SDN 中一种基于多级流表的功能组合方法

段 通,兰巨龙,胡宇翔,刘释然
(国家数字交换系统工程技术研究中心,河南郑州 450002)

摘 要: 软件定义网络(Software-Defined Network,SDN)中的网络应用往往需要实现多种功能以满足上层业务需求,而如何对运行在控制器上的功能模块进行编排以完成数据包的多功能组合处理是一个仍待解决的问题.针对该问题,本文提出基于多级流表的功能并行和串行组合方案;其次,提出与任意多级流表交换机相适配的功能组合算法;最后,在 Ryu 控制器中添加功能组合模块并基于 NetFPGA-10G 节点完成了功能组合的原型实现.仿真实验与结果表明,与现有功能组合方案相比,所提功能组合方法降低了流处理时延及表项存储开销.

关键词: 软件定义网络; 多级流表; 功能组合; Ryu 控制器
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2016)11-2682-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2016.11.017

A Function Composition Method of Software-Defined Network Based on Multiple Tables

DUAN Tong, LAN Ju-long, HU Yu-xiang, LIU Shi-ran
(*National Digital Switching System Engineering & Technology Research Center, Zhengzhou, Henan* 450002, *China*)

Abstract: Network applications in Software-Defined Network (SDN) are always required to implement several functions to meet the network service demands, but how to arrange the function modules running on SDN controller to achieve multiple function packet processing is a problem to be solved. To compose the network functions, we introduced parallel and sequential composition schemes based on multiple tables; secondly, we put forward functional composition algorithms that can adapt with any multiple-table openflow switches; finally, we devised and implemented a Ryu-based control module which is used to do the function composition. Experimental results show that the method can reduce the storage cost and processing delay effectively compared with the existing schemes.

Key words: software-defined network; multiple tables; function composition; Ryu controller

1 引言

软件定义网络[1](Software-Defined Network,SDN)将控制功能转移到控制平面并为应用层提供可编程接口,从而实现了灵活的网络功能定制.现有网络业务往往需要对数据包进行多重功能处理(比如路由、监控、接入控制和负载均衡等),如果在单个网络应用内涵盖这些功能,会导致应用开发复杂度高、难以适应多种多样的业务需求[2].对此,C Monsanto 等人[2,3]提出将 SDN 应用转化为多个可独立处理数据流的功能模块的组合,该方法在降低应用开发复杂度的同时也提高了应用的灵活性.其中如何对运行在控制器之上的功能模块进行组合,以使得多个功能模块生成的表项同时对数据包产生作用,是业界研究的重点.

为实现 SDN 中的功能模块组合,文献[3]中提出并行的模块化编程语言 Frenetic,将功能模块生成的流表项进行组合以完成对数据包的多功能并行处理.作者随后又提出 Pyretic[2]编程语言,Pyretic 支持串行功能模块的组合,应用通过下发组合策略的方式将功能模块生成的表项组合起来,实现功能模块对数据流的多功能串行处理.以上研究基于 OpenFlow 单级流表[4]实现,面临组合表项数量较大、功能组合复杂度较高等问题.

收稿日期:2015-05-22;修回日期:2015-10-13;责任编辑:马兰英
基金项目:国家 973 重点基础研究发展规划计划(No. 2012CB315901, No. 2013CB329104);国家自然科学基金(No. 61372121);国家 863 高技术研究发展计划(No. 2013AA013505)

Hesham Mekky 等人[5]基于网络应用感知[6]在 SIGCOMM'14 上提出在交换机软件部分添加存放功能组合策略的应用表,通过数据流与应用表的匹配依次调用策略指示的上层应用,最终形成功能组合表项,达到功能组合的效果. 该功能组合方法不仅要修改数据平面和控制平面之间的部分交互协议,还要不停调用上层应用,复杂度较高. 此外,文献[7]将功能组合分散到多个交换机内进行,其方法系统复杂度高、且未规范使用多级流表,并不符合 SDN 控制器发展的趋势. 文献[8~10]利用 SDN 的集中式控制将数据流导向网络功能部署的节点,实现对数据流的功能组合操作,但其功能部署于中间件内,并未考虑在控制器之上的功能应用. Heng Pan 等人提出 FlowAdapter[11]的交换机实现架构,旨在利用多级流表进行规则的合并和拆分,从而将控制器下发的多级流表转化为与底层适配的多级流表. 其方法虽然不是针对功能组合,但其表项合并和拆分的方法为多个功能的表项组合提供了参考.

基于以上分析,本文提出基于 OpenFlow 多级流表[12]的 SDN 功能模块组合方法(Network Function Composition,NFC)及其具体组合算法,并且在 Ryu 控制器中添加功能组合模块实现了所提功能组合方法的原型系统. 本文主要创新点有:(1)利用多级流表进行 SDN 功能组合;(2)提出了与任意多级流表交换机适配的功能组合算法;(3)在 SDN 控制器内实现了功能组合模块.

2　概念描述

作为 SDN 最成熟的交换机-控制器交互协议,OpenFlow1.0 协议采用十二元组构成的单级流表结构,支持对数据流操作的功能定制. 单级流表的匹配字段长度长达 252 位,鉴于一般网络功能所需的匹配字段长度不足 100 位,因此其在实际部署和应用中面临的主要问题之一就是表项存储资源开销问题. 对此,标准化组织提出基于多级流表的交换机结构[12],它将流表进行特征提取,进而将匹配过程分解成多个步骤,形成流水线的处理形式,不仅减小了表项的存储开销,也增加了上层配置的灵活性. 本文采用多级流表实现 SDN 中的网络功能组合.

2.1　功能组合概念描述

首先,描述本文所使用的基本概念,作为后续功能组合方案及算法设计的基础.

定义 1　功能,指在 SDN 控制器上运行的程序模块(或应用),可通过控制器提供的北向接口向控制器下发功能表项,以完成对数据流的相应处理. 不同网络功能的包处理过程所涉及到的匹配域种类和动作类型不同,如常用安全功能的 ACL 表需要以 IPv4 协议的源/目的 IP 地址作为匹配域,动作类型为转发或丢弃;NAT 功能则以 IP 地址作为匹配域,动作类型为修改 IP 字段.

定义 2　功能并行组合,指通过将功能生成的表项进行合并,达到多个功能同时作用于数据包的效果. 例如对于功能 A 和功能 B,用 $A|B$ 表示 A 和 B 的并行组合:若数据包能够匹配功能 A 生成的表项,则执行功能 A 的动作;若能够匹配功能 B 生成的表项,则执行功能 B 的动作;若同时两者同时满足,则 A 和 B 的动作均需执行.

定义 3　功能串行组合,指将功能生成的表项进行合并,达到多个功能相继作用于数据包的效果. 例如对于功能 A 和功能 B,用 $A>>B$ 表示 A 和 B 的串行组合:若数据包能够匹配功能 A 生成的表项,则执行功能 A 的动作;经过功能 A 处理后的数据包继续匹配功能 B 的表项,若匹配成功则执行功能 B 的动作.

定义 4　组合策略,指定需要对相应数据包进行的功能组合处理. 组合策略由目标集和功能集组成,其中目标集指定所需处理的数据流,功能集指定数据流处理所需的功能及其组合方式. 例如组合策略 P = match(srcip = p)[$A|B$]表示对源 IP 为 p 的数据流执行 $A|B$ 的功能操作.

2.2　功能组合方案

多级流表以其灵活的流表跳转和连接特性为功能组合提供了更加灵活的方案,通过示例对基于多级流表的功能组合方法 NFC 进行说明. 为方便与 Frenetic、Pyretic 等经典方法进行对比,本文采用文献[2]中提供的 3 种功能的示例表项(见图 1(a)、(b)、(c)),利用多级流表分别进行并行和串行功能组合,组合结果见图 1(d)、(e).

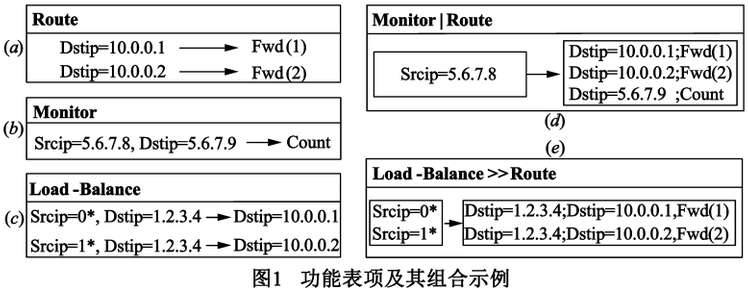


图1　功能表项及其组合示例

其中 Route 功能根据目的 IP 匹配进行端口转发,如图 1(a);Monitor 功能根据源/目的 IP 进行计数,如图 1(b);Load-Balance 功能将来自不同主机的数据流分别重新分配路由,如图 1(c). 假设交换机多级流表中源 IP 和目的 IP 的匹配域分别位于表 1 和表 2 中,则对于 Route 和 Monitor 的并行组合,首先匹配源 IP,若匹配成功则跳转至表 2;若未匹配到源 IP,则通过 Table-miss 项跳转至表 2,以实现路由的同时完成对数据流的监控,如图 1(d). 对于 Load-Balance 和 Route 的串行组合,考虑到 Load-Balance 功能会修改数据包的目的 IP,导致对 Route 的匹配产生影响,因此在表 2 的表项中将 Route 目的 IP 的匹配置换成 Load-Balance 的目的 IP 匹配,以达到负载均衡和路由的串行处理效果,如图 1(e).

3 功能组合算法

基于以上标准化概念描述,本文提出基于多级流表的 SDN 功能组合问题.

定义 5 功能组合问题. 给定交换机多级流表结构、组合策略和功能表项,在数据包满足策略目标集的前提下,根据各级流表的匹配域对功能表项进行合并,并下发到交换机中,以实现对数据包的并行或串行处理.

为形式化地描述功能组合问题,本文将使用下列符号:

表 1 主要的符号定义及其意义

符号	含义
N	交换机的流表级数
M_f	$M_f=\{m_1,m_2,\cdots,m_K\}$,匹配域集合,表示共有 K 个匹配域
T_i	第 i 级流表,$0<i\leqslant N$
C_i	T_i 包含的匹配域集合,$C_i\subseteq M_f$,$0<i\leqslant N$
A	$A=\{a_1,a_2,\cdots,a_L\}$,多级流表所支持的动作集合
$f(v)$	表示值 v 所属的匹配域,$f(v)\in M_f$
F	一条功能表项,$F=\{v_1,v_2,\cdots,v_k;a\}$,其中 $f(v_i)\in M_f, a\in A$

假设 1 各级流表内的匹配域无重叠. 多级流表将各个匹配域分散到多个流表中,从而减小了表项空间耗费. 若各级流表之间包含相同的匹配域,将会增加控制器下发表项时的难度,同时也违背了多级流表设计的初衷. 因此除非特殊厂商的要求,各级流表包含的匹配域不同.

令功能 P 的一条表项 $F_P=\{p_1,p_2,\cdots,p_{k1};a_p\}$,功能 Q 的一条表项 $F_Q=\{q_1,q_2,\cdots,q_{k2};a_q\}$,则 $P\mid Q$ 和 $P>>Q$ 的表项组合算法分别描述如下.

3.1 并行组合算法

由于功能并行组合不用考虑功能 P 处理对功能 Q 匹配产生的影响,因此并行组合的关键在于匹配字段放置和动作字段放置. 对于匹配字段的放置,需要判断匹配字段所属的流表,并使用 Goto 指令对各匹配域所在的流表进行连接;对于动作字段的放置,需要判断所属流表编号最大的匹配域,并将动作字段放置在该匹配域所属的流表内.

算法 1 并行组合算法

```
0. Function Map (F)   /* F={v1,v2,…,vk;a} */
1. for i=1,i<=N,i=i+1
2.   for j=1,j<k,j=j+1
3.     if f(vj) ∈ Ci
4.       if f(vj-1) ∉ Ci
5.         assign vj to a new entry to Ti
6.       endif
7.       if f(vj+1) ∈ Ci', (i'≠i)
8.         assign < Goto Ci' > to the action of the current entry
9.       endif
10.      assign vj to the match-field of the current entry of Ti
11.  endif
12.  end for
13.  if f(vk) ∈ Ci
14.    assign vk to the match-field of the current entry of Ti
15.    assign  <a>  to the action of the current entry
16.  endif
17. endfor
18. end Function
19. Map(FP),Map(FQ)
```

考虑功能表项最多包含 K 个匹配域,因此并行组合算法的复杂度为 $O(NK)$,也即复杂度与多级流表级数和匹配域个数成线性关系.

3.2 串行组合算法

由于功能 P 的数据包处理可能影响到功能 Q 的数据包匹配,因此需要首先判断功能 P 的动作会不会影响功能 Q 的匹配:若产生影响则先对受到影响的匹配字段进行操作,然后进行表项放置;否则对 P 和 Q 的表项依次放置,同时考虑功能 Q 的流表在前的情况.

算法 2 串行组合算法

```
0. if ap = <set vpi to vpi', > & vpi' ∈ FQ
1.   FP>>Q={(p1,p2,…,pk1,q1,q2,…,qk2)/vpi';ap,aq}
2.   Map(FP>>Q)
3. else if f(pk1) ∈ Ci & f(qk2) ∈ Ci' & i>i'
4.   FP>>Q={p1,p2,…,pk1,q1,q2,…,qk2;ap,aq}
5.   Map(FP>>Q)
6. else
7.   Map(FP),Map(FQ)
8. endif
```

对于串行组合算法,要考虑功能 P 对功能 Q 可能产生的影响,若 P 对 Q 的匹配域产生影响,则算法复杂

度为 $O(2NK)$,否则算法复杂度为 $O(NK)$. 因此串行组合算法的复杂度也为 $O(NK)$.

3.3　多个功能的组合方法

以上算法针对两个功能的组合,其方法对于三种或三种以上的功能表项组合同样适用. 如对于三个网络功能的表项 F_P、F_Q、F_R,并行组合算法在算法 1 的末行加入 $Map(F_R)$ 即可;串行组合算法则先将算法 2 中得到的 $F_{P>>Q}$ 看做一个功能的表项,再按照算法 2 进行 $F_{P>>Q}$ 与 F_R 的合并. 三个以上功能组合方法以此类推. 超过两个功能组合时,由于组合后的功能表项可能继续与其他表项再次组合,因此组合算法的复杂度最坏可达 $O(N^2)$. 考虑到 OpenFlow 多级流表级数一般在 10 级以内,因此即使 $O(N^2)$ 的复杂度也对控制器的计算能力也构不成主要影响因素.

4　基于 Ryu 控制器的原型实现

文献[5]将功能组合模块放置在交换机软件内,并修改 OpenFlow 数据流消息,从而使得交换机软件可以向控制器调用所需组合的功能. 该种在数据平面调用控制平面功能的方法的困难之处在于需要修改数据平面和控制平面的标准交互协议,导致复杂度较高且难以推广. 本文提出在控制平面增加功能组合模块,通过直接调用控制器内的功能应用实现组合.

4.1　方案设计

本文所提功能组合方法 NFC 的系统整体方案设计如图 2 所示,该架构分为数据平面和控制平面,其中数据平面由多级流表交换机组成,完成对数据流的处理;控制平面内部增加功能组合模块,根据接收到的组合策略调用上层应用,完成组合表项的下发.

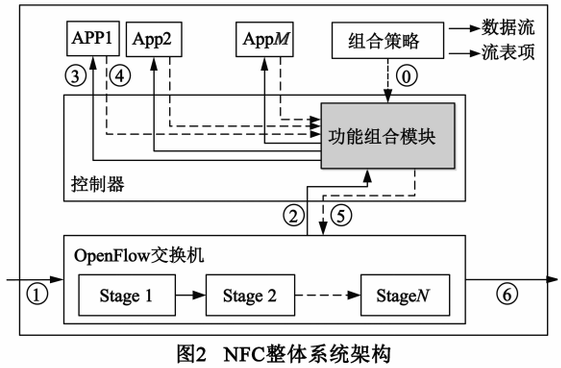


图2　NFC整体系统架构

功能组合模块的引入改变了 SDN 控制器处理数据流的方式,具体处理步骤如图 2 所示. 首先管理员或应用向功能组合模块下发组合策略,用于选定组合所需功能及组合类型,见步骤 0. 数据流到达交换机后,与交换机中流表进行匹配,若匹配成功则执行相应数据包处理;若无匹配则上传至控制器,见步骤 1. 数据包上传至控制器后交由功能组合模块处理,见步骤 2. 功能组合模块接收到数据包后根据组合策略将数据包发送至相应功能应用,见步骤 3. 若数据包不是组合策略需要处理的数据包,则将数据包交给控制器其他模块进行处理. 功能应用接收到数据包后通过功能计算得到相应表项,下发至控制器,见步骤 4. 功能组合模块接收到应用下发的功能表项计算功能组合表项并下发至交换机,见步骤 5. 数据流后续数据包直接在交换机内根据下发的流表项进行处理并转发,见步骤 6.

4.2　原型系统实现

目前,本文的多级流表节点采用项目组已有的在 NetFPGA - 10G[13] 平台上的开发成果,支持四级流表的流水线处理. 原型系统实现框架如图 3 所示,底层多级流表各匹配域分布如图 4 所示. 每级流表的动作模块可完成协议规定的动作,如转发、丢弃、修改字段、添加 VLAN 标签等,可实现如路由、转发、安全、接入控制、负载均衡、NAT 等功能. 与 NetFPGA 相连的主机作为交换节点的软件部分,负责对硬件的流表下发和与控制器的交互.

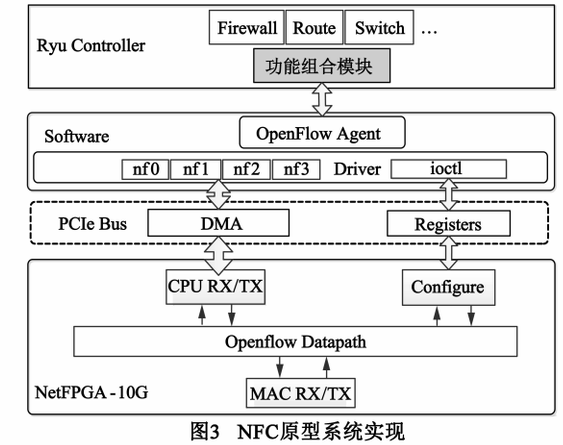


图3　NFC原型系统实现

控制平面采用支持多级流表的 Ryu[14] 控制器,该控制器自带一些编写好的应用模块,并且为应用组件开发定义了良好的 API,使得开发者可以简单地创建新的网络管理与控制应用. 本文利用 Ryu 已有的功能模块(Firewall, Router, Switching hub)以及新开发的功能组合模块完成功能组合的原型验证,见图 3.

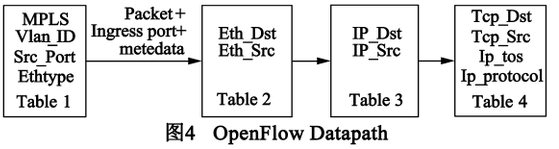


图4　OpenFlow Datapath

5　组合性能仿真与分析

本节从两方面对本文所提功能组合方法进行验

证. 首先,与文献[6]中基于数据平面的功能组合模块做性能对比测试,验证基于控制平面的功能组合的性能优势;然后,通过与其他编程语言[2,3]的功能组合机制做对比实验,验证 NFC 在资源占用和计算复杂度方面的优势.

5.1 性能分析

为验证 NFC 的性能优势,通过实验与文献[6]中基于数据平面实现的功能组合模块(Application-aware)进行对比分析. 使用网口带宽为 10Gbps 的数据包测试仪发送数据流,图 5 对比了两种方案的平均数据包处理时延,图中[1]代表 Application-aware,[2]代表 NFC.

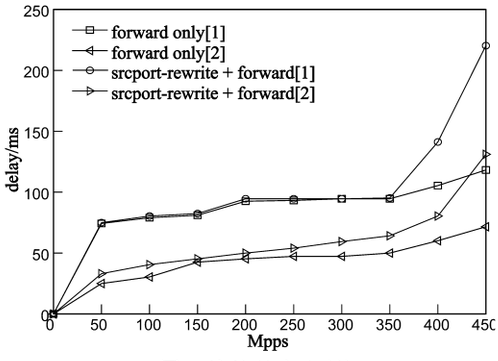


图5 流处理时延对比

实验结果表明,相比于 Application-aware 在数据平面内添加功能组合模块,NFC 在处理数据流时的时延明显较小,这是由于在控制器内调用功能模块比在数据平面内调用控制器内的功能模块所需的时间少. 此外,从实验结果可知,NFC 的源地址改写所带来的额外时延要比 Application-aware 大,这是由于对于单纯的转发功能,NFC 依然要经过控制器内的功能组合模块进行判断;而在无功能组合时,Application-aware 不需调用应用表,也不需控制器的参与,因此在低流速率情况下,Application-aware 有无功能组合的处理时延相差较小.

5.2 组合效率分析

多级流表相比单级流表的处理流程更为复杂,因此其组合表项的计算复杂度有所提高;但同时多级流表分散了各个匹配域,使得表项存储开销有大幅下降. 实验选取文献[2]中提供的三种功能实例表项(Monitor、Route、Load Balancer),分别随机生成 1000 ~ 7000 次组合策略,仿真平台同 5.1 节. 图 6 对比了 NFC 与 Frenetic、Pyretic 组合表项生成时间.

仿真结果表明,相比于 Frenetic 和 Pyretic,NFC 生成组合表项的时间复杂度较高. 这是由于相比于单级流表的表项组合,多级流表的表项组合需要针对各级流表的匹配域进行判断及表项拆分,这增加了额外的计算复杂度.

图 7 对比了针对上述三种功能表项,NFC 与 Frenetic、Pyretic 方法生成组合表项的流表存储空间占用(按照一般的交换机设计,流表项中的匹配字段放置在 TCAM 中,动作字段放置在 SRAM 中). 在给定的三种功能的示例表项情况下,串行功能组合和并行功能组合所占用存储空间相同,因此仅列出一项. 实验结果表明,由于多级流表对单级流表的各个匹配域进行了拆分,因此其匹配域所占用的存储空间明显比单级流表表项小;但由于多级流表级数增多,因此动作字段所占存储空间相对较大. NFC 在 SRAM 资源使用与 Frenetic 相当的情况下 TCAM 资源节省了 90%,在 SRAM 资源使用比 Pyretic 高的情况下 TCAM 资源节省了 75%. 而由于 TCAM 的存储成本要比 SRAM 高出很多,因此 NFC 有效节省了交换机的存储资源开销.

考虑到相比于计算复杂度方面高出的 20% 的开销,NFC 在交换机流表存储资源方面节省的 75% 的开销更为宝贵[15],因此 NFC 在实用性上优于现有的基于单级流表的功能组合方法.

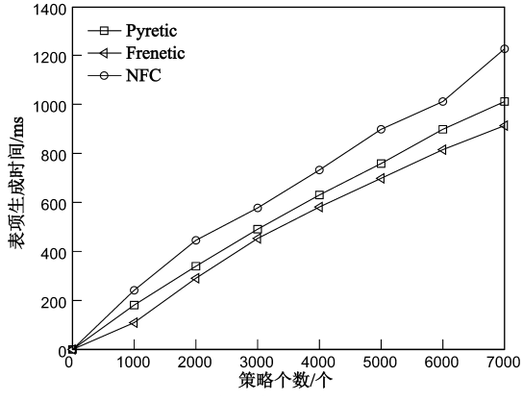


图6 表项生成时间对比

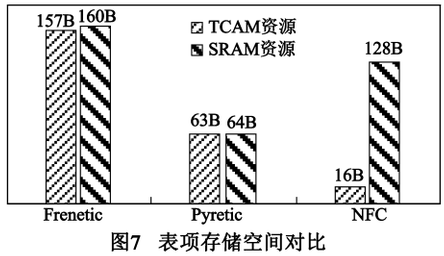


图7 表项存储空间对比

6 结论

本文提出基于多级流表的 SDN 功能组合方案并设计了并行和串行组合算法,且在 Ryu 控制器中添加功能组合模块并基于 NetFPGA-10G 节点实现了功能组合的原型系统. 实验结果表明,所提功能组合方法有效降低了流处理时延及流表存储空间耗费. 本文所提功能组合方

法对 SDN 中的功能模块组合具有一定实用价值.

参考文献

[1] McKeown N. Keynote talk: software-defined networking [A]. Proceedings of the IEEE INFOCOM[C]. Rio de Janeiro, Brazil: IEEE, 2009. 1 – 11.

[2] Monsanto C, Reich J, Foster N, et al. Composing software-defined networks [A]. Proceedings of NSDI' 13 [C]. LOMBARD, IL: USENIX, 2013. 1 – 14.

[3] Foster N, Harrison R, Freedman M J, et al. Frenetic: a network programming language [A]. Proceedings of ICFP [C]. Tokyo, Japan: ACM, 2011. 279 – 291.

[4] Mckeown N, Anderson T, Balakrishnan H N, et al. Openflow: enabling innovation in campus networks [J]. ACM SIGCOMM Computer Commnication Review, 2008, 38 (2): 69 – 74.

[5] Hesham M Fang H, Sarit M, et al. Application-aware data plane processing in SDN[A]. Proceedings of SIGCOMM' 14[C]. Chicago, USA: ACM, 2014. 13 – 18.

[6] Koponen T, Amidon K, Balland P, et al. Network virtualization in multi-tenant datacenters[A]. Proceedings of NSDI' 14[C]. Seattle USA: USENIX, 2014. 203 – 216.

[7] 段通,兰巨龙,程国振. 基于元能力的 SDN 功能组合机制[J]. 通信学报,2015,36(5):2015178-1-2015178-11.
Duan Tong, Lan Ju-long, Cheng Guo-zhen. Functional composition in software-defined network based on atomic capacity [J]. Journal on Communications. 2015, 36 (5): 2015178-1-2015178-11. (in Chinese)

[8] Fayazbakhsh S K, Chiang L, Sekar V, et al. Enforcing network-wide policies in the presence of dynamic middlebox actions using flowtags[A]. Proceedings of NSDI' 14[C]. Seattle USA: USENIX, 2014. 533 – 546.

[9] Qazi Z A, Tu C C, Chiang L, et al. SIMPLE-fying middlebox policy enforcement using SDN [A]. Proceedings of SIGCOMM' 13 [C]. HongKong, China: ACM, 2013. 27 – 38.

[10] Zhang Y, Beheshti N, Beliveau L, et al. StEERING: a software-defined networking for inline service chaining[A]. Proceedings of the 21st IEEE International Conference on Network Protocols[C]. Gottingen, Germany: IEEE, 2013. 1 – 10.

[11] Heng Pan, Hongtao Guan, Junjie Liu, et al. The flowadapter: enable flexible multi-table processing on legacy hardware [A]. Proceedings of the 2nd ACM SIGCOMM workshop on Hot topics in software defined networking [C]. NewYork: ACM, 2013. 85 – 90.

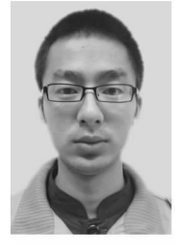
[12] OpenFlow v1. 1 project[EB/OL]. http://archive. openflow. org/wk/index. php/OpenFlow_v1. 1, 2012.

[13] NetFPGA-10G project [EB/OL]. https://github. com/NetFPGA/NetFPGA-public/wiki, 2013.

[14] Ryu controller project[EB/OL]. https://osrg. github. io/ryu, 2014.

[15] Curtis A R, Mogul J C, Tourrilhes J, et al. DevoFlow: scaling flow management for high-performance networks [A]. Proceedings of the ACM SIGCOMM 2011 conference[C]. New York, USA: ACM, 2011. 254 – 265.

作者简介



段　通　男,1992 年生于河南驻马店. 现为国家数字交换系统工程技术研究中心硕士研究生. 主要研究方向为可编程网络数据平面.
E-mail: duantong21@126. com

兰巨龙　男,1962 年生于河北张北. 现为国家数字交换系统工程技术研究中心总工程师、教授、博士生导师. 主要研究方向为新一代信息网络关键理论与技术.
E-mail: ndscljl@163. com