



基于 SDN 的网络虚拟化平台及其隔离性研究

张凯, 裘晓峰, 朱新宁

(北京邮电大学网络体系构建与融合北京市重点实验室, 北京 100876)

摘要:随着云计算的高速发展,虚拟化正成为近年来在企业级环境下广泛应用的技术。除服务器及存储虚拟化外,在 SDN 和 OpenFlow 技术的推动下,网络虚拟化成为热点,基于 SDN 的网络虚拟化平台也应运而生。在分析基于 SDN 的网络虚拟化设计目标的基础上,对近年来业界推出的 4 种典型 SDN 网络虚拟化平台的系统架构和原理进行了对比,并对其安全和隔离性进行了分析总结。

关键词:SDN;网络虚拟化;OpenFlow;隔离性

中图分类号:TP393

文献标识码:A

doi: 10.11959/j.issn.1000-0801.2016234

Research on SDN based network virtualization platform and its isolation

ZHANG Kai, QIU Xiaofeng, ZHU Xinning

Beijing Key Laboratory of Network System Architecture and Convergence, Beijing University
of Posts and Telecommunications, Beijing 100876, China

Abstract: The rapid development of cloud computing has recently prompted extensive application of virtualization in enterprise-level environments. In addition to server and storage virtualization, network virtualization attracts more and more attention with the application of SDN and OpenFlow. Therewith, software-defined network platforms designed for network virtualization were presented. Firstly, the objectives of the SDN based network virtualization platform were analyzed. Subsequently, the system architectures and principles of four typical network virtualization platforms based on SDN were compared. At last, the safety and isolation mechanism provided by the network virtualization platforms were investigated.

Key words: software defined networking, network virtualization, OpenFlow, isolation

1 引言

云计算的诞生意味着资源的存储、计算都向着虚拟化、动态化方向发展。与云计算中广泛应用的服务器虚拟化类似,网络虚拟化是基于软件抽象地从物理网络元素中分离网络流量的一种方式,它抽象隔离了网络中的交换机、网络端口、路由器以及其他物理元素的网络流量,在一

个共享的物理网络资源上构建多个可独立部署和管理的虚拟网络^[1]。

在基础网络设施中如何使虚拟网络快速适应物理网络资源的调配和负载变化,并提供稳定的网络资源响应成为网络虚拟化面临的主要问题。基于 OpenFlow 的 SDN (software defined networking, 软件定义网络)通过分离网络的控制面与转发面,并将所有网络转发设备的控制逻辑集中



到一个远程控制中心,使得网络控制转发策略可以基于全局的视角上统一决定,从而为解决这一问题提供了可能。

网络虚拟化和 SDN 都极大地增加了网络的灵活性和可编程特性。将这两种技术相结合,可以进一步增加网络资源利用的效率。同时,基于 SDN 的网络虚拟化技术实现网络切片,根据不同网络业务的需求,提供具有不同性能和不同服务能力的虚拟网络,也成为 5G 核心网演进的重要功能之一。近几年业内提出了多种基于 SDN 进行网络虚拟化的方案,加入的网络虚拟化平台作为虚拟层,可以实现构建于同一物理网络之上的多个隔绝的虚拟网络独立操作。本文针对 4 种经典的 SDN 虚拟化平台进行了研究,并对其进行安全隔离性分析。

2 基于 SDN 的网络虚拟化平台设计目标

基于 SDN 的网络虚拟化平台作为网络的虚拟层,使网络运维人员可以创建和管理多租户的 vSDN (virtual SDN, 虚拟软件定义网络),各租户通过该平台共享底层的基础设施。

基于 SDN 的网络虚拟化平台通常设计应满足如下的设计目标。

- 透明性:虚拟化平台对于底层的物理网络以及上层虚拟网络中的控制器而言,都是透明的。其目的是使上层控制器和底层设备无需关心虚拟层的操作方式,因而设计不会受到虚拟层的限制。
- 灵活性:虚拟化平台的设计应使虚拟网络始终保持灵活性,以适应动态变化的物理网络。如虚拟网络应具有处理网络故障以及虚拟节点或者虚拟链路迁移等能力。
- 扩展性:虚拟化平台要做到对于其上虚拟 SDN 网络拓扑的高度可扩展性。当虚拟网络中创建虚拟拓扑和进行扩展时,虚拟化平台应具有高效地将虚拟拓扑映射到物理拓扑的能力。
- 隔离性:网络虚拟化应使网络资源对于在物理基础设施上共存的虚拟网络之间具有隔离性。对于 SDN 中的网络资源,可以从带宽、拓扑、流量、网络设备的 CPU 和转发表 5 个方面进行资源划分和隔离。

3 基于 SDN 的网络虚拟化平台研究现状

在 SDN 网络虚拟化技术热潮的不断推动下,业界提出了多种基于 SDN 的网络虚拟化平台^[2]。主要介绍 4 种被业

界广泛关注的虚拟化平台,并对其架构和原理进行分析。

3.1 FlowVisor

3.1.1 FlowVisor 架构

与创建和运行虚拟机的管理程序 Hypervisor 在 PC 机中位于硬件和软件之间类似,FlowVisor^[3,4]使用 OpenFlow 作为硬件抽象层,处于控制平面和数据平面之间,如图 1 所示。FlowVisor 在物理硬件网络设备和多个 OpenFlow 控制器之间起着透明代理的作用。FlowVisor 使用 OpenFlow 协议控制多个 OpenFlow 控制器和交换机通信,每个控制器对应一个网络切片,而控制器和交换机并不知道 FlowVisor 的存在。图 1 中 Open Roads 和 PlugN Serve 均为构建在 NOX 网络操作系统上的虚拟网络控制器。

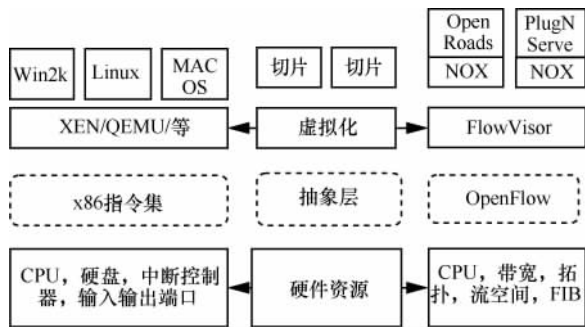


图 1 FlowVisor 在虚拟化网络中的位置

3.1.2 FlowVisor 虚拟化原理

OpenFlow 控制器给 OpenFlow 交换机发送的命令都将经过 FlowVisor 的处理以实现虚拟化网络分片的功能。FlowVisor 的操作过程如图 2 所示^[4]。

- 当 OpenFlow 控制器发送命令给 OpenFlow 交换机时,该命令将由 FlowVisor 的分片器单元进行处理。一个分片器对应一个虚拟网络控制器。

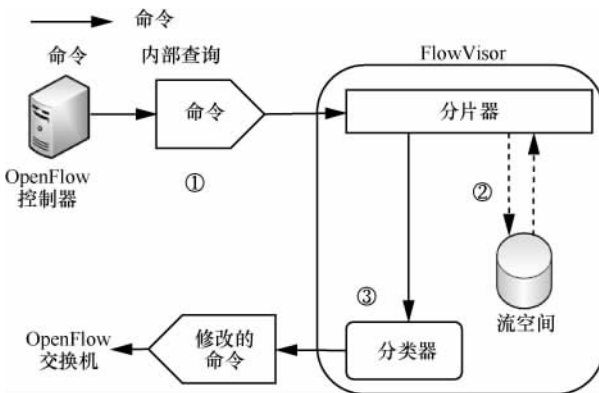


图 2 FlowVisor 的操作过程^[4]

- 分片器根据其流空间规则,对所接收到的命令进行验证,以确定是否符合该虚拟网络的规定,并在需要时对命令进行修改。
- 最终产生的命令经由分类器发送给 OpenFlow 交换机。分类器负责管理与 OpenFlow 交换机交互的命令和消息,每个 OpenFlow 交换机对应一个分类器。

3.2 VeRTIGO

3.2.1 VeRTIGO 的功能

VT(VeRTIGO)^[5]虚拟网络平台是对 FlowVisor 的功能扩展,通过增加一系列的功能模块来满足不同虚拟化场景的需求,如可以实现包含虚拟链路和虚拟节点的虚拟网络,或将整个网络抽象为单一路由器或交换机的单个虚拟节点。VeRTIGO 对物理网络的抽象化是基于两个基本的虚拟元素,即虚拟链路和虚拟端口来实现的。

- 虚拟链路由不相邻的交换机及其之间的链路构成。这种情况下 OpenFlow 控制器看到的是虚拟的网络拓扑,而不会得知其是如何由物理拓扑生成的。
- 虚拟端口由互连后不同交换机的远程端口构成。此时,OpenFlow 控制器看到的只是这个虚拟交换机节点的虚拟端口,而不知道底层是如何映射的。

3.2.2 VeRTIGO 架构及工作原理

VeRTIGO 的架构如图 3 所示。VeRTIGO 利用 OpenFlow 协议,通过其控制信道与 OpenFlow 控制器和 OpenFlow 交换机进行通信。VeRTIGO 的各个功能模块介绍如下。

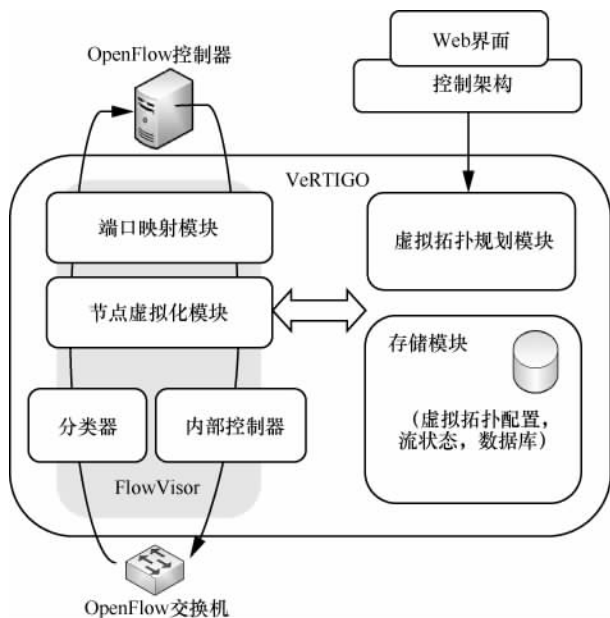


图 3 VeRTIGO 架构

(1) 分类器模块

分类器模块利用从配置文件中获得的信息对来自 OpenFlow 交换机的 OpenFlow 消息进行分类。在将网络虚拟化为单个交换机节点的场景中,该模块只转发节点的虚拟端口信息给 OpenFlow 控制器,其他的映射信息则由 VeRTIGO 的内部控制器来处理。而在将网络虚拟化为虚拟网络的场景中,只有虚拟链路的端点可以和 OpenFlow 控制器通信,而其他的节点则由 VeRTIGO 直接控制其流表,因此对 OpenFlow 控制器而言并不可见。所以 OpenFlow 控制器只获取了虚拟网络的拓扑信息,而对于实际拓扑以及映射信息完全不知,中间的处理过程均交由 VeRTIGO 内部控制器来处理。

(2) 节点虚拟化模块

将网络虚拟化为单个交换节点时, VeRTIGO 需要实现节点和 OpenFlow 控制器间的控制信道。节点虚拟化模块的作用就是将物理网络和 VeRTIGO 间的多路信道复用为一个虚拟控制信道,用于 VeRTIGO 和 OpenFlow 控制器间的信息交互。

(3) 端口映射模块

端口映射模块实现虚拟端口号与实际物理端口的映射。

(4) 内部控制器模块

内部控制器的作用是控制虚拟网络或虚拟交换节点中隐藏的交换机。这些隐藏的交换机流表由内部控制器模块下发,因而外部的 OpenFlow 控制器不会对其进行任何操作。

(5) 存储模块

存储模块用来存储虚拟节点和拓扑的配置信息,并记录流在网络中的统计信息和头字段。

(6) 虚拟拓扑规划模块

该模块实现了一个路径选择算法,用来高效地把物理节点和链路汇聚整合为虚拟拓扑。

(7) 用户接口与控制架构

提供用户接口以简化虚拟网络的配置。例如,可以配置流空间、虚拟节点、虚拟链路等信息,并将这些信息传递给虚拟拓扑规划模块来生成虚拟拓扑。

3.3 FlowN

3.3.1 FlowN 架构

图 4 所示为 FlowN^[6]的网络架构,其设计思想主要基于以下两点考虑。

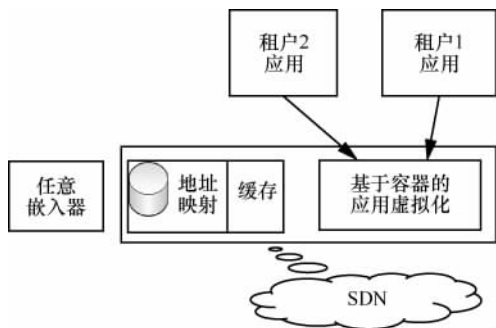


图4 FlowN 架构

- 虚拟网络租户可以通过设计自定义的控制器软件实现对地址空间的完全控制和任意的虚拟拓扑。因此 FlowN 采用一种类似于 LXC (Linux container, Linux 容器) 的基于容器的虚拟化方法, 运行共享的控制器平台 (如 NOX 网络操作系统), 而不是对每个租户运行分离的独立控制器。
- 通过使用先进的数据库技术来完成虚拟地址和物理地址空间之间的映射, 以确保虚拟节点和物理节点数增加时的可扩展性。

3.3.2 FlowN 虚拟化原理

与 FlowVisor 不同, FlowN 不是通过在底层物理交换机和租户的控制器之间利用对 OpenFlow 消息的翻译和映射来进行虚拟化, 而是实现了一个改进的 NOX 控制器, 通过运行多个 NOX 应用, 每个应用对应自己的地址空间、虚拟拓扑和事件处理程序进行虚拟化。

FlowN 实现了租户 NOX API 调用和物理网络 NOX API 调用间的映射。例如, 当一个分组到达 FlowN 控制器时, FlowN 会运行它自己的分组到达事件处理程序, 通过判断其所属租户, 调用该租户的事件处理程序。当租户的事件处理程序发起一个 API 调用时, FlowN 会拦截这个请求, 并进行虚拟网络组件与物理网络组件的映射。

3.4 OpenVirteX

3.4.1 OpenVirteX 架构

OpenVirteX(OVX)^[7]也是一个利用 OpenFlow 来生成虚拟网络的网络虚拟化平台。OpenVirteX 的系统架构如图 5 所示。生成的虚拟网络可以根据租户的需求自定义任意的网络拓扑、地址空间等网络参数。这些配置请求使用了一个和 OVX 松耦合的工具 (在图 5 中标识为 SDN 嵌入器), 通过 API 调用传送到 OVX 中。当租户想自定义一个虚拟网络时, 首先把自定义的虚拟网络配置信息请求发送给网

络嵌入器, 网络嵌入器会结合从 OVX 获取的物理网络的信息, 并根据映射算法来生成一个虚拟网络到物理网络的映射。然后再将该映射信息发送给 OVX, 由 OVX 生成基于物理拓扑的虚拟网络。

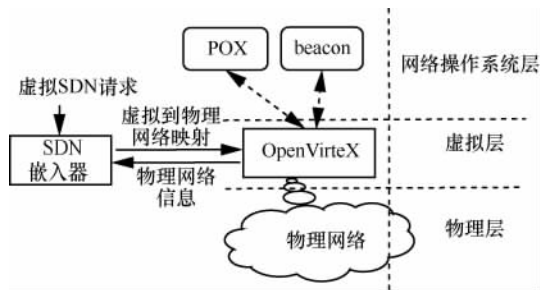


图5 OVX 架构

3.4.2 OpenVirteX 的内部结构

OVX 的内部结构如图 6 所示。在 OVX 的内部, 实现了对虚拟网络元素 (包括链路、端口、节点、地址等) 和对应的物理网络元素的解耦。虚拟网络元素和物理元素之间的映射关系是由外部的嵌入器模块根据相应的映射算法生成并由 OVX 维护。所有的虚拟网络单元被映射到至少一个物理单元。但映射本身并不规定网络虚拟化的具体实现方法。由于每个虚拟单元只是指向某个物理网络单元的指针, 因此可以在运行时禁用、使能、修改和/或配置虚拟网络单元。在网络元素被解耦和虚拟化之后, 虚拟网络和物理网络的工作就被相对隔离出来。

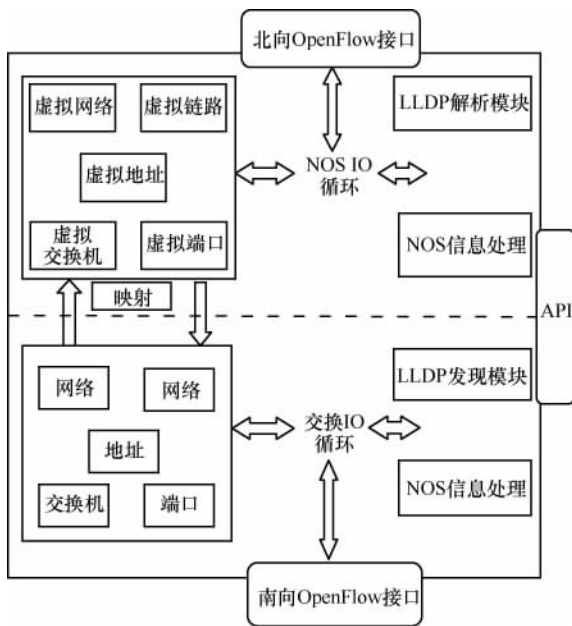


图6 OVX 内部结构

3.5 基于 SDN 的网络虚拟化平台的方案对比

表 1 对上述 4 种基于 SDN 的网络虚拟化平台的设计思路和方案进行了对比和总结。

此外,虚拟化平台的加入,最直接的影响即增大网络通信的传输时延。为了评估 4 种网络虚拟化平台的时延性能,利用 cbench 工具来模拟真实环境中“流建立”过程所占用的时延。结果显示,在虚拟网络和交换机节点从 10 逐渐增加到 100 的过程中,OVS 具有较好的稳定性,对比非虚拟化环境,其时延增加了约 0.27 ms。FlowVisor 与 FlowN 增加时延均约为 0.6 ms,但 FlowVisor 上升幅度更大^[7]。在另一组对照实验中,VerTIGO 的平均时延要比 FlowVisor 高约 0.3 ms^[5]。

4 基于 SDN 的网络虚拟化平台隔离性分析

4.1 虚拟网络流数据的隔离性

虚拟网络中的数据在进行传输中通常使用地址空间来区分不同虚拟网络的数据,虚拟化平台将网络数据传送到与之相匹配的特定虚拟网络中。

这一过程存在因地址空间隔离性问题导致的安全威胁。

比如,FlowVisor 中的虚拟网络共享了一套地址空间规则^[4],在不同的虚拟网络进行数据匹配时,可能使不同的规则匹配到相同的网络流数据,导致网络数据的隔离性被破坏。

因此,网络流数据的隔离性问题是设计网络虚拟化平台必须要考虑的问题之一。针对该问题,FlowN 和 OpenVirteX 采取了相应的改进策略,见表 2。

4.2 虚拟网络拓扑的隔离性

虚拟网络拓扑的隔离性可以分为两个方面:① 不同虚拟网络之间的拓扑隔离;② 虚拟网络与底层物理网络的拓扑隔离。

首先,对于虚拟网络之间的拓扑隔离,要做到各个虚拟网络拓扑中的节点与链路的隔离,各虚拟网络的控制器仅能得到该虚拟网络的拓扑信息。其次,对于虚拟网络与底层物理网络拓扑的隔离,应做到虚拟网络与底层物理网络的解耦,使租户可以自定义拓扑,从而摆脱物理网络中节点和链路拓扑的限制。

对于文中介绍的 4 种 SDN 网络虚拟化平台,其拓扑隔离机制见表 3。

表 1 基于 SDN 的网络虚拟化平台设计对比

对比项	FlowVisor	VerTIGO	FlowN	OpenVirteX
虚拟化方式	基于虚拟管理器	基于虚拟管理器	基于容器	基于虚拟管理器
支持数据隔离	否	否	是	是
支持自定义拓扑	否	是	是	是
使用数据库技术存储映射	否	是	是	否
网络状态可恢复	否	否	否	是
映射模块可分离	否	否	否	是

表 2 FlowN 和 OpenVirteX 的地址空间隔离策略

虚拟化平台	地址空间隔离策略
FlowN	实现了从虚拟地址到物理地址的映射。为了区分不同租户的数据,边缘交换机封装收到的数据分组,并增加一个额外的分组头,用于标识该数据属于哪个虚拟网络,该封装过程和增加的额外分组头信息对控制器和上层应用而言是透明的
OpenVirteX	租户能够为其终端主机分配地址空间,不同的租户在同一物理网络中存在 IP 地址块的重叠。为区分不同虚拟网络的主机,OVX 生成了全局标识的租户 ID,这个 ID 来自地址空间中的某一个字段。因此,根据这一字段的值,可以对数据所属的虚拟网络进行隔离

表 3 不同虚拟化平台的拓扑隔离策略

虚拟化平台	隔离性能	拓扑隔离策略
FlowVisor	①	在 LLDP 过程中,捕获底层交换机上发给控制器的分组到达消息,并且记录发送该消息的交换机和端口等信息,以此获得各虚拟网络的全局视图
VerTIGO	①②	利用映射算法,实现虚拟拓扑与物理拓扑的解耦
FlowN	①②	原理同 VerTIGO
OpenVirteX	①②	原理同 VerTIGO



4.3 物理网络交换机 CPU 的隔离性

虽然 SDN 虚拟化平台可以创建诸多虚拟网络,并在每个网络上创建互不影响的节点和链路。但实质上,这些虚拟网络中的网络资源来自于同一个底层物理网络。因此,底层物理网络资源的隔离性考虑也是一个重要因素。

物理网络硬件中的 CPU 资源是低能耗的,因此很容易发生超负荷工作的情况。在 SDN 中,占用交换机 CPU 负载的来源主要有 3 种,即新网络流建立、处理控制器的请求以及内部状态保持。不同 CPU 负载来源下的隔离威胁见表 4。

FlowVisor 针对这 3 个威胁交换机 CPU 资源的事件进行阈值设定,保证单位时间内每个事件的发生频率都在允许范围内。当超过设定的阈值时,FlowVisor 会强制结束该事件的请求,以保证交换机 CPU 资源不被过度使用。

4.4 物理网络交换机带宽的隔离性

对于底层网络的交换机而言,其带宽被上层的各个虚拟网络所共用。当某个租户占用过多带宽时,可能导致其他租户的带宽不足以满足正常的网络通信。因此,如何实现底层物理网络交换机的带宽隔离也是虚拟化平台必须解决的问题之一。

针对底层交换机带宽隔离性的要求,目前主要是利用 QoS 方案来进行解决。例如,FlowVisor 利用现有的交换机带宽隔离机制,对数据分组进行 VLAN (virtual local area network, 虚拟局域网) 优先级标记,从而实现带宽隔离。VLAN 标签有 3 bit 的 PCP(priority code point, 优权代码点) 字段,用来把数据包匹配到 8 个不同优先级的队列中。OpenFlow 协议具有管理 VLAN 标签和 PCP 字段的能力。因此可以通过对网络中的所有流进行 VLAN 标记,设置特定的优先级,确定其所属的队列。FlowVisor 重写了所有虚拟网络中的流表,增加了“设置 VLAN 优先级”的操作,使得虚拟网络中的每一条流都匹配到一个优先级队列。

除了利用 VLAN PCP 来实现隔离性以外,还可以利用 OpenFlow1.0 及以后的版本中提出的 QoS 特征和队列定

义,实现细粒度的 QoS 控制。

4.5 物理网络交换机中流表项的隔离性

虚拟化平台创建的虚拟网络共享了底层的物理网络。对于不同虚拟网络产生的流表项也使用了相同的交换机资源来进行存储。在这种情况下可能存在因流表项的隔离性问题导致的安全威胁,例如某一租户产生了大量的流表项,占用了交换机资源,使得其他租户的流表存储空间不足,从而影响其他租户正常的网络通信。

针对流表项隔离性问题,FlowVisor 的解决方案为每个虚拟网络分配一个关于流表项的预设限制。每个虚拟网络的流表项超过这个预设值时,就会收到“table full”的错误信息,从而避免了某个虚拟网络过度占用流表项资源。

4.6 实验评估

在 4 种网络虚拟化平台中,FlowVisor 最早被提出,同时很多虚拟化平台的设计和开发均在其基础上实现并进行性能提升。因此,对 FlowVisor 的隔离性进行评估具有参考意义。

首先,对 FlowVisor 的带宽隔离性能进行评估实验。利用 iperf 工具向交换机 (HP ProCurve5400) 发送 TCP 和 UDP(1 Gbit/s)流量。这两类流量作为不同租户的流量,用来测试信道被占用情况。在使用 FlowVisor 带宽隔离机制前,UDP 流量几乎占据了所有的带宽,而 TCP 只占据 1.2% 的链路带宽;在启用 FlowVisor 带宽机制,并输入最低带宽保障门限后,TCP 流量的平均带宽达到了 64.2%,UDP 流量平均带宽为 28.5%^[3]。这说明 FlowVisor 具有很好的带宽隔离性。

其次,为了评估交换机 CPU 的隔离性,分别对网络中最常见过程占用交换机 CPU 利用率进行评估。

- 检测控制器发送交换机端口状态请求时交换机 CPU 利用率: 当控制器恶意地发送过量端口状态请求时,交换机利用率在 30 s 内达到 100%,此时交换机已处于完全占用状态。当使用 FlowVisor 隔离机制后,CPU 的占用率不随控制器请求量的增加而增大,维持在 10%左右^[3]。

表 4 交换机 CPU 的隔离威胁

交换机 CPU 的隔离性威胁主要来源	威胁原理
新网络流的建立	当网络中有数据分组到达交换机并且没有匹配到任何流表项时,新流建立消息就会被发送至控制器,这个过程消耗了交换机的 CPU 资源。如果网络中新流建立过于频繁,会使交换机频繁发送新流建立消息到控制器,可能导致 CPU 资源的耗尽
处理控制器的请求	OpenFlow 控制器向交换机发送的请求,如配置流表、询问状态信息等都会消耗 CPU 资源
内部状态保持	交换机在维护内部的计数器、处理事件、更新计数器时,消耗 CPU 资源

- 检测新的数据流建立时交换机 CPU 的利用率:结果显示,发送大量流建立请求时,交换机 CPU 利用率为 100%,在使用 FlowVisor 后,CPU 平均利用率维持在 50%左右。上述实验说明 FlowVisor 提供的 CPU 隔离机制可以明显地降低交换机工作瓶颈,为多租户共享底层基础网络提供了基础保障。

最后,通过两个实验来检测 FlowVisor 流数据的隔离性^[4]。实验环境为 FlowVisor、POX 控制器以及两台电脑(3.2 GHz 英特尔内核,i7 处理器,Debian Wheezy 操作系统),并利用 iperf 工具进行流量生成。第一个实验是流量丢弃攻击,攻击方(控制器)通过修改交换机中的流规则来达到丢弃所匹配的网络流量的目的。实验结果显示,在攻击前,接收方可以收到发送方发送的 10 Mbit/s 流量。攻击发生后 5 s,接收方收到的数据迅速减少为 0。另一个实验为流量注入攻击,攻击者通过重写流规则,实现获取其他虚拟网络流量的目的。实验结果显示,在攻击发生后 5 s,攻击方接收的流量从 3 Mbit/s 增加到 10 Mbit/s。

以上两个攻击实验暴露了 FlowVisor 存在的流数据隔离性的问题,恶意控制器或其他攻击方可以通过修改流规则来达到控制网络流量的目的。由于 VeRTIGO 沿用了流数据隔离机制,因此存在同样的隔离问题。FlowN 与 OpenVirtX 分别使用不同的机制(见第 4.1 节)解决了该问题。

5 结束语

云计算对网络的灵活性和可编程特性的需求使得 SDN 和网络虚拟化技术得到了飞速的发展,基于 SDN 的网络虚拟化平台也应运而生,但是近年来对于基于 SDN 的虚拟化平台设计还不够成熟。分析了当下较为突出的 4 种 SDN 虚拟化平台的设计原理和架构,并重点分析了 SDN 虚拟化过程中需要考虑的隔离性问题。因此,在今后的基于 SDN 的网络虚拟化平台设计中,可以利用现有的虚拟化平台的设计原理和优势,并全面地考虑 SDN 虚拟化过程中存在的隔离性问题,设计出功能更全面、性能更优的基于 SDN 的网络虚拟化平台。

随着 SDN 虚拟化研究的不断深入,可以预见基于 SDN 的网络虚拟化平台的技术会更加成熟,功能更全面而系统,从而更好地提供网络基础设施服务。

参考文献:

- [1] 陶松. 基于 SDN 的网络虚拟化安全研究[J]. 电脑知识与技术, 2015, 11(5): 23-25.
- [2] TAO S. On security of SDN-based network virtualization[J]. Computer Knowledge and Technology, 2015, 11(5): 23-25.
- [3] BLENK A, BASTA A, KELLERER W. HyperFlex: an SDN virtualization architecture with flexible hypervisor function allocation[C]//IEEE International Symposium on Integrated Network Management, May 11-15, 2015, Ottawa, Canada. New Jersey: IEEE Press, 2015: 397-405.
- [4] SHERWOOD R, GIBB G, YAP K K, et al. FlowVisor: a network virtualization layer[EB/OL]. (2009-01-29) [2016-05-20]. http://www.researchgate.net/publication/238109224-Flow_Visor-A_Network.
- [5] COSTA V T, COSTA L H M K. Vulnerabilities and solutions for isolation in FlowVisor-based virtual network environments[J]. Journal of Internet Services & Applications, 2015, 6(1): 1-9.
- [6] ROBERTO D C, MATTEOG, ROBERTO R, et al. VeRTIGO: network virtualization and beyond[C]//2012 European Workshop on Software Defined Networking, October 25-26, 2012, Darmstadt, Germany. New York: ACM Press, 2012: 24-29.
- [7] DRUTSKOY D, KELLER E, REXFORD J. Scalable network virtualization in software-defined networks[J]. IEEE Internet Computing, 2013, 17(2): 20-27.
- [8] ALI A, MARC D L, AYAKA K, et al. OpenVirtX: make your virtual SDNs programmable[C]//ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, August 22-24, 2014, Chicago, IL, USA. New York: ACM Press, 2014: 25-30.

[作者简介]



张凯(1992-),男,北京邮电大学硕士生,主要研究方向为网络安全、网络虚拟化等。



裴晓峰(1969-),女,博士,北京邮电大学副教授,主要研究方向为网络安全和网络架构等。



朱新宁(1970-),女,博士,北京邮电大学副教授,主要研究方向为无线网络及其数据挖掘技术。