运营技术广角

面向 SDN 的动态网络策略部署与实现

董黎刚,何博翰,徐倜杰,费硕成,王伟明

(浙江工商大学信息与电子工程学院,浙江 杭州 310018)

摘　要:软件定义网络(SDN)为未来网络业务的管理要求提供了一种新的解决方案。以动态服务功能链作为服务功能编排模式、以高级网络编程语言作为服务功能部署工具,并结合虚拟网络映射设计了业务到策略、从控制器到网络节点的部署方案。基于 SDN 三层结构,提出动态策略管理系统。动态策略管理系统可以根据反馈的网络状态信息调整网络策略,并对同一节点上的网络策略进行冲突检测,根据不同的冲突类型选择组合方式,有效避免了网络策略冲突,实现网络服务的自适应部署。最后,通过实验验证了动态策略管理系统从业务到策略的完整部署过程。

关键词:SDN;高级网络编程语言;服务功能链;网络策略;业务部署

中图分类号:TP393　　文献标识码:A

doi: 10.11959/j.issn.1000-0801.2016273

Deployment and implementation of dynamic network policy in SDN

DONG Ligang, HE Bohan, XU Tijie, FEI Shuocheng, WANG Weiming

School of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou 310018, China

Abstract: Software defined networking (SDN) provides a new solution for management requirements of the future network. Based on dynamic service function chain as the creation model of service function, advanced network programming language as a service deployment tools, and combined with virtual network mapping theory, a deployment scheme which could take services to policies, and take controllers to network nodes was designed. Based on the three-layer structure of the SDN, a dynamic policy management system was proposed. It can implement self-adaption deployment of network service, which based on the feedback information of the network state. Besides, it can detect network policy conflicts on the network note and choose different ways of combination to avoid policy conflicts. Finally, the complete deployment process of the dynamic policy management system was verified through some experiments.

Key words: SDN, advanced network programming language, service function chain, network policy, service deployment

收稿日期:2016-07-30;**修回日期**:2016-09-29

基金项目:国家自然科学基金资助项目(No. 61379120, No. 61402408)

Foundation Items: The National Natural Science Foundation of China(No. 61379120, No. 61402408)

1 引言

SDN是一种新型可编程的未来网络架构[1]。通过其核心技术OpenFlow[2]将路由器、交换机等网络设备的控制平面从数据平面中分离出来,为网络管理者提供了全局的网络视图,实现了网络设备的集中化、精细化管理。网络管理者不再局限于单一网络设备的配置,可以通过业务部署的形式对整个网络进行全局的配置和管理,使大规模网络管理变得更加方便和智能。

业务部署是SDN研究的热点之一,随着当前高级网络编程语言的不断成熟和网络虚拟化[3]的不断推进,业务部署正向多租户模式[4]的方向发展。为不同的用户提供满足特定要求的服务,实现从业务需求到网络策略的部署。其中实现业务和网络的无缝对接是当前研究的难点所在。

针对上述问题提出基于服务功能的业务需求转化和部署方法,实现从业务需求到服务功能、服务功能实例、网络策略、流表的完整过程。并且在NetCore[5]、Pyretic[6]/Kinetic[7]等高级策略语言的基础上,设计出基于状态判断的动态策略管理系统,利用服务功能链,组合多个基于状态的网络服务功能来完成业务需求,与此同时对同一网络节点上的服务功能实例进行冲突检测,根据不同的冲突类型选择不同的组合方式,有效避免冲突,从而使控制器可以根据网络状态的变化实现多租户网络策略的自适应部署。

2 相关工作

服务功能(service function,SF)[8]负责的是对接收分组进行具体的处理,每一个服务功能都能实现一项独立网络功能。服务功能可以发生在一个协议栈的不同OSI层。服务功能可以理解为一个虚拟元件,可以被嵌入物理网络器件中。服务功能链(service function chain,SFC)[8]则是一个抽象服务功能的有序集合。

目前主流的服务功能编排方式有增强网关、静态服务功能链和动态服务功能链3种[9]。

增强网关[9]的设计思想是,将网络域中的数据分组汇集到网关,然后再路由到互联网上,由于数据分组都要经过网关,因此将复杂的服务功能都集中放在网关上,形成增强网关。网关上的多项服务功能之间以数据流串联的形式组合在一起,需要进行处理的数据分组将按序完成服务功能的处理。按照现在的理论体系,增强网关其实是后续的服务功能链技术在网关设备内部的一种实现方式。增强网关虽然已经考虑了网络对数据分组流的部分处理要求,但是从网络架构的角度来说,增强网关的扩展性非常局限。

为了提供一个相对开放的部署环境,2013年国际互联网工程任务组(Internet Engineering Task Force,IETF)的SFC工作组提出了静态服务功能链理论[10],即将服务功能或复杂的流处理中间件部署在网关后面的网络设备上,采用串行方式进行处理。以静态服务功能链方式部署第三方的服务功能[9],实现了一个更加开放的业务部署环境,各项业务不再限制具体厂商和功能。但静态服务功能链同样存在着不足。

- 由于服务功能严格按照串联方式进行处理,服务功能链上的功能顺序固定,难以复用,即使可以复用也需要进行复杂的配置。
- 服务功能部署在网络设备上,因此数据流接收服务顺序严重依赖于拓扑,如果一旦部署,难以添加、删除这些服务功能,也很难改变服务功能的顺序。
- 对于设定好的静态服务功能链,网络中任何经过的数据流都必须逐跳遍历服务功能链上的服务功能,即使一些数据流本不需要经过一些服务功能。这一缺陷导致了静态服务功能链的资源利用率大大降低。

为了提高网络资源的利用率和服务灵活性,IETF的SFC工作组又于2014年提出“动态服务功能”概念,希望在静态服务功能链基础上,提供更加开放的服务功能环境和可操作的流处理平台。并且在发展过程中提出多种动态服务功能部署的方式。其中一种方式是采用叠加(overlay)网络[9]的方式将服务功能从物理路由拓扑中抽离出来,使得业务层部署的服务功能不再依赖拓扑,能够自由移动、插入或删除,同时使SDN技术结合数据流控制,根据数据流的不同维度、不同细粒度需求,编排不同的服务功能链及服务功能链路径。另一种方式是网络服务报头(network service header,NSH)[11],新的数据中心网络和云架构需要更加灵活的服务功能部署模型[12],而NSH为动态创建服务功能链提出了新服务平面协议(service plane protocol)[12],能够更加简易地在一系列设备上部署服务功能。IETF的PCE(path computation element)工作组于2014年提出了内嵌PCE的动态服务功能链[13],通过使用PCE计算服务功

能路径(service function path, SFP)[14]来动态编排服务功能链。动态服务功能链使得网络管理员可以根据用户属性、服务属性、网络属性等不同层面的需求定制用户数据流经过的服务功能。

上述 3 种服务功能编排模式都能根据业务需求有效地编排服务功能,实现业务需求各具优劣点。本文利用服务功能链机制,提出基于状态判断的动态策略管理系统,组合多个基于状态的服务功能来完成业务需求。

3　动态策略管理系统

网络策略管理是指控制器根据管理员提供网络业务需求编排服务功能链,并根据编排好的服务功能链部署网络策略的过程。网络策略管理是一项复杂且容易出错的工作,由于网络状态在不停地变化,网络管理员必须经常手动配置网络策略来应对不同网络设备、资源触发的网络事件(安全警报、系统错误、网络故障),本文设计了一个基于状态判断的动态策略管理系统来实现策略管理的自动化,从而将网络管理员从繁琐的配置工作中解放出来。

3.1　基于状态判断的动态策略管理

目前 SDN 整体架构可以分为 5 层[15],如图 1 所示,自上而下分别是业务层、北向接口层、控制层、南向接口层和基础设施层。SDN 的网络结构由根控制器、本地控制器和网络节点组成。根控制器管理多个本地控制器,而本地控制器直接控制若干个网络节点,一个本地控制器可能受多个根控制器管理。本文提出的动态管理系统主要组成部分在业务层、北向接口层和控制层上。图 1 标注的部分为动态策略管理系统组成部分。

业务层由服务功能编排器和服务功能目录组成。与普通的静态网络策略不同,动态策略管理系统中的每项服务功能都能根据底层网络环境的不同,下发适应当前网络状态的网络策略来部署网络。该层还需要实现对服务功能的编排工作,根据用户需求建立完成不同的服务功能链。北向接口层可以使网络管理员灵活地实现多种网络策略应用。目前北向接口有 REST API 以及 Pyretic、Kinetic、NetCore、Frenetic 等网络编程语言。动态策略管理系统是通过 Pyretic/Kinetic 网络编程语言来描述网络策略。控制层提供类似于操作系统的基础服务,比如拓扑统计、消息通知、最短路径转发等网络控制功能。动态策略管理系统在控制层有两项功能,一是利用消息管理器来监测服务功能中敏感的网络参数,并通过事件形式将信息返回给上层服务功能模块;二是利用策略管理器来管理由根控制器设计的服务功能实例链,组合同一网络节点上的多项服务功能实例,合成统一的网络策略后再从南向接口下发给网络设备。

本文中提出的动态策略管理拥有以下几个特点。

(1)动态策略管理系统利用服务功能链,组合多个基于状态的网络服务功能来完成业务需求

一个网络要实现一项网络业务需求通常需要配置多个网络服务功能(如负载均衡、流量限制、防火墙等)来共同实现。动态策略管理系统使用服务功能链以串联、并联

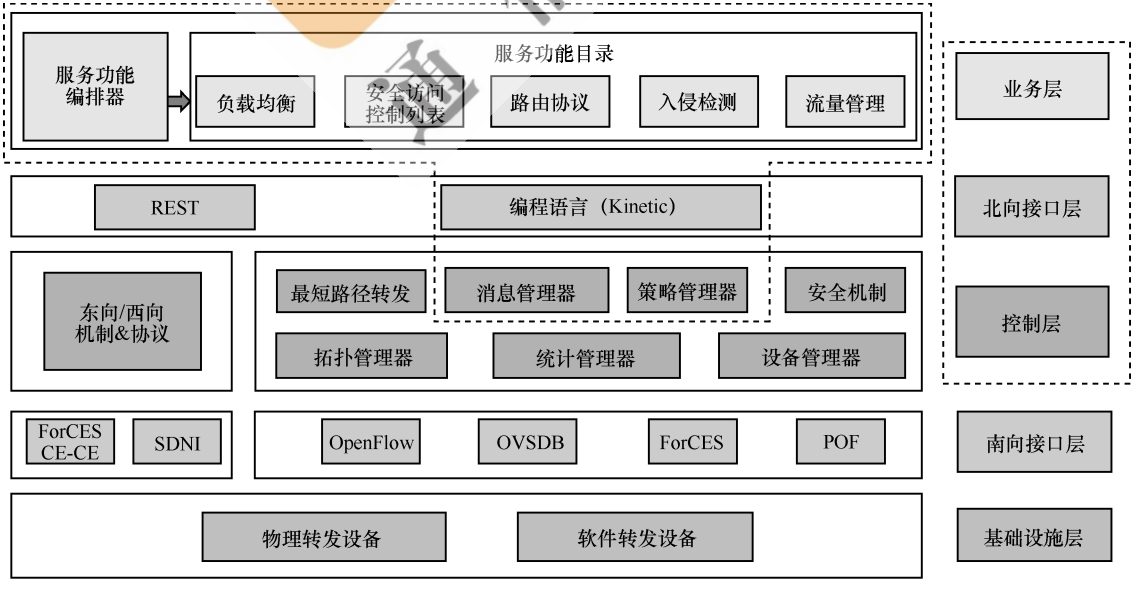


图 1　SDN 系统结构

的形式组合多个基于状态的服务功能共同实现业务需求。

(2)动态策略管理系统支持服务功能的动态部署和管理

管理系统通过网络监视区来监视当前网络状态,并通过现有的网络状态选择合适的网络策略来配置当前网络,以提高对当前网络环境的适应能力,为用户提供除连通性之外更加优质的网络服务。

(3)支持联合配置及服务功能链组合

一个网络要实现一整套网络配置,往往是由多个服务功能链共同协助完成的(例如,安全服务功能链和流量控制服务功能链分别配置策略到网络上)。此时,动态策略管理系统需要整合同一网络节点上来自不同服务功能实例链的服务功能实例,使每个服务功能实例链都可以独立表达,同时在部署策略时可以检测出策略组合时可能出现的冲突和重叠,进行处理。

3.2 动态策略管理系统结构

动态策略管理系统由服务功能区、网络监视区、策略组合区3部分组成,如图2所示。动态策略管理系统主要完成服务功能链创建、服务功能的编排、服务功能实例对应的网络策略组合以及策略转化流表的工作。

服务功能区位于SDN架构中的业务层(根控制器),整个服务功能区由两部分组成,分别是服务编排管理器和服务功能目录。服务编排管理器按照用户需求构建网络源端点和目的端点之间的服务功能链,并根据网络节点资源状况和服务功能链特征信息创建服务功能实例链。同时服务编排管理器会将服务功能链和与其关联不同分类的数据流告知本地控制器,使不同的数据流接收不同的网络服务。服务功能目录是承载网络域中可用服务功能的实例池。对于每一个服务功能链的要求,服务编排管理器会从服务功能池中提取所有需要服务功能来编排服务功能链,并在实例池中记录下此服务功能使用的相关信息[16]。有一定网络编程经验的管理员可以自己编写服务功能,加入服务功能目录中。

服务功能区有3个基本功能。一,网络管理员可以使用网络策略语言来编写服务功能来实现自己期望的网络服务,但并不负责实施。同时管理员可以根据业务需求添加多个不同的服务功能。二,服务功能区可以根据全网业务需求通过服务功能链形式将服务功能进行编排,多个服务功能协同完成业务需求。三,每个服务功能中有一套策略选择机制,每个状态对应一个不同的网络策略集合,服务编排管理器可以将网络监视区收集上报的触发事件反馈给服务功能,使策略选择机制选择合适策略下发。

网络监视区如图3所示,位于控制层的消息管理模块,监视区中有两个主要组成部分,分别是网络状态监测器(sFlow)和事件触发器(JSON)。网络状态监测器主要负责连续实时收集整个网络中的抽样数据分组,对全网每一个端口和链路上的的流量、时延等网络参数进行监控,当监控的数据到达服务功能的状态转换的阈值时,网络状态监测器会将监测的指标发给事件触发器。事件触发器[17]会发送触发事件到对应的服务功能模块,服务功能通过自带的策略选择机制选择适合当前状态下的网络策略下发。动

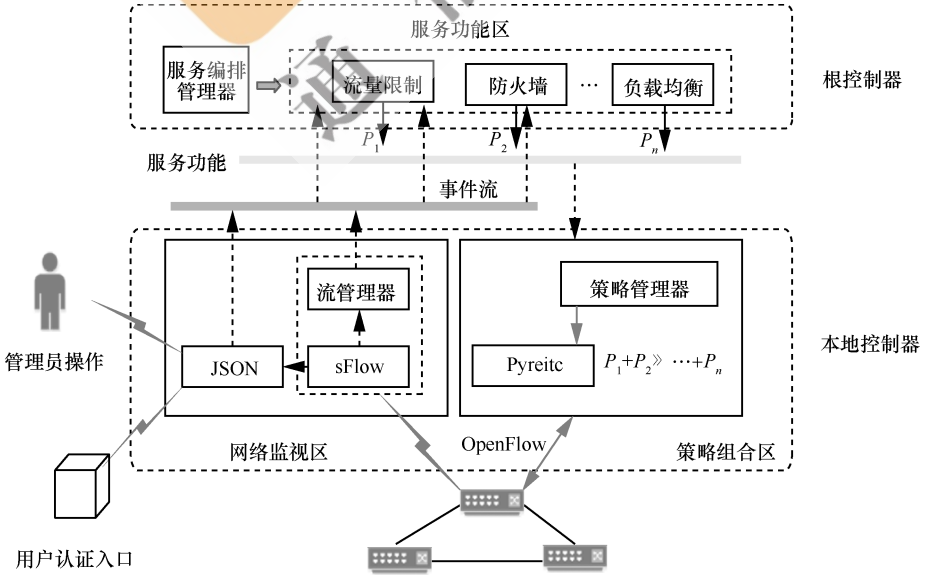


图2 动态策略管理系统结构

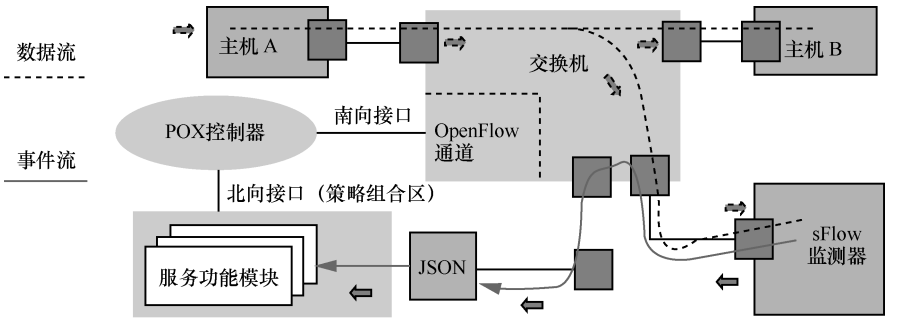


图 3　网络监视区结构

态策略管理系统中通过网络监视区提供了两种事件触发实现方式：sFlow 触发事件和人工触发事件。sFlow 事件触发器可以实现精确收集网络信息并发送触发事件的自动管理过程，反应时间比较短。通过事件触发器（JSON）进行人工触发事件，可以保证管理员手动管理触发事件的情况。网络状态监测器位于本地控制器中，它负责持续监测 CPU、带宽、链路时延、数据流速等网络状态，当发现服务功能敏感的网络状态变化超过阈值，网络状态监测器将阈值数据作为处理函数发送至事件触发器，事件触发器向服务功能发出新的事件的使其转换状态。

策略组合区由控制层的策略管理器与北向接口中的 Pyretic/Kinetic 高级编程语言组成。策略组合是服务功能部署过程中一个重要的环节，服务编排管理器通过状态选择机制确定当前状态下的策略后，就要实现策略的下发和部署，因此服务功能的部署本质上是网络策略的部署。由于整个网络有多条服务功能实例链，每个网络节点上都有多个服务功能实例对应的网络策略要部署，因此策略管理器需要负责组合同一网络节点上多个服务功能实例，读取网络节点上服务功能实例的条件和动作，根据条件范围和动作异同确定多个服务功能实例的组合类型，采用串行、分类并行、复制并行、合并或排他的组合方式，并将组合好的服务功能实例对应的网络策略编译生成流表。实现服务功能实例组合的主要目的是为了避免不同服务功能实例对应的策略之间产生冲突，因此在对策略进行组合的过程中执行冲突检测，并根据冲突类型使用不同的组合方式完成策略的组合，有效避免了策略间的冲突。最后将组合好的策略编译成 OpenFlow 流表下发到交换机。

4　动态策略部署方法

具体介绍使用动态网络策略管理系统部署网络策略的方法。

4.1　动态策略管理系统的业务需求转化和部署步骤

动态策略管理系统实现业务需求转化和部署方法，其特征在于，在 SDN 网络结构中，实现抽象的网络业务请求到具体的网络设备配置策略的转换并部署，并根据网络状态动态调整部署的网络策略方案，其方法包括以下步骤，如图 4 所示。

（1）根控制器根据用户提出的网络业务需求创建服务功能链，同一服务功能链中服务功能间的关系为：分类并行、复制并行、串行。

（2）本地控制器向根控制器上报底层网络节点的资源状况。

（3）根控制器根据底层网络节点的资源状况和服务功能链信息设计服务功能实例链。

（4）服务功能实例链中的每一个服务功能实例解析为一个针对多状态的网络策略的选择机制，每个网络策略对应一种网络状态下的一套网络配置。

（5）根控制器针对当前网络状态选择合适的网络策略下发给对应的本地控制器。

（6）本地控制器对同一网络节点上来自不同服务功能实例链的服务功能实例进行组合，组合类型有：无冲突、强冲突、弱冲突；组合方式有：串行、分类并行、复制并行、合并、排他。

（7）本地控制器对同一网络策略中的复杂条件策略转化为只含“与”和“非”操作的若干简单条件策略。

（8）本地控制器将组合后的策略编译为 OpenFlow 流表。

（9）本地控制器将流表下发至底层网络节点，部署策略，如果部署成功跳转至步骤（5），否则跳转至步骤（2）。

4.2　服务功能的组合方式

一条服务功能链由多个抽象有序的服务功能组成[17]，在服务功能链中服务功能有以下 3 种不同的组合方式。

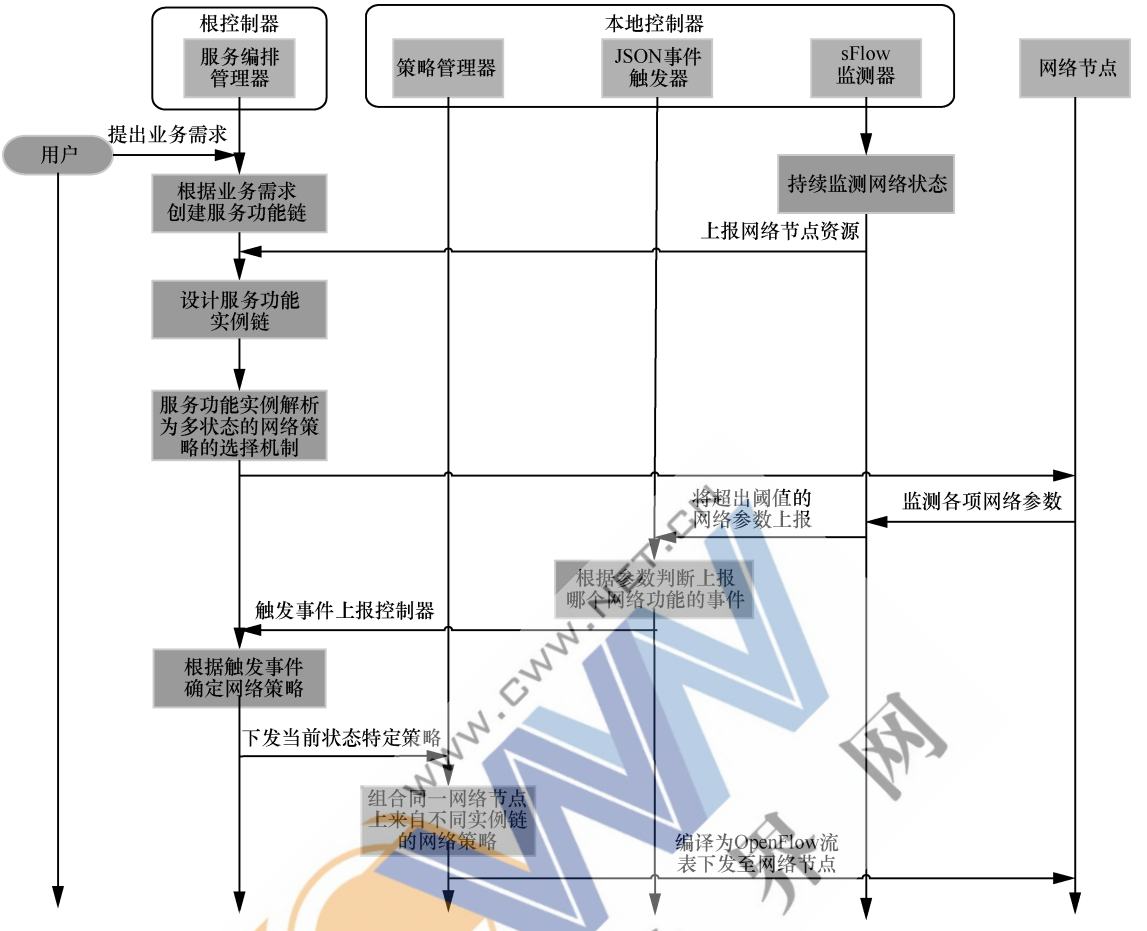


图 4　动态策略管理系统的业务需求转化和部署步骤

(1)串行组合(运算符号“>>”)

前一个的功能输出作为下一个的输入。例如:SFC=firewall (防火墙)>> Auth >>load-balance(负载均衡),首先经过的数据分组先执行防火墙操作，完成后执行用户验证,最后执行负载均衡服务。

(2)分类并行(运算符号“||”)

数据分组将根据分类要求分成若干流,分别经过并行的服务功能。例如,服务功能链,当数据流经过这条服务功能链时,根据两个服务功能的条件规定,一部分数据分组接受防火墙功能处理,另一部分数据分组接受入侵检测功能服务。

(3)复制并行(运算符号)

部分数据分组需要同时接受多个服务功能处理,这些数据分组将复制多份,分别经过每个并行的服务功能。例如,当数据流经过这条服务功能链时,交换机对数据流执行复制操作,一条数据流接受防火墙服务功能处理,另一条接受“负载均衡”服务功能处理。

4.3　服务功能实例链设计

服务功能链仅仅是逻辑层面表述抽象的业务需求,并不能实现具体服务功能到网络设备的映射,因此系统需要根据服务功能链信息以及网络节点的资源状况确定实例链编排的可行性,然后设计出服务功能实例链用于部署策略。每个网络节点上可以部署若干个服务功能实例。设计服务功能实例链的实施过程如下。

(1)向用户获取创建服务功能链的特征,其中包括服务功能的优先级、接入类型、位置、接入次数、顺序,所需CPU 和带宽、服务功能之间的关系等。

(2)根据服务功能链的特征信息,通过本地控制器获取可选网络节点的资源状况,包括节点位置、CPU、带宽、链路时延、支持的服务功能。

(3) 基于服务功能链特征信息和网络节点资源状况,优化计算出服务功能实例链的特征信息,并且完成服务功能实例到网络节点的映射,为部署服务功能实例链做好准备。

4.4 服务功能实例链的组合

动态策略管理系统通过服务功能实例中的多状态策略选择机制来实现策略状态的转换。服务功能实例链建立之后，服务编排管理器根据多状态策略选择机制中的状态参数以及状态转换阈值，为每个服务功能实例建立“状态参数—阈值”表，为每个网络节点建立“参数—服务功能实例”表、为每个服务功能建立“状态参数监视”表。系统接收触发事件中的状态参数，查询链表并实现状态的选择。根控制器通过服务功能中的状态选择机制将对应的策略下发到本地控制器。

用以下例子对上述 3 类表进行详细描述。

SF1 是“用户认证(authentication，Auth)”服务功能，用于管理和授权上网人员；SF2 是“流量管理(traffic management，TM)”服务功能。在交换机 S1 上执行服务功能实例“用户认证”“流量管理”。服务功能实例链为 SFC=Auth(S1)>>TM(S1)。根控制器为每个服务功能实例建立一个“状态参数—阈值”表，如图 5 所示。

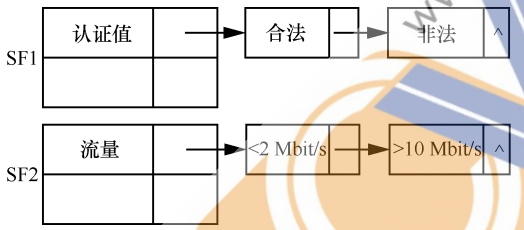


图 5　“状态参数—阈值”表

根控制器为网络节点 S1 建立“参数—服务功能实例”表，如图 6 所示。

最后，根控制器为每一条服务功能链建立一张状态参数监视表，如图 7 所示。

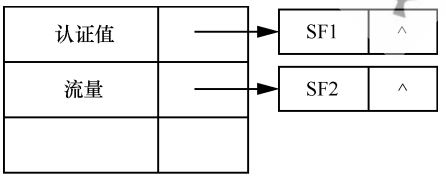


图 6　“参数—服务功能实例”表

通过查询“状态参数监视表”，根控制器通知本地控制器，将每个服务功能敏感的网络状态参数和阈值发送给与其实例相关的网络节点。比如根控制器通过查询“状态参数监视表”获取“用户认证”服务功能敏感的网络参数是“认证值”以及合法与非法两个阈值。根控制器将这些信息下发到网络节点 S1 对应的本地控制器。

如果本地控制器将每一条服务功能实例链的策略单独下发到网络节点，不同实例链间的策略将会产生策略冲突，因此本地控制器对同一网络节点上来自不同服务功能实例链的实例进行组合，组合类型分为无冲突、强冲突、弱冲突 3 种，组合方式有串行、分类并行、复制并行、合并、排他，见表 1。

(1)本地控制器根据参与组合的服务功能实例的条件和动作来判断策略组合的类型

无冲突组合类型：参与组合的服务功能实例条件无交集，或者条件有交集且动作一样。强冲突组合类型：参与组合的服务功能实例条件有交集，且至少一方的动作为分组丢失。弱冲突组合类型：参与组合的服务功能实例条件有交集，且动作均为不同目的的转发。

(2)本地控制器根据参与组合的服务功能实例的组合类型和动作来判断功能间的组合方式

当服务功能实例间的动作一致时，组合方式为：合并。当服务功能实例间组合类型为无冲突，动作却不一致时，组合方式为分类并行。当服务功能实例间的组合类型为强冲突时，组合方式可以根据用户要求选择排他(只执行其中一方的动作)或者串行。当服务功能实例间的组合类型为弱冲突时，组合方式可以根据用户要求选择排他、串行或者复制并行。

4.5 复杂条件策略的转化与编译

本地控制器判断每条策略中的条件是否为包含“或”“差”和“括号”符号的组合条件，拥有组合条件的网络策略称为复杂条件网络策略。将该网络节点上所有复杂条件网络策略提取出所有不含“或”“差”和“括号”符号的简单条

服务功能实例编号	本地控制器编号	网络节点编号	“状态参数—阈值”表地址
SF1	1	S1	
SF2	1	S1	

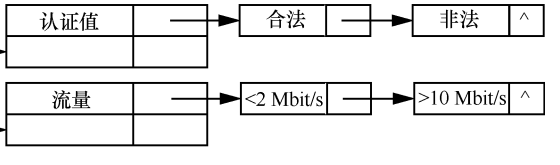
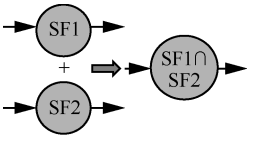
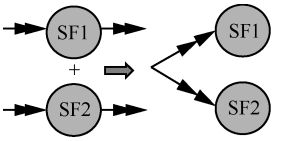
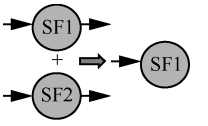
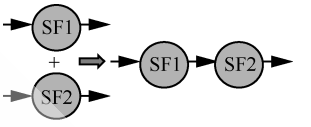
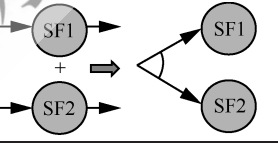


图 7　服务功能实例链 SFC 的状态参数监视表

表 1　服务功能实例组合类型及处理方式

序号	条件	动作	组合类型	处理方式	实例
1	无交集	一致	无冲突	合并	
2		不一致	无冲突	分类并行	
3	有交集	一致	无冲突	合并	同实例 1
4		不一致,至少一方的动作为分组丢失	强冲突	排他	
5				串行	
6		动作为不同目的的转发	弱冲突	排他	同实例 4
7				串行	同实例 5
8				复制并行	

件网络策略。以某个“用户认证”服务功能为例。

“用户认证”服务功能为复杂条件策略：

Auth(S1):match(SrcAddr =10.20.0.0/16\(SrcAddr:10.20.4.5))[fwd(1)]

从复杂条件网络策略可以提取出 2 个简单条件：

e1= SrcAddr =10.20.0.0/16；

e2= SrcAddr=10.20.4.5

接着，为每条简单条件策略的末尾加上缺省条件(所有简单条件策略都未覆盖的部分)，形成简单条件策略序列：

T1(s1,e1)=<SrcAddr =10.20.0.0/16:true>#“true”代表执行策略动作。

<*:false>#“*”表示缺省条件，“false”代表不执行策略动作。

T1(s1,e2)=<SrcAddr=10.20.4.5:true>

<*:false>

任选两个简单条件策略序列进行条件组合生成简单条件策略序列在执行条件组合时把“或”“差”转换成“与”“非”进行运算。“用户认证”服务功能中两个简单条件策略序列进行组合，并对结果按照条件覆盖范围大小由小到大进行排序，形成不含“或”“差”的条件策略序列。

T1(s1,e1∩¬e2)=<(Srcaddr=10.20.0.0/16∩Srcaddr=10.20.4.5):false># 同时满足两个条件的情况不执行策略动作。

<(Srcaddr=10.20.0.0/16):ture># 满足此条件下执行策略动作。

<(Srcaddr=10.20.4.5):false># 满足此条件下不执行策略动作。

<*:false># 缺省条件不执行策略动作

为简单条件策略序列添加对应的动作。Ω 代表缺省简单条件策略的动作，可以根据管理员需求定义为“分组丢失”或者“重定向”，此处定义为分组丢失。以下为“用户认证”化简后的简单条件策略序列：

P1 (s1,e1∩¬e2)=<(Srcaddr=10.20.0.0/16∩ Srcaddr=10.20.4.5):Ω>

<(Srcaddr=10.20.0.0/16):fwd(TM,TA)>

<(Srcaddr=10.20.4.5):Ω>

<*:Ω>

至此，复杂条件策略的转化和编译结束。

4.6 策略编译为流表

在复杂策略转化完成之后,本地控制器将组合后的策略编译为 OpenFlow 流表。本文使用的是 POX 控制器,能够将组合编译好的策略转化成 OpenFlow 流表,并自动下发到 OpenFlow 交换机上。其中将策略编译成流表的过程可分成两步。首先,将策略条件和动作翻译为对应流表元组项,每一个条件或动作对应一个元组项。根据表 2 中的对应关系,将服务功能实例中简单条件网络策略序列的条件和动作翻译成元组项。

表 2　策略元组对应

含义	流表项	策略条件或动作
目标地址	dl_dst	Dstaddr
源地址	dl_src	SrcAddr
目标端口	tp_dst	Dstport
源端口	tp_src	Srcport
服务类型	nw_tos	tos
数据分组类型	dl_type	ethtype
协议类型	nw_proto	protocol
所在 VLAN 编号	dl_vlan	vlan_id
VLAN 优先级	dl_vlan_pcp	vlan_pcp
转发动作	output	fwd

接着将每一条策略转化为一条流表项,并根据策略在策略序列中的先后顺序为流表项添加优先级。每一个服务功能实例对应一张流表。例如把“负载均衡”服务功能实例下的若干策略转化为流表。策略和对应的流表如图 8 所示。

策略:
P3(s1,e1)=<(Dstaddr=10.20.7.0/24):fwd(4)>
流表:
cookie=0x0,duration=59.411s,table=0,n_packets=0, n_bytes=0, idle_age=59, priority=59995,vlan_tci=0x0000 actions=drop # 缺省条件匹配的数据分组,执行分组丢失操作。
cookie=0x0,duration=59.411s,table=0,n_packets=0,n_bytes=0, idle_age =59,priority =59996,arp,vlan_tci =0x0000,arp_tpa = 10.20.7.0/24 actions=output:4 # 目标地址为 10.20.7.0/24 的 ARP # 数据分组,从端口 4 转发出去。
cookie=0x0,duration=59.411s,table=0,n_packets=0,n_bytes=0, idle_age=59, priority=0 actions=CONTROLLER:65535 # 无法匹配的转发到控制器,重定向。

图 8　策略和对应的流表

最后,本地控制器将流表下发至底层网络节点,完成策略部署。

5　动态策略部署验证实验

为了测试动态网络策略部署的效果,搭建了基于 Pyretic/Kinetic 项目和 Open vSwitch(OVS)交换机的测试平台,测试平台在 Linux 系统下运行。该测试平台需要两台 PC,一台装有 3 块网卡,作为 OVS 交换机,另一台只有一块网卡作为 SDN 控制器。另外还需要一台网络测试仪(SmartBits-600)用来产生变化数据分组,以触发设置的状态阈值,从而达到状态判断的动态网络策略部署的效果。

5.1 实验设计

本实验将完成由用户认证(Auth)及入侵检测(IDS)两个服务功能串行组合的服务功能链。

按照图 9 的拓扑将物理 PC 连接好。

在整个实验环境的搭建过程中遇到很多诸如版本兼容性及不同版本之间的性能差别问题,通过多次的尝试最终确定为:Pyretic/Kinetic、Open vSwitch 2.0.2、sFlow-RT、SmartFlow。

Pyretic/Kinetic 安装在图 9 中的物理机 PC1 上,IP 地址为 10.20.215.86. 安装方法是将官网(http://resonance.noise.gatech.edu/)提供的已集成 Kinetic/mininet/POX 的虚拟机镜像下载到本地,导入 VMware 中。

OpenvSwitch 安装在有 3 块网卡的 PC 机上,在图 9 中为 PC2,3 块网卡名称分别为:eth0、eth1、eth2。其中 eth0 用于此主机连接网络,eth1 和 eth2 作为 OVS 交换机的两个端口,PC2 的网络地址为:10.20.215.184。根据实验需求,首先要创建一个 OVS 交换机,在终端输入:# ovs-vsctl add-br br0,就可以创建一个名称为 br0 的 OVS 交换机。创建交换机之后,系统会产生一个虚拟网桥,名称就是 br0。之所以产生这个虚拟网桥,是为了实现交换机功能。然后将本地的两个物理网卡分别挂载到 br0 交换机上。接着设置 br0 交换机要连接的 SDN 控制器以及连接方式,OVS 提供了 3 种与控制器连接的方式,分别为 SSL、TCP、UNIX。在实验中,选择用 TCP 连接方式与控制器连接。最后,由于要监控交换机上每个端口的网络流量信息,所以需要在 br0 交换机上创建一个 sFlow agent。因此,紧接着要在 PC2 上安装 sFlow-RT。

最后使用一台 SmartBits-600 用来发送数据分组。对 SmartBits-600 进行以下简单的配置。

首先打开 SmartFlow 将 SmartFlow 软件与SmartBits-600 设备连接起来。进行 Cards 选项的配置,在实验中将使用

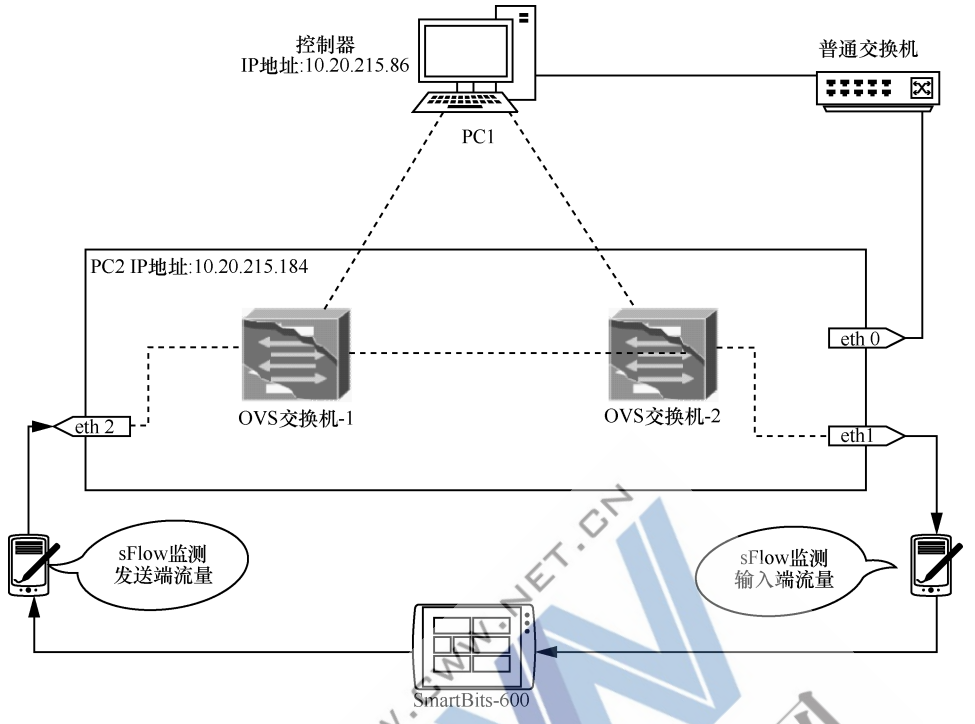


图 9　测试用例拓扑

SMB1 1-1 和 SMB 1 1-2 两个最高速度为吉比特的端口。由于 OVS 交换上挂载是两个百兆的物理网卡，综合考虑到虚拟交换机的性能及实际情况，将两个端口都设置为 10 Mbit/s。接着就是配置 Ipv4Networks 选项，设置端口的 IP 地址，产生数据分组的 IP 地址、网关地址、掩码地址等。在本次实验中，将两个端口的 IP 地址分别设为：SMB 1 1-1:10.20.215.135、SMB 1 1-2:10.20.215.136。数据分组的 IP 地址分别设为：10.20.215.111 和 10.20.215.222，所属网段 IP 地址设为：10.20.215.0，网关 IP 地址设为各自的端口 IP 地址。接着就是要按照实验的要求对数据流进行设置。然后对数据分组的发送个数的起始值、增长速率、最大值以及一次测试的完成时间进行设置，在本实验中将上述值分别设为：0.1%、0.1%、100%、30 s。

到此，本实验所有配置完成。

5.2　实验操作与结果

首先在 IP 地址为 10.20.216.86 的 PC1 上启动 Pyretic/Kinetic 提供的用户认证与入侵检测串行组合的应用，在应用启动的同时自动运行 POX 控制器：

```
$ cd /home/pyretic/pyretic/kinetic
$ pyretic.py pyretic.kinetic.examples.auth_ids
```

（1）用户认证功能测试

首先进行用户认证功能测试，由于用户认证功能属于管理员行为，因此采取人工触发事件来切换状态变化。默认策略为所有用户都未通过验证，可以看到 OVS 上的流表动作都显示分组丢失（actions=drop），如图 10 所示。

在下发策略之后 $ python json_sender.py -n auth -l True--flow="{srcip=10.20.215. 112}"-a 127.0.0.1 -p 50001，再次查看流表可以看到源地址为 10.20.215.112 的流表项动作已变为从 2 号端口转发（actions=output:2），如图 11 所示。

通过 sFlow-RT 的界面，可以更直观地看到流量变化的过程。位于图 12 的窗口采集的是发送端口的数据分组流

```
root@zero-All-Series: /home/zero
root@zero-All-Series:zero# ovs-ofctl dump-flows sflow
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=9.717s, table=0, n_packets=82, n_bytes=10168, idle_age=0,
priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=0.004s, table=0, n_packets=0, n_bytes=0, idle_age=0, prior
ity=0,ip,in_port=1,dl_src=00:00:01:00:00:01,dl_dst=00:00:02:00:00:01,nw_src=10.2
0.215.111,nw_dst=10.20.215.222,nw_proto=0,nw_tos=0 actions=drop
 cookie=0x0, duration=0.133s, table=0, n_packets=0, n_bytes=0, idle_age=0, prior
ity=0,ip,in_port=1,dl_src=00:00:01:00:00:02,dl_dst=00:00:02:00:00:02,nw_src=10.2
0.215.112,nw_dst=10.20.215.223,nw_proto=0,nw_tos=0 actions=drop
root@zero-All-Series:zero#
```

图 10　未通过认证时 OVS 默认状态的流表

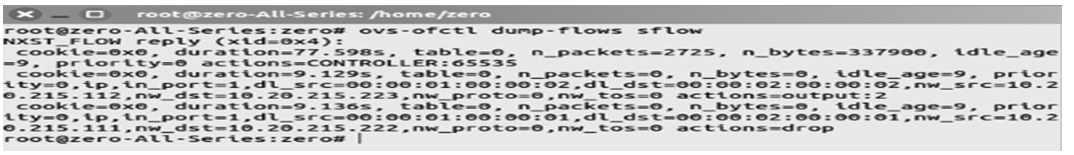


图 11　认证终端 10.20.215.112 后 OVS 的流表

图 12　用户认证中发送端数据分组流量

量，不管策略是否下发，数据分组始终按照预先设定的规则发送。位于图 13 的窗口采集的是输入端口的数据分组流量。SmartBits-600 每次发分组持续 1 min 后需要有一段准备时间为下一分钟的发分组做准备，因此流量图上每隔 1 min 会出现间隙。一开始下发两个用户（10.20.215.111、10.20.215.112）认证成功的策略，可以看到第一次数据分组顺利被接收端收到。在第二次数据分组发送时，下发两个用户认证失败的策略，接收端显示无流量接收。直到第 4 次数据分组发送，下发用户 1（10.20.215.111）认证失败的策略，用户 2（10.20.215.112）依旧认证成功，可以看到接收端的流量为发送端的一半。第 5 次数据发送时，再次下发用户 1 认证成功的策略，接收端流量恢复与发送端流量相同的状态。

（2）入侵检测功能测试

通过编写一个 ExecuteSender.java 的程序，实现获取端口流量并下发策略的功能，从而实现根控制器根据当前节点的网络状态自动部署策略给本地控制器的功能。在这个 Java 程序中，设定的流量阈值为 40，即一旦某一用户的流量超过 40 就认为有入侵，策略为丢弃所有该用户所有发送的数据分组。

图 14 中的窗口采集的是发送端口的数据分组流量，位于图 15 的窗口采集的是接收端口的数据分组流量。可以看到当发送端流量按照预先设定的规则逐渐大于阈值 40 时，策略被自动触发，接收端收到的流量立刻降为0。查看 OVS 的流表，也清楚地显示出转发动作为分组丢失（actions=drop），如图 16 所示。两个实验显示验证成功。

6　结束语

设计了一套基于状态判断的网络策略管理方法，帮助网络管理员描述和管理实时变化的网络，并借助多状态选

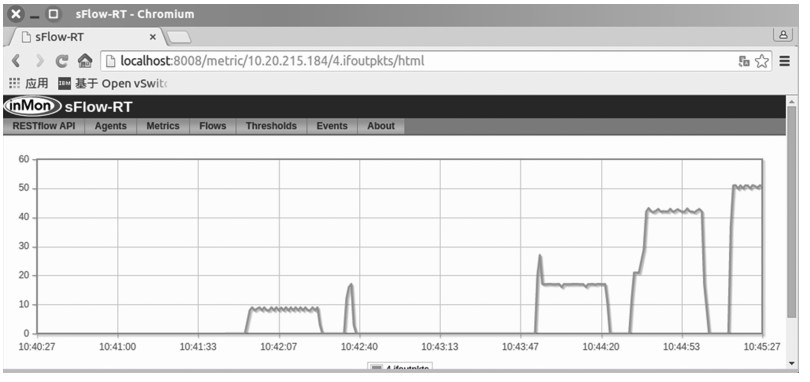


图 13　用户认证中接收端数据分组流量

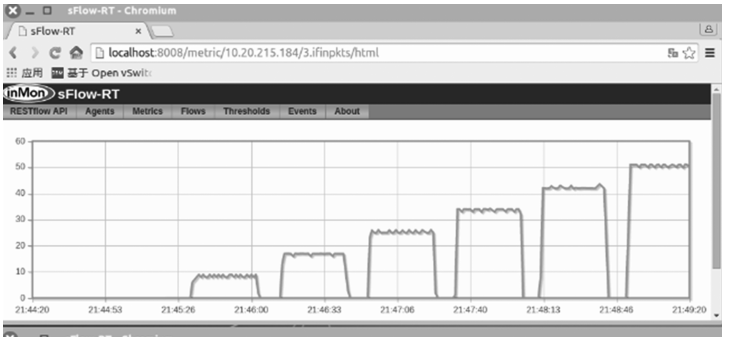


图 14　入侵检测启动后发送端流量采集

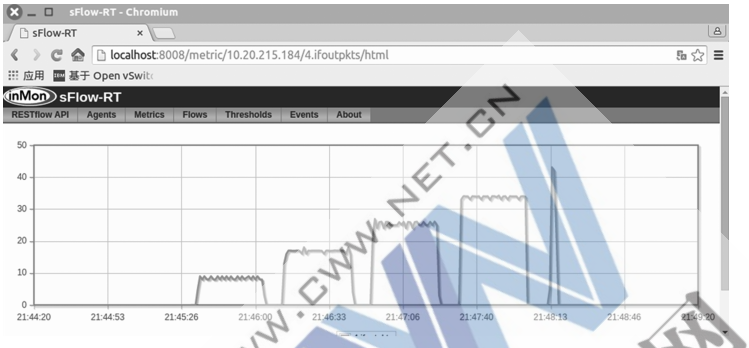


图 15　入侵检测启动后接收端流量采集

```
root@zero-All-Series: /home/zero
root@zero-All-Series:zero# ovs-ofctl dump-flows sflow
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=9.583s, table=0, n_packets=231, n_bytes=28644, idle_age=0,
 priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=0.006s, table=0, n_packets=0, n_bytes=0, idle_age=0, prior
ity=0,ip,in_port=1,dl_src=00:00:01:00:00:01,dl_dst=00:00:02:00:00:01,nw_src=10.2
0.215.111,nw_dst=10.20.215.222,nw_proto=0,nw_tos=0 actions=drop
root@zero-All-Series:zero#
```

图 16　入侵检测启动后 OVS 流表

择机制使网络策略可以自动适应当前网络环境，减少网络配置的工作量。本文从服务功能链的创建、服务功能的编排、网络状态的监测、事件状态触发、服务功能实例链的映射、服务功能实例及网络策略的组合等几方面全面阐述了业务部署的过程。

动态策略管理系统将用户的业务需求转化为服务功能链，并实现了多条服务功能链在本地控制器的组合，服务功能链上的每一个服务功能都具有状态转换的机制，可自适应地下发应对不同网络状态的配置策略。最终服务功能实例通过北向接口的 Pyretic/Kinetic 将策略转化为 OpenFlow 流表下发到各网络节点。

本文对控制器中的策略的动态管理问题进行了研究分析，但由于时间和笔者水平有限，本文对策略管理问题上还存在多个值得深入探讨的问题。

（1）目前动态策略管理系统中，从用户的业务需求转化为服务功能链的过程，必须由管理员提供服务功能链特征表来实现服务功能链的设计，是一个手动的过程。在以后的过程中将通过收集服务功能的编排规则，实现服务功能链自动化设计。

（2）目前动态策略管理系统在实现多条服务功能链组合的方式是将同一网络节点上来自不同实例链的服务功能实例进行组合，并进行冲突检测。将在以后的工作中探索服务功能实例组合进行定量评价。

参考文献：

[1] 郑毅，华一强，何晓峰. SDN 的特征、发展现状及趋势[J]. 电信科学，2013，29(9)：102-107.
ZHENG Y，HUA Y Q，HE X F. Characteristics，development and future of SDN[J]. Telecommunications Science，2013，29(9)：102-107.

[2] OpenFlow Switch Consortium. OpenFlow version 1.5.0 switch specification[S]. 2015.

[3] CHOWDHURY N M M K，BOUTABA R. A survey of network virtualization[J]. Computer Networks，2010，54(5)：862-876.

[4] GUO C J，SUN W，HUANG Y，et al. A framework for native multi-tenancy application development and management [C]// IEEE International Conference on E-Commerce Technology，and

Enterprise Computing, E-Commerce, and E-Services, IEEE Computer Society, July 23–26, 2007, Tokyo, Japan. New Jersey: IEEE Press, 2007: 551-558.

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

[6] REICH J, MONSANTO C, FOSTER N, et al. Modular SDN programming with Pyretic [J]. USENIX; login, 2013, 38 (5): 128-134.

[7] KIM H, REICH J, GUPTA A, et al. Kinetic: verifiable dynamic network control [C]//12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15), May 4–6, 2015, Berkeley, CA, USA. New Jersey: IEEE Press, 2015: 59-72.

[8] HALPERN J, PIGNATARO C. Service function chaining (SFC) architecture[EB/OL]. (2015–11–10)[2016–06–30]. https://datatracker.ietf.org/doc/rfc7665/.

[9] 薛淼, 符刚. 基于 SDN/NFV 的 Service Chaining 关键技术研究[J]. 邮电设计技术, 2015(2): 1-6.
XUE M, FU G. Research on key technologies of service chaining based on SDN/NFV [J]. Designing Techniques of Posts and Telecommunications, 2015(2): 1-6.

[10] JOHN W, PENTIKOUSIS K, AGAPIOU G, et al. Research directions in network service chaining [C]//2013 IEEE SDN for Future Networks and Services (SDN4FNS), Nov 11–13, 2013. Trento, Italy. New Jersey: IEEE Press, 2013: 1-7.

[11] QUINN P, ELZUR U. Network Service Header [EB/OL]. (2016–05–26)[2016–06–30]. https://datatracker.ietf.org/doc/draft-ietf-sfc-nsh/05/.

[12] GUICHARD J, SMITH M, KUMAR S, et al. Network service header (NSH) context header allocation (Data Center)[EB/OL]. (2016–02–16)[2016–06–30]. https://datatracker.ietf.org/doc/draft-guichard-sfc-nsh-dc-allocation/.

[13] DHODY D, BOUCADAIR M. PCEP Extensions for Service Function Chaining (SFC)[EB/OL]. (2016–03–07)[2016–06–30]. https://datatracker.ietf.org/doc/draft-wu-pce-traffic-steering-sfc/.

[14] BOUCADAIR M. Service function chaining (SFC) control plane components & requirements[EB/OL]. (2016–5–20)[2016–06–30]. https://datatracker.ietf.org/doc/draft-ietf-sfc-control-plane/.

[15] SZYRKOWIEC T, MUNOZ R, VILALTA R, et al. Demonstration of SDN based optical network virtualization and multidomain service orchestration [C]//2014 Third European Workshop on Software Defined Networks (EWSDN), Sep 1–3, 2014, Budapest, Hungary. New Jersey: IEEE Press, 2014: 137-138.

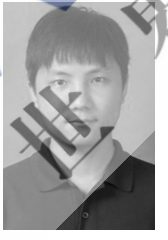
[16] CANINI M, KUZNETSOV P, LEVIN D, et al. Software transactional networking: concurrent and consistent policy composition[C]// The Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Aug 12–16, 2013, Hong Kong, China. New York: ACM Press, 2013: 1-6.

[17] QUINN P, NADEAU T. Problem statement for service function chaining[EB/OL]. (2015–04–12)[2016–06–30]. https://datatracker.ietf.org/doc/rfc7498/.

[作者简介]

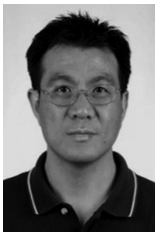
董黎刚(1973–),男,博士,浙江工商大学信息与电子工程学院院长、教授、硕士生导师,中国电子学会高级会员,浙江省计算机学会理事,浙江省高校学科带头人,主要研究方向为新一代网络和分布式系统。

何博翰(1993–),男,浙江工商大学硕士生,主要研究方向为软件定义网络和未来网络架构。



徐倜杰(1991–),男,浙江工商大学硕士生,主要研究方向为软件定义网络。

费硕成(1988–),男,浙江工商大学硕士生,主要研究方向为软件定义网络和未来网络架构。



王伟明(1964–),男,博士,浙江工商大学教授、硕士生导师,网络与通信工程研究所所长,浙江省新世纪“151”人才工程培养第一层次,主要研究方向为网络通信技术。