软件定义网络发展的理论研究

黎志勇 李 宁（中山大学信息科学与技术学院 广州 510006）

摘 要：近年来，软件定义网络SDN（Software Defined Networking）已经成为科研机构、运营商、云服务提供商、大型数据中心以及企业网用户最为关注的网络技术。SDN作为一种新兴的控制与转发分离并直接可编程的网络架构，能摆脱传统网络设备对网络控制功能的捆绑，实现网络智能管理、快速部署与维护以及灵活扩展等功能。为此，我们将回顾SDN发展的早期工作，对其体系架构与发展的阶段性研究状况进行归纳总结，并对SDN未来的发展进行展望。

关键词：软件定义网络；SDN；网络架构

1 引言

SDN作为一种新兴的基于软件技术的网络架构，是由美国斯坦福大学Cleanslate研究组提出的[1]，其最大的特点是控制平面与数据平面的松耦合性、网络状态控制的集中化支持和实现底层网络设施对上层应用的透明化。SDN具有灵活的软件编程能力，促使网络的自动化管理与控制能力得到空前的提升，可以有效解决当前网络系统面临的资源规模受限、组网灵活性不足、难以快速部署业务等问题。

在过去的几年中，SDN在业内已经获得显著发展。许多商用交换机纷纷支持SDN技术，最初的厂商包括思科、惠普、NEC和IBM，后来支持者数量急剧增多。许多不同控制器平台逐渐涌现，研究人员使用这些平台创造了很多应用，例如动态接入控制、服务器负载均衡、网络虚拟化、高效节能网络、虚拟机无缝迁移和用户移动性研究[2]。Google公司的广域流量管理系统[3]和Nicira公司的网络虚拟化平台等早期商用成功案例吸引了业界人士的高度关注，标志着SDN正式进入商用化阶段。SDN技术是未来互联网的发展方向，它改变了传统网络的静态化现状，并与以服务器虚拟化为代表的动态化发展趋势相吻合，能够为云计算、物联网、大数据、移动互联网以及更多的创新业务提供有力的网络支持[4]。

本文将从SDN发展的早期工作出发，对SDN体系架构与发展的阶段性研究状况进行归纳总结，并对SDN未来的发展进行展望。

2 SDN 体系架构

SDN技术正在改变我们设计和管理网络的方式。软件定义网络体系架构如图1所示，自底向上，SDN体系架构分为基础设施层、控制层和应用层。其中，控制层中控制软件与基础设施中的交换/路由等网络设备经由控制数据面接口(也被称为南向接口) 交互，与应用层各种 APP 经由开放应用程序编程接口API

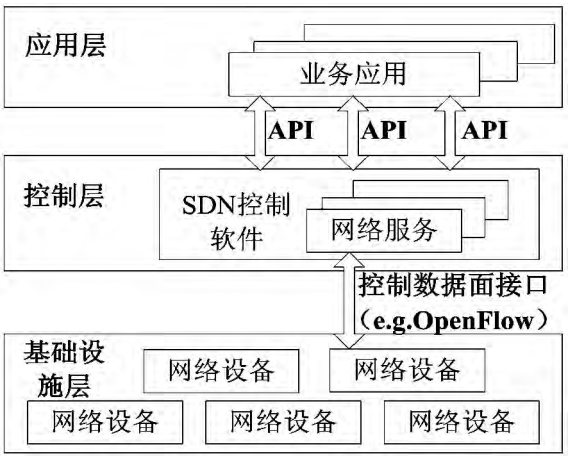

图1 SDN体系结构

(也被称为北向接口) 交互; 网络基础设施由原交换/路由设计中的转发面而抽象为数据面角色。

SDN将控制平面 (决定如何操纵数据流的走向) 和数据平面 (根据控制平面做出的决策转发数据流) 分离 ;SDN将网络环境合并成一个控制面,使得单个软件控制程序能够控制多个数据平面元素。SDN控制平面能够通过南向接口直接控制网络中数据平面元素 (比如路由器、交换机等) 的状态。OpenFlow是此类南向接口的一个突出的例子。一个OpenFlow交换机具有一个或多个具有包处理规则的转发表。每条规则匹配数据流的一个子集并执行符合规则的流量的某些动作,这些动作包括丢弃、转发和泛洪。基于导入到控制器中的规则,OpenFlow交换机能够像路由器、交换机、防火墙和网络地址转换器等网络设备一样运行。

3 SDN发展史

在过去的几年里SDN所取得的成就是有目共睹的,但是很多与SDN有关的想法已经发展了近二十年。早期电话网络将控制平面与数据平面清晰地分离,以简化网络管理和新服务的部署,SDN在某些方面正重现这种想法。然而,比起专为电话服务的封闭网络,SDN提供的开放接口能够在控制平台和应用上驱动更多的创新。在其他方面,SDN类似主动网络的研究,阐述了一个可编程网络的愿景。但SDN还涉及之前计算机网络中控制平面与数据平面分离的工作。

计算机网络的可编程化能够驱动网络管理的创新,降低部署新服务的门槛。回顾软件定义网络发展的早期工作,如图2所示,笔者将这段历史分为三个阶段:

⑴ 主动网络 (上世纪90年代中期到21世纪初)

⑵ 控制平面与数据平面分离 (大致从2001年到2007年)

⑶ OpenFlow和网络操作系统 (大致从2007年到2010年)

3.1 主动网络

上世纪90年代早中期,伴随着因特网的发展,科学家察觉应用程序的发展远远超过了文件传输和电子邮件。面向普通民众的更多样化的应用程序驱使着研究人员热切地测试和部署提高网络服务的新想法。为此,研究人员根据小型实验室设置和大型网络仿真行为来设计和测试新的网络协议。

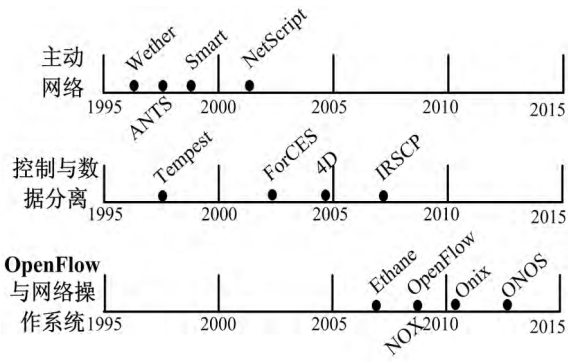

图2 SDN发展历程

对此,类比独立PC机能够重新编程,一些网络研究人员开始追求着开放网络控制的另一种方法。具体地讲,传统网络是无法实现"可编程"的。主动网络代表了一种对网络控制的激进的做法,通过设置一个可编程的接口,将资源暴露给个人网络节点,支持用户自定义功能建设并应用到通过该节点的数据包的处理。这种做法震惊了互联网社区中许多主张网络核心设备简单化的研究人员。

主动网络研究项目[5]探讨了一种传统因特网协议栈所提供服务的激进的替代方法,这些协议栈包括IP协议栈和另一种在上世纪九十年代占主导地位的异步传输模式 (ATM) 。从这种意义上说,主动网络是一系列颠覆性 (clean-slate) 网络架构方案中的开拓者,这些方案后续包括GENI、美国国家科学基金会的FIND、欧盟的FIRE。

主动网络社区[6]主要推行两种编程模式:

⑴ 胶囊模式 :运行在节点上的代码由数据包进行带内传输。

⑵ 可编程路由器/交换机模式 :运行在结点上的代码由带外机制建立。

胶囊模式与主动网络的联系最为紧密,该模式设想通过网络进行新数据平面功能的安装,在数据包中携带代码,使用高速缓存提高代码分发效率[7]。可编程路由器/交换机模式则将可扩展性的控制权直接交给网络运营人员。

3.2 控制平面与数据平面分离

在21世纪初,日益增加的网络流量和对网络可靠性、可预测性和网络性能的更高要求导致网络运营商不断寻求更好的解决特定网络管理功能的方

法，例如对路径的控制以传输数据流量（俗称“流量工程”）。这种通过常规路由协议进行流量工程的方法相当原始[8]。一些与骨干网运营商有日常交流的研究人员发现了运营商对这种方法的无奈。这驱使了研究人员决定研究要么被标准驱动，要么可以使用现有协议立即部署的务实的新方法。

具体地讲，传统的路由器和交换机等网络设备的控制平面和数据平面之间紧密集成。这种耦合使得调试配置问题、预测或控制路由行为等各种网络管理任务非常难以执行。为了应对这些挑战，将控制平面与数据平面分离的各种努力开始出现。

随着上世纪90年代互联网的蓬勃发展，骨干网的链路速度也快速增长，促使设备供应商将数据包转发逻辑直接实现在硬件上，从控制平面软件上分离出来。互联网服务提供商（ISP）也在努力寻求方法来管理规模和范围日益增大的网络，以及满足更高可靠性和新服务的需求。在这两个趋势的同时，大规模计算平台的快速发展也意味着，相比一两年前已经部署好了的路由器或交换机的控制平面处理器，现在的服务器通常具有更多的内存和处理资源。这种趋势催化了两个创新：

⑴ 控制平面与数据平面之间的开放接口，例如已经被IETF标准化了的ForCES（Forwarding and Control Element Separation）接口[9]和Linux中针对内核级数据包转发功能的Netlink接口[10]。

⑵ 网络中的逻辑中央控制，这种创新在路由控制平台[11]RCP（Routing Control Platform）、软路由架构[12]以及IETF的路径计算单元协议[13]中也出现过。

这些创新被ISP网络中管理路由的技术需求所驱动[14]。学术界也出现了一些早期的关于控制平面与数据平面分离的提案，包括ATM网络[15]和主动网络[16]。与主动网络中的早期研究相比，这些项目关注点在于网络管理中的紧迫性问题，重点是通过网络管理员，而不是终端用户和研究人员的创新；控制平面，而不是数据平面的可编程性；全网可视化和控制，而不是设备级别的配置。

网络管理应用程序包括根据当前流量负载选择最优网络路径，在规划路由变更时最小化瞬时干扰，给予用户网络更多的流量控制，重定向或丢弃可疑的网络攻击。几个运行在电信服务提供商网络的控制程序在骨干网中使用传统的路由器，虽然在此期间大部分路由管理工作由单个ISP管理，但是有些工作也会设法提出跨越多个管理域的灵活的路由控制。

此后，为了进一步加强控制平面与数据平面的分离，研究人员开始探索针对逻辑中央控制策略的颠覆性的网络架构。期间有两个项目具有代表性。一个是4D项目[17]，4D项目主张四层网络架构模型——数据平面：根据配置规则处理数据包；发现平面：收集拓扑和流量测量信息；传输平面：安装数据包处理规则；决策平面：包括能够将网络层目标转换为数据包处理状态的逻辑中央控制器。另一个是Ethane项目[18]，Ethane项目创建了用于企业网中接入控制的逻辑中央、流级别的解决方案。Ethane减少了基于高层次安全策略的控制器填充的流表切换。Ethane项目在斯坦福大学计算机系的成功部署为OpenFlow的出现设置了铺垫。尤其是Ethane中的简单交换机设计理念，后来成为了OpenFlow API的原始基础[19]。

3.3 OpenFlow和网络操作系统

2000年之后，受到实验性基础架构项目（如PlanetLab[20]和Emulab[21]）的鼓舞，研究人员和资助机构对大规模网络实验的想法产生了极大的兴趣。

在OpenFlow出现之前，SDN的相关理念面临着完全可编程网络的理想与真实世界实际部署之间的矛盾。OpenFlow通过比早期路由控制器实现更多的功能和发展现有的交换硬件，在这两个目标之间寻求平衡。虽然依托现有的交换机硬件在一定程度上限制了灵活性，但是OpenFlow几乎是可以立即部署的，这让SDN计划既务实又大胆。

OpenFlow交换机上有包含数据包处理规则的转发表，每条规则都有一个模式（在包头匹配一个比特），一个操作列表（例如丢弃、泛洪、转发到特定接口，修改包头域或发送数据包到控制器），一组计数器（跟踪字节和数据包的数量）以及一个优先级（消除重叠模式规则之间的歧义）。当收到一个数据包，OpenFlow交换机将识别最高优先级匹配规则，执行相关操作，增加计数器[22]。

在SDN范畴中，网络操作系统NOS（Network Operating System）特指运行在控制器上的网络控制平台，实际为控制软件，通过在NOS上运行不同的应用程序能够实现不同的逻辑管控功能。从整个网络来看，网络操作系统应该是抽象网络中的各种资源，为

网络管理者提供易用的接口，建立网络管理和控制的应用。目前，较为流行的NOS有NOX，Beacon，Onix等。

在基于NOX[23]的OpenFlow网络中，NOX是控制核心，NOX通过维护网络试图来维护整个网络的基本信息，如拓扑、网络单元和提供的服务，运行在OpenFlow交换机上的应用程序完成对整个网络的控制和管理。在基于Beacon[24]的OpenFlow中，采用模块化功能实现了事件和多线程操作的处理平台，可定制易于扩展的界面框架。在基于Onix[25]的OpenFlow网络中，Onix在可扩展性、可靠性方面有了很大的改进，通过多控制器分域进行管控的思想，提出一套面向大规模网络的分布式SDN部署方案，目前已经作为很多组织机构构建商业应用的基础平台。

4 总结

SDN作为一种新兴的网络体系架构，从最初的主动网络到21世纪初的控制平面与数据平面的分离，再到当今的OpenFlow和网络操作系统，其发展过程经历了20多年的变化，日趋成熟。

作为未来5年内IT领域十大关键技术之一的SDN，不仅继承现有网络技术，也可独立于现有网络技术单独发展;不仅顺从当前新的应用趋势，也符合控制、转发分离的思想。目的是对现有复杂的网络控制面进行抽象简化，使能控制面独立创新发展，使得网络面向应用可编程。

SDN未来的展望可以分为三个方面，一来是数据中心网络的部署，如何实现一个高效、稳定可控的网络一直是数据中心的研究重点；二来是面向大规模网络的部署，目前SDN部署环境主要面向校园网、企业网和数据中心，还缺乏大规模网络部署的相关经验；三来面向未来互联网研究的部署，未来互联网研究将在增加网络可控性的基础上逐渐展开,SDN技术有可能发展成为面向未来互联网的新型设计标准。

参考文献

[1] McKEOW N N, ANDERSON T,BALAKRISHNAN H,*et al*. Open-Flow:enabling innovation in campus networks[J]. ACM SIGCOMM Computer Communication Review,2008, 38(2) : 69-75

[2] Nick Feamster, Jennifer Rexford, Ellen Zegura. The Road to SDN: An Intellectural History of Programmable Networks [J]. Queue-Large-Scale Implementations. 2013,11 (12):1-13

[3] Sushant Jain, Alok Kumar, Subhasree Mandal,*et al*.B4:experience with a globally-deployed software defined wan [C]. Proceeding of the ACM SIGCOMM 2013 conference on SIGCOMM. 2013,43(4):3-14

[4] 范伟.软件定义网络及应用[J].通信技术. 2013,46(3):67-70

[5] Ken Calvert. Reflections on network architecture: an active networking perspective [J]. ACM SIGCOMM Computer Communication Review. 2006,36(2):27-30

[6] Wetherall, Guttag, John. ANTS: a toolkit for building and dynamically deploying network protocols [J]. Open Architectures and Network Programming. 1998,3(4):117-129

[7] Tennenhouse, Smith, Sincoskie. A survey of active network research [J]. IEEE Communications Magazine. 1997,35(1): 80-86

[8] 刘军，雷振明. 以太网流量工程研究[J]. 计算机工程与应用. 2002(15):8-11

[9] RFC 3746, Forwarding and Control Element Separation (ForCES)[S], IETF, 2004

[10] RFC 3549, Linux Netlink as an ISP Services Protocol[S], IETF, 2003

[11] N. Feamster, H. Balakrishnan, J. Rexford. The Case for Separating Routing from Routers [C]. Proceeding of the ACM SIGCOMM workshop on Future directions in network architecture. 2004:5-12

[12] T. V. Lakshman, T. Nandagopal, R. Ramjee, *et al*. The SoftRouter Architecture [C]. Proceeding of the 3rd ACM Workshop on Hot Topics in Networks. 2004:1-6

[13] RFC 4655, A Path Computation Element (PCE)-Based Architecture[S], IETF,2006

[14] J. Biswas, A. A. Lazar, JF Huard, *et al*. The IEEE P1520 Standards Initiative for Programmable Network Interfaces [J]. IEEE Communications Magazine. 1998,36 (10):64-70

[15] Jacobus, der Merwe, Sean Rooney. The Tempest-A Practical Framework for Network Programmability [J]. IEEE Network. 1998:20-28

[16] Jonathan M., Kenneth L., Sandra L. Activating Networks: A Progress Report [J]. 1999,32(4):32-41

[17] A. Greenberg, G. Hjalmtysson, A. Maltz, *et al*. A Clean Slate 4D Approach to Network Control and Management [J]. ACM SIGCOMM Computer Communication Review.

(下转第33页)

参考文献

[1] 马锋明等.计算机应用与软件[M]. 北京 2010
[2] 罗可,吴一凡.数据库安全[M]. 北京2008
[3] 朱鹏翔,刘文煌.基于CRM动态数据仓库．计算机工程与应用,2002
[4] 陈波,谭运猛.计算机与数据工程[M]. 北京 2005
[5] 钱雪忠,数据库系统原理[J].中国电力教育.2010

作者简介:王爱东,黑龙江省移动通信公司高级工程师,主要研究和开发数据通信网络设备软件设计与数据库系统维护的优化。郝高麟,哈尔滨电信规化设计院高级工程师,研究方向为新一代通信网络设计与维护的优化。■

(上接第29页)

2005,35(5):41-54
[18] M. Casado, J. Freedman, J. Pettit, *et al*. Ethane: Taking Control of the Enterprise [C]. Proceeding of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications. 2007,37(4):1-12
[19] M. Casado, T. Garfinkel, A. Akella, *et al*. SANE: A Protection Architecture for Enterprise Networks [C]. Proceeding of the 15th USENIX Security Symposium. 2006: 137-151
[20] Andy Bavier, Scott Karlin, Steve Muir. Operating System Support for Planetary-Scale Network Services [C]. Proceeding of the First Symposium on Networked Systems Design and Implementation. 2004:29-41
[21] B. White, J. Lepreau, L. Stoller. An Integrated Experimental Environment for Distributed Systems and Networks [C]. Proceeding of the 5th symposium on Operating systems design and implementation. 2002,36(4):255-270
[22] 左青云,陈鸣,赵广松. 基于OpenFlow的SDN技术研究[J]. 软件学报. 2013,24(5):1078-1097
[23] Gude N, Koponen T, Pettit J, Pfaff B, Casado M, McKeown N, Shenker S. Nox: Towards an operating system for networks. ACM SIGCOMM Computer Communication Review, 2008,38(3):105-110
[24] Beacon. 2012. http://www.beaconcontroller.net
[25] Koponen T, Casado M, Gude N, Stribling J, Poutievski L, Zhu M, Ramanathan R, Iwata Y, Inoue H, Hama T, Shenker S. Onix: A distributed control platform for large-scale production networks. In: Proc. of the 9th USENIX Conf. on Operating Systems Design and Implementation (OSDI).Vancouver: USENIX Association,2010.http://dl.acm.org/citation.cfm?id=1924968

作者简介：黎志勇，中山大学信息科学与技术学院硕士研究生,主要研究方向为计算机网络与信息处理;李宁,博士,副教授,主要研究方向为网络安全。■