**SDN关键技术及趋势**

**摘 要：**随着信息通信技术中大量新型业务(如移动互联网、社交网络、云计算和大数据)的出现，未来网正面临着新的挑战,而随时访问性,高带宽,动态管理是至关重要的。然而,基于专有设备手动配置的传统方法是繁琐且易出错的,而且他们不能充分利用网络基础设施的能力。最近,软件定义网络(SDN)已经被称为未来互联网最有前途的解决方案之一。SDN具有两个显著的特点,包括控制平面从数据平面中解耦并且为网络应用程序开发提供了可编程性。因此,SDN被认为能提供更有效的配置,更好的性能和更高的灵活性以适应创新的网络设计。本文总结了SDN活跃研究领域的最新进展。我们首先通过介绍SDN的起源提出一个普遍接受的SDN定义。然后我们简要的介绍了SDN逻辑架构及其技术特征。接着详细介绍了SDN关键技术及其相关领域的研究成果。最后我们描述了我们将来面临的挑战和SDN的发展趋势。

**关键词：**软件定义网络； OpenFlow；关键技术；

**Key technologies and Development of SDN**

**Abstract:** Emerging mega-trends (e.g., mobile, social, cloud, and big data) in information and communication technologies (ICT) are commanding new challenges to future Internet, for which ubiquitous accessibility, high bandwidth, and dynamic management are crucial. However, traditional approaches based on manual configuration of proprietary devices are cumbersome and error-prone, and they cannot fully utilize the capability of physical network infrastructure. Recently, software-defined networking (SDN) has been touted as one of the most promising solutions for future Internet. SDN is characterized by its two distinguished features, including decoupling the control plane from the data plane and providing programmability for network application development. As a result, SDN is positioned to provide more efficient configuration, better performance, and higher flexibility to accommodate innovative network designs. This paper surveys latest developments in this active research area of SDN. We first present a generally accepted definition for SDN with introducing the origin of SDN. We then briefly present its logical architecture and technical characteristics. We then dwell on its key technologies, and the related research results. Finally, we describe the challenges we face and discuss future research directions of this technology.

**Key words:** Software-defined networking, OpenFlow. Key technologies

**引言**

随着社交网络、移动互联网、物联网、云计算[1]等业务领域的快速发展，大数据[3][4]正日益成为当前的焦点，其面向的海量数据处理也对网络提出了更高的要求。大数据应用依赖于预先定义好的计

算模式，在集中化的管理架构下运行，存在着大量的数据批量传输及相关的聚合/划分操作。数据的聚 合和划分通常发生在一台服务器和一个拥有众多

服务器的服务器组之间，这也是大数据应用中最典型的网络流量模式。例如，在用于大数据处理的MapReduce算法的执行过程[2]中，来自众多mapper服务器的中间结果需要集中汇总到一台reducer服务器上进行归约（Reduce）操作，而MapReduce的洗牌（Shuffle）过程更是由mapper和reducer之前的多次数据聚合组合而成。大数据处理过程中的每一次聚合都将导致大量服务器之间的海量数据交换，从而需要极高的网络带宽支持，而如果按照超额认购（oversubscribe）带宽的方式为每台服务器预留网络资源，将导致网络成为瓶颈，同时造成资源浪费。因此，对于大数据业务而言，他更需要对网络进行快速、频繁的实时配置，按需调用网络资源。

但是，传统的网络却难以满足云计算、大数据，以及相关业务提出的灵活的资源需求，这主要是因为它已经过于复杂从而只能处于静态的运作模式。当前，网络中存在着大量各种各样的互不相干的协议，它们被用于在不同间隔距离、不同链路速度、不同拓扑结构的网络主机之间建立网络连接。因为历史原因，这些协议的研发和应用通常是彼此分离的，每个协议通常只是为了解决某个专门的问题而缺少对共性的抽象，这就导致了当前网络的复杂性。正是因为上述的复杂性，传统网络通常都是维持在相对静态的状态，网络管理员通常都要尽可能地减少网络的变动以避免服务中断的风险。

正是在这一背景下，业内形成了“现在是创新思考互联网基本体系结构、采用新的设计理念的时候”的主流意见[3]，并对未来网络的体系架构提出了新的性质和功能需求[4]。软件定义网络[5](Software-Defined Networking，SDN)的出现为人们提供了一种崭新的思路。

本文从SDN起源和概念出发，分析了SDN的逻辑架构与关键技术，在此基础上提出了SDN技术在未来的发展中面临的挑战并总结了可能的研究方向。

**1．SDN起源与概念**

**1.1 SDN起源**

SDN起源于斯坦福[6]大学启动的名为Clean Slate Design for the Internet项目。Clean Slate项目的最终目的是要重新发明英特网，旨在改变设计已略显不合时宜，且难以进化发展的现有网络基础架构。在2006年，斯坦福的学生Martin Casado领导了一个关于网络安全与管理的项目Ethane[7]，该项目试图通过一个集中式的控制器，让网络管理员可以方便地定义基于网络流的安全控制策略，并将这些安全策略应用到各种网络设备中，从而实现对整个网络通讯的安全控制。受此项目(及Ethane的前续项目Sane)启发，Martin和他的导师Nick McKeown教授(时任Clean Slate项目的Faculty Director)发现，如果将Ethane的设计更一般化，将传统网络设备的数据转发(data plane)和路由控制(control plane)两个功能模块相分离，通过集中式的控制器(Controller)以标准化的接口对各种网络设备进行管理和配置，那么这将为网络资源的设计、管理和使用提供更多的可能性，从而更容易推动网络的革新与发展。于是，他们便提出了OpenFlow的概念，并且Nick McKeown等人于2008年在ACM SIGCOMM发表文献[8]，首次详细地介绍了OpenFlow的概念。该篇论文除了阐述OpenFlow的工作原理外，还列举了OpenFlow几大应用场景，其核心思想是将传统网络设备的数据转发（Data Plane）面和路由控制（Control Plane）面相分离，通过集中控制器（controller）以标准化接口对各种网络设备进行配置管理。这种网络架构为网络资源的设计、管理和使用提供了更多的可能性，从而更容易推动网络的革新与发展。由于OpenFLow 开放了网络编程能力，因此 Ethane 被认为是 SDN技术[8]的起源。

**1.2 概念**

SDN的产生与OpenFlow协议密切相关。现在业界普遍将基于OpenFlow协议的SDN视为狭义SDN。这一概念也是业界的默认概念。

随着SDN的发展，越来越多的厂商加入SDN的研究行列。由于不同行业、不同应用对SDN有着各自不同的需求，出现了许多各具特征的SDN定义[9]-[13]。在网络科研领域，利用SDN快速地部署和试验创新的网络架构与通信协议；大型互联网公司希望SDN提供掌握网络深层信息的可编程接口．以优化和提升业务体验；云服务提供商希望SDN提供网络虚拟化和自动配置，以适应其扩展性和多租户需求；ISP希望利用SDN简化网络管理以及实现快速灵活的业务提供；企业网用户希望SDN实现私有云的自动配置和降低设备采购成本。基于这些需求，在思科等厂商的推动下，IETF、IEEE等标准组织去除了SDN OpenFlow的必然联系，保留了可编程特性，从而扩展出SDN的广义概念。即泛指基于开放接口实现软件可编程的各种基础网络架构，进而将具备控制转发分离、逻辑集中控制、开放API 3个基本特征的网络纳入SDN的广义概念中。

**2. SDN逻辑架构及技术特点**

**2.1 SDN逻辑架构**

开放网络基金会（ONF）[14]是一家非营利的组织机构，成立于2011年。OFN致力于SDN的发展和标准化，是当前业界最活跃、规模最大的SDN标准组织。ONF提出的SDN架构如图1所示。

图1 SDN 的逻辑架构

图 1 为 SDN 的逻辑架构[14]。该逻辑架构分为3层。

基础设施层（Infrastructure Layer）主要由网络设备（Network Device）即支持0penFlow协议的SDN交换机组成，它们是保留了传统网络设备数据面能力的硬件，负责基于流表的数据处理、转发和网络状态收集。

控制层（Control Layer）主要包含0penFlow控制器及网络操作系统(network operation system，NOS)，负责处理数据平面资源的编排、维护网络拓扑、状态信息等；控制器是一个平台，该平台向下可以直接与使用OpenFlow协议的交换机(以下简称SDN交换机)进行会话；向上，为应用层软件提供开放接口，用于应用程序检测网络状态、下发控制策略。

位于顶层的应用层（Application Layer）由众多满足用户需求的应用软件构成，这些软件能够根据控制器提供的网络信息执行特定控制算法，并将结果通过控制器转化为流量控制命令，下发到基础设施层的实际设备中。从而完成动态接入控制、无缝切换、负载均衡和网络虚拟化等功能。

SDN 网络控制器与网络设备之间通过专门的控制面和数据面接口连接，该接口是支持 SDN 技术实现的关键接口。目前，SDN 的研究重点之一是对该接口的定义和规范，很多研究将该接口等同于现有网络中用于管理不同厂商设备的南向接口（Southbound Interface），但重新定义了其需要承担的功能，如网络编程、资源虚拟化、网络隔离等；同时，在应用层与网络基础设施层之间定义了类似于传统网络设备上用于设备制造商或网络运营商进行设备接入和管理的北向接口（North-bound Interface），并明确了该接口在路由、网络设备管理、网络策略管理等方面的能力要求。此外，为支持不同的网络控制系统之间的互通，有研究还定义了支持网络控制系统之间互联的东西向接口（East-west Interface）和其在支持网络域间控制、互操作、网络部署等方面的功能需求[15][16]。根据上述论述，OpenFlow协议、网络虚拟化技术和网络操作系统是SDN区别于传统网络架构的关键技术。

**2.2 技术特点**

SDN 的出现打破了传统网络设备制造商独立而封闭的控制面结构体系，将改变网络设备形态和网络运营商的工作模式，对网络的应用和发展将产生直接影响。从技术层面分析，SDN的特点主要体现在以下几个方面[17]：

* 数据面与控制面的分离，简化了网络设备，通过控制面功能的集中和规范数据面和控制面之间的接口，实现对不同厂商的设备进行统一、灵活、高效的 管理和维护。
* 开放网络编程能力，以API 的形式将底层网络能力提供给上层，实现对网络的灵活配置和多类型业务的支持，提高对网络和资源控制的精细化程度。
* 支持业务的快速部署，简化业务配置流程，具有灵活的网络扩展能力，降低设备配置风险，提高网络运营效率。
* 更好地支持用户个性化定制业务的实现，为网络运营商提供便捷的业务创新平台。
* 实现网络的虚拟化，将传输、计算、存储等能力融合，在集中式控制的网络环境下，有效调配网络资源支持业务目标的实现和用户需求，提供更高的网络效率和良好的用户体验。

**3. SDN关键技术**

**3.1 OpenFlow**

OpenFlow标准的名称是OpenFlow Switch Specification，因此它本身是一份设备规范，其中规定了作为SDN基础设施层转发设备的OpenFlow交换机的基本组件和功能要求，以及用于远程控制器对交换机进行控制的OpenFlow协议。本文主要以OpenFlow v1.0 为主介绍OpenFlow协议的基本架构，OpenFlow协议搭建SDN的设计思想和整体架构如图2所示。



图2 OpenFlow交换机的设计思想和整体架构

如图2所示，OpenFlow交换机利用基于安全连接的OpenFlow协议与远程控制器相通信[18]。其中，流表（Flow Table）是OpenFlow交换机的关键组件，负责数据包的告诉查询和转发。另外，OpenFlow交换机还需要通过一个安全通道与外部的控制器进行通信，这个安全通道上传输的是OpenFlow协议，它将负责传递控制器和交换机之间的管理和控制信息。因此，流表、安全通道、及OpenFlow协议是OpenFlow v1.0的核心部分。

**3.1.1 流表**

所谓流表，其实可被视为是OpenFlow对网络设备的数据转发功能的一种抽象。在传统网络设备中，交换机和路由器的数据转发需要依赖设备中保存的二层MAC地址或者三层IP地址路由表，而OpenFlow交换机中使用的流表也是如此，不过在它的表项中整合了网络中各个层次的网络配置信息，从而在进行数据转发时可以使用更丰富的规则。OpenFlow流表的每个表项由3部分组成：用于数据包匹配的包头域（Header Fields），用于统计匹配数据包个数的计数器（Counters），用于展示匹配的数据包如何处理的动作（Actions）。

随着OpenFlow的演进，匹配域的覆盖范围越来越广，以满足更灵活的转发策略。随着版本的升级，匹配域的数量不断增加，应用方式也进行了调整。其相应变化如表1所示。

表1 OpenFlow流表匹配域的变化示意

|  |  |  |
| --- | --- | --- |
| 规范  版本 | 匹配域数量 | 匹配域主要变化 |
|
| v1.0 | 12 | 入端口 |
| 以太网：源，目的，类型，VLAN |
| IPv4:源，目的，协议，ToS位 |
| TCP/UDP:源端口，目的端口 |
| v1.1 | 15 | 元数据，SCTP, VLAN标记 |
| MPLS:标签，数据类型 |
| v1.2 | 36 | OpenFlow扩展匹配(OXM) |
| IPv6：源，目的，流标签，ICMPv6 |
| v1.3 | 39 | PBB，IPv6扩展字段 |
|
| v1.4 | 43 | - |
|

如表所示,每个版本的OpenFlow规范引入了新的匹配字段，包括以太网,IPv4 / v6,MPLS,TCP / UDP等[19]-[23]。然而,只有部分匹配字段是必选的。同样,许多动作和端口类型是可选的。通过一个基于type-length-value(TLV)格式的OpenFlow扩展匹配(OXM)在OpenFlow version 1.2中引入可扩展性功能后使得不再需要添加更多新的匹配字段[20]。为了提高整体协议的可扩展性,OpenFlow version 1.4中，TLV格式也已经添加到端口、流表和队列中[23]。

**3.1.2 安全通道**

OpenFlow采用的是集中控制方式，控制器需要利用OpenFlow协议对交换机进行流标的配置，因此在它们之间传送信息的通道非常重要。通道是连接OpenFlow交换机到控制器的接口，控制器通过这个接口管理和控制OpenFlow交换机，同时也通过这个接口接收来自OpenFlow交换机的消息。

在具体的通道实现中，OpenFlow v1.0要求承载OpenFlow协议传送的通道必须是安全的，并规定通道需要采用TLS（Transport Layer Security，安全传输协议）技术。

**3.1.3 OpenFlow协议**

OpenFlow协议是用来描述控制器和OpenFlow交换机之间交互所用的信息接口标准，其核心是OpenFlow协议信息的集合。OpenFlow协议支持三种消息类型：controller-to-switch、asynchronous（异步）和symmetric（对称）。而每一类消息又可以拥有多个子消息类型。其中，controller-to-switch消息由控制器发起，用来管理或获取OpenFlow交换机状态；asynchronous消息由OpenFlow交换机发起，用来将网络事件或交换机状态变化更新到控制器；symmetric 消息可由交换机或控制器发起。

**3.2 SDN交换机及南向接口技术**

SDN的核心理念之一就是将控制功能从网络设备中剥离出来，SDN交换机只负责网络高速转发，保存的用于转发决策的转发表信息来自控制器，SDN交换机需要在远程控制器的管控下工作，与之相关的设备状态和控制指令都需要经由SDN的南向接口传达，从而实现集中化统一管理、资源的优化利用，提升网络管控效率。

工作在基础设施层的SDN交换机虽然不再需要对逻辑控制进行过多考虑，但作为SDN网络中负责具体数据转发处理的设备，为了完成高速数据转发，还是要遵循交换机工作原理。本质上看，传统设备中无论是交换机还是路由器，其工作原理都是在收到数据包时，将数据包中的某些特征域与设备自身存储的一些表项进行比对，当发现匹配时则按照表项的要求进行相应处理。SDN交换机也是类似的原理，但是与传统设备存在差异的是，设备中的各个表项并非是由设备自身根据周边的网络环境在本地自行生成的，而是由远程控制器统一下发的，因此各种复杂的控制逻辑(例如链路发现、地址学习、路由计算等等)都无需在SDN交换机中实现。

SDN交换机可以忽略控制逻辑的实现，全力关注基于表项的数据处理，而数据处理的性能也就成为评价SDN交换机优劣的最关键指标。另外，考虑到SDN和传统网络的混合工作问题，支持混合模式的SDN交换机也是当前设备层技术研发的焦点。同时，随着虚拟化技术的出现和完善，虚拟化环境将是SDN交换机的一个重要应用场景，因此SDN交换机可能会有硬件、软件等多种形态。例如，OVS(Open vSwitch，开放虚拟交换标准)交换机就是一款基于开源软件技术实现的能够集成在服务器虚拟化Hypervisor中的交换机，具备完善的交换机功能，在虚拟化组网中起到了非常重要的作用。

**3.3 SDN控制器及北向接口技术**

控制层是SDN的大脑，负责对底层转发设备的集中统一控制，同时向上层业务提供网络能力调用的接口，在SDN架构中具有举足轻重的作用，SDN控制器也是SDN关注的焦点。从技术实现上看，控制器除了南向的网络控制和北向的业务支撑外，还需要关注东西的扩展，以避免SDN集中控制导致的性能和安全瓶颈问题，SDN控制器也在南向、北向、东西向上引入了相应的核心技术，有效解决与各层通信以及控制集群横向扩展的难题。

当前，业界有很多基于OpenFlow控制协议的开源的控制器实现，例如OMNI [24], Trema [25], Ryu[26], Floodlight [27], NOX [28], [29], and OpenDaylight[30]等，它们都有各自的特色设计，能够实现链路发现、拓扑管理、策略制定、表项下发等支持SDN网络运行的基本操作。虽然不同的控制器在功能和性能上仍旧存在差异，但是从中已经可以总结出SDN控制器应当具备的技术特征，从这些开源系统的研发与实践中得到的经验和教训将有助于推动SDN控制器的规范化发展。

另外，用于网络集中化控制的控制器作为SDN网络的核心，其性能和安全性非常重要，其可能存在的负载过大、单点失效等问题一直是SDN领域中亟待解决的问题。当前，业界对此也有了很多探讨，从部署架构、技术措施等多个方面提出了很多有创见的方法。文献[31]-[33]通过不同SDN控制器部署策略来避免单点失效问题。但是仍存在三个主要限制因素。针对文献[31]-[33]存在的问题，文献[34]提出了Survivor，一个新颖的控制器部署策略。

SDN控制器对网络的控制主要是通过南向接口协议实现,包括链路发现、拓扑管理、策略制定、表项下发等，其中链路发现和拓扑管理主要是控制其利用南向接口的上行通道对底层交换设备上报信息进行统一监控和统计;而策略制定和表项下发则是控制器利用南向接口的下行通道对网络设备进行统一控制。

SDN北向接口是通过控制器向上层业务应用开放的接口，其目标是使得业务应用能够便利地调用底层的网络资源和能力。通过北向接口，网络业务的开发者能以软件编程的形式调用各种网络资源;同时上层的网络资源管理系统可以通过控制器的北向接口全局把控整个网网络的资源状态，并对资源进行统一调度。因为北向接口是直接为业务应用服务的，因此其设计需要密切联系业务应用需求，具有多样化的特征。同时，北向接口的设计是否合理、便捷，以便能被业务应用广泛调用，会直接影响到SDN控制器厂商的市场前景。文献[35]设计了新的网络编程语言和配置脚本，实现对SDN网络的自动化管理；文献[36]研究了如何在与云控制器集成中给出统一的云编排系统实践方案。

与南向接口方面已有OpenFlow等国际标准不同，北向接口方面还缺少业界公认的标准，因此，北向接口的协议制定成为当前SDN领域竞争的焦点，不同的参与者或者从用户角度出发，或者从运营角度出发，或者从产品能力角度出发提出了很多方案。据悉,目前至少有20种控制器,每种控制器会对外提供北向接口用于上层应用开发和资源编排。虽然北向接口标准当前还很难达成共识，但是充分的开放性、便捷性、灵活性将是衡量接口优劣的重要标准，例如REST API就是上层业务应用的开发者比较喜欢的接口形式。部分传统的网络设备厂商在其现有设备上提供了编程接口供业务应用直接调用，也可被视作是北向接口之一，其目的是在不改变其现有设备架构的条件下提升配置管理灵活性，应对开放协议的竞争。

控制器负责整个SDN网络的集中化控制，对于把握全网置资源视图、改善网络资源交付都具有非常重要的作用。但控制能力的集中化，也意味着控制器局的安全性和性能成为全网的瓶颈;另外，单一的控制器也无法应对跨多个地域的SND网络问题，需要多个SDN控制器组成的分布式集群，以避免单一的控制器节点在可靠性、扩展性、性能方面的问题。目前，用于多个控制器之间沟通和联系的东西向接口还没定义标准，但专家表示，一些非常成熟的集群技术可以被运用到SDN网络中来解决上述难题。

**3.4 网络虚拟化技术**

随着服务器、桌面、应用、存储等虚拟化技术的广泛应用，网络虚拟化成为云计算和数据中心技术发展的迫切需求。网络虚拟化的目的是为了在共享的同一物理网络资源上划出逻辑上独立的网络，以满足多租户、流量隔离和逻辑网络自由管控的应用趋势。开源项目FlowVisor[37][38]从Network Hypervisor角度出发实现网络虚拟化，通过划分流表空间产生独立的网络分片。各个网络分片上的网络流量是相互隔离的，用户可在各个分片上进行互不干扰的各种流量模型和协议创新等实验研究。现在FlowVisor已经被广泛应用到多个研究机构的实验平台上，并在全球OpennetSummit(开放网络峰会)上为各种SDN创新应用提供了共享同一套物理网络资源的演示环境。Koponen et al.基于SDN技术提出了一个网络虚拟平台（network virtualization platform，NVP）[39]实现多用户共享数据中心。

**3.5 应用编排和资源管理技术**

SDN网络的最终目标是服务于多样化的业务应用创新。因此随着SDN技术的部署和推广，将会有越来越多的业务应用被研发，这类应用将能够便捷地通过SDN北向接口调用底层网络能力，按需使用网络资源。

SDN推动业务创新已经是业界不争的事实，它可以被广泛地应用在云数据中心、宽带传输网络、移动网络等种种场景中，其中为云计算业务提供网络资源服务就是一个非常典型的案例。众所周知，在当前的云计算业务中，服务器虚拟化、存储虚拟化都已经被广泛应用，它们将底层的物理资源进行池化共享，进而按需分配给用户使用。相比之下，传统的网络资源远远没有达到类似的灵活性，而SDN的引入则能够很好地解决这一问题。

SDN通过标准的南向接口屏蔽了底层物理转发设备的差异，实现了资源的虚拟化，同时开放了灵活的北向接口供上层业务按需进行网络配置并调用网络资源。云计算领域中知名的OpenStack[40]就是可以工作在SDN应用层的云管理平台，通过在其网络资源管理组件中增加SDN管理插件，管理者和使用者可利用SDN北向接口便捷地调用SDN控制器对外开放的网络能力。当有云主机组网需求(例如建立用户专有的VLAN)被发出时，相关的网络策略和配置可以在OpenStack管理平台的界面上集中制定并进而驱动SDN控制器统一地自动下发到相关的网络设备上。

因此，网络资源可以和其他类型的虚拟化资源一样，以抽象的资源能力的面貌统一呈现给业务应用开发者，开发者无需针对底层网络设备的差异耗费大量开销从事额外的适配工作，这有助于业务应用的快速创新。

**4. 总结与展望**

虽然SDN相对传统网络具有诸多优势，代表着网络发展的方向，但目前仍存在一系列尚待解决的问题。

**4.1 SDN面临的挑战**

(1)运算压力大、软件复杂度高、系统稳定性存在隐患

SDN架构下．控制器需要为每一条流制定优化的路由策略．其运算压力之大可想而知，且这一压力会随控制网元数量的增加呈几何级上升。此外，由于不同应用会在SDN系统建立不同的逻辑网络，各应用程序彼此会妨碍对方的功能．资源竞争将会非常激烈。而从计算机程序的发展历史来看，为了协调各程序的运行，提高资源利用效率，往往会导致资源分配算法的复杂度和运算量呈指数级上升，进而存在成为系统瓶颈的可能。同时，为了实现网络的可编程性。应用程序会被赋予大量对环境的控制权，而这很容易导致系统崩溃。因此，如何在软件复杂度和运算效率之间取得平衡是SDN面临的一大挑战。

(2)控制器接口尚未标准化

目前，ONF仅定义了控制器连接交换机的南向接口，而尚未定义控制器之间的接口及控制器开放给应用程序的北向接口。原因是该组织认为现在标准化这些接口为时尚早，且可能会扼杀网络基础架构中关键组件的创新。但这也无疑增加了各厂商设备间互通的难度，一定程度上延缓了SDN的商用化进程。

(3)网络集中控制固有的安全性隐患

由于SDN采取集中控制架构，因此必然面临着集中所带来的“单点失效”等固有缺陷。

(4)从现有硬件平台向虚拟化网络的平滑演进、兼容性和长期共存的挑战

由于SDN采用OpenFlow协议相对简单，因此在与网络上长期留下来的、多厂商的、多种类的系统、设备在兼容性方面的性能还有待实践检验。

(5)市场利益复杂化成为SDN发展的一大阻力

由于SDN尚处于发展的初级阶段，且各参与方有着不同的利益和目的，对SDN的看法也存在着巨大的差异。目前．业界各家厂商发布的SDN策略、解决方案之间也是千差万别，市场上已经呈现出一些厂商各行其是的现象。业界要做到统一SDN标准。实现各厂商产品相互兼容。还需要一个长期的过程。

**4.2 SDN的发展趋势**

鉴于存在上述复杂的原因，要预测SDN的发展趋势变得十分困难。但可以从两个角度进行考察。

从SDN设备发展的角度来看，由于存在明确的标准且功能相对简单，大部分厂商都已推出了支持OpenFlow的交换机产品。可以预见下一阶段，多数厂商以及标准组织会将关注重点转移到更加复杂的控制器上，推动SDN向进一步商用化发展。

从SDN的应用领域角度来看，数据中心无疑是SDN第一阶段商用的重点。数据中心由于具有流量大、流量模型简单、与其他网络相对隔离等特点，非常适于SDN技术特点的发挥。而且目前大部分数据中心正面临“云”化变革，这为SDN推广提供了难得的机遇。因此，业界普遍将数据中心视为SDN目前最主要的应用领域。

**参考文献**

[1] P.Mell and T. Grance, “The NIST definition of cloud computing (draft),”NIST Special Publication, vol. 800-145, p. 7, 2011.

[2] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[3] J. Gantz and D. Reinsel, “Extracting value from chaos,” IDC, Framingham, MA, USA, White Paper, Jun.2011.[Online].Available:http://www.emc.com/ collateral/analystreports/idcextracting-valuefrom-chaos-ar.pdf

[4] J. Manyika et al., “Big data: The next frontier for innovation, competition, and productivity,” McKinsey Global Inst., Mumbai, India, pp. 1–137, 2011.

[5] “Software-defined networking: The new norm for networks,” Palo Alto, CA, USA, White Paper, Apr. 2012.[Online].Available:https://www.Opennetwork ing.org/images/stories/down-loads/white-papers/wp-sdnnewnorm.Pdf

[6] K. Greene, ‘‘10 Breakthrough Technologies: Software-defined Networking MIT Technol. Rev., 2009.[Online].Available:https://www.technologyreview.com/article/412194/tr10-software-defined-networking/.

[7] M. Casado et al., “Ethane: Taking control of the enterprise,” in Proc.Conf. SIGCOMM Appl., Technol., Archit., Protocols Comput. Commun.,2007, pp. 1–12

[8] MeKEOWN N，ANDERSON T，BALAKRISHNAN H，et al．OpenFlow：enabling innovation in campus networks[J]．ACM SIGCOMM Computer Communication Review，2008，88(2)：69-75

[9] “Software-defined networking: The new norm for networks,” Palo Alto,CA, USA, White Paper, Apr. 2012. [Online]. Available: https://www.opennetwork ing.org/images/stories/downloads/white-papers/wp-sdnnewnorm.pdf

[10] T. Nadeau and P. Pan, Software Driven Networks Problem Statement,Oct. 2011, Internet Draft. [Online]. Available:http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white\_paper\_c11-481360.pdf

[11] Open Networking Summit. [Online]. Available: http://www.opennetsummit.org/archives/oct11/site/tutorials.html

[12] Hot Topics in Software Defined Networking (HotSDN). [Online]. Available: http://conferences. sigcomm.org/sigcomm/2012/hotsdn.php

[13] European Workshop on Software Defined Networks. [Online]. Available:http://www.ewsdn.eu /previous/ewsdn12.html

[14] Open Networking Foundation (ONF). [Online]. Available: https://www.opennetworking.org/

[15] H. Yin *et al.*, SDNi: A Message Exchange Protocol for Software Defined Networks (SDNS) across Multiple Domains, Jun. 2012, Internet draft.[Online].Available:http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white\_paper\_c11-481360.pdf

[16] H. Xie et al. “Software-defined networking efforts debuted at IETF 84,”IETF J., Oct. 2012. [Online]. Available:http://www.internetsociety.org/fr/node/4570

[17] 袁广翔. 软件定义网络技术发展与应用研究[J].现代电信科技,2013(4):45-50.

[18] N. McKeown et al., “OpenFlow: Enabling innovation in campus networks,”SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pp. 69–74, Mar.2008

[19] OpenFlow Switch Specification, Version 1.0.0 (Wire Protocol 0x01).[Online]. Available: http://www.openflow.org/documents/openflowspec-v1.0.0.pdf

[20] OpenFlow Switch Specification, Vrsion 1.1.0 Implemented(Wire Protocol 0x02). [Online]. Available: http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf

[21] OpenFlow Switch Specification, Version 1.2 (Wire Protocol 0x03).[Online]. Available: https://www.opennetworking.org/images/stories/downloads/openflow/openflow-spec-v1.2.pdf

[22] OpenFlow Switch Specification, Version 1.3.0 (Wire Protocol 0x04).[Online]. Available: https://www.opennetworking.org/images/stories/downloads/specification/openflow-specv1.3.0.pdf

[23] OpenFlow Switch Specification, Version 1.4.0 (Wire Protocol 0x05).[Online]. Available: https://www.opennetworking.org/images/stories/downloads/specification/openflow-specv1.4.0.pdf

[24] D. Mattos et al., “OMNI: OpenFlow management infrastructure,” in Proc. Int. Conf. NOF, 2011, pp. 52–56.

[25]Trema.[Online].Available:http://trema.github.com /trema/

[26]Ryu.[Online].Available:http://osrg.github.com /ryu/

[27]Floodlight.[Online].Available:http://www.projectfloodlight.org/

[28] N. Gude et al., “NOX: Towards an operating system for networks,”SIGCOMMComput.Commun. Rev.vol. 38, no. 3, pp. 105–110, Jul. 2008.

[29]NOX.[Online].Available:http://www.noxrepo.org/

[30]OpenDaylight.[Online].Available:http://www.opendaylight.org/

[31] Y. Zhang, N. Beheshti, and M. Tatipamula, “On resilience of splitarchitecture networks,” in Proc. GLOBECOM. IEEE, 2011, pp. 1–6.

[32] Y. Hu, W. Wendong, X. Gong, X. Que, and C. Shiduan, “Reliabilityaware controller placement for software-defined networks,” in Proc. IM. IEEE, 2013, pp. 672–675.

[33] M. F. Bari, A. R. Roy, S. R. Chowdhury, Q. Zhang, M. F. Zhani, R. Ahmed, and R. Boutaba, “Dynamic controller provisioning in software defined networks,” in Proc. CNSM. IEEE, 2013, pp. 1–8.

[34]Muller L F, Oliveira R R, Luizelli M C, et al. Survivor: an enhanced controller placement strategy for improveing SDN survivability[C]//Global Communications Conference (GLOBECOM), 2014 IEEE. IEEE, 2014: 1909-1915.

[35] FOSTER N，HARRISON R，FREEDMAN M J，et a1．Frenetic：a network programming　language [J]．ACM SIGPLAN Notices，2011，46(9)：279－291．

[36]BENSON T，ＡKELLA A，SHAIKH A，et al．Cloudnaas：a cloud networking platform for enterprise applications[C]//Proc of the 2nd ACM Symposium on Cloud Computing．201l：8．

[37] FABIAS F N N，SALVATH J J，CERQUEIRA E C，et a1．A proposal management of the legacy network environment using OpenFlow control plane[C]//Proc of Network Operations and Management Symposium．2012：1143-1150．

[38] R. Sherwood et al., ‘‘Carving research slices out of your production networks with OpenFlow,’’ SIGCOMM Comput. Commun. Rev., vol. 40, no. 1, pp. 129–130, Jan. 2010.

[39] T. Koponen et al., ‘‘Network virtualization in multi-tenant datacenters,’’ in Proc. 11th USENIX Symp. Netw. Syst. Design Implement., Apr. 2014, pp. 203–216.

[40] A. Corradi, M. Fanelli, and L. Foschini, ‘‘VM consolidation: A real case based on openstack cloud,’’ Future Generat. Comput. Syst., vol. 32, pp. 118–127, 2014.