MicroNF：基于微服务的异构网络功能虚拟化框架

孙晨[1,2,3]，毕军[1,2,3]，郑智隆[1,2,3]，王舒鹤[1,2,3]，胡宏新[4]

（1. 清华大学网络科学与网络空间研究院，北京 100084；2. 清华大学信息科学技术学院，北京 100084；
3. 北京信信息科学与技术国家研究中心, 北京 100084；4. 克莱姆森大学，南卡罗来纳州 29634）

摘 要：网络功能虚拟化（NFV）带来灵活性的同时也面临极大的问题，因为承载网络功能的虚拟机可能会引入显著的性能开销。针对此问题，提出了一种名为 MicroNF 的新型高性能可编程框架，将可编程硬件基础设施与 NFV 中的传统软件基础设施相结合，以实现高性能和灵活性。MicroNF 利用微服务—— 一种软件架构中的新设计方法，重新构建 NFV，以实现服务之间的功能可重用性并提高性能。基于 OpenStack 和 ONetCard 的实验平台实现了 MicroNF。实验结果表明，与基于 DPDK 的软件实现相比，MicroNF 将服务链的平均转发时延降低了 70%。

关键词：网络功能虚拟化；异构基础设施；微服务；服务链；模块化

中图分类号：TP302

文献标识码：A

doi: 10.11959/j.issn.1000−436x.2019131

MicroNF: a microservice-based hybrid framework for NFV

SUN Chen[1,2,3], BI Jun[1,2,3], ZHENG Zhilong [1,2,3], WANG Shuhe[1,2,3], HU Hongxin[4]

1.Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China
2. School of Information and Technology, Tsinghua University, Beijing 100084, China
3. Beijing National Research Center for Information Science and Technology, Beijing 100084, China
4. Clemson University, South Carolina 29634, USA

Abstract: Network function virtualization (NFV) brought significant flexibility. However, such flexibility came with considerable compromises, since virtual machine carried monolithic functions could introduce significant performance overhead. A novel high-performance and programmable framework called MicroNF was proposed, which combines programmable hardware infrastructure and traditional software infrastructure in NFV to achieve both high performance and flexibility. In particular, microservice, a new design approach in software architecture, was leveraged by MicroNF to re-architect NFV to enable functional reusability among services and improve performance. MicroNF was implemented in a test bed based on OpenStack and ONetCard. Experimental results show that MicroNF reduces the forwarding latency of a service chain by an average of 70% compared with DPDK-based software implementation.

Key words: network function virtualization, hybrid infrastructure, microservice, service chain, modularization

1 引言

计算机网络中有着种类繁多的网络功能，用于实现网络安全、网络监控、流量调度等。在传统网络中，网络功能通过专有硬件来实现，基于专有硬件的网络功能通常被称为中间件，添加新的网络功能通常需要添加新的中间件。然而，中间件没有通用的硬件框架，并且在高峰负载时难以扩展，因此，网络功能虚拟化（NFV, network function virtualization）的概念被提出，以解决中

收稿日期：2019−03−07；**修回日期**：2019−05−09

基金项目：国家重点研发计划基金资助项目(No.2017YFB0801701）；国家自然科学基金资助项目（No.61872426）

Foundation Items: The National Key Research and Development Program of China (No.2017YFB0801701), The National Natural Science Foundation of China (No.61872426)

间件的局限性。NFV 利用虚拟化技术，通过软件的形式实现网络功能，从而增强服务交付的灵活性，降低总体成本[1]。NFV 中的网络功能称为虚拟化网络功能（VNF, virtualized network function），常见的 VNF 包括网络地址转换（NAT, network address translation）、负载均衡、防火墙等。然而，基于软件实现的 VNF 性能较低，主要有以下 2 个原因。

1) 基于软件的 VNF 存在 200 μs~1 ms 的转发时延，这对于一些网络应用而言是不可接受的，例如，基于算法的股票交易和要求超低时延（几微秒）的高性能分布式存储器缓存[2]。

2) 根据本文对一家主流网络运营商的调研，目前该运营商部署的 VNF 是由几乎没有相关性的独立团队构建的单一、重型的实例。然而，多个 VNF 通常可以部分地共享相似的处理逻辑[4]，例如，负载平衡和访问控制都需要解析数据分组的分组头来匹配规则。入侵检测系统（IDS, intrusion detection system）和 7 层防火墙都可以对流或数据分组执行深度分组检测（DPI, deep packet inspection）。如果没有模块重用，一个数据分组可能会在整个服务链中多次执行同一个操作，这种动作的重复会进一步损害 NFV 的性能。

针对问题 1)，最近的一些工作提出使用可编程硬件加速器来支持 NFV。部分工作[2~5]揭示了基于软件的中间件与商用硬件之间的差距，并提出将弹性的现场可编程门阵列（FPGA, field-programmable gate array）引入基于 OpenStack 的 NFV 以提高性能。但是，现有解决方案仅利用了高性能硬件来支持 NFV，并未发挥可提供丰富资源和高灵活性的商用服务器的优点。实际上，硬件设备上的 TCAM（ternary content addressable memory）资源相当有限且昂贵，难以支持复杂且资源密集型的 VNF，例如 NAT。

针对问题 2)，本文注意到，微服务的概念在软件工程领域越来越受关注[6]。微服务被定义为通过消息交互的最小独立过程。复杂的 VNF 可以分解为更简单的微服务，而一个微服务可以被不同的 VNF 重用。在 NFV 环境中，微服务的模块化和可重用性可以消除重复的处理阶段并缩短时延。

因此，本文提出一种高性能和可编程的架构——MicroNF，利用可编程硬件和传统软件基础架构来支持 VNF 的高性能和灵活性。MicroNF 可以容纳最先进的硬件平台，如 FPGA、P4 和 RMT、ClickOS 等软件平台[7]，只要它们向外提供统一的控制接口即可。此外，MicroNF 通过组合和重用多个硬件或软件的微服务来构建所需的服务链，以消除重复并提高性能，而非构建功能重叠的单片 VNF。本文在 MicroNF 中设计了一个策略解析器层，用于硬件和软件的微服务管理，以隐藏 NFV 协调器底层基础架构的异构性。

本文的主要贡献如下。

1) 提出了一种高性能 NFV 框架——MicroNF，该框架利用异构基础设施，结合了硬件和软件功能，同时支持 VNF 的高性能和灵活性。此外，MicroNF 将单片 VNF 分解为硬件或软件微服务，并通过在 VNF 之间重用微服务来提高性能。

2) 在 MicroNF 中设计了一个策略解析器，以隐藏 NFV 协调器的基础设施异构性。策略解析器还负责 VNF 分解和微服务链接以形成服务链。

3) 实现了基于 OpenStack 和 ONetCard 的 MicroNF 原型。实验结果表明，与纯单片软件 VNF 链相比，MicroNF 中具有微服务增强功能的异构服务链可以实现时延平均降低 70%。

2　MicroNF 架构

基于 NFV 网络的 MicroNF 架构如图 1 所示。其中，编排器做出关于服务链的决定；策略解析器根据链接要求执行 VNF 分解和微服务链接，并将策略提供给相应的模块，包括控制硬件（H-MS, hardware function management service）与软件（S-MS, software function management service）微服务生命周期的微服务管理器，以及可以基于 SDN 控制器实现的转发策略实施器。策略实施器可以在微服务实例之间引导流量。可编程硬件基础架构与软件基础架构结合构建异构基础架构，支持异构服务链。

2.1　策略解析器

传统网络中，编排器负责编排服务并直接通知相关 VNF 管理器，部署 VNF 实例并将其链接在一起形成服务链。然而，在 MicroNF 中需要组装并重用硬件或软件微服务来构建 VNF 并形成服务链，这为服务链的生成提出了 2 个问题：如何选择软硬件实例；如何将软硬件实例混合编排。

为了解决上述问题，本文在 MicroNF 框架中设计了一个策略解析器。策略解析器接收来自编排器的服务链决策，并将策略传递给最合适的微服务管

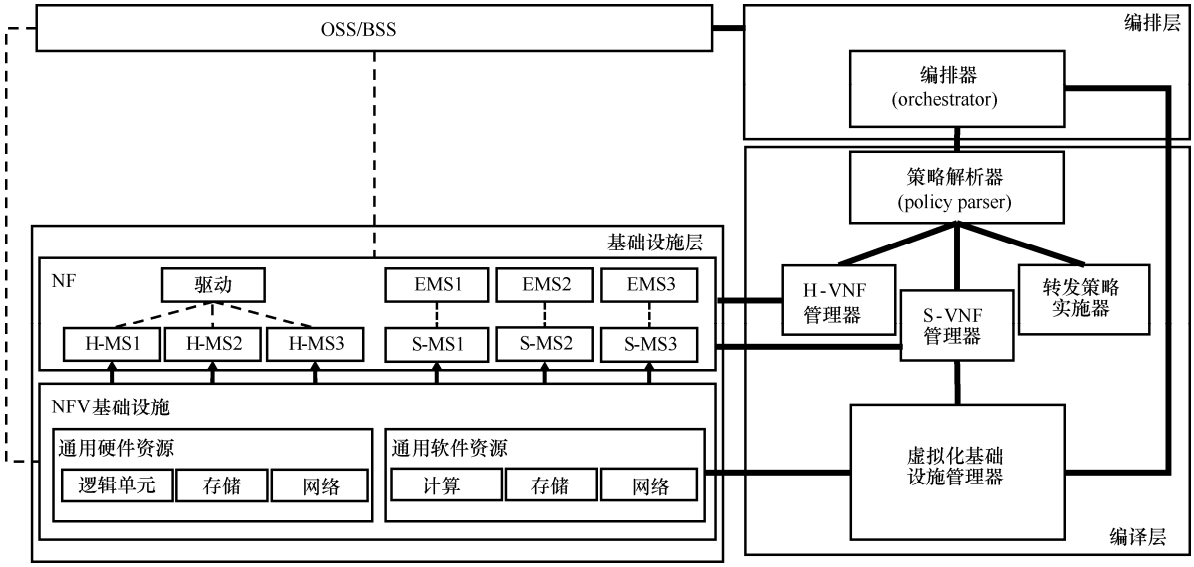


图 1 MicroNF 架构

理器（MSM, micro service management module）或转发策略实施器。为了形成服务链，编排器只需简单描述其对 VNF 的要求，策略解析器就可以自动将 VNF 分解为微服务，并通知相关的 MSM 和转发策略实施器部署服务链。

2.2 VNF 分解

MicroNF 中的策略解析器负责将 VNF 分解为微服务，并利用微服务的可重用性来消除重复并缩短服务链时延。VNF 的分解示例如表 1 所示。

表 1 VNF 分解示例

VNF	微服务
防火墙	分组头解析器+规则匹配器+动作执行器
IDS/IPS	分组头解析器+正则规则匹配器+动作执行器
长流检测器	分组头解析器+计数器+规则匹配器
负载均衡	分组头解析器+散列函数+动作执行器
NAT	分组头解析器 +规则匹配器+分组头修改器
代理	分组头解析器+分组头修改器
VPN	分组头解析器+规则匹配器+分组头修改器
DPI	分组头解析器+负载检查器+动作执行器

2.3 微服务组装与链接

VNF 分解过程将 VNF 分成若干个微服务，但是服务链中的多个 VNF 可能会使用同一个微服务。为了提高服务链性能，本文考虑利用微服务的可重用性来删除重复的微服务，并用所有剩余的非冗余微服务来构建微服务链。最后，策略解析器将服务链转换为逐跳转发策略，并通知转发策略实施器。

为了充分利用微服务的可重用性，微服务的处理结果可以被传递至该服务器链中的后续微服务，但是目前在 NFV 中的 VNF 之间没有上下文传递机制。网络服务分组头（NSH, network service header）[8]是一种被广泛接受的 NFV 服务交付和链接机制，但其中没有足够的空间用于传递大量处理结果。接下来，本文将讨论基于 NSH 的设计方案。

若某一微服务需要将其处理结果传递给服务链中的 2 个微服务，则它可以将处理结果附加到目标微服务的现有 NSH 中，如图 2 所示。具体而言，微服务应首先解析数据分组携带的所有 NSH 并识别其目标微服务。在处理分组之后，微服务将处理结果注入目标微服务 NSH 的“上下文字段”中。随后的目标微服务将从 NSH 导出上下文，并利用上下文信息继续完成自己的逻辑任务。

2.4 MSM 的统一控制接口

为了适应 MicroNF 中的异构基础设施，本文在 MicroNF 中为所有平台的 MSM 设计了一个统一的

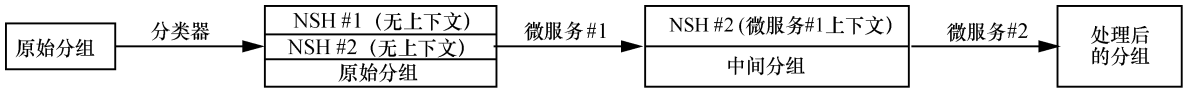


图 2 使用 NSH 上下文头的微服务上下文传递

控制接口集，以避免策略解析器承担管理具有不同接口和内部逻辑的异构 MSM 的负担。控制接口集具体如下。

• deploy_instance (instance)：用于部署实例 instance。

• destruct_instance (instance_ID)：用于销毁编号为 instance_ID 的实例。

• configure_instance (instance_ID, configuration)：用于配置编号为 instance_ID 的实例。

• migrate_instance (old_ID, new_ID, flows)：将编号为 old_ID 的实例上的部分流迁移至编号为 new_ID 的实例。

• get_instance_state (instance_ID)：用于获取编号为 instance_ID 的实例的状态信息。

通过 deploy_instance 和 destruct_instance 接口可部署和销毁微服务实例。策略解析器可以通过 configure_instance 接口配置所有实例，还能够指示特定 MSM 通过 migrate_instance 接口将一些数据流从旧实例迁移到新实例。运行时，协调器需要通过 get_instance_state 接口查询实例状态。

图 3 为 DPI 微服务的生命周期。策略解析器调用 deploy_instance 接口部署 DPI；然后通过 configure_instance 接口发出 2 个规则，如实例过载，使用 deploy_instance 启动 2 号实例并使用 migrate_instance 从 1 号实例迁移到 2 号实例；最后，策略解析器将使用 destruct_instance 对 1 号实例进行清理。

3　系统实现

本文构建了基于 ONetCard 的 MicroNF 基础设施，在硬件资源 ONetCard 上实现了带状态数据平面的抽象（SDPA, stateful data plane abstraction）[9]，用于支持各种硬件网络功能或微服务。本文实现了几个微服务的硬件和软件版本，包括分组头解析器、规则匹配器、动作执行器、分组头修改器和计数器，并用构建了 3 个 VNF，包括硬件和软件版本的 NAT、防火墙和长流检测。本文基于 OpenStack 和 OpenDaylight 实现了 MSM、MicroNF 策略解析器和协调器。OpenStack 用于实现 S-MSM（software MSM），使用 H-MSM（hardware MEM）模块扩展了 OpenDaylight，来管理硬件微服务实例。MicroNF 策略解析器在 OpenDaylight 中被编写为模块，执行 NSH 服务链。具体如图 3 所示。

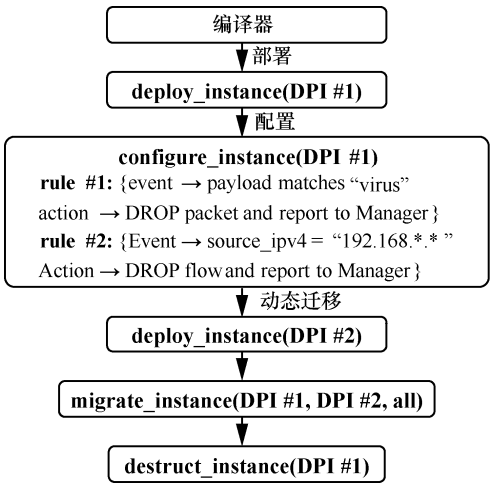


图 3　由控制接口支持的 DPI 生命周期

4　系统性能评估

4.1　异构 VNF 链与软件 VNF 链的对比

本文通过生成由硬件和软件 VNF 组成的异构链来评估 MicroNF 的性能。在硬件上实现了状态防火墙和 NAT，分别在有数据平面开发套件（DPDK , data plane development kit）增强软件和没有 DPDK 增强软件的情况下完成了长流检测的软件实现，并将它们的性能与纯 DPDK VNF 链进行了比较。使用 Ixia 测试器生成固定大小的数据报文，并调整数据流量发送速率为 1～10 Gbit/s，并将相同的流量发送到这 2 个 VNF 实例，测量处理延时和吞吐量。实验结果如图 4 所示。根据实验数据，未优化软件的处理时延可能比硬件高 119～154 倍，而 DPDK 增强软件的平均时延比硬件的平均时延高 5.4 倍。未优化软件的吞吐量比硬件低 91%，DPDK 增强软件可以实现与硬件一样高的吞吐量。由硬件和 DPDK 增强软件组成的异构服务链在处理时延方面优于纯 DPDK 解决方案（降低 40%～67%）。

4.2　软件微服务链与软件 VNF 链的对比

为了评估微服务引入 NFV 带来的性能提升，本文生成了一个具有 DPDK 增强软件的微服务软件链，并将其性能与单片的服务链进行了比较。2 个链都包括相同的软件 VNF 集合，包括 NAT、状态防火墙和长流检测。从图 5 可以看出，基于微服务的服务链的转发时延比基于单片 VNF 的服务链低约 40%。这表明将 VNF 分解为微服务并删除重复的微服务可以在很大程度上缩短转发时延。

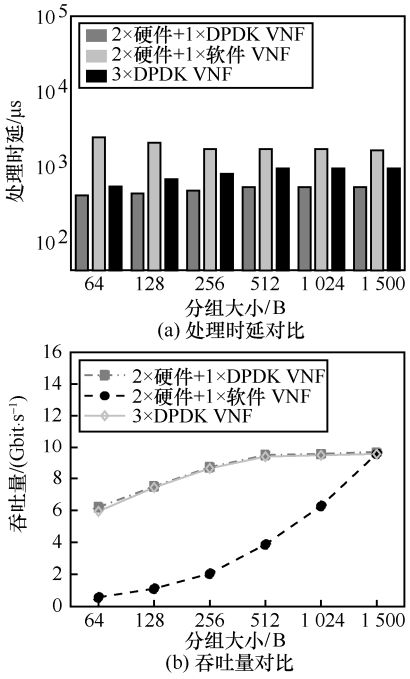


图 4　异构 VNF 链与软件 VNF 链的性能对比

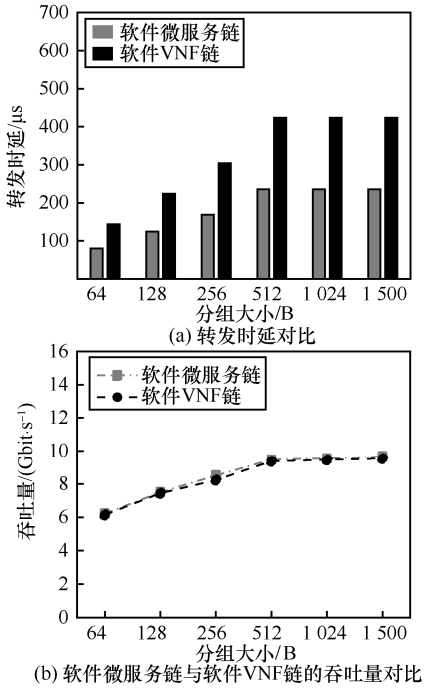


图 5　软件微服务链与软件 VNF 链的性能对比

4.3　MicroNF 链与软件 VNF 链的对比

为了全面评估 MicroNF 的性能增强程度，本文生成了一个异构微服务链，并将其性能与单片软件 VNF 链进行比较。对于异构微服务链，本文在硬件上实现了分组头解析器、规则匹配器和动作执行器，在软件上实现了计数器和分组头修改器。对于单片软件 VNF 链，本文在硬件上实现了带状态防火墙和 NAT，并且在具有 DPDK 增强功能的情况下完成了长流检测的软件实现。实验结果如图 6 所示。MicroNF 使服务链平均转发时延降低 70%。

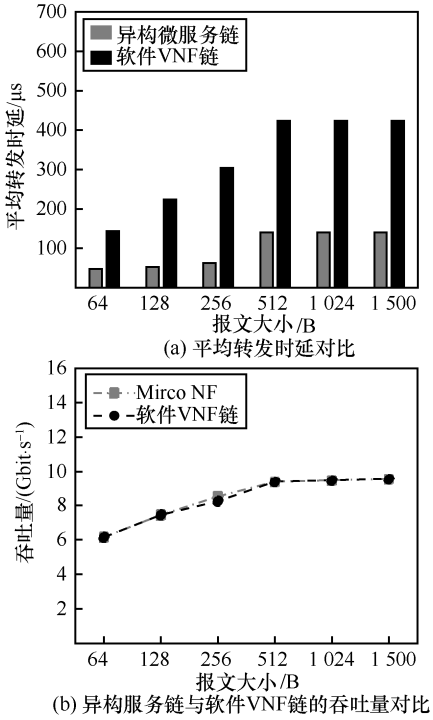


图 6　异构微服务链与软件 VNF 链的对比

5　相关工作

最近一些相关工作已经认识到将硬件和软件相结合来实现高性能和灵活性的优势[10-12]。Rohan 等[10]提出了 Duet，它使用现有的交换机硬件和一小部分软件来构建高性能的负载均衡器。然而，Duet 仅限于实现负载均衡 VNF。Hendrik 等[11]提出了 VNF-P，可以在专用硬件和通用软件上支持 VNF。

但是，专用硬件在 VNF 管理的灵活性方面仍然受到限制。Andrew 等[12]提出了利用 FPGA 扩展数据中心的服务器来提高性能。Arpit 等[13]提出结合可编程交换机和基于软件的流处理器，实现了高效的网络监控功能。Georgios 等[14]提出结合可编程交换机和软件基础设施，支持高性能 NFV 服务链。相比之下，MicroNF 是一种通用框架，可以基于最先进的可编程硬件、软件平台提供高性能和灵活性，并在框架中引入策略解析器层，从而支持多种网络功能，减轻编排器与基础架构相关的管理负担，提高可扩展性。

OpenBox[4]中提出的想法接近于微服务的概

念。但是，它并没有在 NFV 的环境下讨论这一想法。Muhammad 等[15]设计了一种可重用的网络协议栈，供网络功能调用而无需考虑底层分组处理逻辑。Guyue 等[16] 设计了一种 TCP 协议栈，以模块的形式供一条服务链上的多个网络功能调用，提高服务链性能。相比之下，MicroNF 不局限于提供一种网络功能模块，而是一种通用的异构高性能框架，提供网络功能分解方式，实现高性能服务链。

6　结束语

本文为 NFV 提出了一种异构高性能框架 MicroNF，通过硬件和软件基础设施的集成可以有效地支持网络功能。设计了统一的控制接口，以支持更灵活、更高效的 VNF 管理。此外，还将微服务的概念集成到 MicroNF 中，从 NFV 的服务链中删除冗余处理逻辑并提高性能。实验结果表明 MicroNF 可以显著提高服务链性能。

参考文献：

[1] ETSI. Network functions virtualization (NFV): architectural framework[J]. SDN and OpenFlow World Congress, 2013, 1(1):1-16.

[2] SUN C, BI J, ZHENG Z, et al. SLA-NFV: an SLA-aware high performance framework for network function virtualization[C]//The Conference of the ACM Special Interest Group on Data Communication. ACM, 2016: 581-582.

[3] BREBNER G. Softly defined networking[C]// The 8th ACM/IEEE Symposium on Architectures for Networking and Communications Systems. ACM, 2012: 1-2.

[4] BREMLER-BARR, ANAT, YOTAM H, et al. OpenBox: a software-defined framework for developing, deploying, and managing network functions[C]//The Conference of the ACM Special Interest Group on Data Communication. AMC, 2016: 511-524.

[5] LI B, TAN K, LUO L, et al. ClickNP: highly flexible and high performance network processing with reconfigurable hardware[C]//The Conference of the ACM Special Interest Group on Data Communication. ACM, 2016:1-14.

[6] NEWMAN S. Building microservices[M]. California: O' Reilly Media, Inc., 2015: 1-280.

[7] MARTINS J, AHMED M, RAICIU C, et al. ClickOS and the art of network function virtualization[C] // The 11th USENIX Conference on Networked Systems Design and Implementation. USENIX Association, 2014: 459-473.

[8] QUINN P, ELZUR U. Network service header[J].IETF Draft, 2015, 1(1):1-40

[9] ZHU S, BI J, SUN C et al. SDPA: enhancing stateful forwarding for software-defined networking[C]//The 23rd IEEE International Conference on Network Protocols. IEEE, 2015: 323-333.

[10] GANDHI R, LIU H, HU Y, et al. Duet: cloud scale load balancing with hardware and software[J]. ACM SIGCOMM Computer Communication Review, 2015, 44(4):27-38

[11] MOENS H, DE F, TURCK. VNF-P: a model for efficient placement of virtualized network functions[C]//10th International Conference on Network and Service Management. 2014: 418-423.

[12] PUTNAM A, CAULFIELD A, CHUNG E, et al. A reconfigurable fabric for accelerating large-scale datacenter services[C]//ACM/IEEE 41st International Symposium on Computer Architecture. ACM/IEEE, 2014 : 13-24.

[13] GUPTA A, HARRISON R, CARNINI M, et al. Sonata: query-driven streaming network telemetry[C] // The Conference of the ACM Special Interest Group on Data Communication. ACM, 2018: 357-371.

[14] KATSIKAS G P, BARBETTE T, KOSTIC D, et al. Metron: NFV Service Chains at the True Speed of the Underlying Hardware[C] // The 15th USENIX Symposium on Networked Systems Design and Implementation. USENIX, 2018: 171-186.

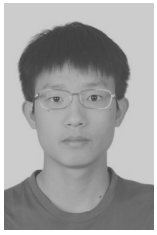
[15] JAMSHED A, MOON G, KIM D, et al. MOS: a reusable networking stack for flow monitoring middleboxes[C] // The 14th USENIX Symposium on Networked Systems Design and Implementation. USENIX, 2017: 113-129.

[16] LIU G, REN Y, YURCHENKO M, et al. Microboxes: high performance NFV with customizable, asynchronous TCP stacks and dynamic subscriptions[C]//The 2018 Conference of the ACM Special Interest Group on Data Communication. ACM, 2018: 504-517.

[作者简介]

孙晨（1992− ），男，黑龙江哈尔滨人，清华大学博士生，主要研究方向为软件定义网络、网络功能虚拟化、网络测量等。

毕军（1972—2019），男，辽宁大连人，博士，清华大学教授、博士生导师，主要研究方向为网络空间安全、软件定义网络、网络功能虚拟化、网络体系结构、源地址验证、域间路由协议、NDN 网络等。



郑智隆（1993− ），男，四川达州人，清华大学博士生，主要研究方向为软件定义网络、网络功能虚拟化等。

王舒鹤（1996− ），男，山西太原人，清华大学博士生，主要研究方向为软件定义网络、网络功能虚拟化、SDN 数据平面可编程等。

胡宏新（1974− ），男，湖北仙桃人，博士，克莱姆森大学助理教授、博士生导师，主要研究方向为软件定义可编程安全、物联网安全与隐私、社交网络安全与隐私、机器学习用于安全与隐私、边缘计算与云计算与移动计算安全等。