 和 OpenDaylight 在加固控制的安全问题上,普遍采用加密认证的方法,如对控制器和交换机之间的认证过程使用 TLS 机制等,没有针对控制器安全加固的研究。

2015 年 2 月,Porras 等[14]第一次提出 SDN 安全控制器模型的概念,构建控制器安全内核(Security-Enforcement

Kernel，SEK），并在 Floodlight 控制器上实现原型 SE Floodlight。SEK 为控制器提供交换机认证管理、基于角色的身份认证、基于 Permission 模型的配置管理，以及流规则冲突解决和安全审计服务，可有效抵抗第 3 节描述的交换机欺骗植入行为。此外，管理人员可以干预由软件决定的数据包转发命令、对控制器的安全行为进行监控等。但是 SE Floodlight 负载增加过大，对控制器的性能要求太高，也使得网络的安全性对控制器的依赖度过高。

（2）南北桥通信安全

基于 OpenFlow 协议的 SDN 使用 OpenFlow 协议作为南向接口（南桥），实现控制器与交换机之间的通信，包括流表更新、配置信息交换等均通过南向接口完成。此外，控制器与上层应用程序之间通过北向接口（北桥）完成信息交互。目前尚无普遍适用的北向协议，常用的有 REST，Ruby API，Python API 等。南北桥是控制命令和网络数据传输的通道，保证南北桥的安全至关重要。

对于南桥的安全性，Khurshid 于 2012 年提出一种协议规则验证工具 VeriFlow[40]。VeriFlow 可以实现控制层与数据层之间的转发规则验证，检测其中违反规则的行为。VeriFlow 以中间代理程序的身份对控制层与数据层之间的通信内容进行解析和监管，来检测这些通信消息对网络有无负面影响。对于检测到的异常行为，使用预先配置的安全操作进行处理，其缺点是增大了控制层与数据层的通信延迟。

对于北桥的安全性，基于角色的认证方法——ForNox[41]可以确保北向接口的通信安全。ForNox 将数据流的安全性分为 3 个等级：管理员级、安全标签级、非安全标签级。采用不同安全等级的数据需要使用对应的私钥进行签名，FortNox 预先存储数字公钥，并使用公钥进行身份认证。其中交换机在接收到对应安全级别的数据流后，对管理员级安全需求的数据流赋予最高的处理优先级，对有安全标签的次之，非安全标签的数据流最后处理。

Fayazbakhsh[42]提出 FlowTags 的解决方法。FlowTags 要求对流添加标签项，使其成为交换其路由和进程的一部分。通过修改标签替代原来的修改网络中一系列的流表项或者策略信息，从而实现网络动态管理，也可以实现流在南北桥传输的可追踪，提升了南北桥的安全性。

此外，针对南北桥的脆弱性问题，OpenFlow v1.3 对安全性进行了改进，包括以下 3 个方面：1）速率限制，可以限制对控制平面的 DDoS 攻击；2）流量聚集，可以将多个流封装到一个流转发，限制了信息泄露；3）更短的流表生命周期，提高了攻击者的攻击难度。

（3）策略安全检查

SDN 网络允许动态地更新网络配置，但是由于网络普遍存在延迟，并且存在一些数据流出错等问题，如何实现策略更新的一致性是亟需解决的问题。此外，如何保证不同的配置策略之间没有冲突和漏洞也是策略安全验证需要关注的问题。

在策略一致性验证方面，Reitblatt[43]提出一种基于强语义保证的抽象网络更新方法，具体的更新操作通过更新运算来完成，为确保每个数据包和每个数据流的一致性，在更新过程中通过更新运算检查来接收到的数据包集合，对这些数据包采用全盘接收或者全盘丢弃的策略保持其一致性，在此基础上再通过形式化的验证方法对更新策略的一致性进行证明。

在策略冲突检测方面，Wang[45]针对 SDN 防火墙应用程序，基于 HSA（Header Space Analysis）构建一种策略冲突检测工具，用于检测防火墙中可能会产生冲突的策略，或者流表中存在的规则冲突。Shin 等[46]基于模型检测的方法提出一种流表入口动态插入安全性检测工具，但是其面临误报率高的瓶颈。FortNox[41]采用一种中间表示的方法完成检测，具体来说首先将当前流表的所有规则转换为 ARR（Alias Reduced Roles）的形式表示，当检测规则是否违反策略时，首先将候选规则集合（cRules）转换成 ARR 形式，然后与预定义的 ARR 规则集（fRules）进行比较，按照预定义的策略可以对每条规则都进行检测。

在策略错误检测方面，NetPlumber[47]作为一种实时策略检测工具，采用 HSA 方法，构建转发规则依赖图，使用该图对全网策略进行检查。NetPlumber 不仅可以处理 SDN 中的规则错误问题，而且对传统网络同样适用。

Wang 等[48]利用 HAS 模型构建冲突检测机制，对防火墙流规则篡改攻击进行防御。通过维护网络的影子流图，根据该图完成防火墙认证策略和流规则策略的冲突检测，当检测到冲突后，由控制器完成流表策略更新，阻塞非法流的转发。作者在 Floodlight 控制器防火墙上完成方法验证，与 VeriFlow 和 NetPlumber 的理论可行性相比，更具有实际应用价值。

（4）应用层脆弱性测试

开发者利用控制器提供的北向接口实现功能完善的应用程序，如 QoS、拓扑管理等。但是应用软件不可避免地会包含一些在具体实现上的错误，如逻辑错误、设计错误等。该类的错误如果在系统部署前没有被检测出来，将会对系统造成安全威胁。NICE[49]将传统的软件脆弱性检测方法用于 SDN 应用程序中，使用符号执行、状态模型检测依据搜索策略完成上述脆弱性检测，采用离线式分析的方法完成约束求解等操作。与上述基于符号执行的细粒度检测方法不同，Kuzniar 提出一种 OpenFlow 的黑盒测试框架 OFTEN[50]。NICE 基于简化的 OpenFlow 交换机模型来构建，而 OFTEN 基于 NICE 实现。

Shin[51]基于 FortNOX 提出了 OpenFlow 应用程序安全编程框架 FRESCO。在该框架下，开发者利用 FRESCO 提供的安全编程库可以实现安全程序的快速开发，并且可以无缝移植到 OpenFlow 应用程序中。

表 7 列出了 SDN 安全加固方法。

表 7　SDN 安全加固方法

加固对象	研究成果
控制器安全性	SE Floodlight[14]
南北桥通信	VeriFlow[40]，ForNox[41]，FlowTags[42]
策略检查	Reitblatt[43]，Wang[44]，Shin[46]，NetPlumber[47-48]
应用层脆弱性测试	OFTEN[50]，FRESCO[51]

4.3　攻击防护

从当前的研究来看，SDN 网络攻击防护手段的研究主要

集中在两个方面:1)针对 SDN 控制器进行防护;2)实现 SDN 网络的系统级防护,如入侵检测系统、防火墙等。本文分别从控制器拒绝服务攻击[55](Distributed Denial of Service Attacking,DDoS)和系统安全防护两个方面介绍 SDN 网络攻击的具体防护手段。

(1)控制器拒绝服务攻击防御

2009 年 12 月,OpenFlow 规范发布了具有里程碑意义的可用于商业化产品的 1.0 版本[52]。其规范了控制器通过流表管理 OpenFlow 交换机的具体标准,流表成为 OpenFlow 交换机转发策略控制的核心数据结构,转发芯片通过查找流表记录对进入交换机的网络数据采取相应的处理操作。OpenFlow 规范的发布也标志着 SDN 商用产品拒绝服务攻击等安全问题的产生。

Braga 等[53]于 2010 年率先在 NOX 控制器上设计一种检测分布式拒绝服务攻击的安全应用程序。控制器从网络流量中提取需要的信息,并从中提取 DDoS 特征,如一段数据流中包的平均个数和平均字节数、单个流的增长曲线等。将提取到的特征输入到自组织图(Self-Organizing Map,SOM)中,SOM 是一种人工神经网络,使用 SOM 配合机器学习的算法可以检测 DDoS 攻击的数据流。该方法已经被实测,具有较高的检测能力,同时具备较低的误报。但是受限于特征提取和机器学习算法的效率,在控制器上部署该系统后会降低控制器的处理效率。

Yao 等[55]于 2011 年提出一种基于原地址判定的解决 SDN 控制器拒绝服务攻击的方法,使用 VAVE(Virtual source Address Validation Edge)机制识别攻击者的 IP Spoof 行为。作者利用 SDN 流量审查分析和动态策略更新的特点,对于不符合转发设备预定义策略的流,将其重定向到控制器,由控制器完成原地址合法性判定;同时控制器会动态更新转发设备的策略,限制接收非法原地址的数据流。该方法存在两个缺陷:1)确信DoS 攻击不会来自于转发设备,在 SDN 中转发设备通常通过虚拟化(如 NFV)软件实现,攻击数据流可能来自于被攻击者控制的转发设备;2)极大地增大了控制器的负载,控制器一方面需要完成 IP 原地址的合法性判定,另一方面需要完成域内所有交换机的策略更新,负载和带宽开销太高。

针对 VAVE 控制器瓶颈的问题,可以使用 ident++[56]协议解决。该协议旨在降低控制器负载,将安全认证、策略更新等分摊到转发设备和主机。ident++通过在主机、转发设备、控制器,以及服务器之间架设第三方通道,当主机经过转发设备对服务器发起访问请求时,该请求被重定向到控制器,控制器使用 ident++要求主机和服务器提供额外的认证信息。认证通过后,控制器将转发的流规则下发到转发设备,由转发设备完成主机请求的转发。但是该方法有以下两个方面的不足:1)虽然可以在一定程度上解决控制器的负载瓶颈,但是需要建立控制器与服务器之间的通信通道,而 SDN 架构中不存在该通道的设计;2)扩大了网络攻击面,额外架设的主机、转发设备、控制器,以及服务器之间的第三方通道增大了攻击者对网络进行攻击的攻击面,ident++可能引入新的安全隐患。

Yeganeh 等[57]于 2012 年提出降低控制器负载的 Kandoo 框架,该框架不仅可以缓解 DDoS 攻击,还可增强网络的鲁棒性。Kandoo 采用两层控制器结构,包括底层控制器平面和顶层控制器平面,其中底层控制器平面负责单一区域内高频度的事件处理;顶层控制器平面负责管控整个 SDN 网络,处理底层控制器无法处理的事件。采用上述方法可以在一定程度上缓解控制器 DDoS 的攻击流,但是该方法仅能降低负载,无法识别 DDoS 攻击流,不能从根本上解决 DDoS 攻击的问题。

2013 年 2 月,RadWare 公司针对 SDN 架构,开发了一款可以防止拒绝服务攻击的软件 DefenseFlow[58],充分利用控制器的数据搜集功能,对攻击行为进行检测。区别于以往的只限于速率方面的检测,该软件还考虑了非速率因素(如流量分布情况)。当流量超过一定阈值,传统的检测机制会判定为攻击行为,但是这些流量可能来自不同的地点,属于正常情况。该方法是 Braga 基于 SOM 方法的升级,但是与 Braga 面临相同的问题,仍然无法解决防御导致的控制器负载急剧增加的难题。

2013 年 8 月,Kreutz[59]分析了针对 SDN 网络的 DDoS 攻击方法,攻击者通过向网络中发送未知数据流,可以使得交换机频繁向控制器发送请求,淹没其他数据流,产生拒绝服务。同时作者也提出 Replication 的解决思路,要求控制器被复制成多个,运行的应用程序也被复制成多个。这样可以抵抗或者减少软硬件故障、恶意行为带来的损失。上述方法可以在一定程度上缓解 DDoS 攻击的问题,但是维护多个控制器副本在时间和空间上的开销过大,此外 DDoS 攻击的问题并未得到正面解决。

2013 年 11 月,Seungwon 等[60]提出 AVANT-GUARD 技术。AVANT-GUARD 通过 Connection Migration 和 Actuating Triggers两项技术对数据平面以及数据到控制平面的安全性能进行了强化,用 Connetion Migration 来防止数据平面的饱和攻击;Actuating Triggers 通过给 SDN 交换机提供条件触发器来提高交换机的响应速度。AVANT-GUARD 可以防御 TCP-SYN 洪泛攻击,但是其代价是牺牲了较大的系统系能,并且无法抵御应用程序层的 DoS 攻击,也无法防御应用层使用 UDP 或者 ICMP 协议的攻击。

针对 DDoS 利用僵尸网络进行攻击的问题,Lim 等[61]提出 DBA(DDoS Blocking Application)的僵尸网络阻断方法。DBA 作为 SDN 控制器应用软件安装在控制器之上,负责监控网络中的每个流量。当检测到 DDoS 攻击时,合法流量会被重定向到另一个备用服务器上,而攻击流则会被当前控制器阻断。由于该方法需要对每个数据流进行监控,因此大规模应用受限;其次 SDN 网络中流规则是主动式安装,对控制器负载要求较高。

Ambrosin 等[62]于 2015 年提出了一种名为 LineSwitch 的方法。LineSwitch 代理来自给定 IP 的所有 TCP 连接,若发现对方发动 SYN 洪泛攻击,则将对方 IP 加入黑名单并阻止所有来自该 IP 的 TCP 数据包。该方法可在产生较小额外开销的前提下有效阻止针对 SDN 网络的 SYN 洪水攻击,且能动态适应网络结构的变化。但是该方法无法解决 IP 伪造或者多代理模式下的 DDoS 攻击。

(2)系统安全防御

在入侵检测方面，SDN 网络由于具备提取全局实时视图的能力，拥有一个逻辑上的管理中心以及可编程的接口，因此可以比较方便地部署集中控制的异常检测系统和防火墙软件。

Giotis 等[63]提出一种基于模块化的、可扩展的 OpenFlow 异常检测框架。该框架包含 3 个组成部分，其中收集模块通过 sFlow 收集网络数据包，采用流监控机制对数据包进行取样，将取样结果输入到异常检测模块进行检测。一旦异常检测模块检测到异常，迁移模块则被激活，迁移模块通过更新流表入口来完成恶意数据流的阻塞。该框架已被证实能够有效检测 DDoS 攻击、蠕虫病毒传播以及端口扫描。

Xing[64]将 Snort 的入侵检测能力与 OpenFlow 网络动态管理的能力相结合，提出一种基于 Snort 代理的主动推送机制的入侵检测系统。SnortFlow 入侵检测系统基于 Xen 的云环境实现。SDNIPS[65]对上述入侵检测能力进行了评估和改进，并与传统网络的入侵检测系统的检测能力进行了对比。

Mehdi[66]将集中防御的思路由核心服务器转移到家庭 SDN 网络中，基于 NOX 控制器提出 4 种异常检测算法并通过程序实现。实验证明，将该算法应用到 SDN 网路中能够获得比传统的基于服务器的 ADS 系统更高的准确率。

2014 年 2 月，Sergei Dotcenko 等提出一种基于模拟逻辑(Fuzzy Logic)的 SDN 信息安全管理方案[67]，基于软计算(Soft Computing)考虑信息安全管理系统相关算法，在 Java 环境下作为 Beacon 控制器的应用，实现了 SDN 入侵检测系统(IDS)的原型，包括静态收集、动态处理和决策模块。

在防火墙方面，2014 年 3 月，针对 SDN 防火墙面临的动态网络策略更新、间接安全侵害检查、有状态监视等问题，通过检查流路径空间实现流追踪机制，并采用依赖打破、升级拒绝、流移除、包阻拦等侵害解析策略，Hu 等[68]基于 Floodlight 实现了一个更可靠的 SDN 防火墙 FlowGuard。

除了上述较为复杂的防火墙外，Michelle Suh 等基于 POX 控制也实现了一个轻量级的防火墙[69]，仅通过查看数据包首部字段来决定是否允许通行。相比于上述防火墙，该轻量级防火墙处理速度更快，用户自定义更为便捷。此外，与传统网络类似，SDN 环境下也有基于二层交换机的防火墙。2014 年 6 月，Tariq Javid 等使用 Mininet 模拟环境，通过修改 POX 控制器中的二层学习型交换机(Learning Switch)代码，实现了一个 SDN 二层防火墙[70]，可根据防火墙规则控制数据流的流向。

在 SDN 的访问控制[44]方面的研究中，Dangovas 等[71]基于 Floodlight 实现了一种 SDN 认证、授权、度量的系统，用于加强 SDN 网络的安全性。该系统可以对 OpenFlow 交换机、用户进行认证和流量绑定。认证过程使用 RADIUS 服务器完成，访问控制器使用 Kerberos 和 LDAP 协议完成，但是该方法对网络性能的影响较大，并且并未明确定义如何通过 LDAP 实现访问控制。针对 AAA 的不足，Toseef 等[72]提出一种跨层模型的认证、授权管理方法 C-BAS，支持用户证书身份认证、角色授权管理等。

在攻击检测方面，除了上述的入侵检测系统、防火墙外，也有基于策略库的攻击检测系统。2015 年 2 月，Dhawan 等[12]提出一种检测 SDN 网络攻击的安全系统 SPHINX，该系统根据 OpenFlow 控制信息对网络中的数据流构建流图，再根据策略库对流图上的行为进行合法性判定，可以检测已知和未知的网络攻击。表 8 列出了从 SDN 三层架构的角度对 SDN 网络的攻击防护方法。

表 8　SDN 网络攻击防护方法

防护对象	研究成果	作用层面		
		控制层	数据层	应用层
控制器拒绝服务攻击	Kreutz[59]	●		
	VAVE[54]，Kandoo[57]，AVANT-GUARD[60]	●	●	
	Ident++[55]，DefenseFlow[58]	●	●	●
	Braga[53]		●	●
	DBA[61]	●		●
	LineSwitch[62]		●	
系统安全防护	Giotis[63]，Mehdi[66]，FLS-DN[67]，SPHINX[12]	●	●	
	Xing[64]，SDNIPS[65]		●	●
	Suh[69]，C-BAS[72]	●		●
	L2FireWall[70]		●	
	FlowGuard[68]，AAA[71]	●	●	●

5　潜在威胁与开放式解决方法讨论

5.1　SDN 潜在的脆弱点

除第 2.2 节和第 3 节的介绍外，SDN 网络仍然存在以下可能的脆弱点。

(1)Debug 调试端口

为方便网络故障检测、排除以及网络调试，OpenFlow 协议预留 Debug 端口供控制器调试使用。但是 Debug 端口并未采用加密和认证协议，用户通过简单的身份伪造即可访问 OpenFlow 交换机，通过使用 DPCTL 控制端，用户可以对交换机中的流表进行增、删、改、查的操作。当前基于 OpenFlow 的 SDN 产品多未对 Debug 端口的管理问题进行安全加固，有的甚至仅简单地对协议中的 Debug 端口进行代码实现，在未来的 SDN 安全防御中，Debug 端口问题会成为一个安全隐患。

(2)隐蔽通道

隐蔽通道是指两个主体之间的任何潜在通信都是隐蔽的，通道主要有两种类型：存储通道和定时通道[73]。若一个进程直接或间接地写一个存储单元，另一个进程直接或间接地读该存储单元，则称这种通道为隐蔽存储通道[74]。若一个进程通过调节它对系统资源的使用来影响另外一个进程观察到的真实响应时间，通过该方式实现一个进程向另一个进程传递信息，则称这种通道为隐蔽定时通道[75]。传统网络中，一些隐藏较深的木马会使用隐蔽通道与控制端通信。通信的方式多是利用协议包头中的可选字段，如 TCP/IP 协议在报文头中的可选字段写入 1 或者 0，通过多个分片组合可以实现比特流的隐蔽信息传递。

当前针对 SDN 的隐蔽通道研究未见研究成果，但是 OpenFlow 协议提供了大量的可选字段，利用这些可选字段可以实现基于存储的隐蔽通信。此外，SDN 产品基于虚拟化的软件实现，植入软件中的木马也可以通过增删流表记录实现

另一种存储隐蔽通道，如与控制端约定某个流表记录存在时传递 1，不存在时传递 0，则木马通过增删改流表记录也可以实现比特流的隐蔽传递。由于当前 SDN 的攻击手段多是为达到网络拒绝服务的目的，因此隐蔽通道问题未引起重视，但在未来 SND 安全研究中，隐蔽通道将会成为另一个安全隐患。

5.2 开放式方法

通过第 4 节对 SDN 安全防御方法的介绍可知，当前的安全防御方法多源于传统网络，如防止网络扫描探测、协议模型检测、策略合法性检查，以及防火墙和入侵检测系统等均是由传统网路演变而来。上述防御方法在一定程度上可以实现攻击防御，但存在以下两点不足：1)未充分利用 SDN 网络动态可编程的特性；2)均是被动式的防御方法，在攻击产生后才启动防护流程，具有滞后性，并且防御的策略具有静态性。

针对上述不足，未来 SDN 安全防御方面的研究可以借鉴移动目标防御的思想[38]，改变当前网络防御相对静态的特性，构建一种可以多维度变化的动态防护系统，提升攻击者的攻击难度，降低网络的可攻击面[76-77]，如 Namal 提出的 OF-HIP 模型[78]，其整合了 HIP(Host Identity Protocol)、OpenFlow，以及 IPSec 3 个协议。OFHIP 使用加密空间，在数据包转发过程中，动态修改主机和 OpenFlow 交换机的 IPv6 地址，提高网络的不确定性，增大攻击者的攻击范围。动态防御方法不仅可以充分利用 SDN 动态可编程的特性实现移动目标防御，也可以有效利用控制器集中控制的特点实现动态策略的部署、更新，同时该方法是主动防御，能够改变当前防御滞后性的不足。

结束语 SDN 是一把双刃剑，在简化网络管理、缩短创新周期的同时，也引入了不可低估的安全风险。本文对当前 SDN 攻防领域的研究进行了介绍，分析了 SDN 的脆弱点，讨论了 SDN 网络攻击和安全防御的方法。通过前文的讨论可知，当前 SDN 的攻击手段远不如防御手段丰富，SDN 的攻击普遍是针对网络中的某个网元而不是整个 SDN 网络进行的，如针对控制器、流表等进行攻击。但是攻击的结果往往会影响到整个 SDN 网络。而针对 SDN 的防御则多是通过构建多维度防护网实现，如防火墙、入侵检测系统等，均需要通过从网络中的不同网元(如交换机、转发策略等)，提取需要的特征或者规则，进而通过相应的安全策略完成网络防护。但是，这种多维度的防护网往往因为某个很小的缺陷而被攻击者突破。未来 SDN 攻防领域的研究仍有许多工作需要深入研究，也有许多研究领域需要开展。

参 考 文 献

[1] FEAMSTER N, REXFORD J, ZEGURA E. The road to SDN: an intellectual history of programmable networks [J]. ACM SIGCOMM Computer Communication Review, 2014, 44(2): 87-98.

[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] Juniper Networks. Contrail: A SDN Solution Purpose Built for the Cloud[EB/OL]. (2015-08-16) [2016-03-10]. http://www.juniper.net/us/en/products-services/sdn/contrail.

[4] Icebeen 谷歌利用 SDN 实现数据中心互联[EB/OL]. (2014-05-31) [2016-03-10]. http://www.educity.cn/net/1417699.html.

[5] 36 氪. Nicira：网络虚拟化—互联网的下一波革命[EB/OL]. (2012-04-22) [2016-03-10]. http://www.199it.com/archives/33042.html.

[6] 邹铮. Facebook 推出其“Wedge”开放数据中心交换机[EB/OL]. (2014-06-23) [2016-03-10]. http://network.chinabyte.com/225/12992725.shtml.

[7] 吴中. 华为发布 2014 系列 SDN 解决方案[EB/OL]. (2014-03-11) [2016-03-10]. http://www.wuzhongzx.com/zxzx/show.php?itemid=718892.

[8] 华为敏捷网络. HNC2015|华为发布全球首个基于 SDN 架构的敏捷物联解决方案[EB/OL]. (2015-05-21) [2016-03-10]. http://www.wtoutiao.com/p/e78rQ1.html.

[9] FABBI M. Ending the Confusion About Software-Defined Networking: A Taxonomy[EB/OL]. (2013-3-10) [2016-03-10]. http://www.gartner.com/id=2367616.

[10] JARRAYA Y, MADI T, DEBBABI M. A survey and a layered taxonomy of software-defined networking[J]. IEEE Communications Surveys & Tutorials, 2014, 16(4): 1955-1980.

[11] 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. (in Chinese)
左青云，陈鸣，赵广松，等. 基于 OpenFlow 的 SDN 技术研究[J]. 软件学报，2013，24(5)：1078-1097.

[12] DHAWAN M, PODDAR R, MAHAJAN K. SPHINX: Detecting security attacks in software-defined networks[C]// Proceedings of the 2015 Network and Distributed System Security Symposium. 2015: 69-85.

[13] SDN AP. ETSI NFV 架构解读[EB/OL]. (2013-10-20) [2016-03-10]. http://www.sdnap.com/sdnap-post/2856.html.

[14] PORRAS P, CHEUNG S, FONG M, et al. Securing the Software-Defined Network Control Layer[C]// Proceedings of the 2015 Network and Distributed System Security Symposium. 2015: 116-130.

[15] Floodlight. Floodlight Project[EB/OL]. (2014-02-21) [2016-03-10]. http://www.projectfloodlight.org/floodlight.

[16] 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. ACM, 2013: 165-166.

[17] SHU Z, WAN J, LI D, et al. Security in Software-Defined Networking: Threats and Countermeasures[J]. Mobile Networks & Applications, 2016, 10(1): 1-13.

[18] 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. (in Chinese)
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.

[19] ALSMADKI I, XU D. Security of software defined networks: A survey[J]. Computers & Security, 2015, 53(1): 79-108.

[20] HONG S, XU L, WANG H, et al. Poisoning Network Visibility

in Software-Defined Networks: New Attacks and Countermeasures[C]// Proceedings of the 2015 Network and Distributed System Security Symposium. 2015:51-67.

[21] ONF. Software-Defined Networking: The New Norm for Networks[EB/OL]. (2013-04-01) [2016-03-10]. http://www.bigswitch.com/sites/default/files/sdn_resources/onf-whitepaper.pdf.

[22] Big Switch networks. The Open SDN Architecture [EB/OL]. (2012-10-08) [2016-03-10]. http://www.bigswitch.com/sites/default/files/sdn overview.pdf.

[23] Open Networking Foundation, OpenFlow Switch Specification [EB/OL]. (2016-01-08) [2016-03-10]. https://www.opennetworking.org/sdn-resources/openflow.

[24] BOZAKOV Z, SANDER V. OpenFlow: A Perspective for Building Versatile Networks[J]. Network-Embedded Management and Applications, 2013, 12(5): 217-245.

[25] LARA A, KOLASANI A, RAMAMURTHY B. Network Innovation using OpenFlow: A Survey[J]. Communications Surveys Tutorials, 2013, 18(99): 1-20.

[26] SONKOLY B, GULYAS A, NEMETH F, et al. OpenFlow Virtualization Framework with Advanced Capabilities[C]// Proceedings of the 2012 European Workshop on Software Defined Networking. IEEE, 2012: 18-23.

[27] AZODOLMOLKY S. **软件定义网络：基于** OpenFlow **的** SDN **技术解密**[M]. **机械工业出版社**, 2014.

[28] XIA W, WEN Y, FOH C H, et al. A survey on software-defined networking[J/OL]. http:ieeexplore.ieee.org/iel7/1973917/06178210/06834762.pdf.

[29] NARISETTY R, DANE L, MALISHEVSKIY A, et al. OpenFlow Configuration Protocol: Implementation for the of Management Plane[C]// Research and Educational Experiment Workshop. 2013: 66-67.

[30] HU F, HAO Q, BAO K. A survey on software-defined network (SDN) and OpenFlow: from concept to implementation[J/OL]. http://pdfs.semanticscholar.org/6292/0d0511da12988322403e1fba98dfa3c95aa34.pdf.

[31] SHAER E, HAJ S. FlowChecker: Configuration Analysis and Verification of Federated OpenFlow Infrastructures[C]// Proceedings of the 3rd ACM Workshop on Assurable and Usable Security Configuration. ACM, 2010: 37-44.

[32] KREUTZ D, RAMOS F, ESTEVES P, et al. Software-defined networking: A comprehensive survey[J]. Proceedings of the IEEE, 2015, 103(1): 14-76.

[33] PICKETT G. Abusing Software Defined Networks [EB/OL]. (2015-10-09) [2016-03-10]. https://www.blackhat.com/us-15/briefings.html.

[34] BENTON K, CAMP L, SMALL C. OpenFlow vulnerability assessment[C]// Proceedings of the Second Acm Sigcomm Workshop on Hot Topics in Software Defined Networking. 2013: 15-21.

[35] RÖPKE C, HOLZ T. SDN Rootkits : Subverting Network Operating Systems of Software-Defined Networks[M]. Springer International Publishing, 2015: 339-356.

[36] WANG H, XU L, GU G. FloodGuard: A DoS Attack Prevention Extension in Software-Defined Networks[C]// Proceedings of Dependable Systems and Networks. IEEE, 2015: 239-250.

[37] YAP K K. n-casting using openflow [EB/OL]. (2014-10-08) [2016-03-10]. http://archive.openflow.org/wp/n-casting-mobility-using-openflow.

[38] JAFARIAN H, SHAER E, DUAN Q. Openflow random host mutation: transparent moving target defense using software defined networking[C]// Proceedings of the First Workshop on Hot Topics in Software Defined Networks. ACM, 2012: 127-132.

[39] KAMPANAKIS P, PERROS H, BEYENE T. SDN-based solutions for Moving Target Defense network protection[C]// Proceedings of the World of Wireless, Mobile and Multimedia Networks. IEEE, 2014: 1-6.

[40] KHURSHID A, ZHOU W, CAESAR M. VeriFlow: Verifying Network-Wide Invariants in Real-Time[C]// Proceedings of the first Workshop on Hot Topics in Software Defined Networks. ACM, 2012: 49-54.

[41] PORRAS P, SHIN S, YEGNESWARAN V. A Security Enforcement Kernel for OpenFlow Networks[C]// Proceedings of ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. ACM, 2012: 10-17.

[42] FAYAZBAKHSH K, SEKAR V, YU M, et al. FlowTags: enforcing network-wide policies in the presence of dynamic middlebox actions[C]// Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. ACM, 2013: 19-24.

[43] REITBLATT M, FOSTER N, REXFORD J. Consistent Updates for Software-Defined Networks: Change You Can Believe in[C]// Proceedings of the 10th ACM Workshop on Hot Topics in Networks. ACM, 2011: 71-76.

[44] MATTOS F, DUARTE B. AuthFlow authentication and access control mechanism for software defined networking[J]. Annals of Telecommunications, 2016, 10(21): 1-9.

[45] WANG J, WANG Y, ZANG L. Towards a Security-Enhanced Firewall Application for OpenFlow Networks[C]// Proceedings of Cyberspace Safety and Security. Springer, 2013: 92-103.

[46] SON S, SHIN S, GU G. Model Checking Invariant Security Properties in OpenFlow[C]// Proceedings of 2013 IEEE International Conference on Communications. 2013: 33-39.

[47] KAZEMIA P, CHANG M, WHYTE S, et al. Real time network policy checking using header space analysis[C]// Proceedings of USENIX Symposium on Networked Systems Design and Implementation. 2013: 69-74.

[48] KAZEMIAN P, CHANG M, ZENG H, et al. Real Time Network Policy Checking Using Header Space Analysis[C]// Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation. USENIX Association, 2013: 99-112.

[49] CANINI M, VENZANO D, REXFORD J, et al. A NICE way to Test OpenFlow Applications[C]// Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 2012: 10-16.

[50] KUZNIAR M, CANINI M, KOSTIC D. OFTEN Testing OpenFlow Networks[C]// Proceedings of the 2012 European Workshop on Software-Defined Networking. IEEE, 2012: 54-60.

[51] SHIN S,PORRAS P,YEGNESWARAN V,et al. FRESCO: Modular compostable security services for software-defined networks[C]// Proceedings of Network and Distributed Security Symposium. 2013:91-97.

[52] ONF. OpenFlow switch specification version 1.0.0[EB/OL]. (2012-12-31) [2016-03-10]. http://archive.openflow.org/wp/documents.

[53] BRAGA R,MOTA E,PASSITO A. Lightweight DDoS flooding attack detection using NOX/OpenFlow[C]// Proceedings of Local Computer Networks (LCN). 2010:408-415.

[54] RAMACHANDRAN S,SHANMUGAM V. Impact of DoS Attack in Software Defined Network for Virtual Network[J]. Wireless Personal Communications,2016,13(1):1-14.

[55] YAO G,BI J,XIAO P. Source address validation solution with OpenFlow/NOX architecture[C]// Proceedings of 19th IEEE International Conference on Network Protocols (ICNP). IEEE, 2011:7-12.

[56] NAOUS J,STUTSMAN R,MAZIERES D,et al. Delegating network security with more information[C]// Proceedings of the 1st ACM Workshop on Research on Enterprise Networking. ACM,2009:19-26.

[57] HASSAS S,GANJALI Y. Kandoo:a framework for efficient and scalable offloading of control applications[C]// Proceedings of the 1st Workshop on Hot Topics in Software Defined Networks. ACM,2012:19-24.

[58] RADWARE. DefenseFlow [EB/OL]. (2013-1-31) [2016-03-10]. http://www.radware.com/Products/DefenseFlow.

[59] KREUTZ D,RAMOS M,VERISSIMO P. Towards Secure and Dependable Software-Defined Networks[C]// Proceedings of the ACM SIGCOMM Workshop on Hot Topics in Software Defined Etworking (HotSDN). 2013:213-219.

[60] SHIN S,YEGNESWARAN V,PORRAS P,et al. AVANT-GUARD:scalable and vigilant switch flow management in software-defined networks[C]// Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. ACM, 2013:413-424.

[61] LIM S,HA J,KIM H,et al. A SDN-oriented DDoS blocking scheme for botnet-based attacks[C]// Proceedings of Ubiquitous and Future Networks (ICUFN). IEEE,2014:63-68.

[62] MORENO A,MAURO C,FABIO D,et al. LineSwitch:Efficiently Managing Switch Flow in Software-Defined Networking while Effectively Tackling DoS Attacks[C]// Proceedings of the 10th ACM Symposium on Information. ACM,2015:199-204.

[63] 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,2013,31(10):73-87.

[64] XING T,HUANG D,XU L,et al. Snortflow:A openflow-based intrusion prevention system in cloud environment[C]// Proceedings of the Research and Educational Experiment Workshop (GREE). IEEE,2013:89-92.

[65] XING T,XIONG Z,HUANG D,et al. SDNIPS:Enabling Software-Defined Networking Based Intrusion Prevention System in Clouds[C]// Proceedings of the International Conference on Network and Service Management. 2014:308-311.

[66] MEHDI A,KHALID J,KHAYAM A. Revisiting traffic anomaly detection using software defined networking[C]// Proceedings of the Procee-dings of the 14th International Conference on Recent Advances in Intrusion Detection. 2011:161-180.

[67] DOTCENKO S,VLADYKO A,LETENKO I. A fuzzy logic-based information security management for software-defined networks[C]// Proceedings of the 16th International Conference on Advanced Communication Technology (ICACT). IEEE,2014:167-171.

[68] HU H X,AHN G J,HAN W,et al. Towards a reliable SDN firewall[C]// Open Networking Summit 2014 (ONS). 2014:23-24.

[69] HUY H,HANZ W,AHNZ G,et al. Building robust firewalls for software-defined networks[C]// 2014 ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking (HotSDN). 2014.

[70] JAVID T,RIAZ T,RASHEED A. A layer2 firewall for software defined network[C]// 2014 Conference on Information Assurance and Cyber Security (CIACS). IEEE,2014:39-42.

[71] DANGOVAS V,KULIESIUS F. SDN-Driven Authentication and Access Control System[C]// The International Conference on Digital Information, Networking, and Wireless Communications (DINWC2014). The Society of Digital Information and Wireless Communication,2014:20-23.

[72] TOSEEF U,ZAALOUK A,ROTHE T,et al. CBAS:Certificate-based AAA for SDN experimental facilities[C]// 2014 Third European Workshop on Software Defined Networks (EWSDN). IEEE,2014:91-96.

[73] LIU X,XUE H,FENG X,et al. Design of the multi-level security network switch system which restricts covert channel[C]// Proceedings of the IEEE 3rd International Conference on Communication Software and Networks (ICCSN). IEEE,2011:233-237

[74] WANG J,WU Z,ZENG T,et al. Covert channel research[J]. Journal of Software,2010,21(9):2262-2288.

[75] LEE S,WANG H,WEATHERSPOON H. PHY covert channels:can you see the idles[C]// Proceedings of the Usenix Conference on Networked Systems Design and Implementation. USENIX Association,2014:173-185.

[76] JAJODIA S,GHOSH A K,SUBRAHMANIAN S,et al. Moving Target Defense II:Application of Game Theory and Adversarial Modeling[J]. Advances in Information Security,2012,100(1): 196-203.

[77] CASOLA V,DE BENEDICTIS A,ALBANESE M. A moving target defense approach for protecting resource-constrained distributed devices[C]// Proceedings of the Information Reuse and Integration (IRI). IEEE,2013:22-29.

[78] NAMAL S,AHMAD I,GURTOV A,et al. Enabling Secure Mobility with OpenFlow[C]// Proceedings of the IEEE Software Defined Networks for Future Networks and Services. IEEE,2013:179-204.