www.zte.com.cn/magazine/English

ISSN 1673-5188
CODEN ZCTOAK

ZTE COMMUNICATIONS

An International ICT R&D Journal Sponsored by ZTE Corporation

June 2014, Vol.12 No.2

SPECIAL TOPIC:
Software-Defined Networking

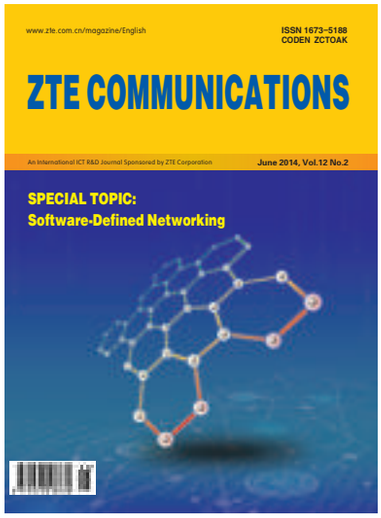
9 771673 518130 06

ZTE COMMUNICATIONS

VOLUME 12 NUMBER 2 JUNE 2014

ZTE Communications Editorial Board

CONTENTS



Special Topic: Software-Defined Networking

CONTENTS

ZTE COMMUNICATIONS
Vol. 12 No.2 (Issue 42)
Quarterly
First English Issue Published in 2003

Supervised by:
Anhui Science and Technology Department

Sponsored by:
Anhui Science and Technology Information Research Institute and ZTE Corporation

Editorial Correspondence:
Add: 12F Kaixuan Building,
329 Jinzhai Road,
Hefei 230061, P. R. China
Tel: +86-551-65533356
Fax: +86-551-65850139
Email: magazine@zte.com.cn

Published and Circulated (Home and Abroad) by:
Editorial Office of
ZTE Communications

Printed by:
Hefei Zhongjian Color Printing Company

Publication Date:
June 25, 2014

Publication Licenses:

ISSN 1673-5188
CN 34-1294/ TN

Advertising License:
皖合工商广字0058号

Annual Subscription:
RMB 80

Software-Defined Networking
Zhili Sun, Jiandong Li, and Kun Yang

Software-Defined Networking

▶ Zhili Sun

Professor Zhili Sun is chair of communication networking at the Centre for Communication Systems Research, University of Surrey, UK. He received his BSc in mathematics from Nanjing University, China, in 1982. He received his PhD in computer science from Lancaster University, UK, in 1991. From 1989 to 1993, he worked as a postdoctoral research fellow at Queen Mary University, London. He has worked in the capacity of principle investigator and technical co-coordinator on many projects within EU framework programs, within the EPSRC, and within industry. He has published more than 125 papers in international journals and conference proceedings and has also authored book chapters. He was the sole author of Satellite Networking: Principles and Protocols, 1st and 2nd editions, published by Wiley in 2005 and 2014 respectively. He was a contributing editor of IP Networking Over Next-Generation Satellite Systems, published by Springer in 2008. He was also contributing editor of the textbook Satellite Communications Systems: Systems, Techniques and Technology, 5th ed., published by Wiley in 2009. His research interests include wireless and sensor networks, satellite communications, mobile operating systems, Internet protocols and architecture, clod computing, SDN, multicast, and security.

▶ Jiandong Li

Professor Jiandong Li received his BS, MS and PhD degrees from Xidian University, China, in 1982, 1985 and 1991. From 1990 to 1994, he was an associate professor at Xidian University and became a full professor in 1994. In 1995, he undertook the role of PhD supervisor at Xidian University. From 2007 to 2012, he was executive vice dean of the Graduate School of Xidian University. From 1997 to 2006, he was dean of School of Telecommunications Engineering, Xidian University. From 2001 to 2003, he was a visiting professor at Cornell University. Professor Li has previously been awarded the National Science Fund Award for Distinguished Young Scholars. He is a senior member of the IEEE, a senior member of the China Institute of Electronics (CIE), and a fellow of the China Institute of Communications (CIC). From 1993 to 1994 and then from 1999 to 2000, he was a member of the Personal Communications Networks Specialist Group for China "863" Communication High Technology Program. He is also a member of the Broadband Wireless Mobile Communication Specialist Group, Ministry of Information Industry, China, and director of the Broadband Wireless IP Standard Work Group, Ministry of Information Industry, China. His main research interests include broadband wireless mobile communications, cognitive and software-defined radio, and wireless ad-hoc networks.

▶ Kun Yang

Professor Kun Yang received his PhD degree from University College London. He received his MSc and BSc degrees from Jilin University, China. He is currently a chair professor in the School of Computer Science and Electronic Engineering, University of Essex, and leads the Network Convergence Laboratory there. Before joining the University of Essex in 2003, he worked for several years at University College London on EU research projects. His main research interests include heterogeneous wireless networks, fixed-mobile convergence, future Internet technology and network virtualization, and cloud computing and networking. He manages research projects funded by sources such as UK EPSRC, EU FP7, and industry. He has published more than 150 journal papers. He serves on the editorial boards of both IEEE and non-IEEE journals. He is a senior member of the IEEE and a fellow of IET.

Software-defined networking (SDN) is a promising technology for next-generation networking and has attracted much attention from academics, network equipment manufacturer, network operators, and service providers. It has found applications in mobile, data center, and enterprise networks. The SDN architecture has a centralized, programmable control plane that is separate from the data plane. SDN also provides the ability to control and manage virtualized resources and networks without requiring new hardware technologies. This is a major shift in networking technologies.

The ITU-T has been engaged in SDN standardization, and the European Telecommunications Standard Institute (ETSI) has been working on network function virtualization (NFV), which complements SDN. The Open Network Foundation (ONF) is a non-profit organization dedicated to promoting the adoption of open SDN. Recently, much work has been done on SDN to meet future network requirements.

Network virtualization creates multiple virtual infrastructures within a deployed infrastructure. These virtualized infrastructures can be created over a single physical infrastructure. Each virtual network can be isolated from each other and programmed to meet user requirements in terms of resource functionality and capacity. This ensures that appropriate network resources are provided to the user.

The SDN framework includes programmable control plane, data-forwarding plane abstraction, and methods to map the virtualized infrastructures onto the underlying physical network infrastructure.

Key issues to be addressed are network resource isolation, network abstraction, topology awareness, quick reconfigurability, performance, programmability, management, mobility, security, and wireless network access.

We received strong response to this call for papers on SDN from network operators, equipment manufacturers, universities, and research institutes. Following a peer-review process, we selected nine papers for inclusion in this special issue.

The first paper, "Network Function Virtualization Technology: Progess and Standardization" discusses the main challenges in SDN faced by network carriers. This paper also discusses current standardization activities and research on NFV related to SDN.

The second paper, "Service Parameter Exposure and Dynamic Service Negotiation in SDN Environments," discusses the ability of SDN to facilitate dynamic provisioning of network services. The paper focuses on two main aspects of the SDN framework: network abstraction and dynamic parameter exposure and negotiation.

The third paper, "SDN-Based Broadband Network for Cloud Services," discusses how SDN/NFV will be vital for constructing cloud-oriented broadband infrastructure, especially within data center networks and for interconnecting between data cen-

ter networks. The authors propose SDN/NFV in broadband access to realize a virtualized residential gateway.

The fourth paper, "D-ZENIC: A Scalable Distributed SDN Controller Architecture," describes a solution to minimizing the cost of network state distribution. This solution is a network control platform called D-ZNEIC that supports distributed deployment and linear scale-out by trading off complexity for scalability

The fifth paper, "Software-Defined Cellular Mobile Network Solutions," describes current research on and solutions for software-defined cellular networks. It also discusses related specifications and possible research directions.

The sixth paper, "SDN-Based Data Offloading for 5G Mobile Networks," describes an integrated 4G/Wi-Fi architecture evolved with SDN abstraction in the mobile backhaul and enhanced components that facilitate the move towards 5G.

The seventh paper, "Integrating IPsec Within OpenFlow Architecture for Secure Group Communication," discusses Internet Protocol security (IPsec) in the context of OpenFlow architecture and SDN.

The eighth paper, "Virtualized Wireless SDNs: Modelling Delay Through the Use of Stochastic Network Calculus," describes a delay model for a software-defined wireless virtual network with some theoretical investigation into wireless SDN.

The final paper, "Load Balancing Fat-Tree on Long-Lived Flows: Avoiding Congestions in Data Center Network," describes a dynamic load-balancing algorithm for fat tree in the context of SDN architecture.

Conference Information

2014 The Second International Conference on Advanced Cloud and Big Data (CBD 2014)

November 20-22, 2014, Huangshan, China
http://cbd.seu.edu.cn/cbd2014

Organizing & Program Committees

General Conference Co-Chairs
- Yi Pan, Georgia State University, USA
- You Chin Fuh, IBM

Program Committee Co-Chairs
- Jianzhou Luo, Southeast University, China
- Guth Tschetter,IBM
- Laurence T. Yang, St. Francis Xavier University, Canada

SponsorsCo-Chairs
- Keith Brown, IBM
- Hao Wang, IBM

LocalCo-Chairs
- Ludeng Ge, Anhui University of Technology, China
- Haijiang Lv, Huangshan University, China

Organization Co-Chairs
- Bo Liu, Southeast University, China
- Xiao Zheng, Anhui University of Technology, China
- Tracy Zhu, IBM
- Beibei Shi, IBM

Publication Co-Chairs
- FangDong, Southeast University, China
- Jinghui Zhang, Southeast University, China

Organized by
- Southeast University

Co-Sponsored by
- IBM
- Anhui University of Technology, China
- Huangshan University, China
- ACM Nanjing Chapter
- Jiangsu Computer Society

Scope of Conference

The International Conference on Advanced Cloud and Big Data (CBD) provides a forum for both academics and practitioners who are working on cloud computing and big data technologies to explore new ideas, share their experience and leverage each other's perspectives. Besides the latest research achievements, this conference also covers some innovative commercial issues of cloud computing and big data, such as the commercial data management system, commercial applications, etc., and gives all participants a chance to identify the new/emerging "hot" trends in this important area. We solicit original papers on a wide range of cloud computing and big data topics that can be divided into three tracks but are not limited to:

Research Track
- Cloud Data Privacy
- Cloud Security
- Cloud Resource Management and Performance
- Cloud Data Management
- Storage Architecture of Cloud
- Green Cloud
- Networking Technologies for Data Center
- Virtualization Technologies
- Big Data Processing (Analytics, Querying, Mining)
- Big Data Storage and Management
- Big Data Graph algorithms

Industry Track
- Cloud Computing Solutions
- Cloud Computing Specifications and Standards
- Big Data Economic Analytics
- Big Data in Business Performance Management
- Big Data in Enterprise Models and Practices

Application Track
- Cloud Computing Platforms
- Mobile Cloud Computing Applications
- Big Data As A Service
- Big Data Platforms
- Big Data Toolkits

Paper Submission

Submitted manuscripts must be formatted in standard IEEE US Letter Format: http://www.ieee.org/conferences_events/conferences/publishing/templates.html and must be submitted via EasyChair (https://www.easychair.org/conferences/?conf=cbd2014) as PDF files. The review version is limited to 8 pages (IEEE proceedings format), including references and illustrations. Submitted papers should not be previously published in or be under consideration for publication in another conference or journal. Submission of a paper should be regarded as an undertaking that, should the paper be accepted, at least one of the authors will attend the conference to present the paper.

Publication

All submitted papers will be reviewed by program committee members and selected based on their originality, significance, relevance, and clarity of presentation. Accepted papers will be all published by Conference Publishing Services (CPS) and will be submitted for indexing to EI (Compendex). Authors of selected papers will be invited to submit revised and expanded version of their papers to be considered for publication in special issues of well-known international journals such as Cluster Computing (SCI), International Journal of Cloud Computing, ZTE Communications, etc.

Contact

Dr. Bo Liu
Southeast University, China
Tel: +86-25-52091013
Email: cbd@seu.edu.cn

Important Dates:
- Submission deadline: **July20, 2014**
- Notification of acceptance: Sep 1, 2014
- Camera Ready Due: Sep 20, 2014
- Registration Due: Sep20, 2014

DOI: 10.3969/j. issn. 1673-5188. 2014. 02. 001
http://www.cnki.net/kcms/detail/34.1294.TN.20140603.1053.001.html, published online June 3, 2014

Network Function Virtualization Technology: Progress and Standardization

Huiling Zhao, Yunpeng Xie, and Fan Shi
(China Telecom Beijing Research Institute, Beijing 100035, China)

Abstract

Network innovation and business transformation are both necessary for telecom operators to adapt to new situations, but operators face challenges in terms of network bearer complexity, business centralization, and IT/CT integration. Network function virtualization (NFV) may inspire new development ideas, but many doubts still exist within industry, especially about how to introduce NFV into an operator's network. This article describes the latest progress in NFV standardization, NFV requirements and hot technology issues, and typical NFV applications in an operator networks.

Keywords

network functions virtualization (NFV); overlay network; virtual extensible LAN (VXLAN); service chaining

1 NFV Standardization Progress

1.1 ETSI NFV Progress

In October 2012, AT&T, British Telecom, Deutsche Telekom, Orange, Telecom Italia, Telefonica, and Verizon established the Network Functions Virtualization Industry Specification Group (NFV ISG) in the ETSI. This group will define the specifications for architecture that supports NFV hardware and software and will create a guide to virtualized network functions. NFV ISG will cooperate with other standards organizations to consolidate existing virtualization technologies and standards.

NFV ISG intends to leverage standard IT virtualization technology and consolidate many different types of network equipment into industry-standard, high-volume servers, switches and storage. Software with particular functions could be installed or uninstalled on hardware in various locations in a network, and new equipment would not need to be installed. Benefits of NFV for network operators and customers include [1]:

- reduced equipment cost and power consumption
- lower capex and opex
- increased speed of deployment and provisioning of new network services
- increased investment margins for new services
- a virtual appliance market that is open to pure software entrants
- encourages more innovation and new services for much lower risk.

NFV ISG now has 184 members, including operators, network equipment vendors, IT equipment vendors, and technology vendors. The NFV ISG has a technical steering committee that manages four working groups and two expert groups. Different working groups and expert groups focus on:

- architecture for the virtualization infrastructure, including infrastructure requirements in the computing, storage, and network domains
- management and orchestration, including NFV platform management functions such as network mapping for end-to-end services, allocation and expansion of hardware resources, and VNF instance tracing
- software architecture, including the implementation environment for VNF
- reliability and availability, including resilience and fault tolerance through VNF load-allocation approaches and VNF instance portability
- security of NFV platforms
- performance and portability, including scalability, efficiency, and migration performance, from dedicated platforms to general-purpose hardware.

In 2013, NFV ISG focused on designing high-level documents. It has released NFV use cases, requirements, architecture, terminology, proof of concept (PoC), and other technical documents as well as NFV White Paper V1.0 and NFV White Paper V2.0. The focus of NFV ISG has shifted from identifying requirements to defining them, and NFV ISG is attempting to achieve feasible results by specific deadlines. In the first half of 2014, NFV ISG has focused on PoC and is looking to collect

and evaluate products and prototypes that satisfy NFV requirements. This will help promote NFV development. By the end of February 2014, nine PoC proposals had been accepted.

Not long ago, NFV ISG also released NFV Phase 2 discussion draft, which specifies the work plan for the first two years of NFV ISG. In this draft, two points should be noted. First, NFV ISG plans to establish an NFV steering board (NSB), which will be a major organizational entity focused on promoting NFV work. Compared with NFV ISG TSC, the NSB will have a more rights. The NSB not only coordinates technologies, as the current TSC does, but it also supervises the progress of NFV ISG. Second, an ad hoc group will be established to replace the existing working groups and expert groups. According to the NFV Phase 2 plan, the objectives and tasks of this ad hoc group will be specified by the fourth quarter of 2014, and the ad hoc group will begin work in 2015.

1.2 Network and NFV Standardization: CCSA Efforts

The China Communications Standards Association (CCSA) pays much attention to network and NFV standardization and guides NFV study and application in China. Software and virtualization have become important trends in the evolution and development of future networks. These two topics are highly complementary, and relevant representative technologies and protocols will be the basis of future networks. However, software-defined networking (SDN), NFV technologies, and the architecture of future networks are all still being studied and depend on the development of relevant technical standards.

TC1 of the CCSA focuses studies standards related to IP and multimedia communication. A large amount of study is being done on the virtualization of data centers, CDNs, and broadband bearer networks. These standards have been completed:

- scenarios and requirements of future data networks (FDNs) (industry standard)
- general requirements of internet data center based on virtualization technologies (industry standard)
- router virtualization technical requirements (association standard)
- impacts of network edge virtualization on MAN (study subject).

Other industry standardization projects that have been initiated include:

- scenarios and requirements of stream-specific state migration in cloud data centers
- orchestration scenarios and technical requirements of FDN services based on cloud computing management platforms
- application scenarios and technical requirements of FDN-based CDN
- application scenarios and technical requirements of FDN-based broadband customer networks
- technical requirements of FDN-based broadband network access servers.

TC3 of the CCSA has established the Software Virtualization Network (SVN) Work Group and has been studying NGN key technologies, equipment, signaling protocols, and network architecture evolution. TC3 is a major technical work committee for telecommunications network architecture. The design of future network architecture and relevant technologies is within the scope of this work group. Intelligent communications network technology, which is a particular focus of TC3, is the foundation of future networks and will profoundly affect the future development the entire ICT network.

The CCSA TC3 SVN group, also called the Software Intelligent Communications Network Work Group, studies the architecture and key technologies of future SDN and NFV networks. CCSCA TC3 SVN undertakes relevant standardization and provides important references for development in this field.

One of the main subjects of TC3 SVN is the requirements, frameworks, and key technologies of SDN-based intelligent communication networks. This subject encompasses:

- general requirements of SDN-based intelligent communication networks
- perception analysis in SDN-based intelligent communications networks
- traffic scheduling in SDN-based intelligent communications networks
- policy control in SDN-based intelligent communications networks
- evolution of existing networks to SDNs.

The other main subject of TC3 SVN is the requirements, frameworks, and key technologies of network virtualization. This subject encompasses:

- general requirements of network virtualization
- virtualized network functions
- virtualized network services
- virtualized evolution of existing networks.

TC3 SVN has also studied the requirements, frameworks, and key technologies of future networks. It has initiated study on the general technical requirements of SDN-based intelligent communication networks, technical requirements of SDN-based intelligent perception systems, control-plane platform virtualization of core networks, and technical requirements of SDN/NFV-based virtualized IMS.

2 Hot Technology Issues in NFV

NFV partly borrows from existing network virtualization technologies and also incorporates new technologies, such as software virtualization and SDN. NFV properly abstracts, splits, and schedules network function sets and involves many technologies. In this paper, only overlay network, virtualized traffic scheduling, virtual cluster, and networking technologies are discussed.

2.1 Overlay Network Technologies

Overlay network technologies are used to implement virtual-

ization over existing network architecture, and the basic overlay network is not greatly changed. Thus, application bearers can be established in the overlay network and are separate from other network bearers. At present, overlay network technologies are mainly used for high-volume interconnection in the internal networks of datacenters. Here, we describe mainstream overlay network technologies.

2.1.1 Virtual Extensible LAN (VXLAN)

VXLAN [2] is an important virtualization technology and subset of IETF standard drafts. VXLAN enables network virtualization by using MAC-in-UDP encapsulation to overlay a layer-2 network onto a layer-3 network. Each VXLAN is identified with a 24-bit VNI. VXLAN encapsulation enables the layer-2 to communicate with any end point as long as the end points are in the same VXLAN segment. These end points may not necessarily be in the same IP subnet, so the problem of limited MAC address capacity in switches is eliminated.

2.1.2 Network Virtualization Using Generic Routing Encapsulation (NVGRE)

NVGRE uses the GRE tunneling protocol encapsulation, defined in RFC 2784 [3] and RFC 2890 [4], to create an independent virtual layer-2 network. In NVGRE, address learning is implemented by the control plane, but NVGRE has previously had no specific implementation solution for address learning until now. Compared with VXLAN, NVGRE is defective in terms of load sharing, i.e., NVGRE cannot implement GRE key-based load sharing. In addition, NVGRE tunnels are end-to-end, so the number of tunnels increases exponentially as the number of terminals increases. As a result, the overhead for tunnel maintenance becomes very large.

2.1.3 Stateless Transport Tunneling (STT)

STT is also an overlay technology used to create a layer-2 virtual network over a layer-2 or layer-3 physical network [5]. In technical terms, STT is very similar to VXLAN. Tunnel end points of STT are also provided by a hypervisor vSwitch; VNIDs of STT are also 24-bit; and STT has a multipath advantage by controlling transmission source packet headers. The difference between STT and VXLAN is that STT fragments data frames before encapsulation. Thus, the hardware acceleration of network cards can be fully utilized for higher efficiency. In addition, STT disguises STT packets as TCP/IP packets, and TCP packet headers do not maintain TCP state information; thus, re-transmission does not occur after packet loss. In this way, STT tunnels are less reliable.

2.2 Virtualized Resource Scheduling Technologies

Virtualized resource scheduling technologies use SDN and NFV to virtualize and intelligently schedule network traffic, service functions, and other resources. Such technologies are mainly used for virtualized traffic scheduling and service chaining.

2.2.1 Virtualized Traffic Scheduling

Virtualized traffic scheduling overcomes the limitation of a distributed IP network routing by using virtualization technologies. It uses centralized route computing and traffic scheduling to dynamically balance traffic and optimize the architecture across the whole network. Virtualized traffic scheduling is mainly used in IP backbone networks to determine 1) how to define the abstraction of IP route function sets, 2) the implementation mode of centralized route decision systems, 3) the reliability of centralized systems, and 4) the real-time algorithms used to compute protection paths in this mode. The current trend for virtualized traffic scheduling is SDN and other new technologies, e.g., adding a PCE/controller system to implement a centralized route-decision system.

2.2.2 Service Chaining

Virtual firewalls, load balancers, gateways, and other service-processing functions in a network are called service function points. By processing traffic at a series of service-function points, a service chain is formed. This process is called service chaining [6]. Unlike virtualized traffic scheduling, service chaining focuses on server programming for controlling traffic forwarding in a virtual network. Because it has been promoted by SDN and NFV, service chaining has received much attention, and it is widely considered to have good prospects.

2.3 Virtual Cluster Technologies

A virtual cluster is formed when virtualization technologies are used to logically combine network elements (NEs) or their internal components in order to meet operational and management requirements. Currently, the study of virtual cluster technologies is focused on homogeneous and heterogeneous virtual clusters.

2.3.1 Homogeneous Virtual Clusters

By expanding the control plane, a homogeneous virtual cluster virtualizes multiple physical devices of the same type into a single logical device. The cluster implements resource sharing and flexible scheduling in these physical devices through a resource controller. By means of pooling, the virtual cluster has a uniform control plane and management plane and uses a unique ID. Compared with the original physical devices, the virtual cluster has much greater capacity and is much more reliable. This technology is mainly used in backbone networks to solve the problem of insufficient single-server forwarding and insufficient throughput in core nodes. Moreover, it can be used for multi-service edge (MSE) pooling in IP networks and mobility management entity (MME) pooling in core networks.

2.3.2 Heterogeneous Virtual clusters

A heterogeneous virtual cluster consolidates different types

of physical devices in distributed mode. Thus, the number of managed or configured NEs and NE types is reduced, and service and network deployment can be made more flexible and efficient. At present, research is focused on virtual clusters of access control devices and switches, optical network unit (OLT) access control devices and home gateways, and routers and optical transport network (OTN).

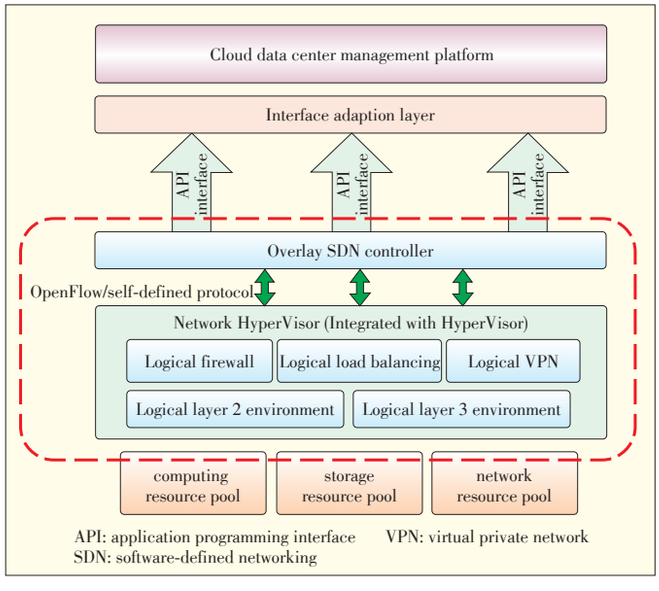
3 Typical Applications of NFV

In the face of market competition, operational requirements, and increasing maintenance costs, network operators have begun to explore NFV and have attempted to it to satisfy specific service requirements in data centers, mobile core networks, and home networks.

3.1 Data Center Network Virtualization

Data center network virtualization [7] comprehensively shields underlying physical network appliances in overlay mode. In a virtualized data center network, physical network resources are shared, and different tenants are isolated by software or programming. Each tenant has a separate network definition, including networking, traffic control, and security management. A cloud data center resource-management platform is connected to an SDN controller through API interfaces. By means of programming, a multitenancy network can be flexibly deployed, and inter-datacenter deployment is also possible. **Fig. 1** shows datacenter network virtualization [8]–[10].

Data center network virtualization does not depend on underlying networks so that security, traffic, and performance policies can be flexibly implemented for different tenants. The network can also be automatically configured because of the programming capabilities [11]. After overlay network technologies are introduced, however, the network architecture becomes more complicated, and the physical network cannot perceive the logical network. In addition, network performance is compromised because the logical network is controlled by software.



▲Figure 1. Data center network virtualization.

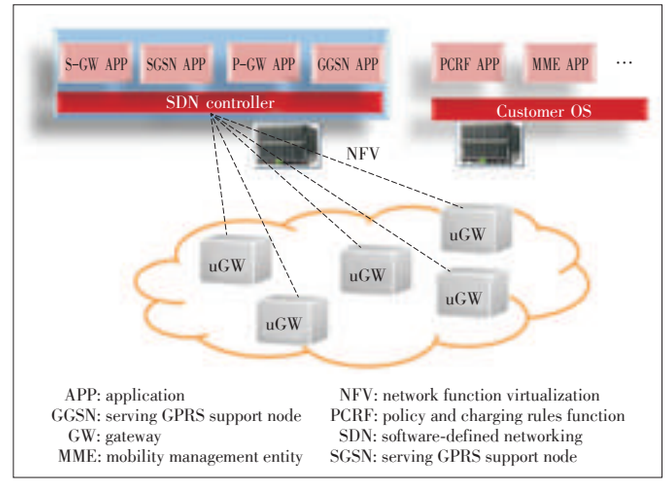
3.2 EPC NE Virtualization

Virtualization of evolved packet core (EPC) network involves the use of a three-layer application + controller + switch architecture. Control functions, including traffic flow and traffic processing, are implemented by applications + controller layers. The switch layer implements stream-based forwarding functions or even integrates DPI and other traffic analysis and processing functions. Control-plane NEs are gradually centralized, and a virtual control cloud is formed in the mobile core network by converging the System Architecture Evolution (SAE) gateway signaling plane with the MME or policy and charging rules function (PCRF) [7]. **Fig. 2** shows EPC NE virtualization [12].

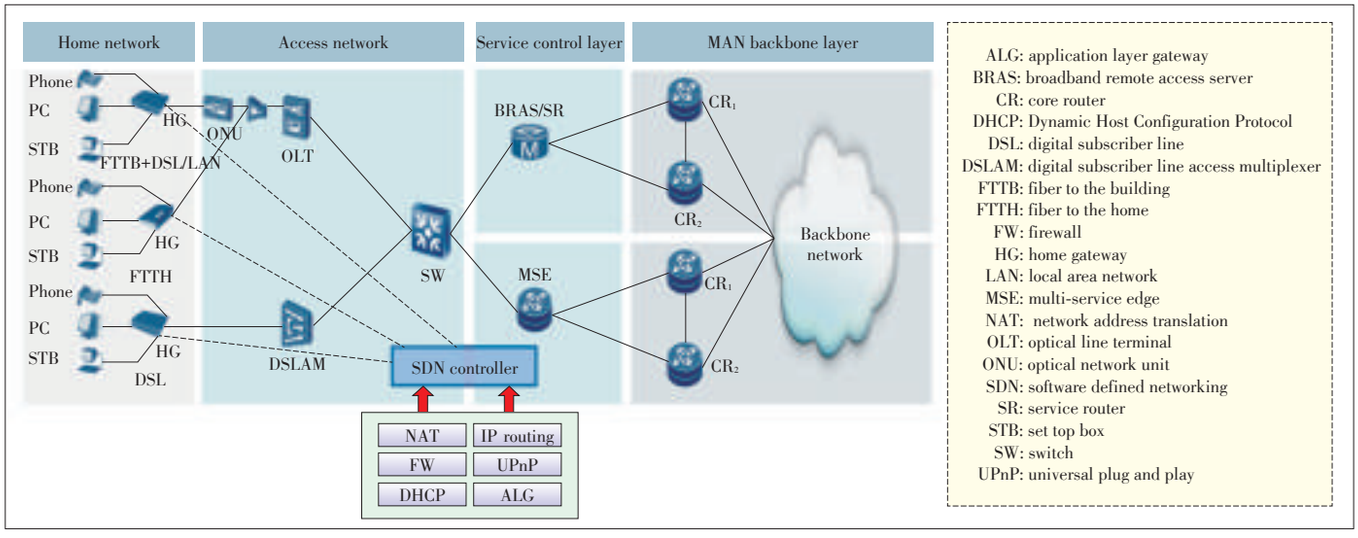
EPC NE virtualization unifies the network hardware architecture with the NFV technology, so that the cost will not increase greatly due to increasing capacity. By separating service control from forwarding and separating software from hardware, EPC NE virtualization allows flexible service deployment and enhancement, and thus reduces CAPEX and OPEX for network operators.

3.3 Home Network Virtualization

Home network virtualization separates control plane functions and service processing functions (such as firewall, address management, device management, and fault diagnosis) of home gateways (HGs) and set top boxes (STBs) in home networks, and migrate these functions to the controller side or cloud end after virtualization. On the HGs and STBs, only physical interfaces and data plane layer 2 forwarding functions are remained [7], [13]. **Fig. 3** shows an application scenario of home network virtualization.



▲Figure 2. Core network single-NE virtualization.



▲**Figure 3. A home network virtualization scenario.**

Home network virtualization simplifies end-user premises. Network operators can provide remote network fault diagnosis without continuously maintaining and upgrading STBs and HGs. Thus, services are more manageable, and less power is consumed. Home network virtualization also makes service deployment more flexible. Operators can deploy new hardware or software quicker and easier so that the time to market is reduced [14].

4 Conclusion

NFV has succeeded in the IT industry and has entered the operator landscape. NFV has many advantages in various scenarios and is a growing trend in the telecom industry.

However, NFV is still being standardized; relevant technical standards are not yet complete and require further in-depth study. Take service chaining for example. The functional points and logical combination sequence of service chaining may differ for different services. Therefore, it is general purpose service chaining applications urgently need to be defined. The content of NVF will improve as ETSI and CCSA continue with their study and formulation of relevant standards.

References

[1] ETSI. (2013, Oct. 17). *Network Function Virtualization — Introductory White Paper* [Online]. Available: http://portal.etsi.org/nfv/nfv_white_paper.pdf
[2] *NFV-INF Network Domain Interworking-Data Plane*, ETSI NFV INF (13) 000056, Aug. 2013.
[3] *Generic Routing Encapsulation (GRE)*, IETF RFC 2784, Mar. 2000.
[4] *Key and Sequence Number Extensions to GRE*, IETF RFC 2890, Sept. 2000.
[5] Vishwas Manral. (2012, Mar. 22). *Stateless Transport Tunneling (STT): Yet another cloud encapsulation or next-generation VxLAN?* [Online]. Available: http://h30507.www3.hp.com/t5/HP-Networking/Stateless-Transport-Tunneling-STT-Yet-another-cloud/ba-p/109559
[6] *Network Functions Virtualisation (NFV) Use Cases*, ETSI GS NFV 001 V1.1.1, Oct. 2013.
[7] *Scenarios and Requirements of Future Data Network*, CCSA Industry Standard, 2013.
[8] Xuan Luo, Baoqing Huang, Jianwen Wei, and Yaohui Jin, "Data-Center-Oriented SDN," *China Education Network*, no. 100, pp. 24–27, Aug. 2013.
[9] Qian Wang, Huiling Zhao, and Yunpeng Xie, "Standardization and Deployment of SDN," *ZTE Technology Journal*, vol. 19, no. 5, pp. 2–5, Oct. 2013.
[10] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Apr. 2008. doi: 10.1145/1355734.1355746.
[11] Baohua Lei, Feng Wang, Qian Wang, Heyu Wang, Yunpeng Xie, and Fan Shi, *Deciphering SDN: Core Techniques and Practical Guide*, Beijing, China: Publishing House of Electronics Industry, 2013.
[12] SDNAP. (2013, Oct. 20). *Architecture of ETSI NFV* [Online]. Available: http://www.sdnap.com/sdnap-post/2856.html
[13] *High Level Requirements and Framework for SDN in Telecommunication Broadband Networks*, BBF SD-313, Mar. 2013.
[14] *Network Located Residential Gateway*, BBF PD-295, Oct. 2012

Manuscript received: 2014-03-05

Biographies

Huiling Zhao (zhaohl@ctbri.com.cn) is director of Cloud Computing Research Center and chief engineer of Beijing Research Institute of China Telecom Corporation Limited. She is also the executive director of China Institute of Communications, chairman of the Information and Communications Network Technology Professional Committee, vice president of Beijing branch of China Institute of Communications, chairman of the Networking and Switching technology Committee of CCSA, and a member of the MEF Board. She has previously been granted special government allowances. She is one of the experts leading broadband network projects and tri-network integration projects in China's 12th Five Year Science Plan.

Yunpeng Xie (xieyp@ctbri.com.cn) is a senior engineer of the Network Architecture and Cutting-Edge Technology Study Group at the Network Technology Department of Beijing Research Institute of China Telecom Corporation Limited. His research interests include SDN/NFV and future networks. He has received one provincial award, and submitted four patents applications. He is a joint author of two monographs, and has published more than 10 papers.

Fan Shi (shifan@ctbri.com.cn) is director of the Network Architecture and Cutting-Edge Technology Study Group at the Network Technology Department of Beijing Research Institute of China Telecom Corporation Limited. He is also the co-chair of MEF China Working Group and the leader of CCSA TC3 SAV working group. His research interests include SDN/NFV and next-generation internet.

DOI: 10.3969/j. issn. 1673-5188. 2014. 02. 002
http://www.cnki.net/kcms/detail/34.1294.TN.20140526.1800.001.html, published online May 26, 2014

Service Parameter Exposure and Dynamic Service Negotiation in SDN Environments

M. Boucadair and C. Jacquenet
(Orange Group, 4 rue du Clos Courtel, Cesson-Sévigné, 35512, France)

Abstract

Software-defined networking (SDN) is a generic term and one of the major interests of the telecoms industry (and beyond) over the past two years. However, defining SDN is a somewhat controversial exercise. The claimed flexibility, as well as other presumed assets of SDN, should be carefully investigated. In particular, the use of SDN to dynamically provision network services suggests the introduction of a certain level of automation in the overall network service delivery process, from service parameter negotiation to delivery and operation. This paper aims to clarify the SDN landscape and focuses on two main aspects of the SDN framework: network abstraction, and dynamic parameter exposure and negotiation.

Keywords

software-defined networking (SDN); service parameter exposure and negotiation; network operation automation; autonomic networking

1 Introduction

Software-defined networking (SDN) has become one of the hottest topics in the telecoms industry over the past two years. Although the definition of SDN is contested, there seems to be rough consensus within the Internet community that SDN promises dynamically programmable, configurable, responsive networks for optimized, somewhat automated network service delivery and operation.

This paper gives a network provider's view of some of the techniques under the SDN umbrella that may be used to introduce a high degree of automation in the preliminary stages of a typical network service lifecycle. This paper describes the dynamic negotiation of service parameters which, when completed, feed the computing intelligence of an SDN architecture so that corresponding resources can be allocated and appropriate policies can be enforced.

The ability to dynamically expose and negotiate the set of parameters that pertain to a given network service is promising and should help introduce a high degree of automation in the overall service-delivery procedure. This will, in turn, facilitate the use of autonomic networking techniques. Currently, most if not all existing network services typically delivered over an Internet Protocol/Multiprotocol Label Switching (IP/MPLS) infrastructure assume little or no negotiation besides price negotiation, if any. A customer is presented with a set of services they may want to subscribe to (e.g., a basic Internet service or VPN service), but the service parameters are hardly detailed.

Most residential customers are unlikely to care about the technical details that define the service they have subscribed to as long as they can access the service with a perceived, often qualitative, QoS. Nevertheless, there are other customers, such as corporate customers or peering network providers, who pay attention to the quantitative definition of the service they have subscribed to as well as the possible penalties for failure to adhere to the terms of the contract. With these kinds of customers, negotiation is often static. Service parameters may vary from one customer to another and from one service to another. There are often long negotiations between when the service production chain kicks in (and the long-awaited order is processed) and when the service is actually delivered.

There is therefore a need to automate the service parameter exposure and negotiation procedure somewhat. Service parameter exposure is 1) the process of capturing the service requirements of an application or customer and 2) presenting the benefits of an underlying infrastructure to a service and customer. Service parameter negotiation involves a customer and network provider mapping the customer's service requirements with the underlying network capabilities. We believe such an advanced procedure requires a standard template that details all the service parameters that may be valued as a function of the service to be delivered, which must conform to the customer's expectations.

In this paper, a network provider is the entity that owns and

This work was supported in part by the EIT SOFTNETS Project.

administers one or many transport domain(s) and is responsible for ensuring connectivity services (e.g., offering global or restricted reachability). A customer subscribes to a service offered by a provider.

The paper is organized as follows: In section 2, we introduce SDN and the techniques we believe it encompasses. In section 3, we introduce connectivity provisioning profile (CPP) [1], which facilitates the exposure and negotiation of service parameters. In section 4, we discuss Connectivity Provisioning Negotiation Protocol (CPNP) [2], which is one of the candidate protocols for conveying service parameter negotiation arguments between two parties (typically a customer and network provider). In section 5, we outline additional areas of investigation that complement the dynamic negotiation of service parameters. These areas are SDN bootstrapping procedures and dynamic service structuring based on ordered sets of elementary service functions, also known as service function chaining (SFC) [3].

2 Introduction to Software-Defined Networking

2.1 Global Framework

Networking environments generally have three planes: forwarding, control, and management. Each of these planes supports various features, e.g., forwarding and routing, traffic engineering (TE), and security and management.

These capabilities need to be configured in the switches, routers, service platforms, etc. according to the services supported by the network and, ideally, according to the customer's requirements in terms of QoS, service robustness and availability, and service resiliency.

In the case of inter-provider network services, implementing such capabilities involves exchanging (configuring) information via various protocols and interfaces located between the forwarding, control, and management planes; between the customer and network (customer-to-network interface, CNI); and between two networks (inter-carrier interface, ICI) (**Fig. 1**).

Subsequent network operations, such as the processing of a customer order, can be triggered by an application. Other network operations, such as connection of an additional site to a VPN infrastructure, can be triggered by customer requests. And yet other network operations can be triggered by internal service engineering modules, e.g., within the context of a maintenance period. Any combination of these is also possible. The configuration information used by participating devices to deliver a service is the product of various combined inputs and includes:

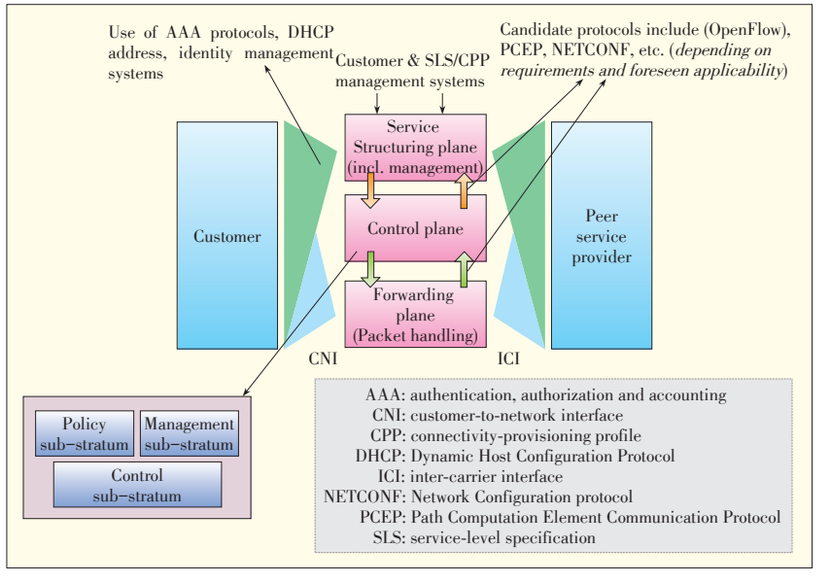
- The actual capabilities supported by the participating devices.
- The completion of the service-parameter negotiation between the customer and the provider, which conforms to customer's requirements and network resource information. Such parameters often follow the guidelines of business development teams. Service parameters can be negotiated according to a specific template, such as a CPP template detailed in section 3.
- The traffic forecasts that can be derived from the number of CPP templates that need to be processed over different time scales (a minute, a day, etc.) but which also relate to the network planning policy that needs to be enforced over several years.
- The number and the nature of the various, possibly service-specific policies enforced by the network provider. Such policies may not only rely on the device capabilities or information derived from the customer requirements; they may also reflect technology-specific recommendations.

2.2 From Service Subscription to Delivery and Operation

Fig. 1 shows a service-structuring layer that is meant to document the nature of the service, including its scope and associated parameters. This service-structuring layer is where negotiation between the customer and network provider takes place [4].

The management plane can then be seen as a substratum of the service-structuring layer, and the management layer is where management information is maintained. Such information is used according to the typical ISO-defined Specific Management Functional Areas (SMFAs), i.e., fault, configuration, accounting, performance and security management information, and is detailed in information and data models.

The control plane is where policy and management substra-



▲**Figure 1. Three-plane representation of networking environments and related interfaces.**

tums reside. These are designed to derive the outcomes of the CPP-formatted service parameter negotiation into policy-provisioning information, such as metrics to be assigned to link interfaces and constraints to be taken into account by, for example, a Constrained Shortest Path First (CSPF) [5] algorithm for TE policy enforcement.

Such policy-provisioning information can either be service- or customer-specific, depending on the nature and number of services that a customer can subscribe to.

This policy-provisioning information is then passed to network devices as configuration information so that the requested service can be delivered. The forwarding of this configuration information may rely upon a variety of protocols that include but are not limited to:

- OpenFlow, which is used exclusively to populate the forwarding information base (FIB) maintained by network devices and is now complemented by the NETCONF-based OpenFlow Management Configuration Protocol [6]
- Path Computation Element Communication Protocol (PCEP) [7], which is used in particular to provision VPN configuration information
- Network Configuration Protocol (NETCONF) [8]
- Common Open Policy Service (COPS) protocol, which is used in particular to support policy provisioning (COPS-PR) [9]
- Interface for Metadata Access Points (IF-MAP) [10]

2.3 The Deterministic Nature of Dynamic Network Operation Procedures

In physics, determinism refers to the principle that the values of a system's variables at a given time determine the values of the same variables at a later time.

In SDN, determinism is a key feature of dynamic, (possibly) automated service-delivery procedures. It is expected that resources and policies used to deliver and operate any given network service will derive from the service parameters that have been negotiated between the customer and network provider.

Indeed, the behavior of systems deployed into operational networks should be predictable and controlled. The outputs and states of those systems should be deterministic and without unexpected behavior, which risks provoking chaotic situations.

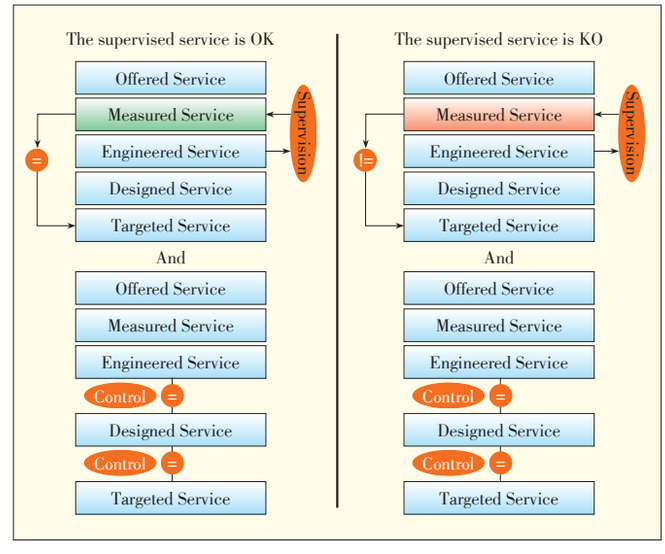
From a deterministic standpoint, a high degree of automation can be introduced into a system only if automation relies on well-known, carefully designed procedures. Such procedures can be decomposed into state machines, policies, etc., which reflect the different behaviors of the system under various conditions. This means that how the service/network behaves in certain circumstances, with particular entries, is known in advance, and the expected result of such behavior is predictable and deterministic.

The path to full network automation is paved with numerous challenges. In particular, it is critically important that automation is well implemented in order to facilitate testing, including validation checks, and troubleshooting. This suggests the need for simulation tools that accurately assess the impact of introducing high-level automation into the overall service delivery procedure and avoid the typical "mad robot" syndrome. This syndrome recently made some Google services unreachable. On January 24, 2014, most Google users who subscribed to log-in services such as Gmail, Google+, Calendar, and Documents were unable to access those services for approximately 25 minutes. For about 10 percent of users, the problem persisted for as long as another 30 minutes.

This recent event further emphasizes the need for a network automation system that relies on deterministic behaviors that can be used during service operation cycles. The behavior exhibited under normal operating conditions should be the input to such an automation system in order to, for example, assess whether a service is up and running as expected. The automation system must also integrate a feedback loop to assess whether policies are properly enforced and whether the set of policies is consistent with service objectives. The automation system can be pre-wired to indicate how it will react when a problem occurs. This deterministic assessment capability can also be implemented inside the supervision system in order to improve fault detection.

The recent Google misadventure is a lesson that should not be forgotten by SDN proponents who claim that service flexibility and agility come at no cost. Automation is where the complexity resides, and so-called "service orchestration intelligence" should not be solicited, regardless of the nature and number of services to be delivered, without accurate modeling of predictable behaviors.

Fig. 2 shows the service lifecycle and control loops that should be implemented in order to assess whether an engineered service matches the service objectives, as expressed by



▲**Figure 2. The importance of control loops.**

business development teams, or customers.

Fig. 2 shows an example of the steps for delivering a service. The challenge facing a network provider is to deliver a service that corresponds to the "measured service" level and that fulfils the clauses of a targeted service. The targeted service level is technology-agnostic; that is, it is described as a set of service requirements that are translated into and accommodated within an architectural and technological view during the designed-service phase.

Once this is accomplished, suitable engineering rules are written up, and the required configuration information is derived. This is the engineered service stage. The advent of SDN contributes to a high level of automation in each step of the service lifecycle.

Completion of the service-parameter negotiation phase, or a dedicated trigger received from an application [11], including an internal application managed by the same provider [12], provide input to the SDN intelligence so that the corresponding service can then be structured according to the service-specific policy-provisioning information derived from the negotiation.

Such policy-provisioning information is then translated into device-specific configuration information. Upon completion of these configuration tasks, the service is delivered to the customer in a completely deterministic manner.

During service operation, certain techniques are used for service fulfillment and assurance. In particular, monitoring techniques are used to verify that policies are properly enforced and to ensure that the delivered service complies with the outcomes of a negotiation with the customer, or what has been requested by a customer (in the case of a subscription mode), or what has been requested by internal business development teams.

2.4 A Tentative Definition of Software-Defined Networking

Fig. 1 underscores some of the current discussions related to SDN. The so-called separation of forwarding and control planes, beyond implementation considerations, has almost become a gimmick to promote flexibility as a key feature of SDN.

Flexibility, which is heavily touted by SDN promoters, is undoubtedly a key objective for network providers. The ability to adapt to a wide range of customer requests and flexibly deliver network services is an important competitive advantage. However, flexibility is much, much more than separating the control and forwarding planes in order to facilitate forwarding decision-making processes.

Here, we define SDN as the set of techniques used to facilitate the design, delivery, and operation of network services in a deterministic, dynamic, scalable manner.

2.5 Meta-Functional Domains of Software-Defined Networking

Such a definition assumes a high level of automation in overall service delivery and operation procedures. From this perspective, SDN techniques can be divided into the following functional meta-domains [13]:

- Techniques for dynamic discovery of network topology, devices, and capabilities along with relevant information models that are precisely document such topology, devices, and capabilities.
- Techniques for dynamically exposing and negotiating service parameters, which are used to measure the level of quality associated to the delivery of a given service or a combination of services. These techniques are not only meant to be used in business roles (e.g., by the customer) but also by applications and services. We assume that these triggers can be implemented using a common information model (section 3) that reflects the set of service-inferred policies. These policies are enforced by the network provider and ease the automation of network operations through dynamically and automatically translating customer, application, and service connectivity requirements to network-management actions.
- Techniques used by dynamic resource allocation schemes and policy enforcement schemes derived from service requirements. These techniques include techniques for automatically setting traffic engineering objectives for a given network.
- Dynamic feedback mechanisms that assess how efficiently a given policy (or set of policies) is enforced from a service-fulfillment and assurance perspective. Such feedback mechanisms have features such as self-tuning and autonomic service diagnosis and repair.

Several approaches can be taken with the proposed SDN framework: application-initiated network programming [11], CPP-inferred [1, section 1], or path computation element-based [7].

Here we focus on the second meta-domain mentioned above, and discuss the benefits and stakes in dynamic service parameter negotiation.

3 Exposing Network Services: The CPP Concept

Defining a clear interface between services, including third-party applications, and network layers has various advantages, such as rationalizing the engineering of network infrastructures. The CPP interface is designed to expose and characterize, in a technology-agnostic way, the IP transfer requirements that need to be met when invoking the IP transfer capabilities of a provider network. These requirements are then translated into IP/MPLS-related technical clauses that, for example, define the class of service and specify the need for recovery means and control-plane protection. At a later stage, these clauses are addressed by the activation of adequate network features and technology-specific actions (e.g., MPLS-TE), Resource Reservation Protocol (RSVP), Open Shortest Path First

(OSPF) or Intermediate System to Intermediate System (IS-IS) configuration, etc.).

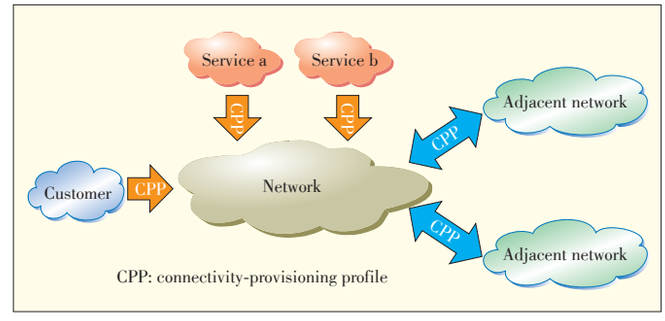
The CPP template is designed to capture connectivity needs and represent and value these requirements in a standardized way. Service- and customer-specific IP provisioning rules may dramatically increase of the number of IP transfer classes that need to be (pre)-engineered in the network. Instantiating each CPP into a distinct class of services should therefore be avoided for the sake of performance and scalability. Application-agnostic IP provisioning is recommended because the requirements and guarantees in the CPP determine which network class of service can be used. From this perspective, the CPP is used to design and engineer a limited number of generic classes so that individual CPP documents, which capture the connectivity requirements of services, applications and Customers, can be easily mapped to these classes. **Fig. 3** shows the connectivity-provisioning interfaces covered by CPP: customer-network service-network, and network-network. Services and applications using CPP via the service-network connectivity-provisioning interface may belong to the same administrative entity managing the underlying network or to a distinct administrative entity managing the underlying network.

A generic CPP template facilitates 1) automation of service negotiation and activation processes, which thus accelerates service provisioning, 2) setting of traffic objectives of TE functions and service-management functions by means of formalized parameters, and 3) improvement of service and network management systems with decision-making capabilities based on negotiated/offered CPPs. The CPP defines the set of IP/MPLS transfer guarantees to be offered by the underlying transport network. It also defines capacity needs and reachability scope, i.e., the set of destinations that can be reached from a customer site, within the context of a given service. Appropriate performance metrics, such as one-way delay or one-way packet delay variation, characterize the IP transfer service. Guarantees on availability and resiliency are also included in the CPP. **Fig. 4** shows the Routing Backus-Naur Form (RBNF) [14] format of the CPP template.

Some important CPP clauses are described below. A full description of all CPP clauses can be found in [1].

A CPP must include the list of customer nodes, e.g., CEs, to be connected to the underlying transport network. For each customer node, a border link or node belonging to the connectivity domain and connected to the customer node needs to be identified. Operations appropriate to the location of the customer node should be performed to retrieve the corresponding border link or "provider node" (e.g., PE). A customer node may be connected to several provider nodes, and multiple customer nodes may be connected to the same provider node.

The scope clause specifies the reachability of each of the involved customer nodes. The scope is a unidirectional parameter, and both directions should be described in the CPP. The reachability scope is the set of destination IP prefixes that can



▲Figure 3. Typical connectivity-provisioning interfaces.

```
<CONNECTIVITY_PROVISIONING_DOCUMENT> ::=
                              <Connectivity Provisioning Component> ...
  <Connectivity Provisioning Component> ::=
                              <CONNECTIVITY_PROVISIONING_PROFILE> ...
  <CONNECTIVITY_PROVISIONING_PROFILE> ::=
                              <Customer Nodes Map>
                              <SCOPE>
                              <QoS Guarantees>
                              <Availability>
                              <CAPACITY>
                              <Traffic Isolation>
                              <Conformance Traffic>
                              <Flow Identification>
                              <Overall Traffic Guarantees>
                              <Routing and Forwarding>
                              <Activation Means>
                              <Invocation Means>
                              <Notifications>
  <Customer Nodes Map> ::=  <Customer Node> ...
  <Customer Node> ::=  <IDENTIFIER>
                       <LINK_IDENTIFIER>
                       <LOCALIZATION>
```

▲Figure 4. RBNF format of the connectivity-provisioning document.

be reached from the customer site. Both global and restricted reachability scopes can be captured in the CPP. A restricted reachability scope means that global (i.e., whole Internet) reachability is not allowed, and only a subset of destinations are reachable.

The QoS guarantee clause specifies performance metrics for the quality of IP transfer experienced by a flow crossing an IP transport infrastructure. This flow is issued by or destined for a (set of) customer node(s). IP performance metrics can be expressed as qualitative or quantitative parameters. When quantitative metrics are used, maximum or average numerical values are provided together with a validity interval that should be indicated in the measurement method.

The availability guarantee clause specifies the percentage of time the IP performance guarantee applies. This clause can be expressed as maximum or average. The guarantee covers QoS deterioration, i.e., IP transfer is available but it is below the agreed performance bounds; physical failure; or general service unavailability.

The capacity clause specifies the capacity that needs to be provided by the underlying network infrastructure. This capacity is bound by a given scope and IP transfer performance guarantees. The capacity may be expressed on a border link basis and for both directions, i.e., incoming and outgoing. This clause includes a traffic limit up to which quantitative perfor-

mance is guaranteed.

The traffic-isolation clause specifies whether traffic issued by or destined for customer nodes should be isolated when crossing the network. This clause is translated into IP engineering policies on activating dedicated tunnels that use IPsec, or establishing BGP/MPLS VPN facilities, or using a combination of these. Activated tunnels or facilities should be consistent with those used to provide guaranteed availability and performance.

The flow-identification clause specifies which information is used to identify the flows that need to be processed in the context of a given CPP. This identifier is used for traffic classification. A flow identifier may comprise source IP address, source port number, destination IP address, destination port number, DiffServ Code Point (DSCP) field, tail-end tunnel endpoint, or a combination of these.

4 Dynamic Resource Allocation According to Service Requirements

4.1 Connectivity Provisioning Negotiation Protocol

CPNP is designed to dynamically exchange and negotiate connectivity provisioning parameters between a customer and provider. CPNP is service-agnostic; that is, it can support additional services, such as storage, as well as the base connectivity service. CPNP is extensible because new methods and negotiation options can be defined as required. CPNP introduces automation into the service negotiation and activation procedures and thus benefits the overall service delivery process.

CPNP negotiation cycles can be triggered by connectivity requirements that are exposed by applications or explicitly requested by customers. Resource allocation and TE objectives can be derived from the outcomes of CPNP negotiation cycles. These TE objectives can be tweaked according to customer requests, available network capacity, and business development guidelines. CPNP can accommodate both technical and business-related requirements. It also supports various negotiation modes, including administrative validation operations.

4.2 CPNP Functional Elements

CPNP operations involve two main functional elements: CPNP client and CPNP server.

The CPNP client is a software instance that sends CPNP requests and receives CPNP responses. A CPNP client creates a quotation order, cancels a quotation order being negotiated, withdraws a pre-negotiated quotation order, or updates a pre-negotiated order.

The CPNP server is a software instance that receives CPNP requests and sends back CPNP responses. The CPNP server processes quotation orders, cancels a quotation order being negotiated, and handles a quotation order withdrawal.

Several models can be used to locate the CPNP client and server functional elements. In one model, the customer deploys a CPNP client while the provider deploys one or several CPNP servers. In a second model, the customer does not enable any CPNP client, but the provider maintains a customer order management portal instead. This model assumes that the same administrative entity, i.e., network provider, is responsible for both the CPNP client and server. In such a model, the customer initiates connectivity-provisioning quotation (CPQ) orders via the portal, and appropriate CPNP messages are then generated and sent to the relevant CPNP server. Once connectivity provisioning parameters have been negotiated and an order has been placed by the customer, network provisioning operations are initiated.

4.3 CPNP Customer Order Processing Models

There are three models for customer order processing: frozen, announcement, and negotiation.

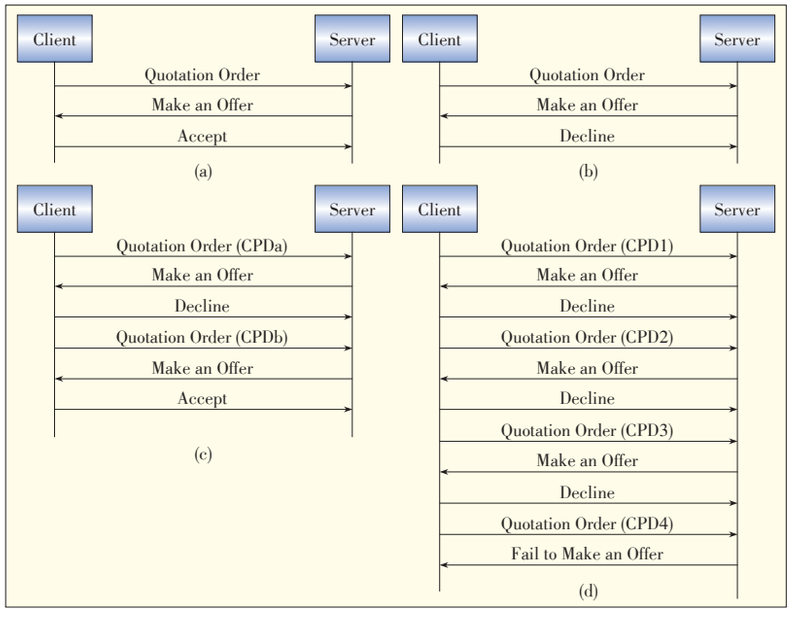
In a frozen model, the customer cannot negotiate the parameters of the connectivity service provided. After consulting a service portfolio, the customer selects the offer they want to subscribe to and places the corresponding order with the provider. On the provider side, handling the order request is quite simple because the service is not customized to the customer's requirements. Rather, it is pre-designed to target a group of customers with similar requirements (and who therefore share the same CPP).

In an announcement model, the provider proceeds to the announcement of a set of service templates. The customer can then initiate a negotiation cycle using these templates and prepare their request order.

In a negotiation model, the customer documents their requirements in a request for quotation, which is sent to one or several providers. These solicited providers then check whether they can meet these requirements and get back to the customer, possibly with an offer that does not exactly satisfy customer's requirements. The customer and provider then negotiate, and the customer places an order.

CPNP uses announcement and negotiation-based models. In particular, it uses a quotation order/offer/answer model where 1) the client specifies their requirements in a provisional quotation order (PQO); 2) the server makes an offer that satisfies or partly satisfies these requirements, or it declines the PQO; and 3) the client either accepts or declines the offer. **Fig. 5** shows a typical CPNP negotiation cycle.

A CPNP transaction, comprising all CPNP messages, occurs between when the first request is sent by the client to the server and when a final response is sent by the server to the client. This final response completes the transaction. The CPNP transaction is bound by a CPNP session. Because multiple CPNP transactions can be maintained by the CPNP client, the client assigns an identifier to each active transaction. This identifier is denoted Transaction-ID and must be randomly assigned according to the best current practice for generating random num-



▲**Figure 5. CPNP negotiation model. (a) 1-step successful negotiation cycle, (b) 1-step failed negotiation cycle, (c) *N*-step successful negotiation cycle, and (d) *N*-step failed negotiation cycle.**

bers. It must not be guessed easily. Transaction-ID is used in the validation of CPNP responses received by the client.

There is only one offer/answer stage in a single CPNP transaction. Nevertheless, multiple CPNP transactions can be handled by the CPNP client.

The CPNP server can be configured in several order-handling modes. In a fully automated mode, no action is required from the administrator when a service is requested. The CPNP server automatically makes decisions about received orders and generates corresponding quotations. In another mode, some or all CPNP server operations are subject to validation by an administrator. This mode requires the administrator to act on some or all requests received by the CPNP server.

The CPNP server may support the option of publishing available services, which are exposed to customers. Dedicated templates can be used for the purpose of announcing services. The CPNP client will use these templates to initiate its CPNP negotiation cycle.

Two key identifiers are used by the CPNP client and server: customer order and provider order. The customer order identifier is assigned by a CPNP client to uniquely identify an order among existing orders. This identifier is included by the CPNP client in all its CPNP messages. The provider order identifier is assigned by the CPNP server to uniquely identify an order among existing orders.

4.4 CPNP Operations

Table 1 lists the operations supported by CPNP, which uses several kinds of connectivity provisioning documents (CPDs) (section 3). A requested CPD is a CPD included by a CPNP client in a PROVISION request. An offered CPD is the document included by a CPNP server in an OFFER message. An offered CPD indicates that the server will accommodate all (or a subset of) the clauses in a requested CPD. The offer also has a validity date. If the CPNP client accepts the offer, the offered CPD is included in an ACCEPT message, and the document is called an Agreed CPD. The Agreed CPD is also included in an ACK message.

4.5 CPNP Client Behavior

To place a CPQ order, the CPNP client initiates a local order object, which has a unique identifier (Customer Order Identifier) that is assigned by the CPNP client. Then, the CPNP client generates a PROVISION request that includes the assigned identifier, possibly an expected response date, Transaction-ID, and Requested CPD. The CPNP client may include additional information elements, such as "cost" or "setup purpose" negotiation options. The setup purpose clause may contain a request for connectivity only for testing purposes and only for a limited period of time. The order can become permanent if the customer is satisfied during the test period.

Once the request has been sent to the CPNP server, the CPNP client sets a timer to the expiration date, which is included in the PROVISION request. If the CPNP server has not answered before the retransmission timer expires, the CPNP client retransmits the request up to three times. If a FAIL message is retuned, the CPNP client may decide to make another request to the same CPNP server, cancel the local order, or contact another CPNP server. If an OFFER message is received, the CPNP client checks whether a PROCESSING message with the same Provider Order Identifier has been received from the CPNP Server. If a PROCESSING message was al-

▼**Table 1. Operations supported by CPNP**

Operation	Used By	Description
PROVISION	Client	Initiates a CPQ order. After receiving a PROVISION request, the server may respond with a PROCESSING, OFFER, or FAIL message.
PROCESSING	Client/ Server	Informs the remote party that the message was received and the order quotation or offer is being processed
OFFER	Server	Informs the client about an offer that best accommodates the requirements in the PROVISION message
ACCEPT	Client	Confirms the acceptance of an offer made by the server
ACK	Server	Acknowledges receipt of an ACCEPT or WITHDRAW message
DECLINE	Client	Rejects an offer made by the server
CANCEL	Client	Cancels an ongoing CPQ order
WITHDRAW	Client	Withdraws a pre-negotiated connectivity provisioning order
UPDATE	Client	Updates an existing connectivity provisioning order
FAIL	Server	Cannot accommodate a requested PQO. This operation can also be used to inform the client about an error encountered when processing the received message. The FAIL message includes a code that gives more information about the error.

ready received for the same order but the Provider Order Identifier does not match the identifier included in the OFFER message, the CPNP client silently ignores the message. If a PROCESSING message with the same Provider Order Identifier was already received and matches the CPNP transaction identifier, the CPNP client changes the state of the order to OfferReceived and sets a timer according to VALIDITY_DATE in the OFFER message.

If an offer is received from the CPNP server, the CPNP client decides whether it will accept or reject the offer. The CPNP client accepts the offer by generating an ACCEPT message, which confirms that the customer has agreed to subscribe to the offer in the OFFER message (**Fig. 6**). The transaction is terminated if an ACK message is received from the server. If no ACK is received from the server, the client proceeds to retransmit the ACCEPT message.

The CPNP client can reject the offer by sending a DECLINE message (**Fig. 7**). If an offer is not acceptable to the CPNP client, it may decide to contact a new server or submit another order to the same server.

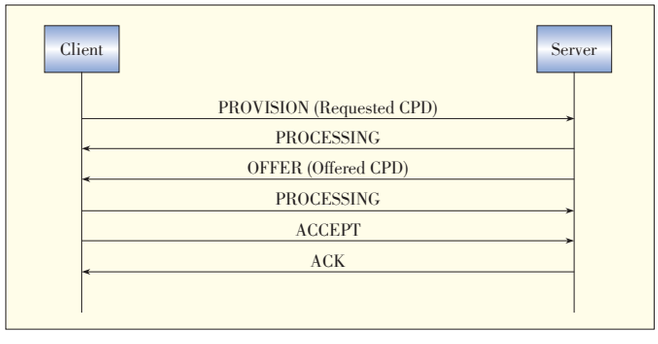
The CPNP client can withdraw a completed order by sending a WITHDRAW message. It can also update a completed order by sending an UPDATE message (**Fig. 8**).

4.6 CPNP Server Behavior

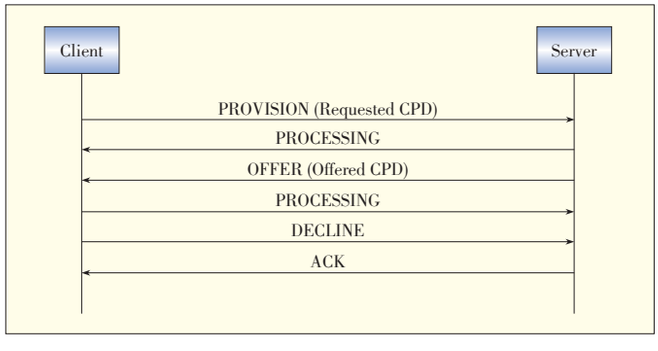
When a PROVISION message is received from a CPNP client, the CPNP server stores the Transaction-ID, generates a Provider Order Identifier, and runs preliminary validation checks. Then, the CPNP server returns a PROCESSING message to notify the client that the quotation order has been received and is being processed. A PROCESSING message can include an expected offer date that tells the CPNP client when an offer will be proposed. The CPNP server runs a decision-making process to decide which offer best suits the received order. The CPNP server makes an offer before the expected offer date.

If the CPNP server can satisfy the request, it sends an OFFER message to the client making the request. This message includes Transaction-ID, Customer Order Identifier (indicated in PROVISION), Provider Order Identifier generated for the order, a Nonce, the offered Connectivity Provisioning document, and an offer validity date (Fig. 6).

If the CPNP server determines that additional resources from another network provider are needed to accommodate a quotation order, it creates a child/children PQO(s) and behaves as a CPNP client in order to negotiate a child/children PQO(s) with the partnering providers. **Fig. 9** shows CPNP messages exchanged in such a case.



▲**Figure 6. A flow of a successful CPNP negotiation cycle.**

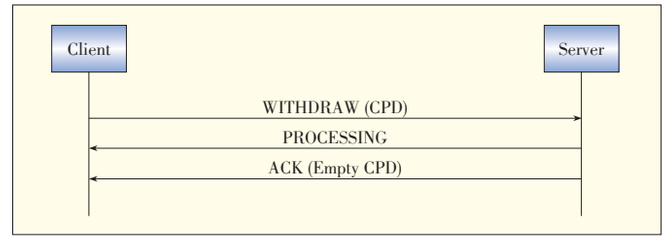


▲**Figure 7. An unsuccessful CPNP negotiation cycle.**

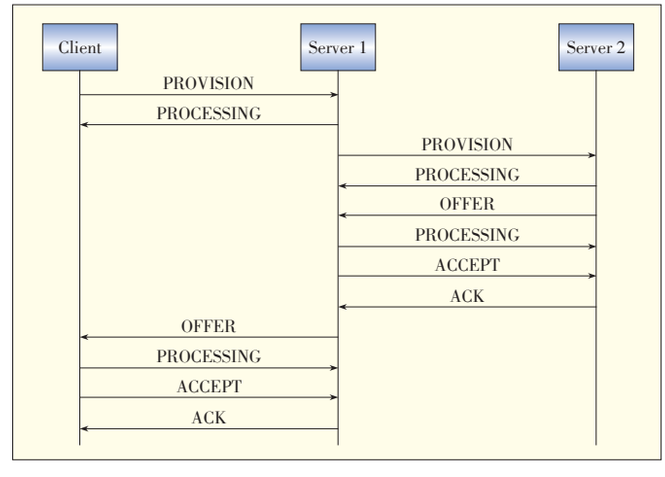
5 Additional Paths to Automated Service Delivery Procedures

5.1 SDN Bootstrapping

The means of dynamically discovering the functional capa-



▲**Figure 8. CPNP withdrawal.**



▲**Figure 9. Inter-provider CPNP negotiation cycle.**

bilities of devices to be programmed by SDN intelligence need to be provided. Acquiring information related to actual network capabilities will help structuring this intelligence so that policy provisioning information can be derived accordingly.

Dynamic discovery may depend on the exchange of specific information via an Interior Gateway Protocol (IGP) or Border Gateway Protocol (BGP) between network devices or between network devices and SDN intelligence in legacy networks. This intelligence can also send unsolicited commands to network devices in order to acquire a description of their capabilities and derive network and service topologies accordingly.

SDN techniques could be used in IGP/BGP-free networking environments; however, in such environments, SDN bootstrapping still requires the following support capabilities:

- dynamic, resilient discovery of participating SDN nodes, including the SDN intelligence, and their respective capabilities. This assumes mutual authentication of the SDN intelligence and participating devices. The integrity of the information exchanged between the SDN intelligence and participating devices during discovery must also be preserved.
- dynamic connection of the SDN intelligence to SDN-capable nodes and avoidance any forwarding loop
- dynamic enabling of network services as a function of device capabilities and (possibly) what has been dynamically negotiated between the customer and network provider
- dynamic checking of connectivity between the SDN intelligence and participating nodes and between participating nodes themselves so that a given network service or set of services can be delivered.
- dynamic assess to the reachability scope as a function of the service to be delivered.
- dynamic detection and diagnosis of failures so that corrective measures can be taken accordingly.

Likewise, the means of dynamically acquiring descriptive information (including base configuration) of any network device that may participate in the delivery of a service should be provided. This helps the SDN intelligence structure the services that can be delivered in light of various factors, such as available resources and resource location.

In networking environments without IGP/BGP, a specific bootstrap protocol may be required to support the previously mentioned capabilities and proper SDN operation. There may also be a need for a specific additional network with discovery and connectivity features.

In particular, SDN design and operation in an IGP/BGP-free environment should provide performances similar to that of legacy environments that run an IGP and BGP. For example, the underlying network should remain operational even if connection with the SDN intelligence is lost.

Furthermore, operators should assess the cost of introducing a new, specific bootstrap protocol compared to the cost of integrating the previously mentioned capabilities into existing IGP/BGP machinery.

Because SDN-related features can be grafted into an existing network infrastructure, they may not all be enabled at once from a bootstrapping perspective, so a gradual approach can be taken instead. A typical deployment example is using an SDN decision-making process as an emulation platform that helps network providers and operators make appropriate technical decisions before their actual deployment in the network.

5.2 Dynamic Service Function Chaining

The current model used by network providers to offer value-added services has reached its limits. This model relies on the invocation of advanced service functions in addition to basic forwarding and routing. Typical examples of such service functions include: NAT, NPTv6, SSL Offload, HOST_ID injection [15], HTTP header enrichment, cache content, and deep packet inspection. Managing and introducing these advanced service functions is hindered by the underlying physical topologies. Furthermore, the model used to deploy these service functions in has a cascaded scheme. This is not optimal in terms of capex and performance. A technique called SFC [3] is a suitable means of invocating a set of service functions in an order. In reference to the SDN framework in section 2, the SDN intelligence can embed the SFC structuring and orchestration functionality.

6 Conclusion

Beyond the current hype surrounding SDN there is a complex combination of multi-metric, service-dependent, computation algorithms, negotiation protocols, and service-modeling languages that presents significant challenges for network providers.

From service parameter exposure to delivery, it is true that automation; flexibility; and adaptive, self-tuning network infrastructures have increased complexity and sometimes led to performance degradation. Such impacts should be quantitatively and qualitatively assessed by network providers.

A dynamic service-parameter negotiation approach is still in its infancy, and various service-specific simulations and validation studies will need to be conducted to refine critical dimensioning figures, e.g., in terms of the amount of traffic exchanged between a customer and provider during service-parameter negotiation.

Furthermore, the robustness of SDN techniques should be analyzed and characterized. The amount of traffic to be exchanged between the SDN intelligence and participating nodes and dynamic policy-enforcement schemes can lead to mad robot situations similar to that recently experienced by Google.

Predictable behavior is key to efficient service-parameter exposure, negotiation, and subsequent translation into technology-specific provisioning information. This requires strictly deterministic approaches with carefully designed information and data models, test cases, and control operations as cornerstones.

To this end, network providers must play a key role in standardizing such tools for the sake of global consistency.

References

[1] M. Boucadair, C. Jacquenet, and N. Wang, "IP/MPLS connectivity provisioning profile," IETF, draft-boucadair-connectivity-provisioning-profile-05, Apr. 2014.
[2] M. Boucadair and C. Jacquenet, "Connectivity provisioning negotiation protocol (CPNP)," IETF, draft-boucadair-connectivity-provisioning-protocol-01, Oct. 2013.
[3] M. Boucadair, C. Jacquenet, R. Parker, D. Lopez, J. Guichard, and C. Pignataro, "Service function chaining: framework & architecture," IETF, draft-boucadair-sfc-framework-02, Feb. 2014.
[4] TM Forum. (2014). *IPsphere* [Online]. Available: http://www.tmforum.org/InDepth/6918/home.html
[5] O. Younis, S. Fahmy, "Constraint-based routing in the internet: basic principles and recent research," *IEEE Communication Surveys & Tutorials*, vol. 5, no. 1, pp. 2–13, Dec. 2009. doi: 10.1109/COMST.2003.5342226.
[6] *OpenFlow Management Configuration Protocol*, ONF OF-CONFIG 1.1.1, Jan. 2013.
[7] K. Kumaki, T. Murai, and P. Jiang, "PCEP extensions for a BGP/MPLS IP-VPN," IETF, draft-kumaki-murai-pce-pcep-extension-l3vpn-12.txt, Oct. 2013.
[8] *Network Configuration Protocol (NETCONF)*, IETF RFC 6241, Jun. 2011.
[9] *COPS Usage for Policy Provisioning (COPS-PR)*, IETF RFC 3084, Mar. 2001.
[10] TCG. (2012, May). *Trusted Network Connect IF-MAP (InterFace to Metadata Access Points) 2.1 Specification* [Online]. Available: http://www.trustedcomputinggroup.org/files/static_page_files/93869B41-1A4B-B294-D0C211E85C7CF901/TNC_IFMAP_v2_1r20.pdf
[11] R. Penno, T. Reddy, M. Boucadair, D. Wing, and S. Vinapamula, "Application enabled SDN (A-SDN)," IETF, draft-penno-pcp-asdn-00, Sept. 2013.
[12] MESCAL. (2005). *Path Computation System* [Online]. Available: http://www.ist-mescal.org/roadmap/pcs.html
[13] *Software-Defined Networking: A Perspective From Within A Service Provider*, IETF RFC 7149, Mar. 2014.
[14] *Routing Backus-Naur Form (RBNF): A Syntax Used to Form Encoding Rules in Various Routing Protocol Specifications*, IETF RFC 5511, Apr. 2009.
[15] *Analysis of Potential Solutions for Revealing a Host Identifier (HOST_ID) in Shared Address Deployments*, IETF RFC 6967, June 2013.

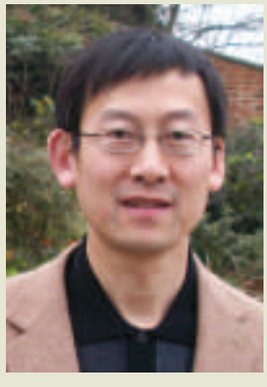
Manuscript received: 2014-02-18

Biographies

Mohamed Boucadair (mohamed.boucadair@orange.com) is a senior IP architect at France Telecom. He has worked for France Telecom R&D and has been part of the team working on VoIP services. He is now working at France Telecom corporate division and is responsible for making recommendations on the evolution of IP/MPLS core networks. He has been involved in IST research projects, working on dynamic provisioning and inter-domain traffic engineering. He has published many journal articles and written extensively on these subject areas. Mr. Boucadair holds several patents on VoIP, IPv4 service continuity, IPv6, etc.

Christian Jacquenet (christian.jacquenet@orange.com) graduated from the Ecole Nationale Supérieure de Physique de Marseille, a French school of engineers. He joined Orange in 1989 and is currently the director of the Strategic Program Office for Advanced IP Networking. In particular, he is responsible of the Groupwise IPv6 Program that aims to define and drive enforcement of the Group's IPv6 strategy. He has authored and co-authored several Internet drafts and RFCs in the field of dynamic routing protocols and provisioning techniques. He has also authored and co-authored multiple papers and books on IP multicast, traffic engineering and automated IP service provisioning techniques.

New Member of *ZTE Communications* Editorial Board



Kun Yang received his PhD from the Department of Electronic & Electrical Engineering of University College London (UCL), UK. His MSc and BSc were in Computer Networks and Computer Science respectively, both from Jilin University, China. He is currently a Chair Professor in the School of Computer Science & Electronic Engineering, University of Essex, UK, and the Head of the Network Convergence Laboratory (NCL) in Essex. Before joining Essex at 2003 he worked at UCL as a Research Fellow on several EU projects such as FAIN, etc. His main research interests include wireless networks/communications, fixed mobile convergence, future Internet technology (such as network virtualization) and cloud computing/networking. He has published 150+ papers in the above research areas. He serves on the editorial boards of both IEEE and non-IEEE journals (such as Wiley and Springer) and (co-)chairs of IEEE conferences. He has been actively involved in research projects funded by European Union (e.g., PURSUIT), UK EPSRC (e.g., PANDA), UK TSB (e.g., PAL) and industries (e.g., British Telecom). He is the Coordinator of EU FP7 Project EVANS and technical leader of several other EU FP7 projects. He is a UK EPSRC Peer Review College Member and is a research proposal review expert on research funding bodies from France, Norway, Canada, Singapore, Hong Kong, China, etc. He is one of the six Executive Committee Members of IEEE InterCloud Initiative. He is a Senior Member of IEEE and a Fellow of IET.

DOI: 10.3969/j. issn. 1673-5188. 2014. 02. 003
http://www.cnki.net/kcms/detail/34.1294.TN.20140530.0848.001.html, published online May 30, 2014

SDN-Based Broadband Network for Cloud Services

Xiongyan Tang, Pei Zhang, and Chang Cao
(China Unicom Network Research Institute, Beijing 100048, China)

Abstract

Over-the-top services and cloud services have created great challenges for telecom operators. To better meet the requirements of cloud services, we propose a decoupled network architecture. Software-defined networking/network function virtualization (SDN/NFV) will be vital in the construction of cloud-oriented broadband infrastructure, especially within data centers and for interconnection between data centers. We also propose introducing SDN/NFV in the broadband access network in order to realize a virtualized residential gateway (VRG). We discuss the deployment modes of VRG.

Keywords

SDN; NFV; Cloud Services; Broadband Network

1 Introduction

The rapid growth of internet-based IT services has had an unprecedented influence on traditional telecom business models. Over the top (OTT) businesses are replacing traditional telecom businesses. OTT services erode the profitability of an operator's traditional voice and SMS services and force the operator to fall back on more fundamental network businesses for revenue growth. From January to September 2013, the revenue of China's big three operators grew by only 0.7% year-on-year whereas income from non-voice business, dominated by mobile traffic and fixed broadband access, rose 16.6% year-on-year. These three operators were responsible for 95.7% of revenue growth in China's domestic telecom industry [1]. The number of point-to-point SMS sent by mobile users declined sharply in 2013, down by 13.7% year-on-year.

From the perspective of the ICT industry, the industry value chain has significantly changed. The basic network, i.e., the pipe, it the operator's core and lifeline, but its value is constantly declining. Concurrently, the value of terminals and the cloud is rising fast. Industry profit continues to be captured by IT enterprises such as Apple, Google, Facebook, and Amazon as well as China's Tencent, Baidu, and Alibaba.

If operator revenue increased in line with traffic growth, then basic network operation would still be a good business. However, nowadays there are also huge challenges and costs associated with providing basic network service. The end result is that more traffic does not generate more revenue. On the one hand, users and applications are demanding more in terms of network bandwidth, and network resources are being rapidly consumed. On the other hand, network unit bandwidth is getting cheaper, which means that network traffic is being decoupled from revenue. Telecom operators have two paths to sustainable development: increase income or reduce costs. With the former, a plateau in user numbers and the difficulty of innovating in the application business mean that operators have to rely more on innovation in areas such as traffic operation, open networks, and collaborative cloud-network terminals in order to increase network value. With the latter, the key is to continuously reduce equipment, construction, and OAM costs as well as improve resource utilization by innovating with technology and optimizing architecture.

OTT services have created great challenges for telecom operators, and both service model and network architecture need to be transformed in order to overcome these challenges. Here, we propose a decoupled network for cloud services, which greatly depend on a broadband network based on software-defined networking (SDN) in order to provide flexible, dynamic connection for customers and cloud data centers.

The rest of the paper is organized as follows: In section 2, we explain the decoupled network architecture that supports current cloud services; in section 3, we describe the role of SDN in a cloud-oriented broadband network as well as some typical SDN applications; in section 4, we discuss SDN-based broadband access network; and in section 5, we conclude the paper.

2 Decoupled Network Architecture for Cloud Services

A network supports and serves various services and applica-

tions and evolves constantly in response to service demand. There are two main trends in ICT business development: user mobility and service end on cloud. In terms of user mobility, smart terminals have become the main tool that people use to access ICT services and are driving the development of mobile Internet and the Internet of Things (IOT). In terms of service end on cloud, ICT services are fully embracing cloud service modes and are the impetus behind changes in network traffic models and system architecture. Although cloud computing only appeared at the end of 2007, its philosophy and modes have quickly penetrated all aspects of ICT service.

Cloud service is a new technological concept and also a kind of new business thinking. Cloud services create new opportunities for telecom operators because they basically exchange computing and storage resources for network resources. Cloud services require data infrastructure such as cloud data centers as well as highly reliable, flexible, smart, ubiquitous broadband networks. Operators should play a leading role in providing cloud service infrastructure, such as data centers and broadband networks, in order to open up new space for growth. According to Gartner Research, in 2013, global cloud service markets were worth $131.7 billion and grew at an annual rate of 18%. This figure is estimated to rise to $244.2 billion in 2017, with growth remaining at 15% or higher.

In the telephony era, network traffic was primarily person-to-person traffic. In the cloud service era, network traffic is primarily communication between smart terminals and the cloud and communication between clouds. From 2000 to 2008, end-to-end file sharing was the main source of internet traffic; however, since 2008, internet traffic has primarily been generated or terminated by data centers. Globally, cloud computing traffic is expected to increase 12-fold between 2010 and 2015, with average annual compounding growth of 66% [2]. Global cloud data traffic first reached the zettabyte level in 2012. In 2015, one-third of data center traffic will be cloud traffic, and in 2016, two-thirds of data center traffic will be cloud traffic [2].

Cloud services rely heavily on Internet data centers (IDCs), which are similar to telephone switches in the telephony age. Future cloud data centers will be centers of data infrastructure, such as server and memory, and also basic network centers. Traditional data centers are mostly located in big cities where there is a concentration of users and good network conditions. However, when selecting the location of new cloud data centers, factors such as land, energy consumption, and climate will be major considerations. The underdeveloped regions of north and west China will become home to future cloud data centers. That is to say, the focus when choosing IDC locations has moved from towards energy efficiency. This will lead to a decoupling of the user center, which encompasses information generators and users, from the data center, which encompasses information storage and processors. In the past, data and network followed users, and the three were tightly bound. In the traditional telephony era, networks mainly served as vehicles for communication between people. In the cloud service era, user centers and data centers are separate, and a "double center" pattern for users and data is formed. Thus, networks serve more as vehicles for communication between users and data applications and are used for delivery of data itself. In the cloud era, basic network services will need to take into account networking between virtual machines (VMs); IDC internal networking, including front-end service networking and back-end storage networking; inter-networking between multiple IDCs, e.g., super IDC networking and edge IDC networking; and connection between users and IDCs. Flexible, dynamic, open networks and quick access to resources are particularly important for cloud services. To better support cloud service development, there must be a transformation from cloud-follows-network to network-follows-cloud.

Newly decoupled network architecture contains the data center domain and data user domain. **Fig. 1** shows the connections within and between these domains.

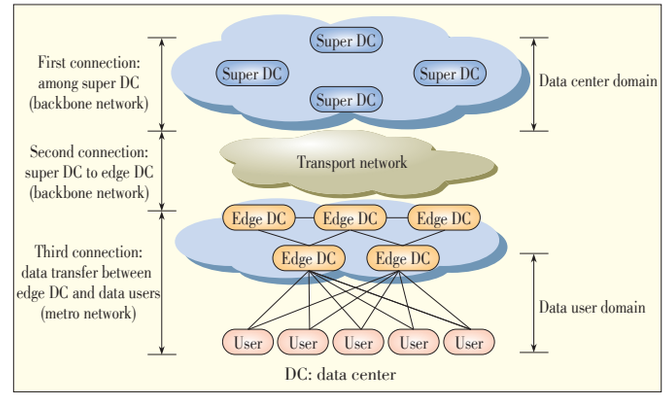
Here, we define the data center domain and data user domain and related connections.

The data center domain includes super data centers and their connections. The main service in the data center domain is data resource transfer and scheduling between servers and their VMs. The data center domain usually crosses the backbone network. The direction of the traffic flow is fixed and centralized, and the volume of traffic can be predicted. The connection between data centers can be full-mesh, ring, or star.

The user domain contains a huge number of end users. The main service in the data user domain is web-browsing, on-demand gaming and video, instant messages, and various application services. These services have high requirement in terms of user experience, i.e., in terms of latency, jitter, and response/backup time.

The first type of connection is between super data centers (super DCs), which are connected to each other via full-mesh, ring, or star topology and high-speed fiber channels.

The second type of connection uses the operator's broadband network to deliver data or content from super DCs to smaller data centers at the edge side of a metro network (edge



▲**Figure 1. Decoupled network architecture.**

DC). The edge DCs usually store part of the content that local users have often accessed.

The third type of connection facilitates data transfer between edge DCs and end users. The traffic in this domain is random, dispersive, and bursty. All data users first visit the nearest edge DC in order to access commonly used content. If this data is not in edge DCs, one edge DC requests data needed by users via the second type of connection.

3 SDN in a Cloud-Oriented Broadband Network

The proposed decoupled architecture lays the foundation for constructing next-generation broadband infrastructure. In cloud services, frequent migration of virtual resources depends on flexible broadband network support. SDN is one of the hottest topics in ICT in recent years [3]–[5]. It is also an important technical concept for building flexible broadband network systems. The concept of SDN originated in the OpenFlow project conducted by Stanford University. The initial motivation for SDN was to break the monopoly of integrated network hardware and software and to enable network equipment to follow computer open industry chain by separating hardware from software [3]. SDN is a new type of network architecture that enables programmable network control by separating the network control plane from the transfer plane and virtualizing the bottom network. SDN, as defined by the Open Network Foundation (ONF), decouples network control and forwarding so that network control becomes programmable, and the underlying infrastructure is abstracted for applications and network services [6]. Narrowly defined, SDN refers to SDN based on OpenFlow standard protocols released by the ONF. More generally defined, SDN refers to various open-interface, software-programmable network architectures, including related standards and technology systems proposed by the IETF, ETSI and other standardization organizations. Network function virtualization (NFV) proposed by ETSI is another important concept supported by many telecom operators. NFV uses software to implement network functions. It can run on industry-standard server hardware and can be moved to or instantiated in various locations in the network without the need to install new equipment. NFV is designed to break up the current network infrastructure model, where building blocks are black boxes vertically integrated by each vendor. NFV complements SDN, and the two concepts and technologies can be combined. In short, SDN/NFV has broken the closed, rigid network system originally formed by proprietary network elements and reduces the cost of network equipment. It has also simplified network OAM and made network services more flexible.

SDN/NFV technology can be used in all layers of the broadband network, including for routing switch, transport, access, and home network [7], [8], and is vital for constructing a cloud-oriented next-generation broadband network. SDN will soon be used for data center internal networks, data center interconnection, virtual residential gateways, IP intelligent edges, mobile backhaul, mobile core network, and more. SDN has broad prospects and has given a profound influence on network development.

At the current stage of technological development, the main application of SDN is in data center networks. Cloud services have imposed higher requirements in terms of the flexibility, automation, and scalability of a data center network. Because traditional data centers have a large number of internal switches, network deployment strategies are complex, cross-domain migration of virtual resources is difficult, and security is difficult to guarantee. SDN meets satisfies the requirements of data center networks. It can also be conveniently deployed in data centers, and the internal network environment is relatively independent. SDN switches in a data center network are good for rapid, synchronized migration of virtual resources and are a good network strategy. SDN switches enable closer collaboration between the network and computing and storage resources, and it facilitates greater control of overall resources. SDN may be further used for wide-area data center interconnection. Google has successfully used SDN in this way and has set the standard for this within the industry. Using SDN to interconnect data centers significantly improves bandwidth utilization, improves link availability, improves network scalability, and lowers network costs. It also simplifies OAM and makes cloud services smoother and more efficient.

More and more internet enterprises and OTT/cloud service providers are using SDN/NFV and other emerging network technologies to build their cloud service infrastructures. The use of SDN/NFV and other technologies enables closer collaboration between networks and cloud services. If telecom operators fail to provide flexible network services that are adaptable to cloud services, providers of these services will rely more on their own network facilities, and this may result in a decrease of data traffic on the operator's network. Operators should be aware that OTT is significantly affecting their application services. The threat of OTTs to an operator's basic network services should not be underestimated. For example, Google currently owns huge data centers in many countries. It has built its own data center network and has also cooperated with operators to jointly invest in the rollout of submarine fiber cables in the Pacific Ocean. In addition, Google has entered the US broadband service market and launched 1 Gbps fiber access, which has a very high price-performance ratio in numerous cities in the United States. Apple has also begun to deploy cloud broadband infrastructure. With the expectation of catching up with Google, Apple is preparing network infrastructure to handle more cloud services and distribute more digital content. In China, major internet companies are also arranging and building cloud service infrastructures, including cloud data centers, content delivery networks (CDN), and related networks. In the current environment, telecom operators have to be open and in-

novative, hasten network transformation through the introduction of new technologies, provide better network services to meet OTT cloud service requirements, and come to a win-win arrangement with OTT providers.

4 SDN in the Broadband Access Network

Broadband access networks are the most important part of a broadband infrastructure. They determine customer experience over last one mile and are also critical for the end-to-end QoE of cloud services. Residential gateways in broadband access are a potential area where SDN/NFV could be applied. There are a number of issues with current fixed-access networks, including high capes, difficulty in introducing new services, and complex OAM. By introducing SDN technology, forwarding and controlling planes can be separated in the access network so that access equipment and services are decoupled. SDN architecture can help a telecom operator build a simple, swift, flexible, value-added access network. In future architecture, the network access point can be simplified as a programmable device. A unified access network control and management platform can be used to realize a simple access point that does not require configuration, that has no faults, and that is not costly in terms of OAM. Clouding service and residential gateways enables flexible service deployment and network evolution. In a word, SDN represents a great step forward for access networks, especially residential networks.

4.1 Residential Gateway Virtualization

Residential gateways are broadband access network interfaces provided by operators. Residential gateways are also the core communication equipment between an internal home network and external public network. A residential gateway is a data processing center inside the home and connects to the external operator networks, where there are devices for broadband access and VoIP services. Operators use these devices to administrate and maintain their residential networks.

In current multiservice network architectures, the user device has complex functions. The access network and metro network are usually designed as a layer-2-based Ethernet transparent network. The benefit of this architecture is that it is highly scalable and low-cost. Functions of the upper two layers, e.g., IP protocol and application/service processing, have to be enabled in residential gateway devices. In such architecture, flexible adjustment and evolution of IP layer functions is sacrificed. Many layer-3 functions and service functions have to be deployed at the residential gateway, which is tightly coupled and restricted by the gateway.

With SDN, most layer-3 network functions are moved from the home network to provider network and hosted in a pool of resources. This is the function of the virtualized residential gateway (VRG). In the provider network, the VRG realizes third (higher) layer functionality that is usually tightly coupled with the physical residential gateway. At the same time, the residential gateway can be simplified to a bridge device with only layer 1 and layer2 functionality (**Fig. 2**).

A simplified residential gateway makes installation, alteration, troubleshooting and replacement of residential gateway devices easier and more cost-effective.

The VRG has had a profound influence on access and home networks. The main advantages of VRG are standardized hardware, differentiated services, automated network, simplified terminal maintenance, quick service deployment, network resource savings, operator control of the home network, and innovation with terminals and services.

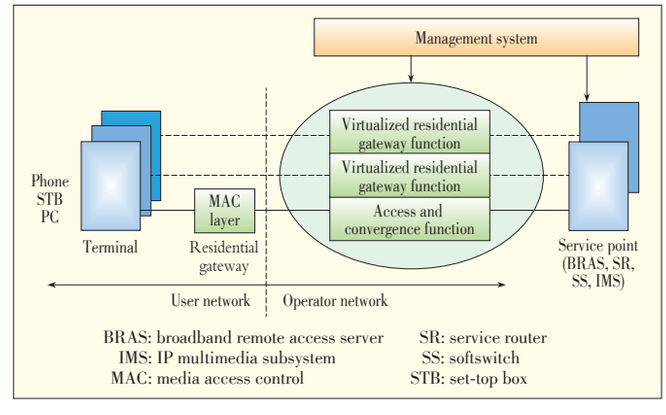
4.2 Realizing the Virtualized Residential Gateway

4.2.1 VRG at the Access Point

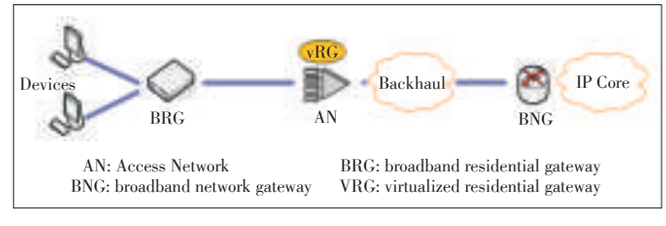
VRG is deployed at the optical line terminal (OLT) with a passive optical network (PON) upstream residential gateway (**Fig. 3**). The first scheme involves adding a VRG service-processing card on the OLT. The second scheme involves realizing VRG functions on the main control panel of the OLT.

4.2.2 VRG Deployed at the Broadband Network Gateway

Fig. 4 shows VRG deployed at broadband network gateway (BNG). BNG nodes are less than access network points, and the broadband remote access server (BRAS) supports the main functions of VRG, e.g., network address transition (NAT) forwarding. Therefore, we only need to estimate the impact on BRAS after VRG functions are added. This scheme also simplifies service procedures; for example, it cancels point-to-point



▲**Figure 2. Network functions in a virtualized residential gateway.**



▲**Figure 3. VRG deployed at an access point.**

protocol over Ethernet (PPPoE) function and procedure. Because VRG functions are maintained by a data network maintenance team rather than an access network maintenance team, this scheme affects the maintenance system.

4.2.3 VRG Independent Deployment

Fig. 5 shows VRG deployed in a metro network. The deployment location can be flexibly chosen according to requirements in terms of VRG processing.

This scheme does not require other equipment to be significantly altered and supports smooth migration according to service requirements. However, some functions, such as NAT, may be deployed redundantly, and service processing should be redefined.

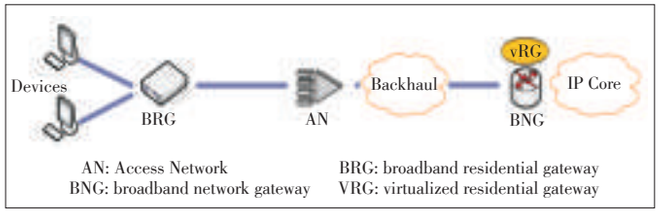
4.2.4 VRG Distributed Deployment

Because some network nodes already have some VRG functions, it is better to deploy different functions, e.g., voice over IP (VoIP), NAT, application layer gateway service (ALG), dynamic host configuration protocol (DHCP), IP over Ethernet (IPoE) and PPPoE, on different equipment. This scheme involves more network devices and more complex service processing, but risk of upgrading legacy devices can be reduced.

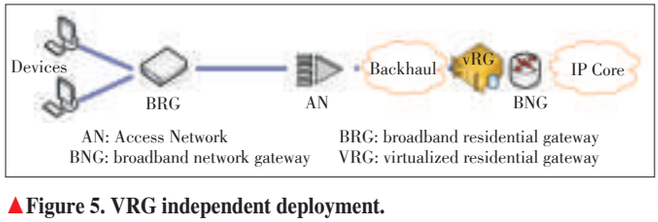
VRG has many advantages over traditional residential gateways; however, several QoS and security issues require attention. On the one hand, some network protocols only run inside the home network, and some services are highly sensitive to timing and need special a QoS guarantee mechanism. On the other hand, some local, private information may be exposed to public network sites, and security in relation to virtualization needs to be taken into account.

5 Conclusion

Cloud service is the future direction of ICT services. Broadband networks have to be transformed in order to better meet the requirements of cloud services. In this paper, we have proposed decoupled network architecture for cloud services. SDN/NFV is vital in the construction of a cloud-oriented broadband network and can be applied in all layers of broadband network for routing switch, transport, access, home network, and more. Currently, SDN is mainly used within data center networks and for interconnection between data center networks. SDN can be introduced into a broadband access network to realize a virtualized residential gateway, and this enables a more flexible, cost-effective broadband infrastructure.



▲Figure 4. VRG deployed at BNG point.



▲Figure 5. VRG independent deployment.

References

[1] Ministry of Indsutry and Information Technology, "Telecommunication economy operation data in Sept 2013," Oct. 2013.
[2] Cisco, "Global cloud index: forecast and methodology, 2012– 2017," Cisco White Paper, 2013.
[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Apr. 2008. doi: 10.1145/1355734.1355746.
[4] G. Goth, "Software - defined networking could shake up more than packets," *IEEE Internet Computing*, vol. 15, no. 4, pp. 6–9, Jul.–Aug. 2011. doi: 10.1109/MIC.2011.96.
[5] S. Shenker, "The future of networking, and the past of protocols," Open Networking Summit, Oct. 18, 2011.
[6] ONF, "Software - defined networking: the new norm for networks," ONF White Paper, Apr. 2012.
[7] D. Verchere, "Cloud computing over telecom network," in *Proc. OFC/NFOEC*, Los Angeles, USA, Mar. 2011, pp. 1–3.
[8] A. Isogai, A. Fukuda, A. Masuda, and A. Hiramatsu, "Global - scale experiment on multi - domain software defined transport network," *10th Int' l. Conf. Optical Internet*, Yokohama, Japan, May, 2012, pp. 8–9.

Manuscript received: 2014-03-24

Biographies

Xiongyan Tang (tangxy@chinaunicom.cn) is now the chief engineer at China Unicom Network Technology Research Institute. He is also the vice-chairman of China Communication Standardization Association TC10. He received his PhD degree in telecom engineering from Beijing University of Posts and Telecommunications in 1994. From 1994 to 1997, he researched high-speed optical communications in Singapore and Germany. Since 1998, he has been working on technology management in telecom operators in China. His research interests include broadband communications, optical fiber networks, next generation networks, and Internet of things.

Pei Zhang (Zhangp7@chinaunicom.cn) received his PhD degree from the Next-Generation Optical Network Laboratory, Beijing University of Posts and Telecommunication, in 2008. He has been researching next-generation high-speed optical transmission, optical access system technology, assessment testing, standard tracking, and other related areas for many years. Over the past few years, he has participated in research on PON, OTN, and packet transport technology with China Unicom. He has published more than 20 papers, applied for seven patents of invention, submitted more than 30 standards documents at ITU - T/FSAN, and written three academic books.

Chang Cao (ccao_bupt@126.com) received his PhD degree from Beijing University of Posts and Telecommunications in 2012. From 2010 to 2011, he was a visiting scholar in the Department of Computer Science, North Carolina State University, US. His main research interests include optical network design and high - speed transmission system evaluation. He has published more than 20 papers and holds five patents.

DOI: 10.3939/j.issn.1673–5188.2014.02.004
http://www.cnki.net/kcms/detail/34.1294.TN.20140521.1114.002.html, published online May 21, 2014

D-ZENIC: A Scalable Distributed SDN Controller Architecture

Yongsheng Hu, Tian Tian, and Jun Wang
(Central R&D Institute of ZTE Corporation, Nanjing 210012, China)

Abstract

In a software-defined network, a powerful central controller provides a flexible platform for defining network traffic through the use of software. When SDN is used in a large-scale network, the logical central controller comprises multiple physical servers, and multiple controllers must act as one to provide transparent control logic to network applications and devices. The challenge is to minimize the cost of network state distribution. To this end, we propose Distributed ZTE Elastic Network Intelligent Controller (D-ZENIC), a network-control platform that supports distributed deployment and linear scale-out. A dedicated component in the D-ZENIC controller provides a global view of the network topology as well as the distribution of host information. The evaluation shows that balance complexity with scalability, the network state distribution needs to be strictly classified.

Keywords

software defined network; OpenFlow; distributed system; scalability; ZENIC

1 Introduction

OpenFlow [1] was first proposed at Stanford University in 2008 and has an architecture in which the data plane and control plane are separate. The external control-plane entity uses OpenFlow Protocol to manage forwarding devices so that all sorts of forwarding logic can be realized. Devices perform controlled forwarding according to the flow tables issued by the OpenFlow controller. The centralized control plane provides a software platform used to define flexible network applications, and in the data plane, functions are kept as simple as possible.

This kind of network platform is constructed on top of a general operating system and physical server. General software programming tools or scripting languages such as Python can be used to develop applications, so new network protocols are well supported, and the amount of time needed to deploy new technologies is reduced. The great interest in this concept was the impetus behind the founding of the Open Networking Foundation (ONF) [2], which promotes OpenFlow.

The main elements in a basic OpenFlow network are the network controller and switch. Generally speaking, the network has a single centralized controller that controls all OpenFlow switches and establishes every flow in the network domain. However, as the network grows rapidly, a single centralized OpenFlow controller becomes a bottleneck that may increase the flow setup time for switches that are further away from the controller. Also, throughput of the controller may be restricted, which affects the ability of the controller to handle data-path requests, and control of end-to-end path capacity may be weakened. Improving the performance of an OpenFlow controller is particularly important in a large-scale data center networks in order to keep up with rising demand. To this end, we propose Distributed ZTE Elastic Network Intelligent Controller (D-ZENIC), a distributed control platform that provides a consistent view of network state and that has friendly programmable interfaces for deployed applications.

The rest of this paper is organized as follows: In section 2, we survey distributed controller solutions; in section 3, we discuss the design and implementation of ZENIC and D-ZENIC; and in section 4, we evaluate the scalability of D-ZENIC.

2 Related Work

Ethane [3] and OpenFlow [1] provide a fully fledged, programmable platform for Internet network innovation. In SDN, the logically centralized controller simplifies modification of network control logic and enables the data and control planes to evolve and scale independently. However, in the data center scenario in [4], control plane scalability was a problem.

Ethane and HyperFlow [5] are based on a fully replicated model and are designed for network scalability. HyperFlow uses WheelFS [6] to synchronize the network-wide state across distributed controller nodes. This ensures that the processing of particular flow request can be localized to an individual controller node.

The earliest public SDN controller to use a distributed hash table (DHT) algorithm was ONIX [7]. As with D-ZENIC, ONIX is a commercial SDN implementation that provides flexible distribution primitives based on Zookeeper DHT storage [8]. This enables application designers to implement control applications without reinventing distribution mechanisms.

OpenDayLight [9] is a Java-based open-source controller with Infinispan [10] as its underlying distributed database system. OpenDaylight enables controllers to be distributed.

In all these solutions, each controller node has a global view of the network state and makes independent decisions about flow requests. This is achieved by synchronizing event messages or by sharing storage. Events that impact the state of the controller system are often synchronized. Such events include the entry or exit of switches or hosts and the updating or altering of the link state. However, in a practical large-scale network, e.g, an Internet datacenter comprising one hundred thousand physical servers, each of which is running 20 VMs, about 700 controllers are needed [4]. The maximum number of network events that need to be synchronized every second across multiple OpenFlow controllers might reach tens of thousands. The demand of which cannot be satisfied by the current global event synchronization mechanisms.

3 Architecture and Design

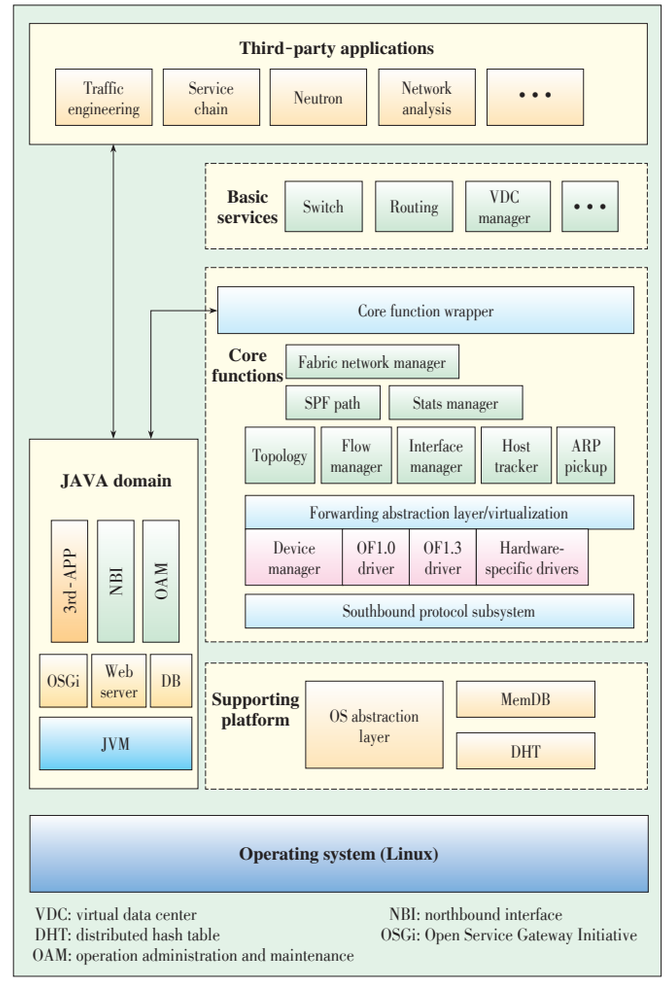
3.1 ZENIC

ZENIC [11] is a logically centralized SDN controller system that mediates between the SDN forwarding plane and SDN-enabled applications. The core of ZENIC encapsulates the complex atomic operations to simplify scheduling and resource allocation. This allows networks to run with complex, precise policies; with greater network resource utilization; and with guaranteed QoS.

ZENIC Release 1 with core and applications developed by ZTE mainly supports the multi-version OpenFlow protocol. In ZENIC Release 2, distributed architecture is introduced into D-ZENIC; which provides a general northbound interface that supports third-party applications.

ZENIC comprises protocol stack layer, drive layer, forwarding abstraction layer, controller core layer, and application layer, which includes internal and external applications (**Fig. 1**). The controller core and self-developed applications are implemented in the C/C++ domain (Fig. 1, right). Maintenance of operations and compatibility between the northbound interface and third-party controller programming interface are implemented in the JAVA domain (Fig. 1, left). This facilitates the migration of applications from other existing controllers.

The forwarding abstraction layer (FAL) defines a unified forwarding control interface in terms of reading and operating state, capacity, hardware resources, forwarding tables, and statistics of the FAL. This layer also manages the derive instances



▲**Figure 1. ZENIC architecture.**

of FAL devices and loads different deriver instances according to a description of the devices. It also makes it convenient to extend southbound protocols, such as NETCONF, SNMP, and IR2S.

Network virtualization support is a built-in feature of ZENIC that supports network partitioning based on the MAC address, port, IP address, or a combination of these. ZENIC Core adopts a 32-bit virtual network identify, with which maximum 2^{32} virtual networks may be supported theoretically. All packets and network states belonging to a specified virtual network are identified, labeled, and sent to an assigned controller.

Each virtual network is isolated from other virtual networks by default, and bandwidth may be shared or reserved. For communication between virtual networks, interworking rules should be configured by the network administrator through the standard northbound interfaces. To simplify configuration, ZENIC provides a default virtual network that enables intra- and inter-virtual network communication.

The controller core functions are responsible for managing network and system resources. This includes topology manage-

ment, host management, interface resource management, flow table management, and management of the network information created by the physical or virtual topology and flow tables. The core functions include not only maintaining the state of network nodes and topology but acquiring the location and state of hosts. In this way, a complete network view can be provided so that further decisions on forwarding and services can be made. Fabric network management is a core function that decouples the access network from the interconnected networks. The core functions include managing Internet packet formats, calculating the complete end-to-end path, and mapping the forwarding policies to corresponding Internet encapsulation labels. Upper applications only make decisions on location and policies of access interfaces inside the SDN control domain. ZENIC supports Internet encapsulation formats such as MPLS and VLAN. In the next stage, VXLAN and GRE will also be supported.

3.2 D-Zenic

To support large-scale networks and guarantee performance, ZENIC Release 2 has a distributed controller architecture and is re-named D-ZENIC (**Fig. 2**). The architecture features two critical approaches to reducing overhead in state synchronization of distributed controllers. In the first approach, the distributed controller reduces message replication as much as possible. In the second approach, controller states should synchronized according to the user's demands. Only a necessary and sufficient state should be replicated.

The controller cluster includes a controller manager and one or more controller node(s). Here, the controller manger is a logical entity which could be implemented on any controller node.

3.3 Controller Cluster Management

In a D-ZENIC system, an improved one-hop DHT algorithm is used to manage the distributed controller cluster. Each controller manages a network partition that contains one or more switches. All OpenFlow messages and events in the network partition, including the entry and exit of switches, detection of link state between switches, and transfer of flow requests by these switches, are handled independently by the connected controller.

The controller synchronizes messages about changes of switch topology across all controllers in the network so that these controllers have a consistent global view of the switch topology. Controllers learn host addresses from the packets transferred by their controlled switches and store this information in the distributed controller network. When a controller receives a flow request, it inquires about the source/destination addresses of the flow in the distributed network, chooses a path for the flow (based on the locally stored global switch topology), and issues the corresponding flow tables to the switches along the path. The controller also provides communication across the nodes, including data operation and message routing for the application.

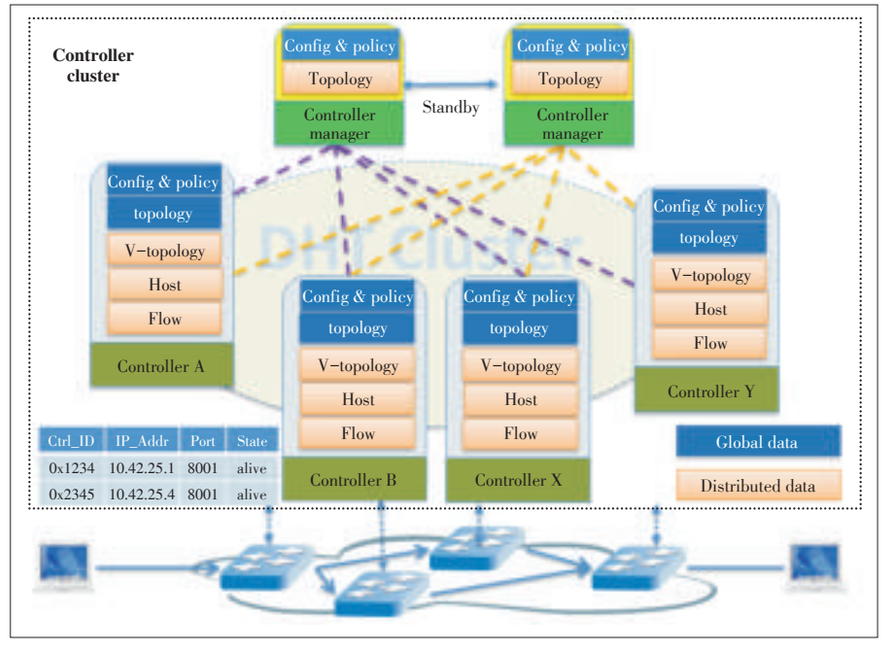
The controller manager, which manages the controller cluster, is responsible for configuring clusters, bootstrapping controller nodes, and providing unified NBI and consistent global data. The controller manager is also responsible for managing the access of the controller node. When a controller node joins or leaves the cluster, the controller manager initiates cluster self-healing by adjusting relationship between specified switches and controller nodes. The controller manager node has a hot standby mechanism to ensure high availability.

In D-ZENIC, the switch should conform to OpenFlow Protocol 1.3 [12]. The switch establishes communication with one master controller node and, at most, two slave controller nodes. It does this by configuring OpenFlow Configuration Point (OCP) on the controller manager. If the master controller fails, a slave requests to become the master. Upon receiving notification from the controller manager, the switch incrementally connects to another available controller.

In D-ZENIC, underlying communication between controller nodes is facilitated by a message middleware zeroMQ [13]. ZeroMQ is a successful open-source platform, and its reliability has been verified in many commercial products.

3.4 DB Subsystem Based on DHT

In an SDN, a switch sends a packet in message to the controller when the flow received by the controller does not match



▲**Figure 2. D-ZENIC deployment.**

any forward flow entry. The controller then decides to install a flow entry on specified switches. This decision is made according to the current network state, which included network topology and source/destination host location. In D-ZENIC, the network state is divided into global data and distributed data according to requirements in terms of scalability, update frequency, and durability.

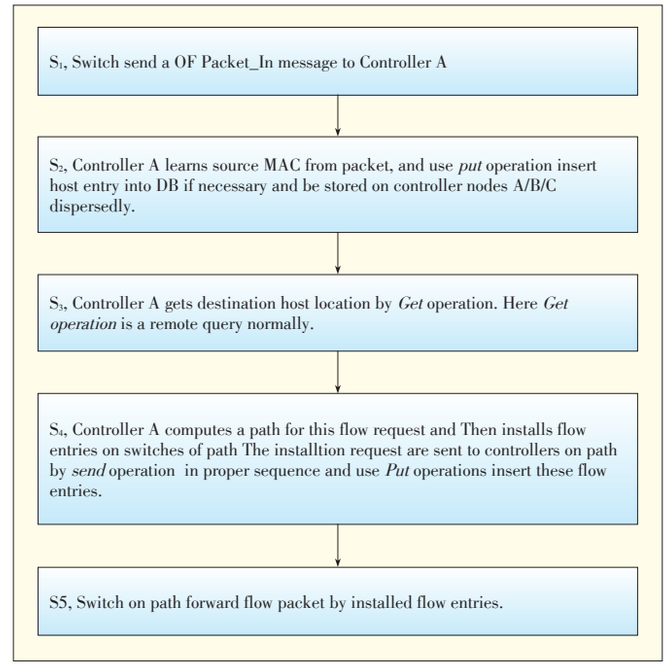
Global data changes slowly and has stringent durability requirements in terms of switches, ports, links, and network policies. Other data changes faster and has scalability requirements in terms of installed flow entries and host locations. However, applications have different requirements in different situations, e.g., because the link load changes frequently, it is important to the traffic engineering application.

The DB subsystem based on DHT is custom-built for D-ZENIC. This subsystem provides fully replicated storage for global data and distributed storage for dynamic data with local characteristics. Furthermore, for higher efficiency and lower latency, the basic *put*/*get* operation is synchronous for global data and asynchronous for distributed data. The DB subsystem based on DHT provides the following basic primitives: 1) *Put*, for adding, modifying, or deleting an entry that has been decentralized; 2) *Get*, to find a matched entry with specified keys; 3) *Subscribe*, applications subscribe change event about topology, host, flow entry, etc.; 4) *Notify*, to notify subscribers of any changes to an entry, and 5) *Publish*, to push an entry to all controller nodes.

3.5 Application Service Implementation

The application deployed on top of D-ZENIC has a consistent network state, so there is no difference in its core code. For example, a joining switch and its port list are automatically inserted by a *Put* operation and then published to all controller nodes. A host entry is inserted by a *Put* operation and then stored into three controller nodes selected using the DHT algorithm. The information of switches, links and hosts would be equally consumed by application services on each controller node.

The steps for processing a basic L2 flow are shown in **Fig. 3**. Of particular note is the use of the DB subsystem based on DHT.



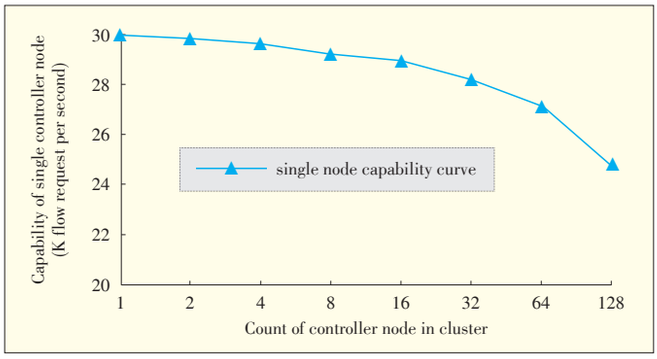
▲**Figure 3. Basic L2 flow implementation in D-ZENIC.**

4 Performance Evaluation

In D-ZENIC, each switch connects to one master controller and two slave controllers simultaneously. According to the OpenFlow specification [12], the master controller fully controls the switches, and the slave controller only receives parts of asynchronous messages. However, slave controller does not need to consider asynchronous events from its connected switches in D-ZENIC because the slave controller can get all switch states from the DB based on DHT. A flow request from a switch is handled locally by its connected master controller, and the performance of the controller is affected by the cost of data synchronization across controller nodes. D-ZENIC has near-linear performance scalability as long as the distributed database in the controller nodes can handle it.

In a general Internet data center [4] comprising ten thousand physical servers, there may be 20 VMs running on each server. Each top of rack (TOR) has 20 servers connected, with 10 concurrent flows per VM [14]. This will lead to 200 concurrent flows per server; that is, there will be 2 million new flow installs per second. The flow entry count is about 200 on the software virtual switch, or 4000 on the TOR switch for per-source/destination flow entry in the switch. Actually, the count of the flow entry will increase tenfold because the expiration time of the flow entry is longer than the flow's lifetime.

Theoretically, the total memory consumed by a controller is bounded by $O(m(Hn+F)+TN)$, where H is the number of host attributes, which include ARP and IP at minimum; F is the number of flow entries on all connected switches to each controller; T is the number of topology entries, which include links and ports of all switches; N is the number of controller nodes; m is the count of data backups in DHT; and n is the index table number of host attributes. Here, m = 3 and n = 2. The amount of CPU processing power consumed by a controller is given by $O(m(p\times H\times n+q\times F)+r\times T\times N)$, where p, q, and r denote the frequency of updates of the host attribute, flow entry, and topology entry, respectively. We estimate the capacity of an individual controller node according to the scale of controller cluster in **Fig. 4**. Assuming that an individual controller can handle 30,000 flow requests per second, the data center previously mentioned should require 67 controller



▲**Figure 4. Capability of a single controller node according to the scale of the controller cluster.**

nodes. If D-ZENIC is deployed, about 2350 extra operation requests per second will be imposed on each controller node, i.e., the processing capacity of individual controller node will fall by 7.83%, and another four controller nodes will be required. We could improve performance by optimizing the distribution database model.

5 Conclusion

Central control and flexibility make OpenFlow a popular choice for different networking scenarios today. However, coordination within the controller cluster could be a challenge in a large-scale network. D-ZENIC is a commercial controller system with near-linear performance scalability and distributed deployment. With D-ZENIC, a programmer does not need to worry about the embedded distribution mechanism. This is an advantage in many deployments. We plan to construct a large integrated test-bed D-ZENIC system that has an IaaS platform, such as OpenStack. This will allow us to thoroughly evaluate the scalability, reliability, and performance of D-ZENIC.

References

[1] McKeown Nick, et al., "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008. doi: 10.1145/1355734.1355746.
[2] *Open Networking Foundation* [Online]. Available: https://www.opennetworking.org/
[3] M. Casado, M. J. Freedman, J. Pettit, et al., "Ethane: taking control of the enterprise," *ACM SIGCOMM Computer Commun. Review*, vol. 37, no. 4, pp. 1-12, 2007. doi: 10.1145/1282427.1282382.
[4] A. Tavakoli, M. Casado, T. Koponen, et al.. *Applying NOX to the Datacenter* [Online]. Available: http://www.cs.duke.edu/courses/current/compsci590.4/838-CloudPapers/hotnets2009-final103.pdf
[5] Amin Tootoonchian and Yashar Ganjali. *Hyperflow: a distributed control plane for openflow* [Online]. Available: https://www.usenix.org/legacy/event/inmwren10/tech/full_papers/Tootoonchian.pdf
[6] J. Stribling, et al., "Flexible, wide-area storage for distributed systems with wheelFS," *NSDI*, vol. 9, pp. 43–58, 2009.
[7] Koponen, Teemu, et al., "Onix: a distributed control platform for large-scale production networks." *OSDI*, vol. 10, pp. 1–6, 2010.
[8] P. Hunt, M. Konar, F. P. Junqueira, et al., "ZooKeeper: wait-free coordination for internet-scale systems," in *Proc. 2010 USENIX Conf. USENIX annual technical conference*, Boston, MA, USA, 2010.
[9] *Open Daylight Project* [Online]. Available: http://www.opendaylight.org/
[10] Marchioni Francesco and Manik Surtani. *Infinispan Data Grid Platform* [Online]. Available: http://www.packtpub.com/infinispan-data-grid-platform/book
[11] Wang Jun, "Software-Defined Networks: Implementation and Key Technology," *ZTE Technology Journal*, vol. 19, no. 5, pp.38-41, Oct. 2013.
[12] *OpenFlow Switch Specification (Version 1.3.0)*[Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf
[13] P. Hintjens. *ZeroMQ: Messaging for Many Applications* [Online]. Available: http://www.pdfbooksplanet.org/development-and-programming/587-zeromq-messaging-for-many-applications.html
[14] A. Greenberg, J. R. Hamilton, N. Jain, et al., "VL2: a scalable and flexible data center network," *ACM SIGCOMM Computer Communication Review - SIGCOMM' 09*, vol. 39, no. 4, pp. 51–62, 2009. doi: 10.1145/1594977.1592576.

Manuscript received: 2014-03-10

Biographies

Yongsheng Hu (hu.yongsheng@zte.com.cn) received his PhD degree from Nanjing University of Science and Technology in 2008. He is a senior engineer in the Central R&D Institute, ZTE Corporation. His current research interests include distributed system, SDN and cloud computing.

Tian Tian (tian.tian1@zte.com.cn) received her Dipl.-Ing degree in electronic and information engineering from Technical University Dortmund, Germany, in 2008. She is a senior standard and pre-research engineer in the Central R&D Institute, ZTE Corporation. Her current research interests include core network evolution and network function virtualization.

Jun Wang (wang.jun17@zte.com.cn) graduated from Nanjing University of Aeronautics and Astronautics, China, and received his MS degree in 2006. He is an architect at the System Architecture department of Central R&D, ZTE Corporation. His research interests include core network evolution, distributed system, and datacenter networking.

DOI:10.3939/j.issn.1673-5188.2014.02.005
http://www.cnki.net/kcms/detail/34.1294.TN.20140521.1111.001.html, published online May 21, 2014

Software-Defined Cellular Mobile Network Solutions

Jiandong Li, Peng Liu, and Hongyan Li
(Information Science Institute, State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China)

Abstract

The emergency relating to software-defined networking (SDN), especially in terms of the prototype associated with OpenFlow, provides new possibilities for innovating on network design. Researchers have started to extend SDN to cellular networks. Such new programmable architecture is beneficial to the evolution of mobile networks and allows operators to provide better services. The typical cellular network comprises radio access network (RAN) and core network (CN); hence, the technique roadmap diverges in two ways. In this paper, we investigate SoftRAN, the latest SDN solution for RAN, and SoftCell and MobileFlow, the latest solutions for CN. We also define a series of control functions for CROWD. Unlike in the other literature, we emphasize only software-defined cellular network solutions and specifications in order to provide possible research directions.

Keywords

SDN; cellular network; radio access network; core network; OpenFlow

1 Introduction

Software-defined networking (SDN) is a compelling technique that has spurred innovative experiments and the evolution of computer networks. It has gained more and more recognition in both academia and industry. SDN gives network designers greater flexibility to separate the control plane, which decides how traffic is handled, from the data plane, which forwards traffic according to decisions made by the control plane [1]. SDN architecture makes the network programmable and facilitates the designing of new protocols. The control plane communicates with the data plane via a well-defined API in order to direct packet-forwarding. A good example of such an API is OpenFlow [2], which was developed at Stanford University and is a milestone in SDN. Before OpenFlow, there was a dynamic tension between those who envisioned a fully programmable network and those who envisioned a more pragmatic network for practical applications.

Researchers started to apply SDN in other fields, such as wireless networking. Both OpenRoads [3] and OpenFlow Wireless [4] were developed at Stanford University. The core idea of these platforms is to make wireless networks open by flattening out some vertical wireless networking techniques. A programmable data plane is created so that subscribers experience seamless handover in a heterogeneous network. However, an open policy does not take into account the specifications of different access techniques, i.e., differences between WLANs and cellular networks, and does not take into account explicit commercial requirements. Gradually, two areas of research emerged: software-defined WLAN and software-defined cellular networks. A typical WLAN solution is Odin [5].

Cellular networks, the main topic of this paper, initially had a relatively complex structure. They are used to guarantee QoS and generate revenue for operators. Before flowing to the Internet, traffic first passes through radio access networks (RANs) and then through core networks (CNs). A RAN comprising users and base stations is generally responsible for radio-related services, such as radio resource mapping and interference management. A CN lies between the base station and Internet and provides packet- or traffic-related carrier service, such as traffic classification and authorization. To facilitate programmability in cellular networks, it is necessary to address the uniqueness of the network [6].

SoftRAN [7], as its name suggests, is an SDN prototype designed to address the challenges in a RAN. Typically, interference management in an LTE mobile network is distributed by coordinating between cells. By separating the control plane from data plane and building a central controller, interference can be effectively migrated between cells. By comparison, the evolution of CN has been more complicated and challenging because fine-grained packet handling can also cause new problems with scalability. SoftCell [8] and MobileFlow [9] are two recent attempts to counter this.

In this paper, we articulate three leading SDN solutions for a cellular network: SoftRAN, SoftCell and MobileFlow. These have been developed over the past two years and have guided new research and applications. We also introduce connectivity management for energy-optimized wireless dense networks (CROWD) [10], which is a collaborative project funded by the

European Commission under the Seventh Framework Programme (FP7). CROWD has a series of control functions that can be mapped into the actions in physical networks. We also discuss challenges related to the evolution of software-defined cellular networks and discuss our recent work.

2 Background

2.1 Software-Defined Networking and OpenFlow

SDN can be viewed in two ways: 1) separation of the control plane from data plane and use of a single software-control program to control multiple data planes, and 2) abstraction of network control in terms of forwarding, specification, and distribution [11]. SDN enables flat, flexible data planes with high-level control planes.

The SDN layers are shown in **Fig. 1**. The north part of the architecture refers to the part above the controller and includes the policy layer and application layer. The south part of the architecture refers to the programmable switches (e.g., OpenFlow). The controller is the brain of the SDN, which obtains information about global resources and network state from south interfaces and makes abstract and global decisions. The controller can be taken as a network operator system, and alternatives include NOX [12] and Floodlight [13].

One of the best-known south-part interfaces is OpenFlow switch, whose protocol has been standardized by the Open Networking Foundation (ONF). OpenFlow is supported by many vendors, including HP, NEC and IBM, and associated switches are available on the market. By separating the control plane from the data plane, OpenFlow provides new possibilities for innovation. The main components of OpenFlow are the flow table and security channel (Fig. 1). The flow table monitors and forwards packets. If the property of the packet matches existing flow entries in the flow table, actions for that entry are performed on the packet. If there is no match, the switch delivers the packet to the security channel and communicates with the controller through the OpenFlow protocol. Finally, the controller determines how to process the packet and updates the entry in the flow table.

2.2 Architecture of an LTE Cellular Network

A typical mobile network comprises two main parts: RAN and CN. The RAN is responsible for radio-related functions, such as scheduling, radio resource and interference management, coding, and multiple antenna schemes. The CN is responsible for authentication, charging, and establishing end-to-end connections [14]. Isolating these functions from the RAN is beneficial to integrating a CN with multiple RANs. Mobile terminals directly connect to the RAN (**Fig. 2**, right). A flat architecture with only one node, an eNodeB, is used in LTE. The eNodeB is a virtual eNode implemented as a site or base station. The eNodeB connects to MME over the control plane (Fig. 2, solid line) and S-GW over user plane (Fig. 2, dashed line). The eNodeB and MME communicate with each other via the X2 interface in order to manage intercell radio resources and coordinate intercell interference.

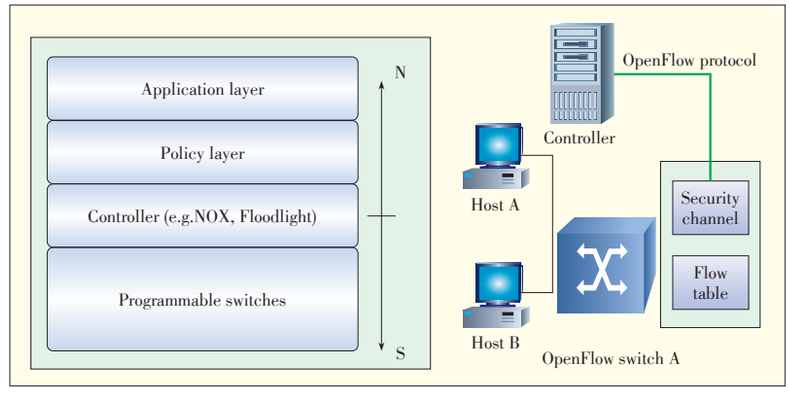
In CN, the mobility-management entity (MME) is the control-plane node. Its functions include connecting and releasing bearers to and from a terminal. The serving gateway (S-GW) is the user-plane node that connects the CN to the RAN. The S-GW acts as a mobility anchor for mobile management and provides statistics for charging. The packet data network gateway (P-GW) is the edge node that connects the CN to the Internet. The P-GW allocates an IP address for a particular terminal.

3 Solutions of Soft-Defined Cellular Networks

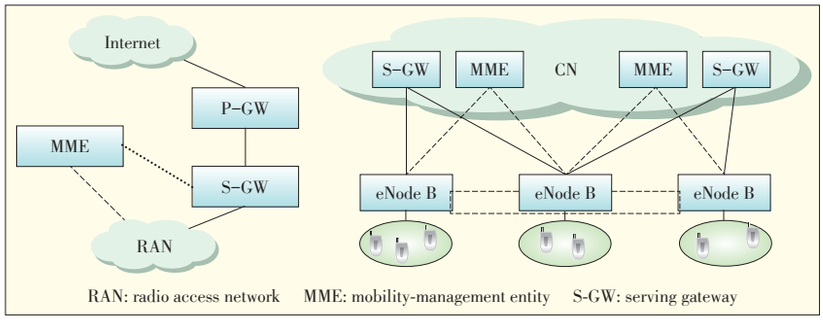
3.1 RAN Solutions

The RAN ensures limited resources are used effectively in radio-related functions. The RAN allocates resources and manages interference, handover, and load-balancing. Currently, the control plane in a RAN is distributed and coordinates intercell interference through message exchange over X2 interface. This is not optimal partly because distributed coordination algorithms generally require iterative and periodic updates of radio resource allocation decisions, and this is hard to get right at scale [7]. SoftRAN boosts the utility by separating the control plane from base stations and forming a high-level central controller.

SoftRAN conforms more to the second SDN concept mentioned in section 2.1: abstraction. Base stations in a geographically nearby area are abstracted as a big virtual station that has a global view of underlying base stations and controls their behavior. SoftRAN defines resources in terms of time, frequency, and base stations. The central controller in the big station determines what spectrum and transmit power will be used at the sites of underlying base stations. The control signaling is de-



▲Figure 1. SDN layers (left) and OpenFlow architecture (right).



▲Figure 2. CN architecture (left) and RAN architecture (right).

fined by the API and exchange between controller and underling base stations via the backhaul.

There are latency problems in the backhaul, and wireless channel conditions may vary rapidly. To address these challenges, SoftRAN also defines a local controller within base stations. This local controller is responsible for decisions that do not affect neighboring sites. Following this principle, some functions are separated and specified; specifically, handovers and downlink transmit power in each channel are arranged by the controller because these have implications for nearby sites. The allocation of a resource block, i.e., a minimum assignable resource unit in LTE, is noticeable. In the downlink, resource blocks can be allocated within base stations because the transmit power has been specified. Downlink resource block allocation is mainly used because it is adaptable to channel variations; however, uplink transmit power is controlled by users in order to counter path loss, and the central controller has no knowledge of the uplink transmit power. In the uplink, resource block allocation dominates in the central controller.

We present the architecture of SoftRAN in **Fig. 3**. A special component is RAN information base (RIB) containing global network state. It includes a weighted interference graph where the weight stands for the average channel conditions between the nodes. Flow Records stores flow-related information such as buffer state. Preferences indicate operator's appetite for different flows.

3.2 CN Solutions

In mobile communications, CN is unique in that it supports fine-grained service. Such support depends heavily on customized policies based on a wide variety of subscriber attributes and application classes [8]. These data services are usually provided by a P-GW, which integrate network functions such as content filtering, traffic optimization, firewalls, and lawful interception [15]. However, combining all these functions on the data plane in a P-GW may make a network inefficient, rigid, and complex. First, all traffic will be forwarded through P-GWs regardless of whether it is device-to-device traffic with latency requirements or video traffic with service-rate requirements. This increases delay and congestion. Second, if an operator wants to cancel an unneeded function in a P-GW, they have to replace the P-GWs. Finally, operators cannot adopt equipment from different vendors.

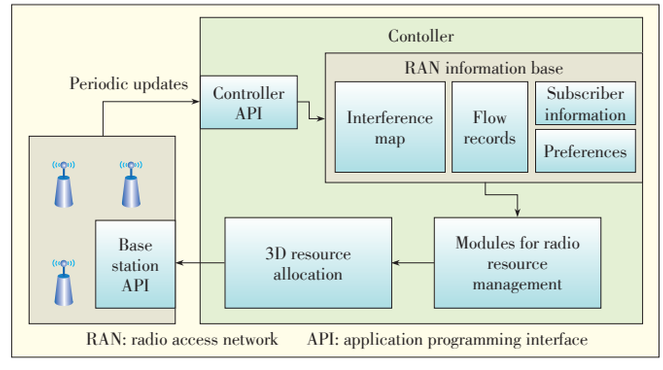
SDN is a good solution for abstracting the control plane. Evolving cellular networks towards SDN presents new challenges, such as inadequate scalability. SoftCell and MobileFlow help overcome such challenges.

3.2.1 SoftCell

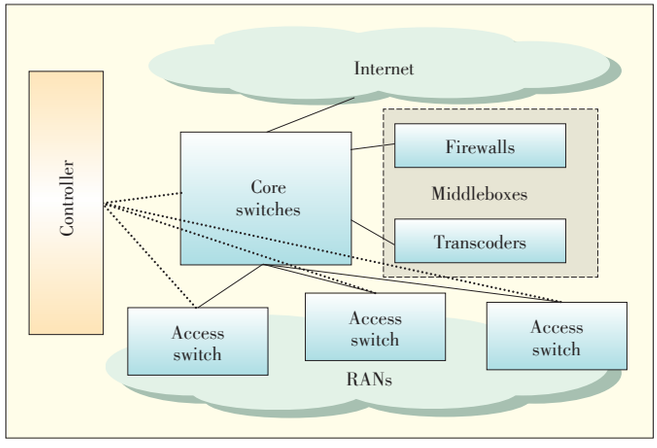
The central idea of SoftCell is to decentralize the functions of a P-GW, offloading them to a serious of switches and middle-boxes, both of which have particular functions and application requirements and are controlled by a central controller. In keeping with the rationale of devolved responsibility, components between a base station and the Internet become low-cost, general equipment. The core network of SoftCell merely comprises middle-boxes, which act as transcoders, web caches or firewalls; access switches, which classify fine-grained packets from users; core switches, which include gateway switches that connect to the Internet and forward packets at high speed; and the controller, which makes global decisions [8]. Thus, this architecture (**Fig. 4**) is flatter and cheaper than a traditional LTE CN architecture. In the SoftCell architecture, there are no special P-GWs or S-GWs (Fig. 2, left), and the control-plane prototype is implemented on top of Floodlight [12].

To enable this architecture and fine-grained policy, the authors of [8] define service policy on top of the controller. This concept is similar to the policy layer in Fig. 1. A service policy designates which traffic (in predicate) should be handled in what way (in action). In **Table 1**, the first clause indicates that video traffic deriving from a user with a gold billing plan must first pass through a firewall before going to the transcoder. The second clause indicates that M2M signaling should be given high priority when passing through a firewall in order to ensure low latency.

Because there are too many different combinations of service requirements for different classes of traffic (e.g., QoS classes, device types, etc.), a data explosion may occur in the flow table. To resolve this issue, SoftCell leverages the aggre-



▲Figure 3. SoftRAN architecture.



▲Figure 4. SoftCell architecture.

gating multidimensional information as well as traditional location- and tag-based routing. Another issue to be addressed is the asymmetry of the architecture, i.e., A CN usually connects to hundreds of or even thousands of base stations while only deploying a few gateway switches that connect to the Internet. Moreover, base stations only deal with the packets of a limited number of active users whereas gateway switches process the packets from the Internet, and the volume of data is usually large. Similarly, if we classify packets on both edges, the latency will be large on the side closest to the Internet. SoftCell addresses this issue by a concept called "smart access edge, dumb gateway edge." In other words, SoftCell classifies packets in access switches when a user invokes a flow and embeds associated messages in the head of the packet. Finally, the authors [8] highlight the problem of dynamics, which a special problem in wireless networks. This problem is solved by deploying a local agent or local controller to form a hierarchical control structure. Actually, this problem can be solved by introducing SoftRAN in a complementary way.

3.2.2 MobileFlow

MobileFlow emphases evolution and provides a blueprint for software-defined mobile networks. It also separates the control plane from the data plane, which is a concept inherited from SDN. MobileFlow has a new entry for supporting the special functions of a mobile network, such as network layer (L3) tunneling and flexible charging. It can also be integrated into OpenFlow networks for basic packet forwarding. A new controller is abstracted in order to manage new entry-based networks, and an OpenFlow controller is introduced to control underlying OpenFlow networks.

In **Fig. 5**, the noticeable enablers are MobileFlow forwarding engine (MFFE) and MobileFlow controller (MFC). MFFEs include all mobile network tunnel processing capabilities. MFFEs also act as wireless access nodes that operate in parallel with existing eNodeBs in order to manage radio bearers, e.g., the one near the RAN. MFFEs have more mobile-related functions than normal switches, such as OpenFlow switches, and are much simpler than a P-GW or router. MFFEs can guarantee QoS and customized services. MFFEs are controlled by an MFC, which is similar to an OpenFlow controller (Fig. 1). MobileFlow also defines a lightweight protocol between the MFC and MFFEs. To support the properties of an existing LTE network, MobileFlow also defines a mobile network application (MNA) north of the control plane. This integrates the functions of existing entities, such as P-GW, S-GW, and MME. Moreover, MobileFlow uses OpenFlow networks. In Fig. 5, the blue flow goes directly to the Internet through an OpenFlow switch. Such a flow can be offload traffic without QoS requirements. The orange flow passes through a set of MFFEs to a specific service.

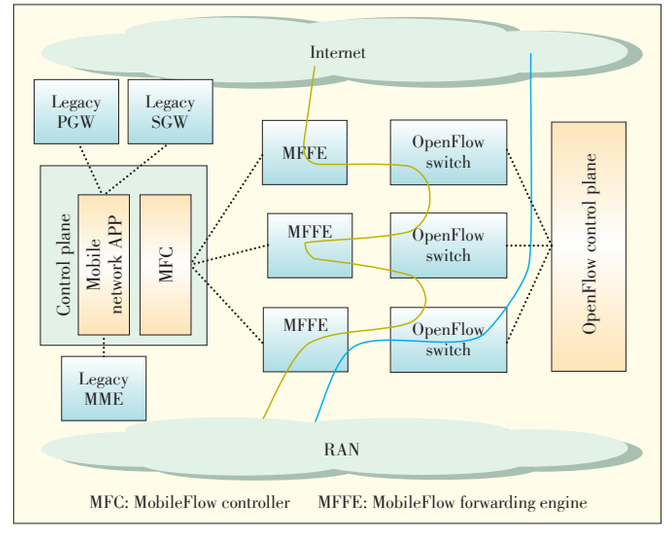
Finally, the authors of [8] design a prototype to verify the flexibility of the MobileFlow architecture and show how existing mobile networks can evolve through a software-defined architecture.

3.2.3 SoftCell vs. MobileFlow

Here, we turn from what SoftCell and MobileFlow support to what they address. Both are based on SDN and provide flexibility and possibilities for new innovation. Both have separate control plane and data plane. Because they both have a software-defined or programmable structure, their architectures can readily be integrated or changed. For example, we can combine MFFEs and OpenFlow switches in MobileFlow: some of these new combined switches will act as access switches and others will act as core switches in SoftCell. An OpenFlow

▼Table 1. A service policy

Priority	Predicates	Service Actions
1	App=video ^ plan=Gold	[Firewall, Transcoder]
2	App=M2M signaling	[HighPriority, Firewall]



▲Figure 5. MobileFlow architecture.

controller can be merged with an MFC to form a super controller in SoftCell.

SoftCell addresses new challenges in software-defined architecture whereas MobileFlow addresses evolution, i.e., coexistence with and gradual replacement of legacy entities. MobileFlow is designed to benefit operators. The authors of [9] suggest that operators start to deploy MFFEs, which can operate with legacy equipment via, for example, GTP/PMIP tunnels, and try out new services in the software-defined part of the mobile network.

3.3 Control-Plane Functions in CROWD

CROWD is an SDN project in Europe. At present, the focus of CROWD is the design of control functions in control plane and the mapping of these functions to the north- or south-end APIs. This ultimately affects physical actions within a network.

CROWD addresses issues in dense heterogonous networks with multiple integrated access networks, such as LTE (macro and small cells) and Wi-Fi. In such networks, there are challenges in terms of mobility, interference, handover, and energy consumption. To help overcome these challenges, CROWD defines a series of control functions in an SDN-based architecture. These functions are valuable for future cellular networks. A CROWD controller has a two-tier design comprising a CRC for global control and a CLC for local control. The functions of these two controllers are listed in **Table 2** and **Table 3** [10].

▼**Table 2. Functions of CRC**

CRC	Applications
topology and network element discovery and monitoring	power-cycling, long-term clustering
controller placement and lifecycle management	long-term adaption of radio parameters, long-term clustering
backhaul management	power-cycling, traffic-proportional backhaul reconfiguration

CRC: CROWD regional controller

▼**Table 3. Functions of CLC**

CLC	Applications
monitoring/filtering	eICIC, access selection, load balancing, WLAN optimization
network discovery	eICIC, access selection, load balancing, D2D offloading
power control setting	eICIC
access selection setting	access selection, load balancing, D2D offloading
scheduling policy control	eICIC, D2D offloading
ABSF control	eICIC
content management	D2D offloading
relay management	access selection, D2D offloading
AP packet retention control	AP cooperation
Wi-Fi parameter setting	WLAN optimization
subframe synching	eICIC, D2D offloading

D2D: Device to Device
CLC: CROWD Local Controller
eICIC: enhanced Inter-cell Interference Coordination

4 Discussion

Software-defined RAN requires a hierarchal control architecture where the local controller can adapt to variable channel conditions. SoftRAN divides the control responsibilities between a high-level controller and local controller. However, to leverage this principle, researchers also need to analyze more specific cases, i.e., dynamic traffic (full buffer or non-full buffer).

Moreover, in the RAN information base contained within the high-level controller of SoftRAN, weighted interference maps are used to abstract the interference relationship between cells; however, practical interference environments are more complex. To describe the interference space distribution and its dynamics, a multi-dimensional interference status space needs to be constructed. Both weighted interference maps and multi-dimensional interference status space reflect the physical interference conditions through abstraction and are the input for radio resource management. This modulus can be informally regarded as the combination of interference management and radio resource management. Because these functions are implemented in the high-level controller with latency considerations, performance can be improved through prediction or by supporting cognitive interference-management schemes based on strategies, Q-learning, interference transfer, and avoidance.

The CN still seems to have a long way to go. Even if SoftCell and MobileFlow have some implementation possibilities, operators can still not be persuaded to replace their legacy structures. After all, these solutions are in the proof-of-concept stage, and from SoftCell in particular, there are many new challenges in software-defined cellular networks. Thus, it is better to use an evolving compatible architecture, such as MobileFlow, at this stage. Meanwhile, researchers need to pay more attention to the uniqueness of and challenges associated with software-defined CN as it relates to SoftCell.

One of our research interests is to figure out the effective interference characteristics approaches, as mentioned previously. We are also concerned about the packet-scheduling problem in software-defined cellular networks. By providing fine-grained services, the packet may be labeled with the requirements of multiple resource types. For example, one packet needing one CPU time unit and 10 KHz bandwidth to be processed may be labeled with <1,10> as resource profile. Those packets with a multi-resource profile will be delivered to particularly functional middle-boxes (Fig. 4).

5 Related Work

In the last two years, there have been many surveys on SDN. In [16] and [17], the authors mainly focus on wireline net-

works, present the architecture of SDN, and present the development of each component. OpenFlow is referred to as a special topic in [18]. The progress of SDN in wireless networks is referred to in [11] and [19]. In [11], the authors divide the technique roadmap into wireless WLANs and cellular networks. Although [11] gives a brief and clear introduction of SDN in wireless networks, it does not elaborate on and compare methods in cellular networks. To the best of our knowledge, this paper is the first work to look into software-defined cellular networks. We describe the latest approaches in this field and provide insights into these approaches.

6 Conclusion and Open Research Areas

In this paper, we have presented state-of-the-art SDN solutions for cellular networks. Specifically, we have elaborated SoftRAN, which is the latest SDN solution for RAN. SoftRAN has a hierarchical control structure, i.e., the high-level controller makes decisions based on global information, and the local controller can adapt to variable channel conditions. SoftRAN is based on the principle of dividing control responsibilities between a high-level controller and local controllers. SoftCell and MobileFlow are CN-related SDN solutions. We have compared these approaches and provided insight into the design of software-defined cellular networks. We have also discussed our recent work in this direction and expect to invoke some new ideas on software-defined cellular networks.

There are still many research areas that can be exploited. Mobile networks tend to be dense and large-scale. Although SoftCell and CROWD address some issues, their performance in particular cases is still unclear. To increase capacity, a distributed massive MIMO system is necessary in the future. SDN-enabled cross-layer MIMO is attractive, especially for the configuration of different beamforming matrixes. Another interesting area is combined use of different access techniques. Existing solutions are still in the proof-of-concept stage.

References

[1] N. Feamster, J. Rexford, and E. Zegura. *The Road to SDN: an intellectual history of programmable networks* [Online]. Available: https://www.cs.princeton.edu/courses/archive/fall13/cos597E/papers/sdnhistory.pdf

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008. doi: 10.1145/1355734.1355746.

[3] K. K. Yap, M. Kobayashi, R. Sherwood, T. Y. Huang, M. Chan, N. Handigol, and N. McKeown, "Openroads: Empowering research in mobile networks," *ACM SIGCOMM Computer Communicatio*n Review, vol. 40, no. 1, pp. 125–126, 2010. doi: 10.1145/1672308.1672331.

[4] K. K. Yap, R. Sherwood, M. Kobayashi, T. Y. Huang, M. Chan, N. Handigol, N. McKeown, and G. Parulkar, "Blueprint for introducing innovation into wireless mobile networks," in *Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures*, New York, USA, pp. 25–32. doi: 10.1145/1851399.1851404.

[5] L. Suresh, J. Schulz-Zander, R. Merz, A. Feldmann, and T. Vazao, "Towards programmable enterprise WLANs with Odin," in *Proceedings of the first workshop on Hot topics in software defined networks*, New York, USA, pp. 115–120. doi: 10.1145/2342441.2342465.

[6] L. E. Li, Z. M. Mao, and J. Rexford, "Toward software-defined cellular networks," in *Software Defined Networking (EWSDN), 2012 European Workshop on*, pp. 7–12. doi: 10.1109/EWSDN.2012.28.

[7] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "SoftRAN: Software defined radio access network," in *ACM SIGCOMM HotSDN Workshop*, New York, USA, pp. 25–30. doi: 10.1145/2491185.2491207.

[8] X Jin, L. E. Li, L. Vanbever, and J. Rexford. *SoftCell: Taking control of cellular core networks* [Online]. Available: http://arxiv.org/abs/1305.3568

[9] K. Pentikousis, Yan Wang, and Weihua Hu, "Mobileflow: Toward software-defined mobile networks," *Communications Magazine, IEEE*, vol. 51, no. 7, pp. 44–53, 2013. doi: 10.1109/MCOM.2013.6553677.

[10] H. Ali-Ahmad, C. Cicconetti, A. de la Oliva, et al., "CROWD: An SDN Approach for DenseNets," *Software Defined Networks (EWSDN), 2013 Second European Workshop on*, vol. 12, no. 2, pp. 25–31, 2013. doi: 10.1109/EWSDN.2013.11.

[11] Qadir Junaid, Nadeem Ahmed, and Nauman Ahad. *Building Programmable Wireless Networks: An Architectural Survey*[Online]. Available: http://arxiv.org/abs/1310.0251

[12] N. Gude, et al., "NOX: Towards an Operating System for Networks", *SIGCOMM CCReview*, Vol. 38, Issue 3, July 2008. doi: 10.1145/1384609.1384625.

[13] *Floodlight Openflow Controller*[Online]. Available: http://floodlight. openflowhub.org/

[14] E. Dahlman, S. Parkvall, and J. Skold, *4G LTE/LTE-Advanced for Mobile Broadband*, UK: Academic Press, 2011, pp. 95–127.

[15] *Cisco PGW Packet Data Network Gateway* [Online]. http:// www:cisco:com/en/US/products/ps11079/index:html

[16] M. Mendonc, B. N. Astuto, X. N. Nguyen, K. Obraczka, T. Turletti, et al.. *A survey of software-defined networking: Past, present, and future of programmable networks* [Online]. Avalibale: http://hal.inria.fr/hal-00825087/

[17] K.Hyojoon and N. Feamster, "Improving network management with software defined networking," *Communications Magazine, IEEE*, vol. 51, no. 2, pp. 114–119, 2013. doi: 10.1109/MCOM.2013.6461195.

[18] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation using OpenFlow: A Survey," *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 1, pp. 493–512, 2013. doi: 10.1109/SURV.2013.081313.00105.

[19] S. Costanzo, L. Galluccio, G. Morabito and S. Palazzo, "Software Defined Wireless Networks: Unbridling SDNs," in *Software Defined Networking (EWSDN), 2012 European Workshop on*, Darmstadt. doi: 10.1109/EWSDN.2012.12.

Manuscript received: 2014-02-15

Biographies

Jiandong Li (jdli@mail.xidian.edu.cn) received his BE, MS, and PhD degrees in Communications Engineering from Xidian University, Xi'an, in 1982, 1985 and 1991. He has been a faculty member of the school of Telecommunications Engineering, Xidian University, since 1985. He is currently a professor and vice director of the academic committee of the State Key Laboratory of Integrated Service Networks. Professor Li is a senior member of IEEE. He was a visiting professor in the Department of Electrical and Computer Engineering, Cornell University, from 2002 to 2003. He was the general vice chair of ChinaCom 2009 and TPC chair of IEEE ICCC 2013. He was awarded as Distinguished Young Researcher award from NSFC and Changjiang Scholar from the Ministry of Education, China. His main research interests include wireless communication theory, cognitive radio, and signal processing.

Peng Liu (liupeng0218@gmail.com) is a PhD candidate at Xidian University, Xi'an, China. He is also a visiting scholar at Columbia University, NY. He received his BS degree in Telecommunications Engineering from Xidian University in 2010. His research interests include resource allocation, interference avoidance, and interference cancellation in heterogeneous networks; and packet scheduling in SDN.

Hongyan Li (hyli@xidian.edu.cn) received her MS degree in control engineering from Xi'an Jiaotong University, China, in 1991. She received her PhD degree in signal and information processing from Xidian University, China, in 2000. She is currently a professor in the State Key Laboratory of Integrated Service Networks, Xidian University. Her research interests include wireless networking, cognitive networks, integration of heterogeneous network, and mobile ad hoc networks.

DOI: 10.3969/j. issn. 1673-5188. 2014. 02. 006
http://www.cnki.net/kcms/detail/34.1294.TN.20140605.0951.002.html, published online June 5, 2014

SDN-Based Data Offloading for 5G Mobile Networks

Mojdeh Amani [1], Toktam Mahmoodi [1], Mallikarjun Tatipamula [2], and Hamid Aghvami [1]
(1. King's College London, The Strand, London, WC2R 2LS, UK;
2. F5 Networks, San Jose, CA 95134, USA)

Abstract

The rapid growth of 3G/4G enabled devices such as smartphones and tablets in large numbers has created increased demand for mobile data services. Wi-Fi offloading helps satisfy the requirements of data-rich applications and terminals with improved multimedia. Wi-Fi is an essential approach to alleviating mobile data traffic load on a cellular network because it provides extra capacity and improves overall performance. In this paper, we propose an integrated LTE/Wi-Fi architecture with software-defined networking (SDN) abstraction in mobile backhaul and enhanced components that facilitate the move towards next-generation 5G mobile networks. Our proposed architecture enables programmable offloading policies that take into account real-time network conditions as well as the status of devices and applications. This mechanism improves overall network performance by deriving real-time policies and steering traffic between cellular and Wi-Fi networks more efficiently.

Keywords

mobile data offloading; LTE/Wi-Fi interworking; policy derivation; network selection; software-defined networking; dynamic policies; 5G mobile networks

1 Introduction

In 2013, mobile phones overtook PCs as the most common Internet access device worldwide, and by 2015, more than 80% of handsets sold in mature markets will be smartphones [1]. According to forecasts, global mobile data traffic will grow 13-fold between 2012 and 2017, which is three times faster than fixed IP traffic [2]. Mobile network operators will carry the bulk of Internet traffic in the future, but they face significant challenges in addressing needs associated with increased traffic demand. To meet this demand, mobile operators are investing in more network capacity. Scarcity of spectrum is forcing such operators to deploy smaller cells and utilize unlicensed spectrum, such as Wi-Fi. Availability of built-in Wi-Fi on smartphones and unlicensed spectrum makes Wi-Fi a natural solution for accommodating increased traffic and maintaining the quality of users' connections.

To this end, mobile data offloading, which refers to the use of complementary network technologies to deliver data originally targeted at mobile/cellular networks, has already become a key solution for meeting traffic demands [3]–[5]. Various platforms, including opportunistic communications, have been considered for mobile data offloading [6]. Furthermore, mobile networks must support various applications, such as voice and streaming, as well as best-effort services on a single IP-based infrastructure. Each of these converged services has quality of service (QoS) requirements, in terms of latency, packet loss and data rates, that must be met through efficient allocation of wireless network resources and cannot that be met only by provisioning the network. One of the main challenges in data offloading is making real-time decisions on offloading the flows and services/applications of different users while taking the condition of available networks and QoS needs of flows into account. In this regard, previous research has focused on providing seamless movement between different access networks during data offloading [7]. Dual stack mobile IP has also been considered to enable simultaneous use of two interfaces [8]. All the technologies involved in mobile data offloading from the LTE network are ready for deployment and, in fact, can be deployed by operators. However, we lack efficient offloading techniques and decision-making criteria.

The mobile networks architecture proposed for 4G/LTE provides easier management compared with earlier architectures because it separates signalling plane functions, such as mobility management, policy-making, and charging. Despite this, today's 4G/LTE architecture is not yet as flexible or programmable as it could be. For example, the 4G/LTE network can provide QoS guarantees and differentiated services to the user through policy-charging and rules function (PCRF) nodes. Policy and charging control (PCC) ensures that the user has guaranteed QoS for a particular subscription and service type. Today's PCCs are user-aware and application-aware but do not have awareness of network congestion.

We propose placing the policy control closer to the wireless access and deriving policies according to radio network infor-

mation as well as previously used user and application information. This can significantly improve mobile data offloading. We propose an abstraction layer based on software-defined networking (SDN) [10] that integrates network resource management (NRM) with radio resource management (RRM) and enables programmable, dynamic policy functions.

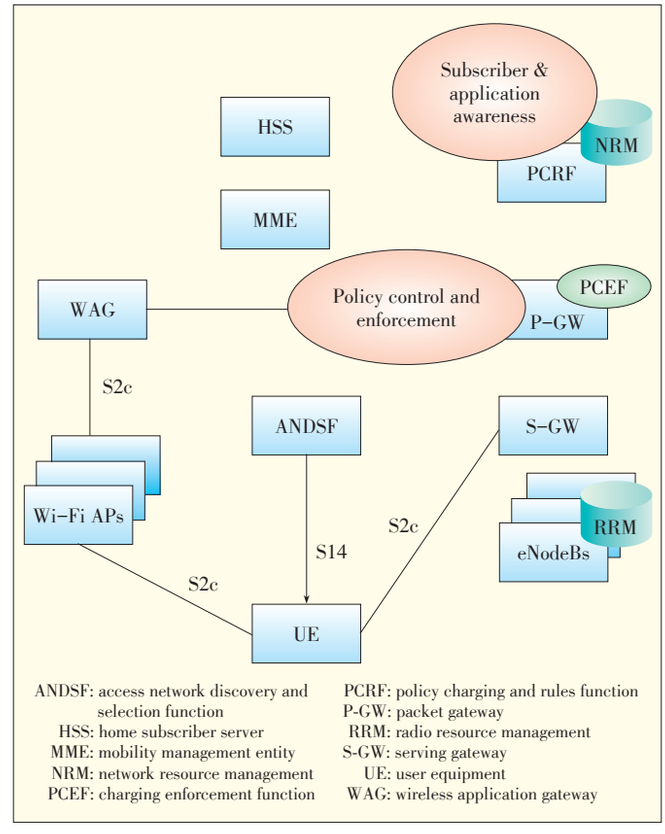
The remainder of this paper is organized as follows. In section 2, we give an overview of current LTE/Wi-Fi interworking architecture. In section 3, we discuss network selection and related challenges. In section 4, we give details of the proposed interworking modification for better QoS. In section 5, we present the policy derivation and offloading mechanism. In section 6, we conclude with a summary of programmable data offloading policies.

2 Cellular and Wi-Fi Interworking Architecture

The goal of mobile data offloading is to dynamically redirect selected traffic towards the lower-cost RAN. 3GPP has been developing new standards and architectures to support the simultaneous use of different cellular access networks, such as LTE and femtocells, as well as non-3GPP access networks, such as Wi-Fi. 3GPP standardization covers the native integration of trusted and untrusted non-3GPP IP access network into the EPC [6]. This standard treats Wi-Fi RAN as valid an access network as any other 3GPP RAN and enables operators to use standard-based EPC components to integrate different types of access networks. It also facilitates interoperability between these networks. Such integration ensures a good level of inter-operability between different access types.

In this paper, we focus on trusted non-3GPP access network architecture, i.e., Wi-Fi access network architecture, which is owned by the cellular operator. S2c and S2a are the two interfaces that provide control and mobility support between non-3GPP access network and packet gateway (P-GW). They also forward Wi-Fi traffic to the EPC (**Fig. 1**). S2c provides mobility and control support between user equipment (UE) and P-GW over the non-3GPP access network. S2a provides control and mobility support between the trusted non-3GPP access network and P-GW. The serving gateway (S-GW) serves between the LTE and mobile access gateway (MAG) of Wi-Fi and reports to the PCRF. The S-GW also sets bearer QoS parameters. Moreover, eNodeB, S-GW and P-GW are involved in other control plane functions, such as location update and mobility, in coordination with the mobility management entity (MME).

Access network discovery and selection function (ANDSF) is an entity within an evolved packet core (EPC) of the system architecture evolution for 3GPP mobile networks. ANDSF helps UE to discover non-3GPP access networks, such as Wi-Fi for data communication, in addition to 3GPP access networks and provides the policies used to access these networks. When combined with ANDSF, the Mobility over GPRS Tunneling Protocol based on S2a enables an operator to benefit from controlled automatic network discovery and selection for the user. This results in a seamless user experience. ANDSF uses a standard S14 interface to communicate information and policies to the UE. This information is organized within nodes of a managed object that contains several nodes, including nodes for discovery information, intersystem mobility policies, and intersystem routing policies. S14 is the only standard interface for ANDSF, and any interaction between the ANDSF server and other network elements is outside the scope of current 3GPP standards.



▲**Figure 1. 3GPP architecture for the non-3GPP IP access integration into the EPC.**

These new measures in the standard enable seamless handover and traffic steering so that users have continuous data service as they roam between cellular, small cell, and Wi-Fi networks. Hence, users can benefit from secure, transparent services regardless of the types of access technology.

3 Network Discovery and Selection

Wireless networks are becoming increasingly heterogeneous. In addition, more mobile devices are capable of simultaneously operating on multiple technologies, i.e., 3G, 4G and Wi-Fi radio. Given this, selecting the best network for a user at any given time and location is crucial for optimizing the access of that

user. The main consideration in any network-selection strategy is network-based information, which informs decisions on network selection and traffic steering, as well as the different ways of distributing this information to devices. Information from traffic-steering policies, the real-time network condition of both cellular and Wi-Fi networks, and subscriber profiles are all necessary for optimal network selection. In addition, the device itself contains important information, such as battery usage, radio conditions and relative motion, which should be considered when making a decision. Thorough consideration of all these parameters makes network selection a very complex problem. In this section, we discuss the possibility of including various decision criteria.

3.1 ANDSF

ANDSF was defined in the recent 3GPP standards as a framework for issuing policies to a device where traffic routing decisions between a cellular and Wi-Fi access network are being made. The device periodically determines the validity of policies in terms of location and time of the day. Many operators are interested in using ANDSF for policy-based network selection and traffic steering. A few of these operators, such as Telefonica, are already trialing the ANDSF server solution. Although ANDSF provides a standard framework for distributing flexible, operator-defined network-selection information and policies, it does not capture additional information in an operator's network, e.g., network condition, which could be useful to derive dynamic policies and communicate them with the device. The key issue in this framework can be addressed by implementing an ANDSF client in devices [11]. This client communicates with an ANDSF server in the network and distributes ANDSF policies to the device so that networks can be dynamically selected and traffic steering decisions can be made [9]. Deriving appropriately dynamic policies and distributing them to the ANDSF client enables an operator to steer traffic between Wi-Fi and cellular networks, create better user experience, and better utilize radio resources.

3.2 Challenges

One of the most important aspects of efficient network selection is the current network condition of all available access networks. For example, the real-time load on the radio link of a cell site significantly impacts the QoS of a given user and can influence the selection of that access network. The 3GPP standard allows user-subscription-dependent policies for ANDSF; however, there is no standardized interface between ANDSF and the user's subscription/profile information, i.e., between ANDSF and the home subscriber server (HSS). In fact, 3GPP enables ANDSF to communicate with the UE via S14 interface. The data obtained from the UE is mainly the UE's location and device type [10]. This information is rather static, but various other dynamic data can significantly affect efficient network selection.

Operators are pursuing solutions in which more dynamic criteria are used in the selective offloading of traffic between all available RANs. On the user side, network-selection criteria that are of interest to the operator include user subscription level and user profile, which includes usage history or caps. Network-driven criteria include different types of Wi-Fi access network, e.g., trusted, public or private, and venue information. Despite this, ANDSF policies are static because ANDSF does not receive any input except that from the UE via the S14 interface. In other words, existing ANDSF policies do not take into account either network condition, such as load and congestion, or UE conditions.

We argue that policies based on an awareness of network, application, and user-profile at the ANDSF can significantly improve both network utilization and user experience. To derive dynamic policies, first we need to determine what type of information must be considered, how to collect and capture this information, and how to derive policies and communicate them to UE for efficient network selection. In the next section, we discuss the challenges related to designing a policy entity that has an awareness of network, application, and user profile, and we present one solution based on it. Our solution is a centralized solution that is, in fact, one way of realizing SDN. An alternative way could be a fully distributed user-centric approach, where the UE collects data and selects a network. We will exploit this family of solutions in our future work.

4 Programmable Data Offloading

We envisage a policy entity that can receive the network, application, and user-profile information; derive the policy, and distribute it to the ANDSF server in the network. Complexity arises from the fact that the network, application, and user-profile data are available at different parts of the end-to-end communication path. For example, the UE is the only entity aware of effective radio condition, throughput over existing connection, active applications, pending traffic, and battery levels. We design a centralized architecture, where policies are driven at a central controller within the network architecture. Here we describe how data is collected and policies derived.

4.1 Decision Criteria and Policy Control

In the architecture in Fig. 1, the policy and charging enforcement function (PCEF) in the P-GW enforces policies and maps service data flows to the bearer that is to be mapped to the underlying transport network. The PCRF is an LTE policy manager that uses operator policies, network information, and user profile stored in the HSS to make decisions according to a set of pre-defined rules and functions. The PCRF is also responsible for QoS authorization, i.e., authorizing the treatment of each traffic flow. Today's policy controls have awareness of users and applications but not congestion within the network [12]. Fine-grain control on various entities in the mobile net-

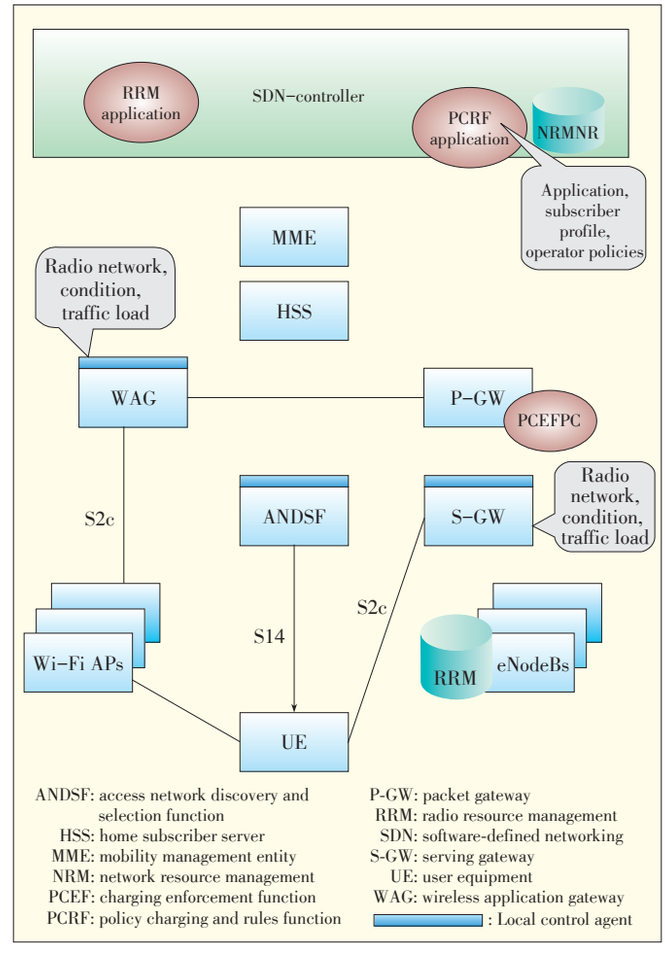
work is crucial so that operators can allocate resources and cheaply maintain and expand the network. It is also important for employing dynamic policies, i.e., for managing the traffic and selectively offload the traffic between different access networks according to the current network condition.

When managing traffic, the policy enforcement point can either be the core network or access network. At the core, policies are enforced in the P-GW, which has deep packet inspection (DPI) functions. This is similar to today's architecture (Fig. 1). Service information does not have to travel at all, and the DPI engine can store subscriber information from the policy controller. This enables congestion and location to be estimated. The central solution described here requires simple integration of the DPI engine into the policy controller. On the other hand, congestion information is extremely dynamic, i.e., it changes rapidly and by the time values sent from RAN are received by the P-GW, the information is no longer valid. Various studies show that bad decisions were made 40% of the time when outdated congestion information was used.

Moreover, estimation based on current traffic is not possible for the DPI engine because:

- The policy controller lacks a feedback mechanism. Simple questions such as "Is 1 Mbps for P2P is enough or are we over penalizing?" cannot be answered.
- The policy controller only has a general indication of reduced throughput, which may occur as the result of poor coverage or congestion. Only the RAN can differentiate between these two causes.
- Quite often, a RAN is shared between different operators, or perhaps the operator is using multiple access points. In these cases, the policy controller cannot see all traffic passing through congested cells.
- Cell capacity depends on the coverage of individual subscribers and varies, even with weather conditions. There may be additional reserved bandwidth for future bearers with guaranteed data rate or other similar cases that affect total cell capacity. In other words, cell capacity varies over time, and the policy controller receives no information about the variations in capacity of the congested cells.

One alternative for addressing these challenges is to move the policy enforcement point to where congestion occurs so that we do not need to transfer dynamic information anywhere. In this way, the scheduler continuously prioritizes data packets and subscriber sessions. The scheduler has perfect knowledge about the location of the user, traffic, and real congestion conditions at the location. Then, the policy controller takes service information from the DPI function and changes QoS parameters such as traffic handling priority, maximum bit rate, guaranteed bit rate, or QoS Class Identifier (QCI). In LTE, there it is also possible to create a dedicated bearer for a specific traffic flow requiring differentiated QoS treatment at the policy-enforcement point. This architecture (**Fig. 2**) makes network management more complex. It is still not clear where the policy control should be located in order to increase efficiency but not significantly increase complexity.



▲**Figure 2. SDN-controlled network selection.**

4.2 SDN Controller and Mobile Data Offloading

As discussed earlier, deriving real-time policies for selective offloading of different services/applications according to the dynamics of network is potentially complex. A programmable interface similar to that in SDN facilitates offloading by providing end-to-end communication between network elements and by pushing corresponding forwarding rules to local elements, i.e., eNodeB and P-GW. In our proposed architecture, the control-plane functionality of the gateways is decoupled and located at the SDN controller as applications. The gateways run local control agents. The SDN-controller derives offloading policy functions and rules by combining information from the RRM and PCRF applications. The radio network condition, defined by wireless condition and traffic load, is measured frequently by the local control agents. This enables an operator to monitor traffic in real time, provide per-subscriber QoS through programmable application modules in the SDN controller, and derive forwarding rules accordingly. These poli-

cies and forwarding rules are periodically sent to the local control agents in the access network and are forwarded to the UE. The LTE and Wi-Fi interworking architecture, including SDN controller and interactions with local control agents are shown in Fig. 2. The two main parts of this architecture are the SDN controller and logical control agents.

The SDN controller in this architecture is an abstraction model that runs programmable applications modules, such as RRM and PCRF. The PCRF application module has subscriber and application information, and the RRM application module collects radio access network conditions, such as traffic load and cell capacity. The SDN controller combines the information from these application modules in order to derive a single set of policies and rules that are sent periodically to the local control agents.

To address the scalability issue and challenges raised in section 3.2, we propose local control agents in the network gateways, i.e., P-GW, S-GW and WAG as well as in the RAN. These local control agents should have some measurement and control capabilities that are authorized by the SDN controller. For example, the agents that run on the gateways can measure QoS parameters, such as delay and resource utilization, and compare the traffic counters with the threshold. These agents can then notify the SDN controller when the threshold has been exceeded. To communicate back with the controller, an interface similar to OpenFlow [14] is required at the local agents. This also enables the agents to exercise simple control, such as changing the weight or priority of a queue, when the traffic counter exceeds the threshold.

5 Policy Derivation and Offloading Mechanism

5.1 Policy Derivation

In order to derive policies, network load information, signal threshold, and operator policy are combined in the SDN controller. This section explains the parameters that are considered when deriving policies in a few different scenarios.

5.1.1 Network Load

In a scenario where the operator controls both a cellular and Wi-Fi networks in a given area and the cellular network is not congested, the operator may prefer to serve customers via the cellular network. As the load on the cellular network increases, potentially impacting user experience, the operator may want to start steering some of the traffic towards the Wi-Fi network. As the cellular network becomes even more congested, the operator may want to steer even more traffic towards the Wi-Fi network. This policy can be made even more effective if the condition of the radio on the cellular network is taken into account. For example, cell-edge users experiencing the worst radio conditions on the network can be steered towards the Wi-Fi network first. Also, when the cellular network is congested, the operator who controls both the cellular and Wi-Fi networks may want to steer users experiencing poor cellular radio from the cellular network to Wi-Fi. As the cellular network becomes more congested, the operator may want to steer more users towards Wi-Fi. Even when the cellular network is not congested, some users may experience poor radio on the cellular network but have access to a Wi-Fi network with acceptable quality and load. In this case, the operator may want to serve that user's traffic via Wi-Fi instead of the cellular network. Additionally, the exact thresholds at which certain users are steered to Wi-Fi depend on the distribution of eNBs and Wi-Fi access points (APs) in the network as well as the instantaneous distribution of UEs in the vicinity. Thus, the mapping between load level and the signal strength at which a user is steered to Wi-Fi is not static.

5.1.2 Signal Strength

A signal strength threshold can also be included in the policy to ensure the correct users are steered towards Wi-Fi while others are kept on the cellular network. For example, when the cellular load is 70%, the signal strength threshold may be −108 dBm, but when cellular load is 88%, the signal strength threshold may be −105 dBm [15]. This signal strength threshold

indicates a minimum received signal strength below which UEs should attach to available and acceptable Wi−Fi APs. Specifically, the UE compares experienced signal strength with the signal strength threshold of the received policies via ANDSF and then selects a network. An operator can first move UEs with poor cellular radio to Wi-Fi APs in a smart way that takes into account the cellular network load as well. Furthermore, the signal strength threshold may be combined with thresholds provided by operator policies, e.g., in a case where different policies are defined for different types of users. Alternatively, the operator might simply want to introduce different tiers of service. For example, the operator policy might seek to steer heavy video users to Wi-Fi while keeping subscribers who complain about Wi-Fi service on the cellular network.

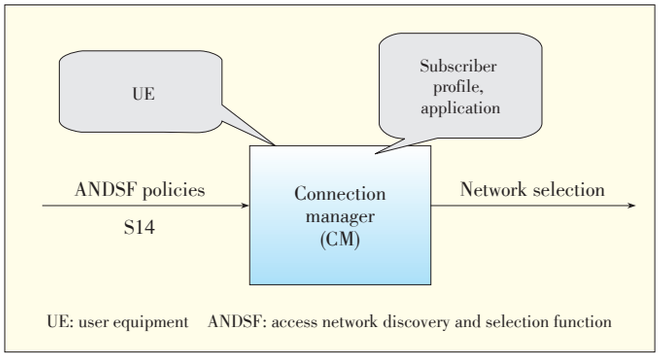
5.1.3 Random Generated Value

Steering large numbers of UEs between cellular and Wi-Fi APs may dramatically affect instantaneous network conditions. For example, if a large number of users were steered towards Wi-Fi, the load on the cellular network would be reduced. This reduction could cause those same UEs to try and reselect the cellular network because it is now a much more desirable access network than before. Because all the users move back to the cellular network, the load increases and causes users to be steered back towards Wi-Fi. Thus, UEs ping-pong between the cellular and Wi-Fi networks as long as this process continues [15]. To avoid the ping-pong effect, a calculated integer can be included in the policy along with load level and signal strength

threshold. This calculated integer is used to steer a subset of targeted UEs together rather than all targeted UEs at once. Therefore, a random integer a is generated in the range of 0-10 and is distributed in the policy. Furthermore, each UE generates a random value b in the same range. If $b < a$ and other conditions are satisfied, then a given UE is steered towards the Wi-Fi AP; otherwise, the UE stays on the cellular network. In this way, a given fraction of the targeted population is steered towards Wi-Fi at any given time. The operator can control this fraction by random distributed values. In the case of different service tiers for different user types, more than one calculated value can be distributed, and each value is targeted at a specific class of user with similar service needs.

5.2 Offloading Mechanism

The local control agents in the RAN and access gateways collect information, such as drop rate, utilization and traffic load, periodically and report it back to the SDN controller. The SDN controller combines this information with the operator's policies and subscriber's profile in the PCRF, derives the policies and forwarding rules, and communicates these policies and rules to the local control agent in ANDSF via an interface similar to OpenFlow [13]. Each mobile device is expected to have a connection manager (CM) [14], which is a functional component that takes ANDSF policies and user preference as input and combines them with the local conditions of UEs in order to offload and steering traffic (**Fig. 3**). The ANDSF dynamic policies include but are not restricted to a variety of parameters, such as operator policies, cellular and Wi-Fi conditions (e.g., traffic load), Wi-Fi quality (including packet loss, RTT and throughput), user subscription profile, time of the day, and location. The local condition of a UE includes movement of the mobile device relative to one or more APs, mobile battery usage, and application requirements (e.g., service continuity and throughput). The relative movement of a device towards an AP can be calculated within the device. There are a number of ways of performing this calculation, such as extrapolation via the rate of change in AP signal strength measured by the UE or estimation by the device according to implementation-dependent mechanisms. The policies are prioritized, and each



▲**Figure 3. UE architecture.**

is validated according to the local condition of the UE. The CM selects the network and steers traffic according to a valid policy. The simplest mechanism may include only two policies, one for when cells are congested and another for when cells are not congested. Both policies would be sent to the UE for use when in a valid area at a designated time of the day and would be updated by the ANDSF only when there is a change in conditions, such as long-term busy hour. Another example could be providing different cellular network traffic load thresholds to the UE according to the type of subscriber. One such threshold, *Cellular_Load_Max*, is for heavy-streaming users, and another threshold can be specified for high-priority users. Therefore, operators can obtain a desirable outcome by defining a policy upon triggering of congestion in a cellular network so that heavy-streaming users are moved to Wi-Fi before high-priority users.

Supporting per-UE or per-cell policies does not mean that the new ANDSF policies should be pushed to UE whenever it changes location or that unique policies need to be maintained for a large number of cells. The frequency with which these policies are updated can be adjusted according to significant changes in network condition and availability of communication resources between the controller and access networks. Hence, during busy hours, a new policy derivation is triggered when a cell is congested. This derivation is based on the received data. The offloading mechanism here offers 1) dynamic policies because NRM and RRM are converted to software applications, 2) robust network-selection mechanism because a precise network condition, such as congestion in the backhaul, is captured, 3) efficient network selection because user preference and application requirements are taken into account, and 4) simplicity because the control plane functions are abstracted and communication between them is simplified.

6 Conclusion

Wi-Fi has become an increasingly popular access mode enabling wireless carriers to meet the capacity demands of mobile data users. The amount of traffic carried over Wi-Fi networks has grown dramatically in recent years and is projected to continue to grow in the years to come. To address the issues related to growing demand, we explore state-of-the-art Wi-Fi/cellular integration and propose a modified architecture that enhances key aspects of this integration and facilitates evolution towards 5G mobile. Such aspects include network discovery and selection, traffic steering, and the effect of policies and rules on traffic steering. In this paper, a programmable policy function derivation mechanism is proposed and enabled through the use of an SDN controller in the mobile backhaul. Our proposed mechanism takes into account real-time network condition as well as existing user and application information in order to control offloading policies and efficiently accommodate traffic on cellular or Wi-Fi access networks. We couple

network resource management with radio resource management in the form of application modules at the SDN controller. This enables us to derive offloading policies that optimize both cellular and wireless resources.

References

[1] Gartner. (2013). *Top 10 Strategic Technology Trends* [Online]. Available: www.gartner.com/newsroom/id/2209615
[2] "VNI global IP traffic forecast, 2012–2017," Cisco White Paper, May 2013.
[3] A. Aijaz, H. Aghvami, and M. Amani, "A survey on mobile data offloading: technical and business perspectives," *IEEE Wireless Communications Magazine*, vol. 20, no. 2, pp. 104–112, Apr. 2013. doi: 10.1109/MWC.2013.6507401.
[4] K. Samdanis, T. Taleb, and S. Schmid, "Traffic offload enhancements for eUTRAN," *IEEE Communications Surveys and Tutorials*, vol. 14, no. 3, pp. 884–896, Jul. 2012. doi: 10.1109/SURV.2011.072711.00168.
[5] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: how much can Wi-Fi deliver?" *IEEE/ACM Transaction on Networking*, vol. 21, no. 2, pp. 536–550, Apr. 2013. doi: 10.1109/TNET.2012.2218122.
[6] B. Han, P. Hui, V. Kumar, M. Marathe, J. Shao, and A. Srinivasan, "Mobile data offloading through opportunistic communications and social participation," *IEEE Transactions on Mobile Computing*, vol. 11, no. 5, Mar. 2012. Doi: 10.1109/TMC.2011.101.
[7] A. De La Oliva, C. Bernardos, M. Calderon, T. Melia, and J. Zuniga, "IP flow mobility: smart traffic offload for future wireless networks," *IEEE Communication Magazine*, vol. 49, no. 10, pp. 124 – 132, Oct. 2011. doi: 10.1109/MCOM.2011.6035826.
[8] 3GPP. (2013). *3GPP TS 23.402 V10.7.0: Architecture Enhancements for Non-3GPP Accesses* [Online]. Available: www.quintillion.co.jp/3GPP/Specs/23402-a70.pdf
[9] "Architecture for mobile data offload over Wi-Fi access networks," Cisco White Paper, 2012.
[10] L. E. Li, Z. M. Mao, and J. Rexford, "Toward software-defined cellular networks," in *European Workshop on Software-defined Networking (EWSDN)*, Darmstadt, Germany, October 2012, pp. 7–12. doi: 10.1109/EWSDN.2012.28.
[11] 3GPP. (2013). *3GPP TS 24.312 V12.3.0: Access Network Discovery and Selection Function (ANDSF) Management Object (MO)* [Online]. Available: www.3gpp.org/ftp/specs/archive/24_series/24.312/
[12] *Access to the 3GPP Evolved Packet Core (EPC) via Non-3GPP Access Networks*, 3GPP TS 24.302 V10.7.0, 2012.
[13] *Policy and charging Control Architecture*, 3GPP TS 23.203 V9.3.0, Dec. 2009.
[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, April 2008. doi: 10.1145/1355734.1355746.
[15] "Integration of cellular and Wi-Fi networks", 4G Americas White Paper, 2013.

Manuscript received: 2014-03-04

Biographies

Mojdeh Amani (mojdeh.amani@kcl.ac.uk) worked in cellular industry after graduating with a BSc degree in telecommunications. She received her MSc in digital signal processing and PhD in telecommunications from King's College London. Her research interests include QoS provisioning, resource management, data offloading in next-generation heterogeneous wireless networks, cloud computing, and software-defined networking. She has an ongoing interest in advances in science and their applications beyond research. She has publications on different aspects of next-generation mobile networks.

Toktam Mahmoodi (toktam.mahmoodi@kcl.ac.uk) works in the Department of Informatics, King's College London. She received her BSc degree in electrical engineering from Sharif University of Technology, Iran, in 2002. She received her PhD degree in telecommunications engineering from Kings College London in 2009. From 2010 to 2011, she was previously a postdoctoral research assistant in the Intelligent Systems and Networks Group, Department of Electrical and Electronic Engineering, Imperial College London. From 2006 to 2009, she was a PhD research assistant in the Core-4 Efficiency Program, Virtual Centre of Excellence in Mobile and Personal Communications (Mobile VCE). She has previously received the IEEE Best Paper Award from IEEE ICC and IARIA E-Energy. She is a member of IEEE and the ACM and has been on the Program Committee of number of IEEE flagship conferences.

Mallikarjun Tatipamula (m.tatipamula@f5.com) is vice president and CTO of service provider and cloud solutions at F5 Networks. He is responsible for innovation and implementation of new and disruptive technologies across the company. Mallik Tatipamula has more than 23 years' experience with telecommunication and networking technologies and has held leadership roles at Ericsson, Juniper, Cisco, and Motorola. He is a fellow of IET and has co-authored more than 100 patents and publications and two books on networking. He has a PhD in information and communication engineering from the University of Tokyo. He received BS degree from the Indian Institute of Technology, Madras, India. He has delivered lectures on networking at Stanford University, Tsinghua University China, and Beijing University of Posts and Telecommunications. He is currently a visiting professor at King's College London.

Hamid Aghvami (hamid.aghvami@kcl.ac.uk) is director of the Centre for Telecommunications Research, King's College London. He leads a team that is working on numerous mobile and personal communications projects supported by government and industry. From 2001 to 2003, he was a member of the Board of Governors of the IEEE Communications Society. From 2004 to 2007, he was a distinguished lecturer of the IEEE Communications Society. He has also been a member, chairman, and vice-chairman of the Technical Program and Organizing Committees of many international conferences. He is the founder of the International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), a major yearly conference attracting around 1000 attendees. He has published more than 700 technical papers and given invited talks and courses worldwide on various aspects of personal and mobile radio communications. He has received numerous awards for his technical contributions to the communications field and for his services to scientific and engineering communities. He is a fellow of the Royal Academy of Engineering and the IET.

DOI:10.3939/j.issn.1673–5188.2014.02.007
http://www.cnki.net/kcms/detail/34.1294.TN.20140604.1151.001.html, published online June 04, 2014

Integrating IPsec within OpenFlow Architecture for Secure Group Communication

Vahid Heydari Fami Tafreshi [1], Ebrahim Ghazisaeedi [2], Haitham Cruickshank [1], and Zhili Sun [1]
(1.Centre for Communication System Research (CCSR), University of Surrey, Guildford, Surrey, GU2 7XH, UK;
2. Department of Systems and Computer Engineering, Carleton University, Ottawa, K1S 5B6, Canada)

Abstract

Network security protocols such as IPsec have been used for many years to ensure robust end-to-end communication and are important in the context of SDN. Despite the widespread installation of IPsec to date, per-packet protection offered by the protocol is not very compatible with OpenFlow and flow-like behavior. OpenFlow architecture cannot aggregate IPsec-ESP flows in transport mode or tunnel mode because layer-3 information is encrypted and therefore unreadable. In this paper, we propose using the Security Parameter Index (SPI) of IPsec within the OpenFlow architecture to identify and direct IPsec flows. This enables IPsec to conform to the packet-based behavior of OpenFlow architecture. In addition, by distinguishing between IPsec flows, the architecture is particularly suited to secure group communication.

Keywords

IPsec; OpenFlow; secure group communication; group domain of interpretation (GDOI); flow-based switching

1 Introduction

As an attempt to embrace the future Internet and its tendency towards software-defined networks (SDN), OpenFlow suggests a move into programmable rather than configurable network deployments. This results in faster innovations through software change rather than infrastructure adaption [1]. OpenFlow works well on the premise that the control plane can be separated from data plane on network packet forwarders and brought into an OpenFlow controller (a server) with centralized network management. All network elements, including routers and switches, are now simple packet forwarders with no complexity. Starting initially with campus networks, data centers such as Google are now extensively reinforced with this evolving architecture [2].

On the other hand, end-to-end security of communication at the IP level is guaranteed by the IPsec framework [3]. As the word "framework" implies, IPsec is not directly limited to any specific security algorithm or technology. Subsequently, the level of security can be tuned by different open standards and combinations to fulfill various immunity requirements of the production environment. Virtual private network (VPN) as a solution for providing a logical channel between two peers over a public and probably insecure network relies on the IPsec for its immunity. Small-office home office (SOHO) scenario or different sites of a corporation which are geographically spread out are other possible use cases to apply VPN remedy over IPsec.

Point-to-point tunnels between two VPN gateways used to be exploited to carry authenticated as well as encrypted traffic from one site to another. However, group domain of interpretation (GDOI) [4]–[6] with IPsec at its core goes even further so that secure communication between various sites called group members (GM) is now possible without any tunnels between these branches.

IPsec as an algorithm-independent framework addresses the confidentiality by encryption as well as the integrity with the aid of hashing as the main security objectives while allows for authenticating the origin of the traffic. Regardless of the core network and its elements, the tunable IPsec protocol with huge install base is simply provisioning the necessary security services for both end entities. Nonetheless, security gained through IPsec is per-packet. This is not deployable to leading future Internet designs such as OpenFlow architecture with flow-based behavior. OpenFlow aims to aggregate different packets into flows and process these flows rather than individual packets. OpenFlow, however, cannot uniquely identify IPsec flows and aggregate/direct these flows accordingly. We provide the ability to distinguish between IPsec flows in order to integrate secure group communication into the OpenFlow architecture. Our ultimate goal is to address this deficiency within OpenFlow by our proposed method. We propose using a security parameter index (SPI) of IPsec within the OpenFlow architecture to uniquely identify and direct IPsec flows.

The rest of the paper is structured as follows. In section 2, we discuss briefly four basic elements used later for our proposed method. This is needed specifically to clarify ambiguities especially those pertaining to a complex protocol like IPsec. The clarification emphasizes characteristics which form the cornerstone of the method in section 3. Subsection 2.1 explores OpenFlow and abstracts the ideas behind this evolving architecture. Subsection 2.2 sanitizes the required features of IPsec itself. The establishment of the secure channel through internet key exchange (IKE) is discussed in 2.3. Subsection 2.4 briefly covers GDOI basics as a cryptographic protocol for group key management for secure group communications. The proposed method is discussed in detail in section 3. Section 4 elaborates on the main use case for the proposed method here which is secure group communication conforming to GDOI standard and its integration into the OpenFlow. Section 5 concludes this paper.

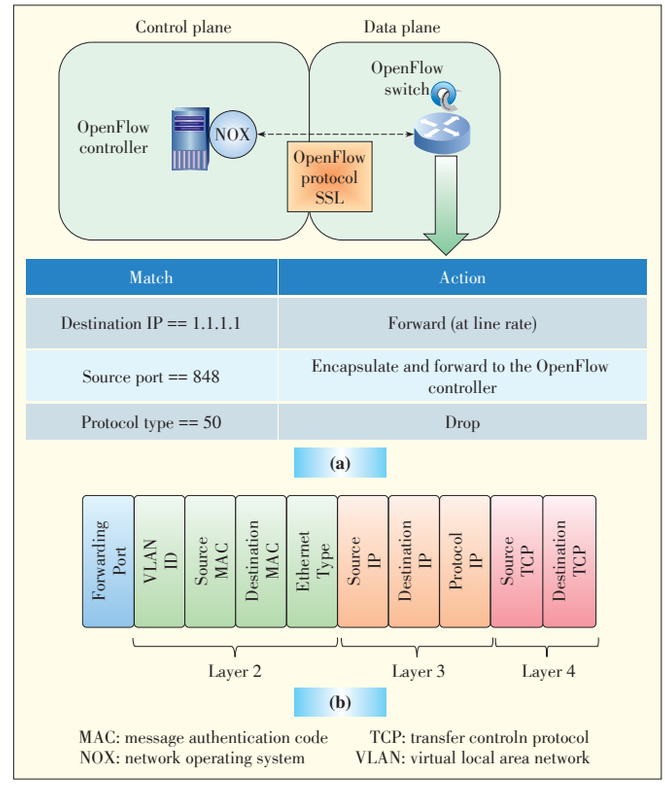
2 Background

2.1 OpenFlow Architecture

OpenFlow improves network programmability and enables packets to be forwarded at a speed approximating the line-rate. This speed is possible due to minimized complexity stemming from the separation of the control plane from data plane [1]. To reduce system complexity, OpenFlow considers all network elements, including routers and switches, as simple, hardware-based packet forwarders. Complexity is thus shifted to the application layer, where software on the OpenFlow controller (a server with sufficient resources) makes various decisions and informs forwarders of the outcomes of these decisions. These outcomes are disseminated as flow-tables across the packet forwarders and define various pairs (match, action). This means that for each incoming packet, if there is a match in the flow-table of the local device, a special action is performed. Three standard actions are Forward, Encapsulate, and Drop. Forward makes the OpenFlow-enabled device act as a router/switch at the line-rate. If no match is found or if the packet is the first in a new, undefined flow, it is encapsulated and forwarded to the OpenFlow controller, where decisions subsequently are made. The packet can also be discarded through a drop action.

The network operating system (NOX) is a programmable interface that facilitates network management by providing an environment for running applications sitting on the OpenFlow controller. The OpenFlow controller communicates with the packet forwarders through the OpenFlow protocol over a secure SSL/TCP channel. With the aid of this open OpenFlow protocol, different routers' and switches' flow-tables can be programmed in a scalable manner. Entries in each flow-table on every OpenFlow packet forwarder are associated with different actions while statistics are being collected. **Fig. 1 (a)** depicts the separation of the control plane from data plane in OpenFlow architecture in addition to the flow-table structure. For instance, if the destination IP address of the incoming packet is equal to 1.1.1.1, the packet is forwarded to a given port. If it has 848 (UDP port for GDOI protocol) as the value for the source port, it will be encapsulated and then forwarded to the controller for further investigation. If the packet is IPsec encapsulating security payload (ESP) packet with type equal to 50, it will be dropped.

The first generation of OpenFlow packet forwarders, called "OpenFlow spec v1.0 conforming switches", defines flow header fields which encompass some features of each incoming packet as illustrated in **Fig. 1 (b)**. When a packet arrives, its header is firstly checked against the Match field and if the header matches any row in the flow-table, the corresponding action is performed. Any combinations amongst these demonstrated 10-tuple can be utilized to define and aggregate flows accordingly. These flow header fields are then exploited in order to specify matches in flow-tables for each incoming packet and perform the corresponding action. However, OpenFlow is currently unable to distinguish between IPsec flows. The authors in [1] emphasize the header fields of "OpenFlow spec v1.0 conforming switch" as the initial and standard header fields with which every OpenFlow switch must comply. This is substantial since later on we introduce our new flow header



▲ **Figure 1. (a) reveals the internal structure of OpenFlow architecture. In (b), flow header fields defined for "OpenFlow spec v1.0 conforming switches" (first generation OpenFlow packet forwarders) are shown. OpenFlow is currently unable to "distinguish between" IPsec flows.**

fields for the OpenFlow interface which is IPsec-aware and also backward compatible to "OpenFlow spec v1.0 conforming switch" header fields.

2.2 IPsec

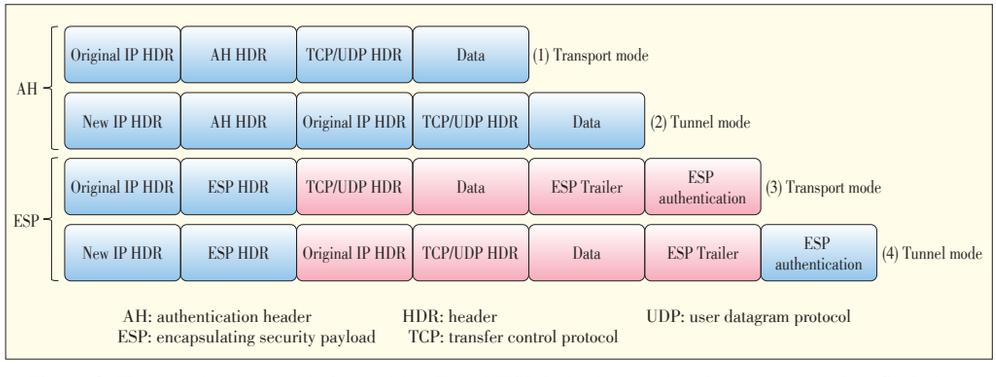
Working at the network layer, IPsec protects the traffic between peers by provisioning encryption as well as authentication from Layer 3 to Layer 7. On the other hand, all the current layer-2 technologies enable the IPsec framework function over them. The IPsec framework comprises five components. Available algorithm choices facilitated for each of these components result in different security solutions with each combination to satisfy various needs.

The first component highlights the IPsec protocol and can support either authentication header (AH) or ESP (protocol type 50 for ESP and 51 for AH).

Each IPsec protocol operates either in transport or tunnel modes. The encapsulations of the IP packet secured by IPsec with AH/ESP in both transport and tunnel operation modes are depicted in **Fig. 2**. Both protocols share provisioning authentication and integrity security services. Nevertheless, confidentiality is not considered in AH. This is crucial to differentiate segments of information which are encrypted and thus unreadable from other readable segments which can be meaningful for the third party in the middle of the conversation (i.e., openflow switch or controller). The second component demonstrates the choice for the encryption/decryption algorithm which pertains to the confidentiality service. As with every cryptographic system, the longer the key, the harder it is for an attacker to break into the IPsec communication. The third component ensures that the IPsec communication is not tampered with in transit and thus provides integrity. The fourth component facilitates authentication of the endpoints in secure communication via IPsec. The last building-block specifies the Diffie-Hellman (DH) algorithm group according to different needs. DH is a public key exchange mechanism that enables both communicating parties to come up with the same key over an unsecured channel. The driven shared key is used by peers for symmetric encryption as well as hashing through message authentication code (MAC) in the second and third components of IPsec, respectively.

In IPsec with AH, a message digest is formed by applying the hashing function to the original IP header and data payload utilizing the shared key. The digest then constructs a new AH header, which is injected into the original packet. The same calculation is performed in the receiving party to find the exact match of hashes. Nonetheless, all the data is transmitted in plaintext. Not considering any encryption mechanism leads to having all the layer-3 information in plaintext and therefore routable. However, the original IP header is also encrypted and unreadable when ESP is in tunnel mode. As we see later, this information in plaintext is immensely valuable for our method to differentiate various IPsec flows. Despite AH, in IPsec with ESP, encryption makes payload and the ultimate transmitters' identifications meaningless to eavesdroppers. Both the IP header and data payload are encrypted in this mode. This is followed by appending a new ESP header (as well as ESP trailer) and ESP authentication fields, including relevant encryption and authentication data to the original packet.

In Fig. 2, IPsec in transport mode merely considers the encapsulation of the data payload and transfer control protocol (TCP)/user datagram protocol (UDP) data (layer-4 and above). Nonetheless, tunnel mode suggests that the whole IP datagram is encapsulated within a new IP packet. On the other hand, while secure communication between gateways demands tunnel mode of the IPsec solution, transport mode facilitates host-to-host immune transmissions. While in AH, the original IP header remains unencrypted, and thus leaves the routing intact, the original TCP/UDP header is encrypted in ESP. Both protocols in different modes have their SPI in plaintext within the ESP/AH header. SPI differentiates various ongoing conversations at the receiving party.



▲**Figure 2. IPsec packet encapsulations with AH and ESP in both tunnel and transport modes; fields in red are encrypted and thus known only to end entities (i.e., not any third party in the middle of conversation including OpenFlow switch or controller). It is also noteworthy that both protocols in different modes have their SPI in plaintext within the ESP/AH header.**

2.3 Internet Key Exchange

The key exchange mechanism in IPsec is accomplished through IKE version 2 protocol [7]. The key exchange process with IKE finally leads to the construction of security association (SA) for IPsec. To establish an IPsec connection, IKE involves two phases. During these phases a set of messages is communicated, either in main mode or aggressive mode, resulting in the establishment of a secure channel between the peers. Phase 1 enables peers to agree on the security proposals generally as

well as the shared secret key and authenticate each other. Upon finalizing a secure tunnel in phase 1, phase 2 negotiates the custom security parameters between peers. On completion of phase 2, an SA is formed in a unidirectional manner. Each SA, as a logical connection, defines the way that the traversing traffic will be processed. Subsequently, the same security processing applies to the traffic associated with every SA.

Because a single SA specifies only two parties in a unidirectional manner, each party holds a security association data base (SADB) comprising multiple SAs, where each SA is associated with a different peer. SPI comprises an arbitrary 32-bit value utilized by a receiving party to differentiate the SA to which an incoming IPsec packet is associated. For a unicast communication, SPI on its own can specify an SA. Other parameters, such as the type of IPsec protocol can come along with SPI to highlight a unique SA. However, [8] emphasizes that the sufficiency of SPI on its own to determine an individual SA to which inbound traffic will be mapped or necessity to exploit other parameters in conjunction with SPI is a local matter. As we will see later, SPI can fall into a domain large enough to uniquely identify an SA. The following tuple illustrates the parameters any combination of which can be used to construct the primary key for SADB locally:

{SPI, IPsec Protocol Type (AH/ESP), Peer IP Address, Transform Set, Secret Key, SA Lifetime}

A combination of the elements in the vector above will shape SADB and determine various SAs stored on each peer.

2.4 GDOI

Group Encrypted Transport VPN solution[6], [9]–[10] with GDOI its heart is deemed to provide revolutionary and ultimate technology that reduces complexity and overheads pertaining to the need for scalable as well as secure transport remedy for always-on and dynamic connectivity of extremely integrated network sites spread over diverged domains. Any-to-any network connectivity is guaranteed to be end-to-end encrypted, authenticated and globally scalable for all applications namely voice, video and data with both unicast as well as multicast traffic. In other words, with the advent of GDOI architecture, the arduous obstacle of complexity pertaining to manageable as well as scalable VPN solutions for an abundance of fully-meshed sites (not only two endpoints) is not out of the question anymore[11].

GDOI as a cryptographic protocol for key management is based on IKE. While IKE ensures pairwise security associations between various peers, GDOI utilizing IKE phase 1 between each GMand a key server (KS) ends up with a single and common SA between all the GMs. Additional to pair-wise SAs with IKE phase 1, GDOI also "interprets" IKE to come up with a single SA for the group security domain. In other words, as the foundation of the GET VPN solution, GDOI defines IKE Domain of Interpretation (DOI). Utilizing UDP port 848, GDOI messages create, delete, and maintain SAs established between authenticated and authorized GMs. KS rekeys the group before current keys downloaded at the time of registration by GMs expire. As **Fig. 3** reveals, regardless of what the core network's technology is (WAN, MPLS, OpenFlow, etc.), each GM initially exchanges a GDOI Register message with KS which leads to downloading required keys and policies via bidirectional arrows. KS at some point in time before current keys expire pushes Rekey message which entails new policies as well as keys to given GMs via unidirectional arrows. In this way, encrypted multicast/unicast conduits are established amongst all GMs, not merely two endpoints, to communicate without any tunneling in place.

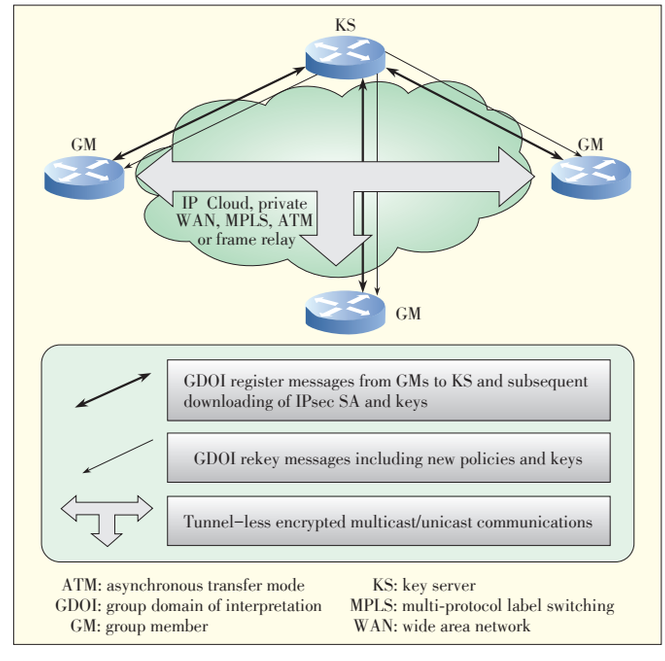
Tunnel-less but secure communication with GDOI for Transport VPN requires GMs to first dispatch registration queries to KS. With the aid of GDOI, KS authenticates and authorizes the given GM and sends back keying materials in addition to the IPsec policy needed for secure GM-to-GM(s) unicasting/ multicasting back to the given GM.

3 Proposed Method and Discussion

3.1 Integration of IPsec within OpenFlow Architecture

Increased control gained through custom forwarding of OpenFlow does enable different flows to be processed in different ways. OpenFlow is advantageous from this wide range of definitions for flows of any combination of header field defined for "OpenFlow spec v1.0 conforming switch" in section 2.

A can highlight a flow. However, when it comes to end-to-



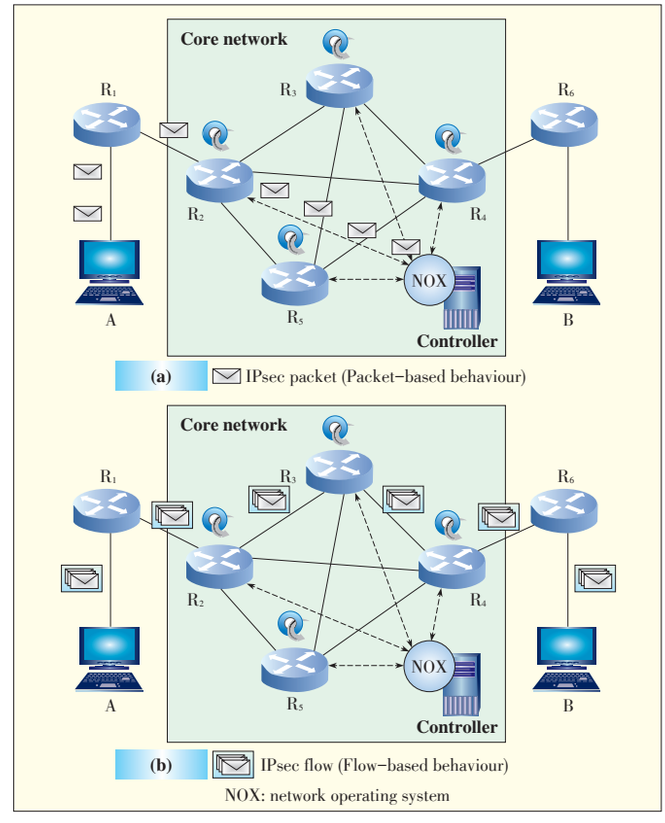
▲ **Figure 3. Upon downloading IPsec policies and keys from KS, GM is now registered with the "IPsec SA for the group" and can exchange unicast/multicast traffic securely with other GMs laying away the KS.**

end IPsec transmission, OpenFlow is unable to detect encrypted IPsec headers, which is discussed in section 2.2, and thus cannot aggregate them into a flow. The only exception is when OpenFlow filters the incoming packets to find a match for the IPsec protocol type, which is not sufficient to uniquely identify a flow because various but irrelevant entities might disseminate IPsec traffic for each other. Encrypted packet headers in IPsec act as a deterrent so OpenFlow switch treats them as a distinct flow [12]. With the standard header fields of "OpenFlow spec v1.0 conforming switch" today, the IPsec ESP-encrypted packets cannot be processed based on layer 4 and above information in both transport and tunnel modes. OpenFlow architecture also cannot deal with decrypting layer-3 information for IPsec with ESP in tunnel mode if any boundary packet forwarder peels the new IP header off before processing further for original layer 3 discovery and then delivery (when VPN tunnel terminates one hop before). To sum up, OpenFlow architecture is unable to aggregate flows of IPsec with ESP in both transport and tunnel modes because layer-3 and above information is encrypted and therefore unreadable for OpenFlow interface. Our method tries to find the distinguishing factor for uniquely identifying IPsec flows and directing these flows accordingly in order to replace the packet-based behavior of OpenFlow architecture towards IPsec with flow-based behavior. We argue that through our proposed method, in the OpenFlow environment we can overcome the abovementioned obstacles in the core network.

Fig. 4(a) shows the baseline scenario in which A tries to establish a secure communication with B via IPsec. It is possible that A acts as a remote access server which serves many clients or shares files with them via IPsec communication (can be KS in GDOI-like implementation). R_2, R_3, R_4 and R_5 form the core network elements in which OpenFlow architecture is employed. R_1 and R_6 can be thought of as security-aware gateways between which IPsec tunnel mode is constructed. In the transport mode of IPsec, they can be seen as local routers while end hosts address immune communication directly.

The dashed arrows indicate the conduits for the OpenFlow controller to securely talk to OpenFlow switches across the core network by OpenFlow protocol. Without our proposed method, IPsec packets from endpoint A to B in the figure reaching R_2 cannot be treated as a flow and should be sent to the OpenFlow controller one by one for decision-making if they are encrypted with ESP (unreadable layer 3 and above information). This will degrade network performance and impose a huge processing burden on the OpenFlow controller within the core network. This is because each IPsec packet is treated with packet-based behavior by being encapsulated and sent to the OpenFlow controller for decision-making one by one. Our goal is to aggregate IPsec packets associated with each secure communication and forward them as flow satisfying arbitrary routing policies of the core network for instance. This might be the case if in an attempt to assign a specific physical route which highly considers security countermeasures and thus is more trustworthy for the IPsec communications (or other traffic engineering tasks such as seeking more available bandwidth), IPsec flows are separated from other flows and then forwarded through this route. Another use case as we will discuss is when more than two endpoints as group members participate in secure group communications over IPsec via GDOI.

Fig. 4(b) shows that R_2 through our method will eventually separate IPsec flow from other incoming traffic sent by R_1, such as http, and direct it via capable and highly trustworthy R_2-to-R_3-to-R_4 links to R_6 as the egress point. In packet-based behavior of OpenFlow architecture, encrypted packets must be encapsulated and then traverse the OpenFlow controller one by one for further processing. Nevertheless, we aim to aggregate IPsec traffic at R_2 and treat it as a flow without involving the OpenFlow controller's resources for processing each packet individually. Specifically, while the problem was that when packets are encrypted using ESP, the flow identifiers are encrypted and hence cannot be used to distinguish flows, we propose using the SPI of IPsec within the OpenFlow architecture as the distinguishing factor for uniquely identifying IPsec flows



▲**Figure 4. (a): Each IPsec packet is treated with packet-based behavior by being encapsulated and sent to the OpenFlow controller for further decision making one by one. (b): Flow-based behaviour through our proposed method, aggregation of given IPsec traffic along with its separation from other IPsec traffic in the core network have been accomplished.**

and directing these flows accordingly.

The functionality of our design is irrespective of IPsec modes or protocols. This makes the remedy flexible enough to cope with all four different encapsulations (Fig. 2). However, because the security database (SDB) construction on the OpenFlow controller is slightly different in transport mode than in tunnel mode, we bring two scenarios here for different modes. The design needs to consider the fact that network elements in the core network are simple packet-forwarders that are security-unaware (backward compatible to "OpenFlow spec v1.0 conforming switches"). In other words, we cannot expect any cryptographic processing on these OpenFlow switches. They are only capable of finding a simple match for each incoming packet against their flow-table and taking a particular action, like forwarding, and subsequent packets of the same match accordingly to treat them as a flow.

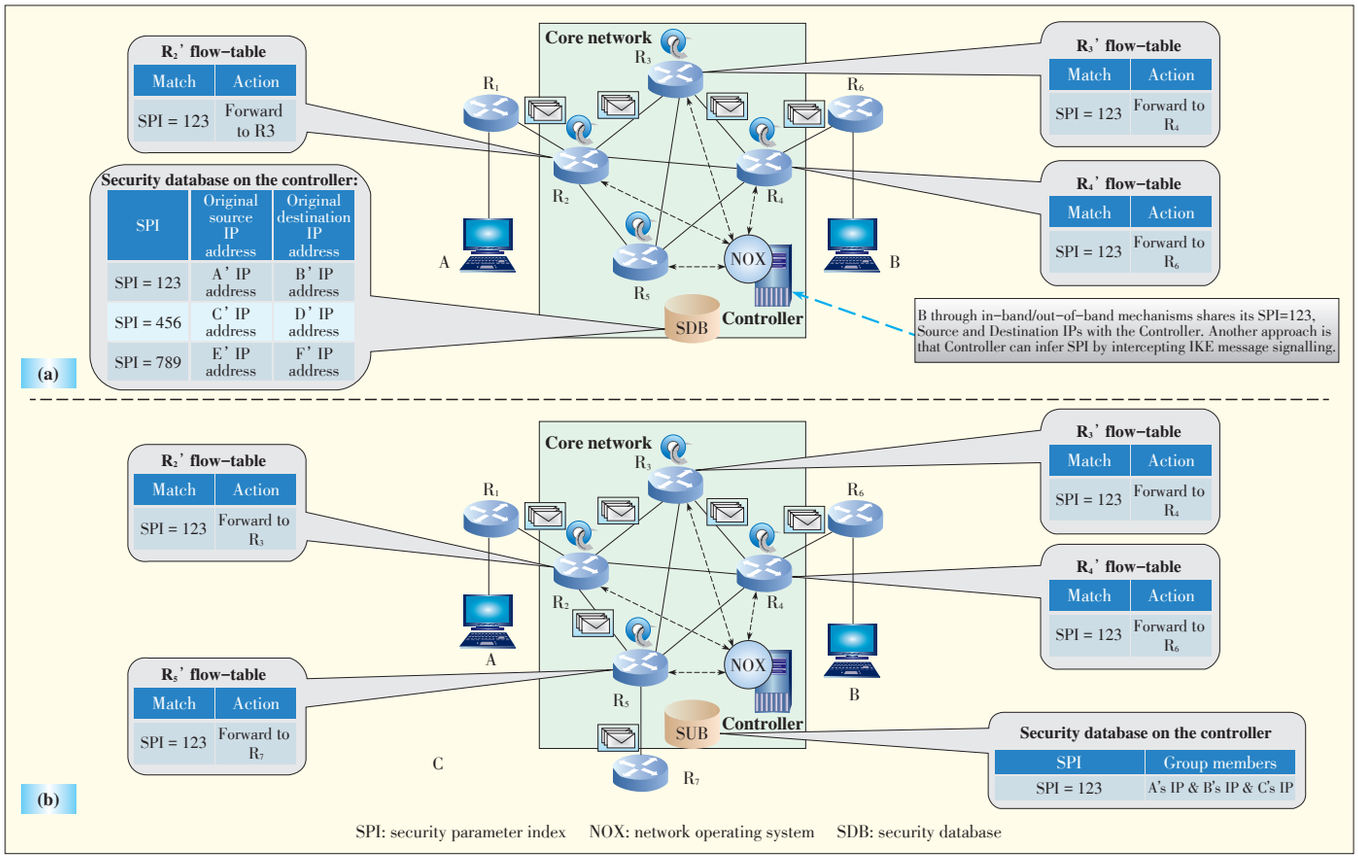
However, this flexibility acquired through the simplicity of OpenFlow architecture cannot distinguish "between" IPsec flows, which is now needed to adapt to secure group communication in GDOI-like architecture for instance. This is due to the fact that the distinguishing factor (if residing in layer-3 or above) is encrypted in the ESP protocol. Each incoming packet encrypted by IPsec with ESP needs to be forwarded to the OpenFlow controller if any information above the IP layer is required for flow-table match-finding.

3.2 Considerations for ESP in Transport Mode

In section 2.3, the first set of messaging between end-devices forms the secure channel over which the transmitters communicate. Once the agreement by end-devices has been reached (IKE phase 2 finished), SAs are established separately for each direction by A as well as B and stored locally in their SADBs. Here, we consider IKE negotiations between endpoints irrespective of the proposed method because SAs need to be constructed prior to treating secure IPsec communication as a flow. Once SAs are established via IKE, the first IP datagram containing the actual secure data onwards can be handled with the proposed design as a flow. Finding a match for header fields listed in Fig. 1b for IPsec on an OpenFlow switch and forwarding based on that fails because end-to-end secure communication ensures that the transmission is unreadable to any entity in the middle when it is ESP for layer-3 and above [12]. On the other hand, these fields are considered as assets accessible only to end-entities who might be reluctant to share them with third parties. The OpenFlow controller initially determines each flow with the aid of the first packet of the communication. This is reasonable because in the beginning, the flow-table has no entry of the flow information before launching the communication. Nevertheless, for IPsec flows, the relevant information is an asset (secret) and thus only both ends have access to it. Because SAs are formed in each direction, each end device is responsible for sharing the required information (here SPI) with the OpenFlow controller prior to traversing the actual flows. Another approach is to let the controller itself infer the SPI because the controller can intercept all IPsec session setup traffic and learn the SPI used between hosts. The SPI acts like a cookie for IPsec where for A-to-B secure communication (two ends, not a group), B firstly determines the SPI value for A-to-B SA and announces it via IKE to A who then carries it in its header field (either AH or ESP header) of IPsec packet(s) to B in plaintext. Consequently, in theory, either A as the data originator should share the received SPI specified by B to the OpenFlow controller or the controller itself infers it directly by intercepting IKE messages.

Upon establishing each SA, the end-device populates its SADB table locally with the relevant security related information. So far, only A and B in Fig. 4 are aware of the security credentials pertaining to IPsec communication between one another. Each SA in each direction can be associated with an SPI number. Subsequently, 2^{32} different SAs can theoretically be established and differentiated between two end-hosts on each site. The SPI is the same for different sequence numbers of the same IPsec communication in a unidirectional manner, and this makes it an appropriate candidate as well as a distinguishing factor among various flow header fields in the design (with more than two entities, SPI also remains the same within a group domain in GDOI). To sanitize it more, bear in mind that IPsec is an immune communication from one sender to another receiver in a one-way direction in which the relevant SA is associated with an SPI carried within AH/ESP headers in plaintext. As a result, for the receiving party, this SPI determines the corresponding SA and thus how the IPsec packet (and resultant flow) will be processed based on the security policy already agreed on mutually via IKE.

Back to **Fig. 5(a)**, we suggest that B shares the SPI with the OpenFlow controller either through in-band (if controller intercepts IKE messages and infers SPI base on them) or out-of-band channels for secure transmission A-to-B before disseminating the actual data. A might have big data and be willing to transmit it in a secure manner to B for instance. The dashed blue arrow reveals the process of handing out the SPI to the OpenFlow controller. Our method requires a SDB on the OpenFlow controller. This SDB contains security related information for IPsec communications. The amount of security credentials shared with the OpenFlow controller is in the end-host's hands. However, our design emphasizes that for flow-based behaviour towards IPsec within OpenFlow architecture, SDB should be populated with SPI at least. In IPsec transport mode, original layer 3 information is also added. Upon sharing SPI with OpenFlow controller by B, the OpenFlow controller must perform an existence check against SDB looking for the announced SPI. If duplicated SPI coexists, the OpenFlow controller should use original layer 3 information as complementary to SPI to uniquely identify the IPsec conversation and update the packet forwarders on the way accordingly. Next, we introduce our new flow header fields for OpenFlow interfaces on the



▲**Figure 5.(a) End-entity B shares its SPI with the controller through in-band/out-of-band mechanisms to add the ability to 'distinguish between' IPsec flows. (b) A, B & C form a group for secure communication in a multicast-like manner, from A to both B and C for instance, based on GDOI.**

switches which contain the new field "SPI" in tuple below in addition to that already mentioned Fig. 1b: {Forwarding Port, VLAN ID, Source MAC, Destination MAC, Ethernet Type, Source IP, Destination IP, IP Protocol, Source TCP, Destination TCP, SPI}

The SPI in plaintext is carried within AH/ESP headers. Therefore, OpenFlow switches are able to detect it directly. The addition of the SPI header field is backward compatible with OpenFlow spec v1.0 conforming switches and does not deem that network elements have any cryptographic capabilities and thus is scalable at the minimum cost.

In a similar way to Fig. 5(a), with **Fig. 5(b)**, A, B and C form a group for secure communication in multicast from A to both B and C with the same method in a GDOI-like manner. The group is associated with SPI = 123 and OpenFlow forwarders are updated accordingly. R_2 now forwards the incoming packets with SPI = 123 to both R_3 and R_5 to form the IPsec flow for the group under the common SA.

3.3 Considerations for ESP in Tunnel Mode

The main difference is that in tunnel mode the original layer 3 information is itself encrypted. Consequently, the OpenFlow controller stores new IP source and destination information in addition to SPI within its SDB at the minimum. This information is needed in case the same SPI has been already installed within SDB and thus more information is required to uniquely identify an IPsec flow. In our scenario, the OpenFlow controller now makes the decision to forward IPsec flows fulfilling its local routing policy and goals by updating appropriate switches' flow-tables while the end to end security is still guaranteed. However, in addition to other header fields, SPI will now also be included for determination of IPsec flows.

To sum up, as the flow header fields defined for OpenFlow spec v1.0 conforming switches indicate in Fig. 1(b), some original flow identification information such as TCP/UDP headers become unavailable with IPsec ESP encrypted traffic for in-path OpenFlow switches to identify/distinguish. With the aid of SPI, which is unencrypted but authenticated in ESP Tunnel Mode, for example, we propose that in-path OpenFlow switches should not only read SPI but can also differentiate IPsec flows accordingly. Using SPI information for classification requires the architecture to embed a mechanism to notify the controller of updates on the SPI values through either in-band or out-of-band mechanisms, such as interpreting IKE negotiations (this can be done prior to actual end-to-end secure communication or through interpreting the first packets of a given IPsec

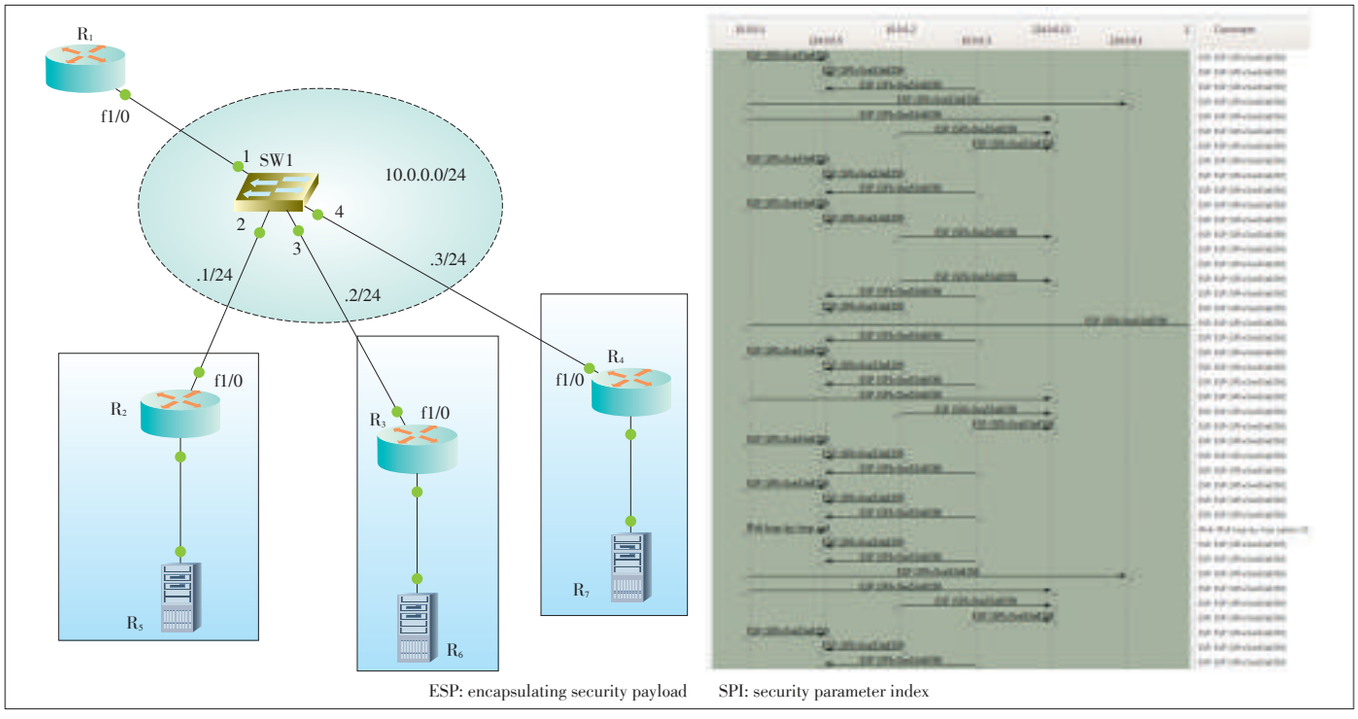
flow by the controller). No matter which, the controller using this mechanism needs to update in-path OpenFlow switches of the given SPI so that it can be read and interpreted for the broad range of intended flow definitions, presuming that our new flow header fields are in place for OpenFlow interfaces on the switches which contain the new field called "SPI" (in addition to the ones defined for OpenFlow spec v1.0 conforming switches displayed in Fig. 1(b)).

Besides the merits achieved by integrating IPsec flows into OpenFlow architecture such as secure group communications based on GDOI standard as discussed in section 4, classifying/policing/shaping IPsec flows let us meet different end-to-end QoS goals in networks as [12] also points out. For instance, usage of SPI within our proposal here enables the OpenFlow switch to perform class-based queuing (CBQ) whereby cryptographically protected traffic among different applications, users (or groups), user affiliations and so forth can be distinguished with an improved level of granularity. Remember that CBQ also accomplishes priority queuing where the preference with which the flows are serviced (can be reliant on service level agreements (SLAs) between different domains for instance) as well as the amount of the queued traffic for them are determined.

4 Use Case: Secure Group Communication Based on GDOI

Traditionally, point-to-point tunnels between VPN gateways were used to carry authenticated and encrypted traffic from one site to another (two ends). For secure group communication with GDOI, encryption/authentication is separated from transport. The merit of this is that secure communication between various sites (more than two) is possible without any tunnels between these branches. This does open the door also for the OpenFlow architecture in the core network eliminating any need for crypto functionality to address transport requirements.

In **Fig. 6** (left), we tried to emulate through Cisco infrastructure [13] GDOI between three nodes, namely, R_2, R_3 and R_4 (can be thought of as A, B and C in Fig. 5(b)) as the GMs. R2, R_3 and R_4 with assigned IP 10.0.0.1/24, 10.0.0.2/24 and 10.0.0.3/24 (all on one subnet), respectively, form a group looking for secure communications through GDOI. R1 will play the role of KS in there. It is likely that the OpenFlow controller serves as the KS. SW1 will represent the core network, which is OpenFlow equipped with our method to respect distinct IPsec flows. GDOI can operate over all the core technologies and therefore must remain infrastructure-independent. The objective here is to eavesdrop on the SW1 after proper GDOI implementation between R_2, R_3 and R_4 via Wireshark to infer the SPI associated with this group domain. Wireshark Flow Graph (Fig. 6). captures all the encrypted communications on the subnet within the group (10.0.0.0/24) after GDOI implementation showing that all the group members share the same SPI for IPsec ESP for the group domain communications. SW1 is required to respect our method through the ability to "distinguish between" IPsec flows using SPI in order to integrate the



▲**Figure 6. Left: baseline scenario in GNS3; R_2, R_3 and R_4 are willing to form a group based on GDOI. Right: Wireshark Flow Graph highlights the captured SPI. The same SPI is used amongst all the group members for secure group communications after proper GDOI implementation.**

notion of secure group communication within SDN. Despite AH, in IPsec with ESP, encryption makes payload and the ultimate transmitters' identifications meaningless to the eavesdroppers. This highlights the main use case for our method within OpenFlow. R_2, R_3 and R_4 were already coded for multicast OSPF as well as PIM to generate some multicast traffic before and after GDOI implementation to highlight the role of this IPsec-based group control protocol. As Fig. 6 (right) reveals, upon finishing GDOI implementation, all the communications originating from GMs (R_2, R_3 and R_4) destined for any multicast address including 224.0.0.5 (for multicast OSPF) or 224.0.0.13 (for PIM multicast) are secured with IPsec ESP while all the communication within this group domain is sharing the same SPI.

5 Conclusion and Future Work

In this paper, we have addressed the deficiency for interworking of OpenFlow with IPsec in both IPsec tunnel as well as transport modes. OpenFlow architecture cannot aggregate flows of IPsec with ESP because layer-3 and above information is encrypted and therefore unreadable. In this paper, we have proposed using the SPI of IPsec within the OpenFlow architecture in order to uniquely identify IPsec flows and direct these flows accordingly. This replaces packet-based behavior of OpenFlow architecture towards IPsec with a flow-based behavior and removes the obstacle of encrypted flow identifiers. We also proposed new flow header fields for OpenFlow switches/interfaces which contain SPI for switching IPsec flows. Sharing SPI with the OpenFlow controller will not jeopardize the immunity of end-to-end IPsec conversation because they are already in plaintext. The proposed method facilitates the ability to distinguish between IPsec flows in order to integrate secure group communication into the OpenFlow architecture. The main use case where identifying "between" IPsec flows can be useful is when secure group communication is required in a similar way to GDOI architecture as discussed.

We will carry out further works on simulating the proposed method in order to evaluate its scalability as well as the performance in the next step based on [14].

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, pp. 69–74, 2008. doi: 10.1145/1355734.1355746.
[2] H. Hu, J. Bi, T. Feng, S. Wang, P. Lin, and Y. Wang, "A Survey on New Architecture Design of Internet," in *Computational and Information Sciences (ICCIS), 2011 International Conference on*, Chengdu, China, 2011, pp. 729–732. doi: 10.1109/ICCIS.2011.57.
[3] *Security Architecture for the Internet Protocol*, RFC 4301 (Proposed Standard), 2005.
[4] *The Group Domain of Interpretation*, RFC 6407 (Proposed Standard), 2011.
[5] *The Group Domain of Interpretation*, RFC 3547 (Proposed Standard), 2003.
[6] Y. Bhaiji, *Network security technologies and solutions*. Indianapolis, IN: Cisco Press, 2008.
[7] *Internet Key Exchange Protocol Version 2 (IKEv2)*, RFC 5996 (Proposed Standard), 2010.
[8] *IP Encapsulating Security Payload (ESP)*, RFC 4303 (Proposed Standard), 2005.
[9] S. Wilkins and F. H. S. III, *CCNP security SECURE 642–637 : official Cert guide (master CCNP SECURE 642–637 exam topics ; assess your knowledge with chapter-opening quizzes ; review key concepts with exam preparation tasks ; practice with realistic exam questions on the CD-ROM)*. Indianapolis, Ind: Cisco Press, 2011.
[10] K. Hutton, M. Schofield, and D. Teare, *Authorized self-study guide : Designing Cisco network service architectures (ARCH)*. Indianapolis, IN: Cisco Press, 2009.
[11] *CCIE security practice labs*. Indianapolis, Ind: Cisco Press, 2004.
[12] V. Fineberg, "A practical architecture for implementing end-to-end QoS in an IP network," *Communications Magazine, IEEE*, vol. 40, no. 1 pp. 122–130, 2002. doi: 10.1109/35.978059.
[13] *Graphical Network Simulator*[Online]. Available: http://www.gns3.net/
[14] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Monterey, California, 2010. doi: 10.1145/1868447.1868466

Manuscript received: 2014-01-25

Biographies

Vahid Heydari Fami Tafreshi (v.fami@surrey.ac.uk) received his BSc in computer software engineering from Shomal Higher Education Institute, Iran, in 2007. He received his Cisco Certified Network Associate (CCNA) and Cisco Certified Network Associate Security (CCNA-Security) certificates from the Cisco Academy at London Metropolitan University, UK, in 2009. He received his MSc in internet computing from the University of Surrey, UK, in 2010. He is currently working and pursuing a PhD degree at the Centre for Communication Systems Research (CCSR), Department of Electronic Engineering, University of Surrey, UK. His main research interests include internet protocols and architecture, network security and multicasting.

Ebrahim Ghazisaeedi (eghazisaeedi@sce.carleton.ca) received his MSc degree in Mobile and satellite communications from the University of Surrey, UK, in 2011. He is currently pursuing a PhD degree in electrical and computer engineering at the Department of Systems and Computer Engineering, Carleton University, Canada. His main research interests are in communication networks, network virtualization, and network optimization.

Haitham Cruickshank (h.cruickshank@surrey.ac.uk) is a senior lecturer at the University of Surrey. He has worked there since January 1996 on several European research projects in the ACTS, ESPRIT, TEN-TELECOM and IST programmes. His main research interests are network security, satellite network architectures, VoIP and IP conferencing over satellites. He also teaches data and Internet networking and satellite communication courses at the University of Surrey. He is a member of the Satellite and Space Communications Committee of the IEEE ComSoc and a chartered engineer and corporate member of the IEE in the UK.

Zhili Sun(z.sun@surrey.ac.uk), Chair of Communication Networking, has been with the Centre for Communication Systems Research (CCSR), Department of Electronic Engineering, Faculty of Engineering and Physical Sciences, University of Surrey since 1993. He got his BSc in Mathematics from Nanjing University, China, in 1982, and PhD in Computer Science from Lancaster University, UK, in 1991. He worked as a postdoctoral research fellow with Queen Mary University of London from 1989 to 1993. He has been principle investigator and technical co-coordinator in many projects within the EU framework programs, ESA, EPSRC and industries, and has published over 125 papers in international journals, book chapters and conferences. He has published a book as sole author titled "satellite networking–principles and protocols" by Wiley in 2005, a book as contributing editors of "IP networking over next generation satellite systems" published by Springer in 2008, and another book as contributing editor to the 5th edition of the text book "Satellite Communications Systems–systems, techniques and technology" published by Wiley in December 2009. His research interests include wireless and sensor networks, satellite communications, mobile operating systems, traffic engineering, Internet protocols and architecture, QoS, multicast and security.

DOI:10.3939/j.issn.1673-5188.2014.02.008
http://www.cnki.net/kcms/detail/34.1294.TN.20140527.0911.001.html, published online May 27, 2014

Virtualized Wireless SDNs: Modelling Delay Through the Use of Stochastic Network Calculus

Lianming Zhang [1], Jia Liu [1], and Kun Yang [2]
(1. College of Physics and Information Science, Hunan Normal University, Changsha 410081, China;
2. Network Convergence Laboratory, University of Essex, Colchester CO4 3SQ, United Kingdom)

Abstract

Software-defined networks (SDN) have attracted much attention recently because of their flexibility in terms of network management. Increasingly, SDN is being introduced into wireless networks to form wireless SDN. One enabling technology for wireless SDN is network virtualization, which logically divides one wireless network element, such as a base station, into multiple slices, and each slice serving as a standalone virtual BS. In this way, one physical mobile wireless network can be partitioned into multiple virtual networks in a software-defined manner. Wireless virtual networks comprising virtual base stations also need to provide QoS to mobile end-user services in the same context as their physical hosting networks. One key QoS parameter is delay. This paper presents a delay model for software-defined wireless virtual networks. Network calculus is used in the modelling. In particular, stochastic network calculus, which describes more realistic models than deterministic network calculus, is used. The model enables theoretical investigation of wireless SDN, which is largely dominated by either algorithms or prototype implementations.

Keywords

wireless software defined networks (SDN); wireless network virtualization; QoS modelling; upper bound delay; stochastic network calculus

1 Introduction

Software-defined networks (SDN) have attracted much attention recently because they enable flexible network management [1], [2]. The bulk of research on SDN has focused on wired networks and OpenFlow [3], [4], but there is an increasing tendency to introduce SDN into wireless networks [5], [6]. SDN brings to wireless networks the same benefits it brings to wired networks, e.g., separation of control and forwarding planes, but it also creates some radio-specific issues [5].

One of the key enabling technologies of SDN is network virtualization. Wireless mobile network virtualization enables physical mobile network operators (PMNO) to partition their network resources into smaller slices and assign each slice to an individual virtual mobile network operator (VMNO). These virtual networks are the managed in a more dynamic, cost-effective way. We call these virtualized individual networks virtual wireless networks (VWNs). VMNOs pay the PMNO using a pay-as-you-use model. Wireless network virtualization has its real-world bearings in mobile cellular networks. Wen et al. summarise some current trends and perspectives in wireless virtualization [7].

The purpose of network virtualization is to provide services to end users. To satisfy user requirements and abide by the service-level agreement with the customer, virtual network operators need to provide quality of service (QoS) in their networks. One important QoS metric is network delay, which is critical for real-time services such as voice. In this paper, we address the delay requirements of different services (flows). A key issue is how a PMNO allocates resources to an individual VMNO in order to satisfy service delay requirements within the VMNO. Guarantee that this delay requirement will be met in a VWN is a challenge to mobile network operators [14],[15].

Before allocating or scheduling resources, it is essential to understand the behaviours of virtual networks, especially in terms delay bounds. Although much work has been done on modelling physical wireless networks themselves, little has been done in the way of modelling virtual networks. Some initial work in this area can be found in [8], but the network being considered is a mesh network. Furthermore, this model does not differentiate physical networks from virtual networks.

As in [9], we partition a physical network node, such as a base station, into multiple slices. This partitioning can be carried out in a dynamic manner using software, i.e., supporting software-defined radio networks. Each slice represents a virtual network node.

The predominant theoretical bases for network modelling

are probability theory and queue theory. In this paper, we use a new modelling tool called network calculus, which is a set of recent developments that enable the derivation of performance bounds in networking [10], [11]. Applications of network calculus are wide-ranging and include QoS control, resource allocation and scheduling, and buffer and delay dimensioning [10]. We have previously researched the use of network calculus in wireless sensor networks [12]. Our recent work [13] extends on this, moving into the new area of wireless network virtualization but using deterministic network calculus. Deterministic network calculus cannot describe service flow distribution or characteristics and thus cannot realistically model real-world scenarios. In this paper, we go one step further and use stochastic network calculus, which enriches the expressiveness of the service flow and is thus a more realistic modelling tool.

The technical aim of this paper is to propose a delay model for VWN under a more realistic service flow model using stochastic network calculus. This paper makes the following main contributions:

- It mathematically describes the different roles of a typical VWN system using stochastic network calculus.
- It describes a delay model for the above virtual wireless network by expressing network delay in its upper bound and in a closed-form manner. The proposed model can help analyze delay guarantee for per-flow granularity.

We do not consider a particular networking technology, such as Wi-Fi or LTE-A. The proposed model is generic enough to be applicable in any network.

2 Related Work

SDN, represented by OpenFlow, has been successful for innovating on network operations and service provisioning. It also reduces complexity in terms of network configuration and management. Costanzo et al. identify the benefits of SDN for wireless and mobile communications, although their exemplar is a wireless personal area network [6].

Network virtualization is a strong enabler of wireless SDN because it provides a flexible, efficient way of deploying customized services on a shared infrastructure [14], [15]. Recently, wireless virtualization has attracted attention because of it benefits in several scenarios [9], [16], [17].

A lot of research has been done on wireless network virtualization [9], but there is a lack of formal modelling of wireless virtual networks. System modelling can be a useful means of studying the fundamental features of a system. In this paper, we aim to fill this gap by providing a model for virtualized wireless networks. In particular, we focus on one important feature of virtual networks: network delay.

There are various approaches to delay-aware resource control in wireless networks. Tao et al. investigate the resource-allocation problem in a multiuser OFDM system with both delay-constrained and non-delay-constrained traffic [18]. However, they do not discuss the affect of the delay mechanism on performance. Another approach is to convert average delay constraints into equivalent average rate constraints using queuing theory [19], [20]. These approaches are linked to a particular resource-allocation or packet-scheduling algorithm and are thus specific to the corresponding algorithms. We provide a more generic model of wireless virtual networks that is agnostic to resource-allocation algorithms and specific network technology. In our recent work [13], we describe a more expressive network-modelling tool, called stochastic network calculus.

Stochastic network calculus is used to analyze performance guarantee in information systems [21], [22]. It has its foundations in the min-plus convolution and max-plus convolution queuing principles, and it has tremendous potential in dealing with queuing-type problems. It complements classical queuing theory [21]. In [22], Ciucu et al. discuss sharp bounds in stochastic network calculus. Stochastic network calculus has can be used to compute per-flow queuing system metrics in a unified manner for a large class of scheduling algorithms. Furthermore, the per-flow results can be extended in a straightforward manner, from a single queue to a large class of queuing networks that are amenable to convolution-form representation in an appropriate algebra.

Here, we summarize representative work in which network calculus is used to model QoS parameters, in particular, delay. In [23], network calculus is used to compute the delay of individual traffic flows in feed-forward networks under arbitrary multiplexing. In [24], the maximum end-to-end delay is calculated, again for feed-forward type networks. In [25], Schmitt et al. propose an analytical framework for analyzing worst-case performance and to dimension resources in a sensor network. In [26]–[28], the authors present research on the deterministic performance bound on end-to-end delay for self-similar traffic regulated by a fractal leaky bucket regulator in an ad hoc network [26], wireless sensor network [27], and wireless mesh network [28]. Working with the concept of flows and micro-flows, Zhang et al. [12] use arrival curves and service curves in network calculus to propose a two-layer scheduling model for sensor nodes. The authors develop a guaranteed QoS model that includes upper bounds on buffer queue length, network delay, and effective bandwidth. In [29], Azodolmolky et al., describe the functionality of the SDN switch and controller and present an analytical model, based on network calculus theory, for delay and queue length boundaries of the SDN switch and buffer length of the SDN controller and SDN switch. In [6], Costanzo et al. present a complete SDN for wireless personal area networks and call it software-defined wireless network (SDWN).

3 Description of System Model

Fig. 1 shows a virtual wireless network with virtual queue, the benefit of which is described in [13]. Each slice is allocated a virtual queue in the hosting physical network or network

node. All these virtual queues share the data rate capacity of the physical network node, i.e., the physical BS under the control of a scheduler. The scheduler takes into account the QoS requirements of the slices when scheduling resources. Each slice, denoted S_1, S_2, ... , S_n, represents a virtual base station.

Fig. 1 highlights the following two key elements in a virtualized network: physical BS and virtual BS (i.e., slice). Each slice represents a virtual mobile network (VMN) and has a slice ID. A slice is used by many end users, i.e., u_1, u_2, $u_3 \cdots u_n$. A user is physically represented by a mobile node in the network and may have multiple flows, i.e., $F_{1,1}$, $F_{1,c1}$, $F_{n,1}$, $F_{n,cn}$. For example, the smart phone may be used to check emails while listening to music online. Here, email and music each represents a flow $F_{n,cn}$. The biggest differentiator between flow types is the delay requirement. Voice flow has more stringent delay requirements than non-real time emails. A flow represents a session, and each flow has an ID.

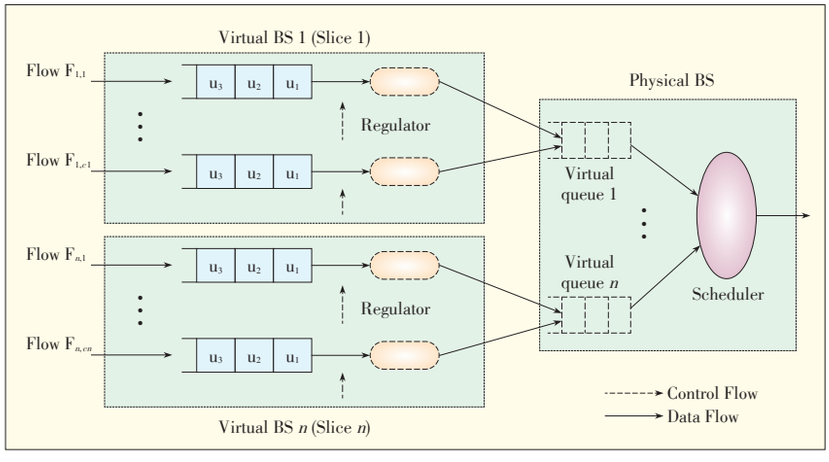
Fig. 1 shows the relationship between the four key elements in a WVN: physical BS, virtual BS, users, and flows. Packets from different users and of the same type (e.g., real-time) are denoted U_i and are put into the same queue in a slice. A leaky bucket source model is used to regulate the flows of each slice queue because this model is simple and practical. A leaky bucket regulator is applied to each slice queue to both regulate the flows so that non-rule flows can be controlled in certain conditions. A flow regulated by the leaky bucket regulator is given by envelope $\alpha(t)$ [12]:

$$\alpha(t) = r \cdot t + b, \qquad \forall t \geq 0 \tag{1}$$

where b is the burst parameter, r is the average arrival rate, and t is time.

4 Proposed Upper Bound Delay Modelling Using Stochastic Network Calculus

In this section, we describe the above wireless SDN system using stochastic network calculus. Then we deduce the delay upper bound for this wireless SDN model. Detailed information about stochastic network calculus can be found in [21], [22], and network calculus in general is described in [10]. Notations used in this paper are listed in **Table 1**.



▲**Figure 1. Virtual wireless network and its roles.**

4.1 System Description Using Stochastic Network Calculus

We model the wireless SDN presented in section 3 using stochastic network calculus. The process of the model is as follows: First, a flow enters a virtual BS and is regulated by a leaky bucket regulator (1). The arrival curve is denoted $\alpha(t)$. Second, we assume a first come, first served (FCFS) strategy for a queue. This is reasonable because the packets in the same queue are of the same service type. Other more comprehensive queuing strategies may be applied here as well. Finally, the aggregated flows from a slice are scheduled in the same way that a generalized processor sharing (GPS) server would schedule them in a physical BS [30]. The system is further explained as follows:

$$\frac{\beta_i(t)}{\beta_j(t)} \geq \frac{\mu_i}{\mu_j}, j = 1,2,...,N \tag{2}$$

$$\beta(t) \cdot \rho = R \cdot (t - T) \cdot \rho = \sum_{j=1}^{N} \beta_j(t) \tag{3}$$

$$\beta_j(t) \leq \alpha_j(t), \ 1 \leq j \leq N \tag{4}$$

$$\alpha_i(t) = \sum_{k=1}^{c_i} \alpha_{i,k}(t) \tag{5}$$

▼**Table 1. Notations**

r	average arrival rate of flows
b	burst parameter of flows
$\alpha(t)$	arrival curve of a flow passing through a physical BS
$\beta(t)$	service curve ofphysical BSs
$\beta_i(t)$	service curve of slice i
μ_i	weight of slice i
ρ	network bandwidth utilization
N	number of slices
R	service rate of physical base stations
T	latency of physical BS
c_i	number of flows in slice i
$\alpha_{i,j}(t)$	arrival curve of the flow j in slice i
D_i	lower bound on network delay of slice i
d_i	given network delay of slice i
$r_{i,j}$	average arrival rate of flow j in slice i
$b_{i,j}$	burst parameter of flow j in slice i
$inf\{\}$	maximum lower bound
$sup\{\}$	minimum upper bound
$exp\{\}$	exponential function
$\Pr[X \geq x]$	probability of $X \geq x$

The parameters in (2) to (5) are shown in Table 1. The flows of the slice i obtain the bandwidth weight. The sum bandwidth of the slices is, at most, the total bandwidth of the physical BS. Each slice entering the physical BS has a certain service curve that is not only decided by the total service curve of the physical BS scheduler but also the arrival curve of the slice.

4.2 Proposed Delay Model

Proposition 1: In an interval [0, t], the least stochastic upper delay bound of physical BSi can be computed using

$$\begin{aligned}\Pr[D_i(t)\geq d_i]\leq \inf_{\delta\geq 0}\Bigg\{\Big\{\exp\{-\delta\varepsilon_i\rho R\cdot(t-T+d_i)\}\cdot \prod_{k=1}^{c_i}\left[\frac{r_{i,k}\cdot t}{r_{i,k}\cdot t+b_{i,k}}\cdot\left(\exp\{\delta\cdot(r_{i,k}\cdot t+b_{i,k})\}-1\right)+1\right]\cdot \prod_{j=1}^{i-1}\prod_{k=1}^{c_i}\left[\frac{r_{j,k}\cdot(t+d_i)}{r_{j,k}\cdot(t+d_i)+b_{j,k}}\cdot \left(\exp\{\varepsilon_i\delta\cdot(r_{j,k}\cdot(t+d_i)+b_{j,k}\}-1\right)+1\right]\Bigg\}\end{aligned} \tag{6}$$

The symbol $\Pr[D_i(t)\geq d_i]$ represents the probability that the delay of the flows passing through the slice i is greater than d_i. When the value of the right side of (14) is at the minimum, we obtain parameter δ. The other parameters are shown in Table 1. Service rate and latency are the two key parameters of the physical BS; the former is equivalent to the network bandwidth, and the latter is the maximum service delay of the physical BS.

Proof: We can derive (7) from (2) and (3):

$$\begin{aligned}\beta(t)\cdot\rho=\sum_{j=1}^{N}\beta_j(t)=\sum_{j=1}^{i-1}\beta_j(t)+\sum_{j=i}^{N}\beta_j(t)\leq \sum_{j=1}^{i-1}\beta_j(t)+\sum_{j=i}^{N}\left(\frac{\mu_j}{\mu_i}\beta_i(t)\right)\leq\sum_{j=1}^{i-1}\beta_j(t)+\frac{1}{\varepsilon_i}\beta_i(t)\end{aligned} \tag{7}$$

From (7), we have

$$\beta_i(t)\geq\varepsilon_i\left(\beta(t)\cdot\rho-\sum_{j=1}^{i-1}\beta_j(t)\right) \tag{8}$$

where $\varepsilon_i=\dfrac{\mu_i}{\sum_{j=1}^{N}\mu_j}$.

From (10) and (17) in [31], we obtain

$$E\left[\exp\{\delta\alpha_{i,k}(t)\}\right]\leq\frac{r_{i,k}\cdot t}{\alpha_{i,k}(t)}\cdot\left(\exp\{\delta\alpha_{i,k}(t)\}-1\right)+1 \tag{9}$$

Substituting (5) into (9) gives

$$E\left[\exp\{\delta\alpha_i(t)\}\right]\leq\prod_{k=1}^{c_i}\left[\frac{r_{i,k}\cdot t}{\alpha_{i,k}(t)}\cdot\left(\exp\{\delta\alpha_{i,k}(t)\}-1\right)+1\right] \tag{10}$$

Then, using Chernoff's Bound Theorem gives

$$\Pr[X\geq x]\leq\exp\{-\delta x\}\cdot E[\exp\{\delta X\}],\ \forall\delta\geq 0 \tag{11}$$

Using network calculus [10] gives

$$D_i(t)\leq\sup_{s\geq 0}\{\inf\{d_i\geq 0:\alpha_i(t)\leq\beta_i(t+d_i)\}\} \tag{12}$$

From (12), we get the following:

$$\Pr[D_i(t)\geq d_i]=\Pr[\alpha_i(t)\geq\beta_i(t+d_i)] \tag{13}$$

Substituting (4) and (8) into (13), we get

$$\begin{aligned}\Pr[D_i(t)\geq d_i]\leq\Pr\left[\alpha_i(t)\geq\varepsilon_i\left(\beta(t+d_i)\cdot\rho-\sum_{j=1}^{i-1}\alpha_j(t+d_i)\right)\right]\\ \leq\Pr\left[\alpha_i(t)+\varepsilon_i\sum_{j=1}^{i-1}\alpha_j(t+d_i)\geq\varepsilon_i\rho\beta(t+d_i)\right]\end{aligned} \tag{14}$$

We can derive (6) from (9), (10), (11) and (14).

5 Numerical Results and Analysis

5.1 Network/Flow Parameter Setup

The two-level model shown in Fig. 1 and described section 3 is used for all physical base stations. The service curves $\beta(t)$ of the physical base stations are given in (8). Slices 1 to 3 are denoted $A_1(t)$, $A_2(t)$ and $A_3(t)$, respectively, and are used to show the evaluation results. We assume that $A_1(t)$ contains three flows: $A_{1,1}(t)$, $A_{1,2}(t)$, $A_{1,3}(t)$; we assume that $A_2(t)$ contains two flows: $A_{2,1}(t)$ and $A_{2,2}(t)$; and we assume that $A_3(t)$ contains one flow: $A_{3,1}(t)$. Here we assume that every flow is regulated by the leaky bucket regulator $\alpha(t)$ (1). Without loss of generality, we specify the reservation of bandwidth weight for each slice according to the size of the flows. The bandwidth weight μ_i of the slices, the average arrival rate $r_{i,k}$, and the burst parameter $b_{i,k}$ of the six flows are shown in **Table 2**.

In terms of evaluation, we investigate network delay as a function of both flow arrival rate and service rate of the physical BS.

5.2 Network Delay

Figs. 2, 3, 4 and **5** show the impact of the same set of variables, i.e., d, R, T and ρ, on $\Pr[D\geq d]$ of a virtual slice. **Figs. 6, 7, 8** and **9** show, respectively, the curved surface of the upper bounds on the delay probability as a function of a) the delay and service rate, b) the service rate and latency, c) the service rate and network bandwidth utilization, and d) the latency and network bandwidth utilization for slice 1.

Fig. 2 shows the delay probability curves as a function of de-

lay. Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ decreases slightly and linearly in relation to d and is almost insensitive to d when traffic for each flow is relatively small. In Fig. 2, if d = 0.52, Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ is 0.1414, 0.0645 and 0.0134, respectively. If d = 0.56, Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ is 0.1224, 0.0527 and 0.0098, respectively. Fig. 3 shows how Pr[$D \geq d$] decreases as R increases and how Pr[$D \geq d$] approaches 0 for all slices. This exponential trend suggests network bandwidth is critical to Pr[$D \geq d$]. In Fig. 3, if R = 20, Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ is 0.6628, 0.5744 and 0.4176 respectively. If R = 100, Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ is 0.1269, 0.0555 and 0.0106, respectively.

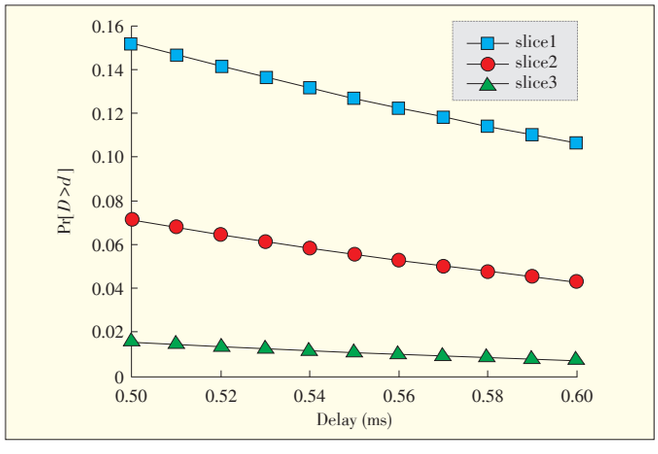
Fig. 6 shows the joint impact of d and R on Pr[$D \geq d$] of $A_1(t)$. Pr[$D \geq d$] increases as both R (network bandwidth) and d decrease. The impact of R on Pr[$D \geq d$] is more obvious than that of d. Fig. 4 shows how Pr[$D \geq d$] increases in relation to T (maximum service delay). Pr[$D \geq d$] starts slow but increases much more significantly as T increases and the network becomes more loaded or even congested. This trend applies to all slices, and the heavier the slice load, the more obvious the trend is. In Fig. 4, if T = 0.1, Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ is 0.1812, 0.0918 and 0.0234, respectively. If T = 0.5, Pr[$D \geq d$] of $A_1(t)$, $A_2(t)$ and $A_3(t)$ is 0.7649, 0.7034 and 0.5741, respectively.

Fig. 7 shows the curved surface of Pr[$D \geq d$] as a function of R and T. Pr[$D \geq d$] decreases as R increases and T decreases. Fig. 5 shows how the effect of ρ on Pr[$D \geq d$] is roughly the same as that of R on Pr[$D \geq d$] (Fig. 3).

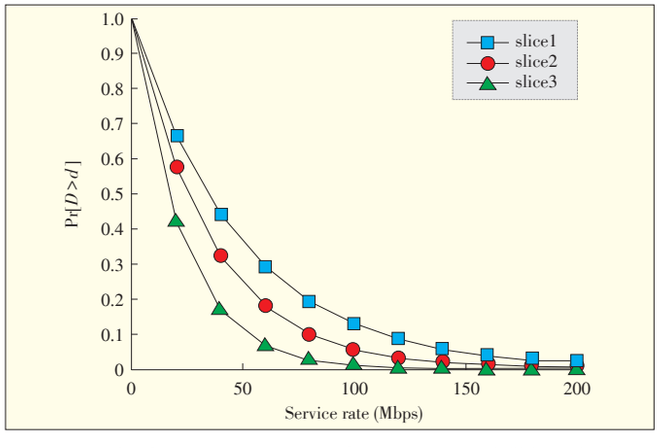
Fig 8 shows the joint effect of ρ and R on Pr[$D \geq d$], and Fig. 9 shows the joint effect of ρ and T on Pr[$D \geq d$]. The joint effect of ρ and R is complex: Pr[$D \geq d$] increases as R decreases (Fig. 3) and decreases as ρ increases (Fig. 5). Pr[$D \geq d$] in-

▼**Table 2. Parameters of the Three Slices and Their Flows**

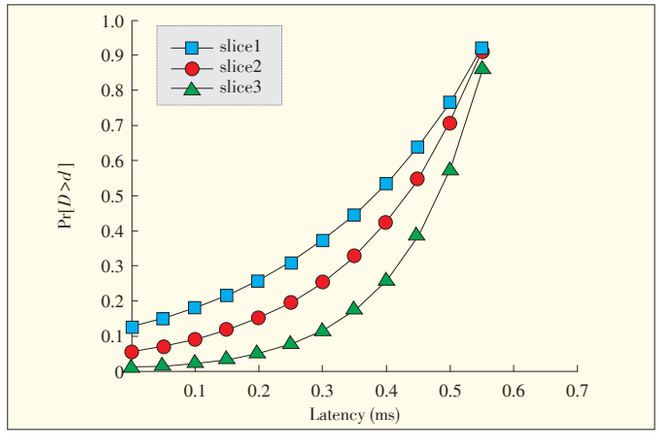
Slices $A_i(t)$	Weight μ_i	Flows $A_{i,k}(t)$	Avg. Arrival Rate $r_{i,k}$(Kb/s)	Burst Tolerance $b_{i,k}$(Kb)
$A_1(t)$	0.45	1	240	400
		2	320	360
		3	400	260
$A_2(t)$	0.35	1	450	420
		2	350	360
$A_3(t)$	0.20	1	700	550



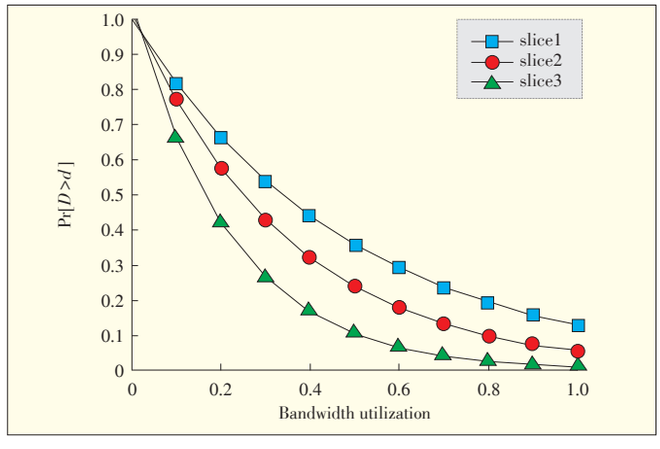
▲**Figure 2. Pr[$D \geq d$] vs. delay for R = 100, T = 0.001, t = 0.025, ρ = 1.**



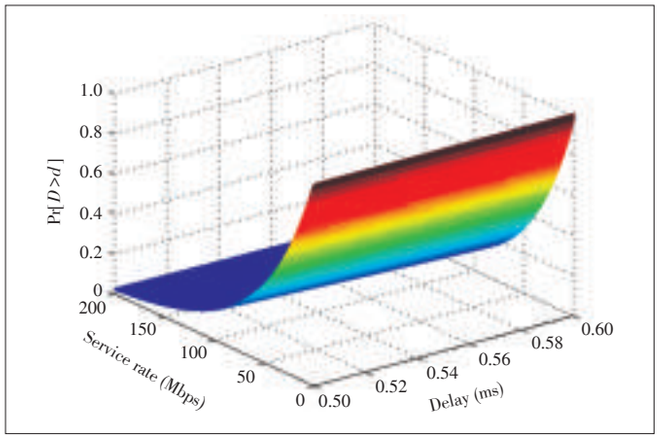
▲**Figure 3. Pr[$D \geq d$] vs. service rate for d = 0.55, T = 0.001, t = 0.025, ρ = 1.**



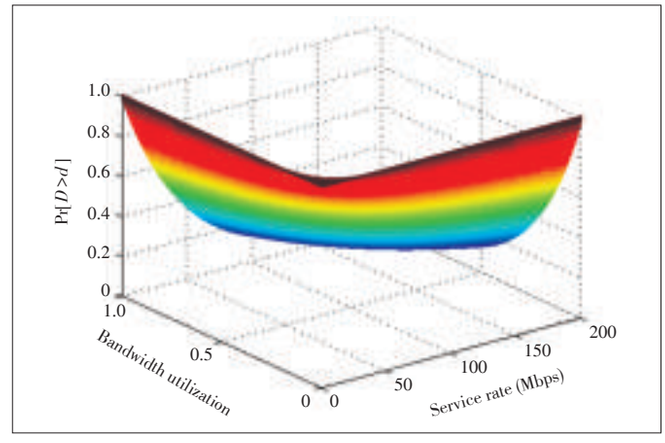
▲**Figure 4. Pr[$D \geq d$] vs. latency for R = 100, d = 0.55, t = 0.025, ρ = 1.**



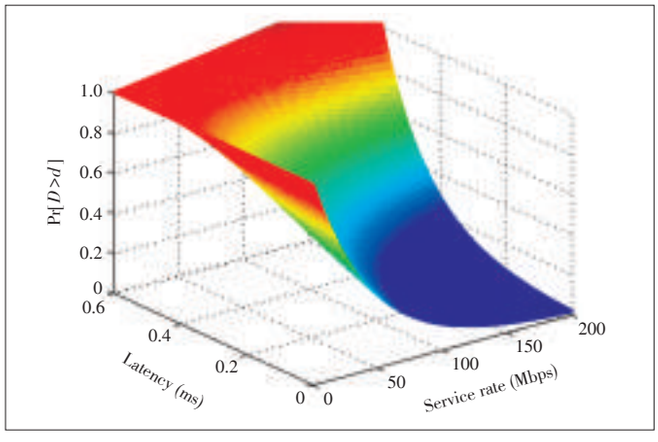
▲**Figure 5. Pr[$D \geq d$] vs. bandwidth utilization for R = 100, T = 0.001, d = 0.55, t = 0.025.**



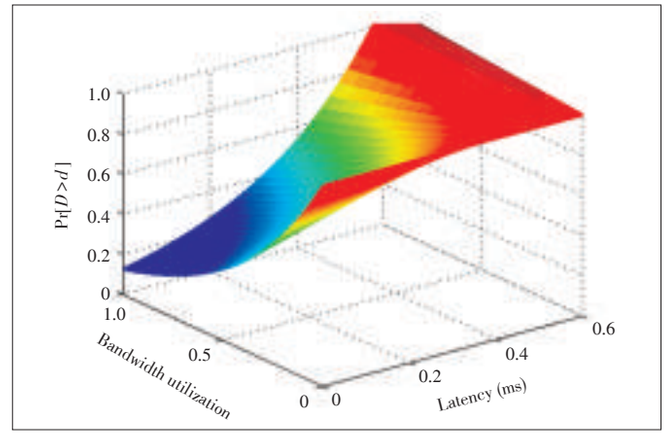
▲Figure 6. Pr[$D \geq d$] of $A_1(t)$ vs. delay and service rate for $T = 0.001$, $t = 0.025$, $\rho = 1$.



▲Figure 8. Pr[$D \geq d$] of $A_1(t)$ vs service rate and bandwidth utilization for $T = 0.001$, $t = 0.025$, $d = 0.55$.



▲Figure 7. Pr[$D \geq d$] of $A_1(t)$ vs. service rate and latency for $t = 0.025$, $d = 0.55$, $\rho = 1$.



▲Figure 9. Pr[$D \geq d$] of $A_1(t)$ vs. latency and bandwidth utilization for $R = 100$, $t = 0.025$, $d = 0.55$.

creases as T increases and decreases as ρ increases (Fig. 9).

In summary, the parameters of the flow regulators and service curves in the physical BS and virtual BS play an important role in modelling guaranteed delay. In particular, Pr[$D \geq d$] decreases as d decreases; Pr[$D \geq d$] decreases as R increases; Pr[$D \geq d$] decreases as ρ increases; and Pr[$D \geq d$] decreases as T decreases. To improve network performance and guaranteed delay in an SDN, certain mechanisms can be used to reduce Pr[$D \geq d$]. These mechanisms may involve rational scheduler parameters, such as network bandwidth and maximum service delay.

6 Conclusion and Future Work

In this paper, we have proposed a simple but realistic model for describing the upper bound delay of a wireless virtual network in the context of SDN. The model takes into account service flows, which represent service types, and virtualized networks, as presented by slices. In particular, we have used a finer system modelling and performance analysis tool, called stochastic network calculus, to describe the proposed model. We also deduced closed-formed formulas for the upper bound delay. In future work, we will propose a scheduling algorithm based on the above QoS model. Another area of future work is to extend the model to include other network parameters, such as throughput.

Acknowledgements

This research was supported in part by the grant from the National Natural Science Foundation of China (60973129). The authors would like to thank there viewers for their valuable comments.

References

[1] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communication Magazine*, vol. 51, no. 2, pp. 114–119, 2013. doi: 10.1109/MCOM.2013.6461195.

[2] S. Sezer, S. Scott-Hayward, P.K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 36–43, 2013. doi: 10.1109/MCOM.2013.6553676.

[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008. doi: 10.1145/1355734.1355746.
[4] K. Choumas, N. Makris, T. Korakis, L. Tassiulas, and M. Ott, "Exploiting OpenFlow resources towards a content-centric LAN," in *Proceedings Of Second European Workshop on Software Defined Networks (EWSDN2003)*, Berlin, 2013, pp. 93–98. doi: 10.1109/EWSDN.2013.22.
[5] C. Chaudet and Y. Haddad, "Wireless software defined networks: challenges and opportunities," in *Proceedings of IEEE InternationalConference on Microwaves, Communications, Antennas and Electronics Systems (COMCAS2013)*, Tel Aviv, Israel, 2013, pp. 1–5.
[6] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo. "Software defined wireless networks: unbridling SDNs," in *Proceedings of European Workshop on Software Defined Networking (EWSDN2012)*, Darmstadt, Germany, 2012, pp. 1–6.
[7] H. Wen, P. K. Tiwary, and T. Le-Ngoc, "Current trends and perspectives in wireless virtualization," in *Proceeding of 2013 InternationalConference on Selected Topics in Mobile and Wireless Networking*, Montreal, QC, Canada, 2013, pp. 62–67. doi: 10.1109/MoWNet.2013.6613798.
[8] R. Matos, C. Marques, S. Sargento, K. A. Hummel, and H. Meyer, "Analytical modeling of context-based multi-virtual Wireless Mesh Networks," *Elsevier International Journal on Ad hoc Networks*, vol. 13, pp. 191–209, 2014. doi: 10.1016/j.adhoc.2011.05.004.
[9] X. Lu, K. Yang, Y. Liu, D. Zhou, and S. Liu. An elastic resource allocation algorithm enabling wireless network virtualization, *Wiley International Journal on Wireless Communications and Mobile Computing*[Online].Available: http://onlinelibrary.wiley.com/doi/10.1002/wcm.2342/full
[10] J. -Y. Le Boudec, and P. Thiran, *Network calculus*. Springer Verlag, 2004.
[11] M. Fidler, "A survey of deterministic and stochastic service curve models in the network calculus," *IEEE Communications surveys & tutorials*, vol. 12, no. 1, pp. 59–86, 2010. doi: 10.1109/SURV.2010.020110.00019.
[12] L. Zhang, J. Wu, and X. Deng, "Modelling the guaranteed QoS for wireless sensor networks: a network calculus approach," *EURASIP Journal Wireless Communication and Networking*, vol. 82, 2011. doi: 10.1186/1687-1499-2011-82.
[13] J. Liu, L. Zhang, and K. Yang, "Modeling guaranteed delay of virtualized wireless networks using network calculus," in *Proceedings of the 10th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MOBIQUITOUS 2013)*, Tokyo, Japan, December 2–4, 2013.
[14] N. M. M. K. Chowdhury, and R. Boutaba, "Network virtualization: state of the art and research challenges," *IEEE Communications Magazine*, vol. 47, no. 7, pp. 20–26, 2009. doi: 10.1109/MCOM.2009.5183468.
[15] G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy, "Network virtualization architecture: Proposal and initial prototype," in *Proceedings of ACM VISA*, NY, USA, 2009, pp. 63–72. doi: 10.1145/1592648.1592659.
[16] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "NVS: A substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1333– 1346, 2012. doi: 10.1109/TNET.2011.2179063.
[17] S. Ahn and C. Yoo, "Network interface virtualization in wireless communication for multi-streaming service," in *Proceedings of International Symposium on Consumer Electronics (ISCE)*, Singapore, 2011, pp. 67– 70. doi: 10.1109/ISCE.2011.5973785.
[18] M. Tao, Y. C. Liang, and F. Zhang, "Resource allocation for delay differentiated traffic in multiuser OFDM systems," *IEEE Transactions on Wireless Communications*, vol. 7, no. 6, pp. 2190–2201, 2008. doi: 10.1109/TWC.2008.060882.
[19] D. S. W. Hui, V. K. N. Lau, and H. L. Wong, "Cross-layer design for OFDMA wireless systems with heterogeneous delay requirements," *IEEE Transactions on Wireless Communications*, vol. 6, no. 8, pp. 2872–2880, 2007. doi: 10.1109/TWC.2007.05716.
[20] C. C. Zarakovitis, Q. Ni, D. E. Skordoulis, and M. G. Hadjinicolaou, "Power-efficient cross-layer design for OFDMA systems with heterogeneous QoS, imperfect CSI, and outage considerations," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 2, pp. 781–798, 2012. doi: 10.1109/TVT.2011.2179817.
[21] Y. Jiang, "Stochastic network calculus for performance analysis of Internet networks– An overview and outlook," in *Proceedings of International Conference on Computing, Networking and Communications (ICNC)*, Maui, HI, 2012. doi: 10.1109/ICCNC.2012.6167501.
[22] F. Ciucu, F. Poloczek, and J. Schmitt, "Sharp bounds in stochastic network calculus," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 1, pp. 367–368, 2013. doi: 10.1145/2494232.2465746.
[23] J. B. Schmitt, F. A. Zdarsky, and M. Fidler, "Delay bounds under arbitrary multiplexing: when network calculus leaves you in the lurch ....," in *Proceedings of 27th IEEE International Conference on Computer Communications (INFOCOM' 08)*, Phoenix, USA, April 2008. doi: 10.1109/INFOCOM.2008.228.
[24] A. Bouillard, L Jouhet, and E. Thierry, "Tight performance bounds in the worst-case analysis of feed-forward networks," in *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM' 10)*, San Diego, USA, 2010, pp. 1316–1324. doi: 10.1109/INFCOM.2010.5461912.
[25] J. B. Schmitt, F. A. Zdarsky, and L. Thiele, "A comprehensive worst-case calculus for wireless sensor networks with in-network processing," in *Proceedings of the 28th IEEE International Real-Time Systems Symposium (RTSS' 07)*, Tucson, USA, December 2007, pp. 193–202. doi: 10.1109/RTSS.2007.17.
[26] L. Zhang, "Bounds on end-to-end delay jitter with self-similar input traffic in ad hoc wireless network," in *Proceedings of 2008 ISECS International Colloquium on Computing, Communication, Control, and Management (CCCM' 08)*, Guangzhou, China, August 2008, pp. 538–541. doi: 10.1109/CCCM.2008.23.
[27] L. Zhang, S. Liu, and H. Xu, "End-to-end delay in wireless sensor network by network calculus," in *Proceedings of 2008 International Workshop on Information Technology and Security (WIST' 08)*, Shanghai, China, December 2008, pp. 179–183.
[28] H. Qi, Z. Chen, and L. Zhang, "Towards end-to-end delay on WMNs based on statistical network calculus," in *Proceedings of the 9th International Conference for Young Computer Scientists (ICYCS' 08)*, Zhang JiaJie, Hunan, China, November 2008, pp. 493–497. doi: 10.1109/ICYCS.2008.347.
[29] Azodolmolky S, Nejabati R, Pazouki M, Wieder P, Yahyapour R, and Simeonidou D, "An analytical model for Software Defined Networking: A network calculus-based approach," in *Global Communications Conference (GLOBECOM2013)*, Atlanta, USA, December 09–13, 2013. doi: 10.1109/NOCS.2009.5071447.
[30] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single node case," *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 344–357, 1993. doi: 10.1109/90.234856.
[31] R. R. Boorstyn, A. Burchard, J. Liebeherr, and C. Oottamakorn, "Statistical service assurances for traffic scheduling algorithms," *IEEE Journal on Selected Areas in Communications Special Issue on Internet QoS*, vol. 18, no. 10, pp. 2651–2664, 2000. doi: 10.1109/49.898747.

Manuscript received: 2014-02-24

Biographies

Lianming Zhang (zlm@hunnu.edu.cn) received his PhD from Central South University, China. He received his BS degree and MS degree from Hunan Normal University, China. He is currently a professor in the school of Physics and Information Science, Hunan Normal University. His current research interests include software-defined networking, complex networks, and network calculus. He completed a two-year postdoctoral fellowship in complex networking at South China University of Technology. He has published more than 90 papers.

Jia Liu is currently an MS student in the College of Physics and Information Science, Hunan Normal University. She also received her BEng. degree from Hunan Normal University. Her research interests include wireless communications, wireless networks, and network calculus. She has published one journal paper in Springer/ACM Mobile Networks and Applications (MONET) and one conference paper in IEEE MOBIQUITOUS 2013.

Kun Yang (kunyang@essex.ac.uk) received his PhD from University College London. He received his BSc degree and MSc degree from Jilin University, China. He is currently a chair professor in the School of Computer Science and Electronic Engineering, University of Essex, and leads the Network Convergence Laboratory (NCL) there. Before joining the University of Essex in 2003, he worked for several years at UCL on several EU research projects. His main research interests include heterogeneous wireless networks, fixed mobile convergence, future Internet technology and network virtualization, cloud computing and networking. He manages research projects funded by various sources such as UK EPSRC, EU FP7 and industries. He has published 60+ journal papers. He serves on the editorial boards of both IEEE and non-IEEE journals. He is a Senior Member of IEEE and a Fellow of IET.

DOI: 10.3969/j. issn. 1673–5188. 2014. 02. 009
http://www.cnki.net/kcms/detail/34.1294.TN.20140515.1111.001.html, published online May 15, 2014

Load Balancing Fat-Tree on Long-Lived Flows: Avoiding Congestion in a Data Center Network

Wen Gao, Xuyan Li, Boyang Zhou, and Chunming Wu
(College of Computer Science, Zhejiang University, Hangzhou 310027, China)

Abstract

In a data center network (DCN), load balancing is required when servers transfer data on the same path. This is necessary to avoid congestion. Load balancing is challenged by the dynamic transferral of demands and complex routing control. Because of the distributed nature of a traditional network, previous research on load balancing has mostly focused on improving the performance of the local network; thus, the load has not been optimally balanced across the entire network. In this paper, we propose a novel dynamic load-balancing algorithm for fat-tree. This algorithm avoids congestions to the great possible extent by searching for non-conflicting paths in a centralized way. We implement the algorithm in the popular software-defined networking architecture and evaluate the algorithm's performance on the Mininet platform. The results show that our algorithm has higher bisection bandwidth than the traditional equal-cost multi-path load-balancing algorithm and thus more effectively avoids congestion.

Keywords

data center network; software-defined networking; load balancing; network management

1 Introduction

In a data center network (DCN), a large number of servers are connected together by high-speed links and switches [1]. A traditional DCN architecture typically has a multi-rooted tree topology. Usually, such architecture has a two-tier or three-tier data-switching structure that can accommodate tens of thousands of servers. However, with the emergence of new services in recent years, the limitations of traditional tree-based DCNs have been exposed. These limitations include poor scalability, low link utilization, and resource slicing. To overcome these limitations, DCN architectures such as fat tree [2], PortLand [3] and BCube [4] have been proposed. Of these, fat tree is the simplest, easiest to deploy, and most used.

In the fat tree, routing tables are configured statically [2]. Although the algorithm for managing table configuration distributes flows evenly over multiple links, the network becomes unbalanced when forwarding for an increasing number of applications. When the network becomes unbalanced, several flows compete for the same links while other links remain idle. Also, the traffic in a DCN comprises many small, transactional-type remote procedure call (RPC) flows and only a few long-lived flows, which require the majority of the bandwidth. If long-lived flows collide on some links, network performance is greatly reduced.

This problem is compounded by the unpredictability of data-forwarding demands because multiple DCN applications randomly request a path at runtime. Because of dynamically changing data flows, the routing in a DCN cannot be changed in time to avoid congestion. Current work on this problem shows that performance can be improved by spreading flows onto multiple paths on every node in a distributed way. However, this does not equate to optimal load balancing across the entire network. Software-defined networking (SDN) has a great advantage in that it provides a global network view and can optimize the network by controlling the data plane in a very centralized way.

SDN [5] separates the control plane from the data plane in traditional routers. The data plane still remains in network devices and is responsible for forwarding packets at high speed. The control plane is passed to the SDN controller and controls the underlying network.

Thus, the underlying network infrastructure is abstracted from applications, and network state and intelligence are logically centralized in the SDN controller. This provides researchers, enterprises, and carriers with unprecedented network programmability, automation, and control and enables them to build

This work is supported by the National Basic Research Program of China (973 Program) (2012CB315903), the Key Science and Technology Innovation Team Project of Zhejiang Province (2011R50010-05), the National Science and Technology Support Program (2014BAH24F01), 863 Program of China (2012AA01A507), and the National Natural Science Foundation of China (61379118 and 61103200). This work is sponsored by the Research Fund of ZTE Corporation.

highly scalable, flexible networks that can rapidly adapt to changing business needs. In other words, we can control the network from a global perspective according to the information maintained by the SDN controller.

SDN creates new opportunities to balance the load of a DCN. Because of the distributed nature of a traditional network, previous research on DCN load balancing has focused on each network node, but not enough consideration has been given to load balancing across the entire network. However, SDN enables centralized control, and the SDN controller can view several aspects of the entire network, e.g., topology, flows, and link utility. This is an advantage when balancing the load across the DCN.

This paper describes an improved fat-tree architecture and dynamic load-balancing algorithm (FTLB). The algorithm regularly collects flow information from edge switches and uses the network view stored in the SDN controller to compute new paths for large flows whose paths conflict with other flows' paths. Then, FTLB tells the switches to modify relevant flow table entries to route large flows through new paths. Our goal is to better balance the load on fat tree by allocating non-conflicting paths for long-lived flows and preventing these paths from colliding on the same physical links.

We evaluate FTLB by emulating a fat-tree DCN topology on Mininet. This topology comprises 128 hosts. We use three communication patterns to generate synthesized data traffic in a DCN. The experiment results show that FTLB improves the bisection bandwidth by up to 50% that which is possible using the traditional equal-cost multipath (ECMP) approach.

The rest of the paper is organized as follows: In section 2, we describe the fat-tree architecture and improved fat-tree based on SDN; in section 3 we propose the FTLB algorithm for load balancing; in section 4, we evaluate the experimental results; in section 5, we discuss related work; and in section 6, we conclude the paper.

2 Architecture

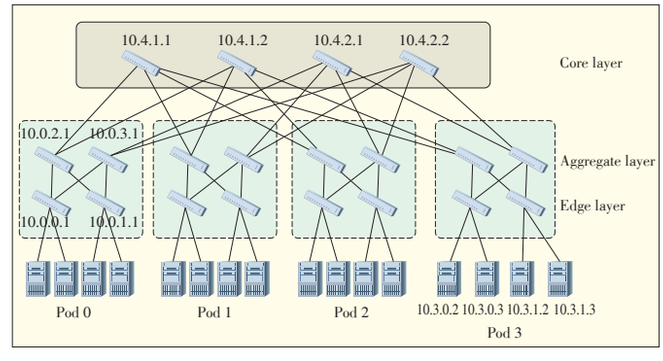
Fat tree is a new DCN architecture proposed by Al-Fares in 2008 [2]. It is based on Clos topology [6] and provides full bandwidth for communication between any two in a DCN. **Fig. 1** shows a k-ary fat tree. The architecture comprises edge switch layer, aggregate switch layer, and core switch layer. A k-ary fat tree contains k pods, each of which contains $k/2$ k-port edge switches and $k/2$ k-port aggregate switches (Fig. 1). Each edge switch use $k/2$ ports to connect to $k/2$ servers, and the remaining $k/2$ ports connect to $k/2$ aggregate switches in the same pod. Similarly, $k/2$ ports of each aggregate switch connect to $k/2$ edge switches, and the remaining $k/2$ ports connect to core switches. Because the total number of aggregate switch uplink ports equals the total number of core switch ports, the number of k-port core switches is $(k/2)^2$. Each core switch has one port that connects to each pod; that is, the i-th port of each core switch connects to pod i.

In the fat-tree, the routing tables distribute flows evenly between aggregate and core switches so that the bisection bandwidth is maximized. To do this, the routing tables use a two-level match: the main routing table uses a prefix match (i.e., left-handed, /m prefix masks of the form 1^m0^{32-m}), and the secondary routing table uses a suffix match (i.e., right-handed, /m suffix masks of the form $0^{32-m}1^m$). Each entry in the main routing table has a potential pointer to a secondary routing table of (suffix, port) entries. A prefix match terminates if it does not contain a pointer to a suffix match. If the prefix-matching search yields a non-terminating prefix, then the matching suffix in the secondary routing table is found and used. Prefix matching is used by edge and aggregate switches to route flows destined towards the pod they stay in. Suffix matching is used by edge and aggregate switches to spread the flow between aggregate and core switches according to current switch ID and destination host ID. Core switches only use prefix matching to route flows to corresponding aggregate switches in the destination pod.

The advantage of the fat tree is that all the network devices are identical, and cheap commodity parts can be used for all the switches in the architecture. Furthermore, the fat tree can be rearranged and is non-blocking. This means that link utilization is relatively high. In addition, a k-ary fat tree can accommodate $k^3/4$ servers if we use 48-port switches. This means that fat tree can contain 48 pods, each of which contains 24 aggregate switches and 24 edge switches. Therefore, the whole topology can contain 27,648 servers, and fat tree is relatively scalable.

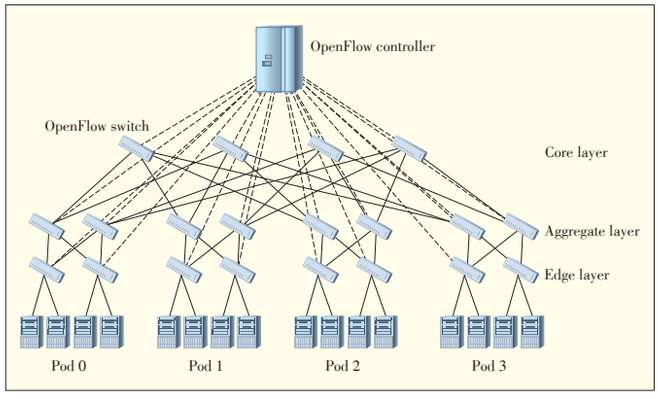
Fat tree provides different paths for the communication between any two servers. However, because of the limited number of core switches and static routing tables, performance may deteriorate when several flows compete for the same link resources, and some links may remain idle. Therefore, we need to make full use of the multiple paths between any two servers and improve the routing algorithm in order to dynamically balance the network load.

In the existing DCN, the load-balancing methods tend to make use of local link information and therefore are only capable of partial load balancing, not overall network load balanc-



▲**Figure 1. Fat-tree architecture.**

ing. However, the SDN controller has an overall view of the DCN, so we can combine DCN with SDN to create a fat-tree architecture based on SDN (**Fig. 2**). The architecture in Fig. 2 is different from traditional fat-tree architecture in that it has a centralized SDN controller, called OpenFlow controller (OF



▲**Figure 2. New fat-tree architecture based on SDN.**

controller) [7], and OpenFlow switches are used as network devices. Each OpenFlow switch connects to the OpenFlow controller through a secure channel using transport layer security (TLS). An OpenFlow controller uses a standard interface provided by OpenFlow switches in order to update the flow tables on those switches and control the forwarding behavior of the underlying network. Further, the OpenFlow controller can obtain statistics and other information, such as topology and port status, from underlying networks to form a view of the whole network.

3 FTLB Algorithm

Several studies have shown that Internet traffic is characterized by a few large, long-lived flows consuming most of the bandwidth, as well as many small, short-lived flows [8], [9]. The method of routing large, long-lived flows can significantly affect network performance and bisection bandwidth; therefore, we need to handle such flows in a special way. Large flows always exist in a network for a long time and have a relatively large number of packets.

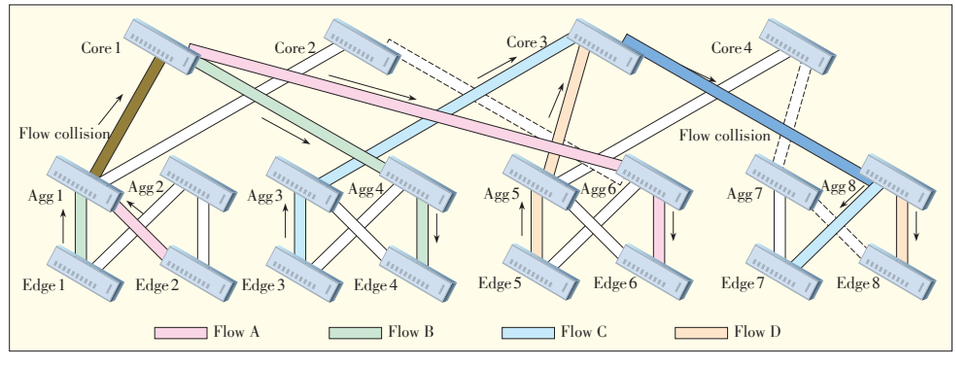
ECMP is currently used to take advantage of multiple paths between any two servers in a fat-tree architecture. Each ECMP-enabled switch maintains a data structure called a flow mapping table that stores mappings of flows to paths. If a new flow arrives but cannot be found in a flow mapping table, it is forwarded along a selected path that corresponds to a hash of the selected fields of the packet's headers modulo the number of paths. Thus, the load is split between multiple paths. In addition, the mapping of the new flow to the selected path is written to the flow mapping table, and subsequent packets belonging to this flow are forwarded along this selected path so that packets do not need to be reordered [10]. However, when using ECMP to select flow paths, several large, long-lived flows can collide on the same links, creating a bottleneck in the network and decreasing network bisection. Therefore, large flows need to be scheduled in order to avoid collision and increase throughput.

Fig. 3 shows a network with four large flows. Flow A and flow B use the same link between aggregate switch 1 and core switch 1 when forwarding packets to core switch. Flows C and D are aggregated on core switch 3 and use the same link between core switch 3 and aggregate switch 8 when forwarding packets to the destination. In other words, flows A and B collide with each other on some links, and flows C and D collide with each other on some links as well. Assuming that the link bandwidth is 100 Mbps and there are collisions between flows, for each of all four flows, its rate is clinched to 50 Mbps, and the bisection bandwidth is also halved. If we schedule flow A to go through core switch 2, and if we schedule flow D to go through core switch 4, all the flows will have full bandwidth, and network bisection bandwidth will be improved.

The proposed FTLB algorithm has three basic steps: 1) the OF controller regularly detects large flows in the network; 2) the OF controller decides which large flows should be scheduled and computes new paths according to the network view stored in the OF controller; 3) the OF controller sends OFPFC_MODIFY messages to corresponding switches to deploy the new paths.

There are two methods for detecting large flows: push and pull. With the former, whenever counters in a flow entry reach the threshold, the switch sends an OpenFlow message to the controller to report a new large flow. With the latter, the OpenFlow controller sends an OpenFlow message to edge switches to periodically retrieve the counters of flow entries. If the value of a counter is greater than the threshold, the flow is marked as large.

FTLB should compute new paths for large flows when they



▲**Figure 3. Different flows compete for the same link resource.**

have been detected. The algorithm makes full use of the multiple paths between any two servers [11] and, to the greatest possible extent, allocates non-conflicting paths for different large flows. The specific method is as follows:

1) All flows are forwarded according to the fat-tree routing algorithm. If a flow does not match any entry in the flow table, it is forwarded to the controller, which computes a path for it. At the same time, new flow entries are inserted into the flow tables. FTLB can only act on large flows, and the paths of new flows are computed with the original routing algorithm.
2) When large flows have been detected, FTLB selects those flows that should be scheduled according to overlapping information between the large flows. If the original path of a large flow does not overlap the path of others, then it should not be scheduled. In this case, we only need to mark the links on the flow path, which means these links have been allocated. For example, if we detect four large flows once, we should only schedule flows B and D, and flows A and flow C do not need to be scheduled (Fig. 3). This decreases the overhead of FTLB.
3) According to the fat-tree architecture, the large flows of which the source and destination are not in the same pod with $k^2/4$ paths. In addition, each flow is only forwarded by a core switch. Therefore, in order to make the large flows do not conflict; we can search $k^2/4$ different paths according to core switches for every large flow that should be scheduled. A path is allocated to a large flow if all links on the path have not been allocated.
4) Large flows of which the source and destination are in the same pod but do not correspond to the same edge switch pass through aggregate switches instead of core switches. Therefore, we only need to search the corresponding paths of $k/2$ aggregate switches. If there is a path whose links have not been allocated, it is allocated to current large flow.
5) If FTLB cannot find a non-allocated path for a new large flow, it also searches the corresponding path of every core switch or aggregate switch to find a path that can be allocated to the minimal number of flows.
6) When a large flow disappears, i.e., when the flow entry of the large flow's new path has timed out, the OpenFlow switch will send a flow-removed message to the controller, which then unmarks the links belonging to the large flow.

Deploying a new path is the last step of FTLB. The OpenFlow controller sends an OFPFC_MODIFY message to the relevant edge of the new path and aggregates switches in order to modify corresponding flow entries. Packets of the large flows are then transported through these new paths.

FTLB computes new paths for large flows by searching paths that correspond to aggregate or core switches. The time to decide whether a path meets the requirements is given by $O(1)$ because a path contains a maximum of four links. For each large flow, the number of paths that FTLB needs to search is $O(k^2)$ at worst and $O(1)$ at best. Thus, for n large flows, the time taken by FTLB is $O(nk^2)$ at worst.

4 Implementation and Evaluation

To determine the feasibility of the proposed FTLB algorithm, we use the Mininet platform to emulate a network with many large flows. Here, we introduce the experimental scenarios and analyze the results.

4.1 Experment Platform

Building a real physical network for experimentation is not ideal because network devices, such as routers and switches, are expensive, and the platform cannot be easily reused. This is a serious waste of resources. The control plane is integrated within the router, making it difficult to develop and test new algorithms. Also, the platform is relatively small, so traffic on platform lacks authenticity.

Mininet [12] is a process-virtualized network experimental platform based on Linux Container and proposed by Nick McKeown of Stanford University. We can use Mininet to build complex network compared favorably with real network hardware environment to experiment our new ideas based on OpenFlow. More importantly, the experimental code can be seamlessly migrated to the real network environment. Therefore, we use Mininet as our experiment platform to validate FTLB algorithm.

4.2 Experiment Scenario

We use an 8-ary fat-tree architecture that contains 128 hosts, 32 edge switches, 32 aggregate switches, and 16 core switches. The switches in fat tree are OpenFlow switches, and the network is controlled by an OF controller.

The performance metric is bisection bandwidth [13], which is the maximum transmission rate through a section if the network is halved by section and the number of nodes in each half is equal. The bisection bandwidth can give an indication of overall network performance—the larger the bisectional bandwidth, the better the transmission of the network.

Because there are no DCN traces, we need to create a group of communication patterns [2]:

- staggered prob (pEdge, pPod). A host sends packets to 1) another host in the same subset with probability pEdge; 2) another host in the same pod with probability pPod; and 3) to any host not in the same pod with probability 1-pEdge-pPod.
- random. A host sends packets to any other host in the network with equal probability.
- stride(i). A host with index m sends packets to host with index $(I + m)$ mod n, where n is the total number of hosts in the network.

In DCN, traffic comprises many small, transactional-type RPC flows, e.g., search results, and a few long-lived flows, e.g., backups and MapReduce tasks. To simulate traffic in DCN, every host generates new flows of different length at a Poisson distributed start time, and the length is exponentially distribut-

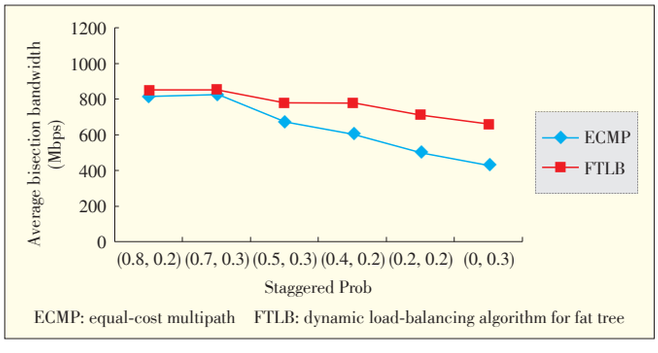
ed. The send rate of every host is updated continuously according to transmission control protocol (TCP) slow-start and additive increase multiplicative decrease (AIMD). If it is in slow-start stage, the send rate is doubled in every tick. When the send rate reaches its threshold, it is halved and moves to the congestion-avoidance stage, where the send rate is increased by addition.

4.3 Evaluation

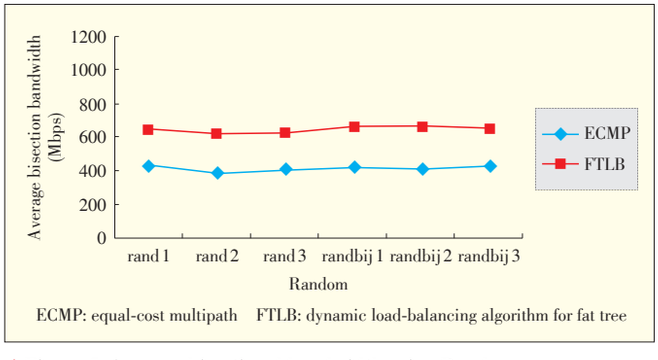
In our experiment, every host constantly measures the incoming bandwidth. Each experiment lasts 60 s, and we measure the average bisectional bandwidth in the middle 40 s. We performed each experiment three times for staggered prob, random, and stride communication patterns. The average was taken as the result of each experiment. **Figs 4, 5** and **6**, show the performance of ECMP algorithm and FTLB algorithm using three different communication patterns.

Fig. 4 shows that FTLB performs better than ECMP. When Staggered Prob is (0.8, 0.2), the two algorithms perform similarly because only 20% of flows whose source and destination are in the same pod but not in the same subnet need to be scheduled. However, when the number of flows sent to the same subnet decreases and the number of flows that need to be scheduled increases, ECMP performance declines sharply. FTLB is therefore very effective for scheduling large flows.

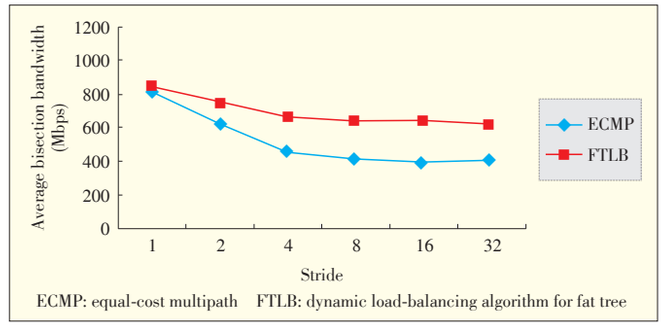
In Fig. 5, the difference in performance between ECMP and



▲**Figure 4. Average bisectional bandwidth using Staggered Prob communication pattern.**



▲**Figure 5. Average bisectional bandwidth using Random communication pattern.**



▲**Figure 6. Average bisectional bandwidth using Stride communication pattern.**

FTLB is relatively large because the number of flows whose source and destination are not in the same subnet is in the majority. The first three groups of data are generated using completely random communication pattern whereas the last three groups of data are generated using a bijective random communication pattern. In a bijective random communication pattern, two hosts send packets to each other. If the two flows moving in the opposite direction between the two hosts are both large flows, they can be scheduled to the same path using FTLB. Thus, link resources can be used more efficiently. Therefore, the performance of FTLB using bijective random communication is slightly better than that using a completely random communication pattern.

As the Stride parameter increases, the number of flows that should be scheduled also increases, and FTLB becomes more efficient than ECMP (Fig. 6).

From the experimental results, we see that FTLB effectively schedules large flows to different paths to avoid network congestion whereas ECMP cannot schedule flows as dynamically because is it static and has local characteristics. Therefore, FTLB provides higher average bisection bandwidth than ECMP.

5 Related Work

The several DCN architectures that have been proposed can be classified into the switch-centric architectures and the server-centric architectures [14]. In the former, switches are used to interconnect the network of servers and have routing intelligence. Such architectures include fat tree [2], Monsoon [15], and PortLand [3]. In the latter, servers with multiple network interface card (NIC) ports have routing intelligence. Such architectures include BCube [4], DCell [1], and MDCube [16]. However, these switch-centric and server-centric approaches lack a means of universal control in order to avoid link congestion.

Researchers also have explored load-balancing algorithms for fat tree (e.g., load balancing algorithm based on packet, ECMP) in a multipath environment. However, these algorithms are implemented on every fat-tree node and have local fea-

tures. Thus, they cannot provide universal load balancing for fat tree. Load-balancing algorithms based on packets can provide improved bisection bandwidth by using round robin or deficit round robin. However, this causes packets to be reordered. ECMP selects a flow path according to the hash of several packet header fields, but the selected path cannot be changed, and several flows can be hashed onto the same path.

Our dynamic traffic-aware algorithm improves the performance of a fat-tree architecture by taking into account the characteristics of DCN flows and using the universal network view stored in the controller.

6 Conclusion

In this paper, we have proposed a novel FTLB algorithm for fat tree. This algorithm schedules large flows in DCN by applying SDN to the fat-tree architecture. A universal view of active flows and network resources is stored in SDN controller. By using this information in the SDN controller, FTLB is more effective than a static load-balancing algorithm. With FTLB, network resources can be more efficiently utilized, and the performance of fat tree can be improved. We decrease the overhead of our algorithm by limiting flows that should be scheduled to large flows that will send more bytes across the network. Through experimentation, we observe that FTLB always outperforms ECMP and provides better bisectional bandwidth than ECMP.

References

[1] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S Lu, "DCell: a scalable and fault-tolerant network structure for data center," in *Proc. ACM SIGCOMM*, Seattle, USA, Aug. 2008, pp. 75–86. doi:10.1145/1402958.1402968.

[2] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proc. ACM SIGCOMM*, Seattle, USA, Aug. 2008, pp. 63–74. doi: 10.1145/1402958.1402967.

[3] R. N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "PortLand: a scalable fault-tolerant layer 2 data center network fabric," in *Proc. ACM SIGCOMM*, Barcelona, Spain, Aug. 2009, pp. 39–50. doi: 10.1145/1592568.1592575.

[4] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: a high performance, server centric network architecture for modular data centers," in *Proc. ACM SIGCOMM*, Barcelona, Spain, Aug. 2009, pp. 36–74. doi: 10.1145/1592568.1592577.

[5] ONF. (2012, Apr. 13). *Software-Defined Networking: The New Norm for Networks* [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[6] C. Clos, "A study of non-blocking switching networks," *The Bell System Technical Journal*, vol. 32, no. 2, pp. 406–424, 1953.

[7] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, Apr. 2008. doi: 10.1145/1355734.1355746.

[8] A. B. Downey, "Evidence for long-tailed distributions in the internet," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, San Francisco, USA, Nov. 2001, pp. 229–241. doi:10.1145/505202.505230.

[9] S. Ben Fred, T. Bonald, A. Proutiere, G. Régnié, and J. W. Roberts, "Statistical bandwidth sharing: a study of congestion at flow level," in *Proc ACM SIGCOMM*, San Diego, USA, Aug. 2001, pp. 111–122. doi: 10.1145/383059.383068.

[10] Ka-Cheong Leung, V. O. K. Li, and Daiqin Yang, "An overview of packet reordering in transmission control protocol (TCP): problems, solutions, and challenges," *IEEE Transaction on Parallel and Distributed Systems*, vol. 18, no. 4, pp. 522–535, Apr. 2007. doi: 10.1109/TPDS.2007.1011.

[11] F. P. Tso, G. Hamilton, R. Weber, C. S. Perkins, and D. P. Pezaros, "Longer is better: exploiting path diversity in data center networks," *IEEE 33rd International Conference on Distributed Computing System*, Philadelphia, USA, Jul. 2013, pp. 430–439. doi: 10.1109/ICDCS.2013.36.

[12] B. Lantz, B. Heller, and N. Mckeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proc ACM HotNets-IX*, Monterey, USA, Oct. 2010. doi: 10.1145/1868447.1868466.

[13] T. Hoefler, T. Schneider, and A. Lumsdaine, "Multistage switches are not crossbars: effects of static routing in high-performance networks," *IEEE International Conference on Cluster Computing*, Tsukuba, Japan, 2008, pp. 116–125. doi: 10.1109/CLUSTR.2008.4663762.

[14] Y. Zhang and N. Ansari, "On architecture design, congestion notification, TCP incast and power consumption in data centers," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 39– 64, Feb. 2013. doi: 10.1109/SURV.2011.122211.00017.

[15] A. Greenberg, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "Towards a next Generation data center architecture: scalability and commoditization," in *Proc. ACM PRESTO' 08*, Seattle, USA, pp. 57 – 62. doi: 10.1145/1397718.1397732.

[16] H. Wu, G. Lu, D. Li, C. Guo, and Y. Zhang, "MDCube: a high performance network structure for modular data center interconnection," in *Proc. Co-NEXT' 09*, Rome, Italy, pp. 25–36, doi: 10.1145/1658939.1658943.

Manuscript received: 2014-03-03

Biographies

Wen Gao (gavingao@zju.edu.cn) is a PhD candidate at the New Generation Network Technology Laboratory at Zhejiang University. His current research interests include network management, software-defined networks, and reconfigurable flexible networks.

Xuyan Li (lixuyanzju@163.com) received her BS degree from Zhejiang University in 2014. Her research interests include software-defined networking and data center networking.

Boyang Zhou (zby@zju.edu.cn) is a PhD candidate at the New Generation Network Technology Laboratory at Zhejiang University. His current research interests include future Internet architecture, software-defined networks, and reconfigurable flexible networks.

Chunming Wu (wuchunming@zju.edu.cn) is a professor in the College of Computer Science at Zhejiang University. He is the director of the New Generation Network Technology Laboratory at that university. His research interests include flexible reconfigurable networks, software-defined networks, network and service testbeds, and innovative security technology for active defense networks.

DOI: 10.3969/j. issn. 1673-5188. 2014. 02. 010
http://www.cnki.net/kcms/detail/34.1294.TN.20140320.1727.006.html, published online March 20, 2014

Formal Protection Architecture for Cloud Computing System

Yasha Chen[1], Jianpeng Zhao[1], Junmao Zhu[1], and Fei Yan[2]
(1. The Institute of North Electronic Equipment, Beijing 10020, China;
2. Department of Computer Science, Wuhan University, Wuhan 430000, China)

Abstract

Cloud computing systems play a vital role in national security. This paper describes a conceptual framework called dual-system architecture for protecting computing environments. While attempting to be logical and rigorous, formalism method is avoided and this paper chooses algebra Communication Sequential Process.

Keywords

formal method; trusted computing; privacy; cloud computing

1 Introduction

Cloud computing relies on shared resources to achieve coherence and economy of scale. It is similar to a utility, such as an electricity grid, over a network. The foundation of cloud computing is converged infrastructure and shared services.

Transitive trust is key to the controlling ability of the Trusted Computing Platform (TCP). The Trusted Computing Group (TCG) states that if the information system starts from an initial root of trust, and every time the transition of the right of control, the trust will be transferred to next components by integrity measurement, thus the platform computing environment is always credible. The trusted platform module (TPM) is a kind of SOC chip and is the root of trust for TCP. For TPM, operation systems and applications are all objects that need to precede the integrity measure because of external needs. So when a new module is loaded in the internal storage, first the kernel of the OS takes charge of determining whether the module is credible. If the loaded module is credible (such as a driver), the kernel of the OS allows it to be loaded. Conversely, if it is not credible, the kernel of the OS refuses to load it. The Transitive trust transmission models presented by TCG are usually BIOS → OS Loader → OS Kernel, finally is passed on to the kernel load area of OS. Using a Linux platform, Sailer [1], [2] fulfilled credible transmission of executable code from OS to applications. Maruyama [3] explored credible transmission mechanism from Grub to OS. Huang Tao [4] showed how to fulfill the credible guide on a server platform. Research on transitive trust is now being conducted by European OpenTC, NGSCB of Microsoft, and Intel's LT technic [5]–[7].

2 Application Description

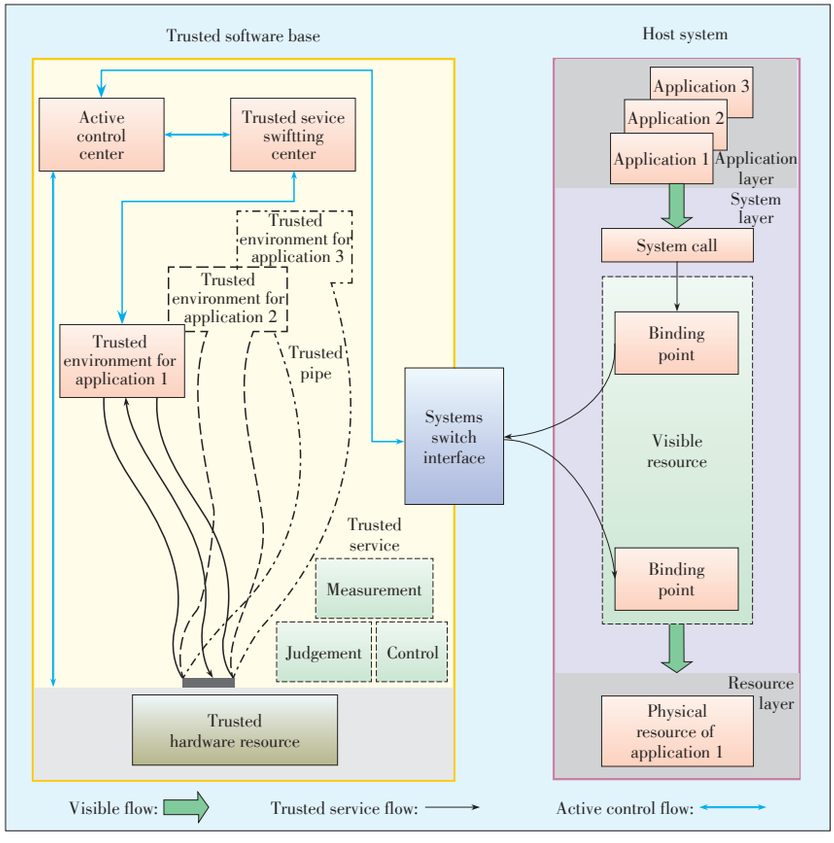
The traditional Von Neumann cloud computing architecture lacks a security mechanism. The TCG has attempted to resolve this problem and has made several breakthroughs by adding trusted hardware [8]–[11]. However, there four are still four issues in terms of essential information system security assurance:

1) Lack of a reasonable security architecture. The current trusted computing single architecture does not separate the trusted computing base from OS. Hence, the architecture can be violated and does not provide adequate protection. There are several dual-system architectures that are based on attribute values and have a passive protection mechanism. The trusted computing function is called passively by the application. Once the security loopholes are attacked there are no restrictions on those illegal usages.
2) Lack of a trusted-resource sharing methodology. With multiple applications sharing the same trusted resources, dynamic calling of the trusted services may lead to the potential conflicts, deadlocks, dispatching problems.
3) Lack of information flow security mechanism among applications in the current OS. An application can easily be called by others, enabling free flow of unnecessary information. This may cause unexpected circulation of information.
4) Lack of verification mechanism between security attributes and practical engineering. The abstract model and real system are different from observation perspective. Security attributes in formalized models are highly conceptual abstractions. The real system lacks transition and explanatory methodology. Consequently, there is a disconnect between practice and theory.

This paper proposes a credible security system architecture to achieve trusted computing core function on OS level to support initiative credible monitoring. This architecture builds a credible software base. The base is logically relatively independent to manage credible resources and computing process by virtual methods. It also supports credible mechanism of monitoring application resource processing behavior by active intercepts on system level.

The features trusted assurance in a dual system are entirely different from those in a single system. We design its functions with initiative control mechanism, credible computing service mechanism and other related auxiliary mechanism. We also

take the function of initiative measure control of Trusted Platform Control Moduel (TPCM) and the double system idea of Trusted Base's Operation System (TBOS) into the basic framework for dual system initiative credible security. The model is shown in **Fig. 1**. Compared with the Trusted Software Stack (TSS) norm of TCG, this system increases credible access,



▲**Figure 1. Trusted assurance of dual system architecture.**

credible computing framework, credible data base, credible resource management, and more. It also includes what is presented in TSS, such as the synchronous access to TPCM, capability of hiding the structuring command stream to applications, and the management of credible hardware resource of TPCM.

The trusted software base (TSB) and host system are logically separated, and they are combined by the system switch interface. For the applications of the host system, application-visible accessing resources are virtual resources by system calls. They really access the physical resources that are mapped from the application level to the source level. Therefore, we put the stuck point in the process of the access of virtual resources, which logically switch the information to the judgment of TSB and then fulfill the virtual resources access. However, for the applications, the flows happen in TSB are unaware. We call the flows transparent to applications.

TSB has initiative measure control function. Because applications use virtual application resource (VAR) by system call, TSB deploys host stuck point (HSP) during the system call service, intercepts the information of applications, and sends this information to TSB to be measured and assessed. The HSP gets the context of application object from the host, gathers credible related information and access-control-related information of application, and changes over to TSB by the system switch interface. TSB then processes credible measure and credible control according to the information. TSB executes credible-measure operation according to credible-measure policy: 1) It determines the credible attribute of related objects according to the measurement and credible-judgment policy and then writes it to the credible context of the object stored in TSB. 2) It determines the mode of control according to the attribute of system action, credible attribute of related object and credible control policy. Then, TSB executes credible control operation. The computing result is returned to the host system by HSP. The access control strategies executed in TSB are common discretionary access control strategies and mandatory access control strategies. This security policy is totally in an environment independent of the host system, and the privileged core process in host system will not interfere with the operation in TSB. In this way, the security issues discussed before are solved.

TSB also offers credible computing function. After judgment, an application that needs the credible service sends the information the credible service need to TSB by HSP. TSB configures exclusive virtual credible service environment for it by building credible pipelines, and sends the computing results to the application by HSP.

3 Architecture Design

3.1 Security Approximation Conditions Based on Non-Interference Attributes

We give the security approximation conditions in non-ideal state; just the approximation attributes of system.

Definition 1. $B(B \subseteq \alpha S)$ is supposed as all the visible operations to system for a certain user. The user can only see the trace shown by his window $B(B \subseteq \alpha S)$, which we call the part of $t(t \in \tau S)$ confined to window B, mark as $t \dagger B$. And,

$$\langle\ \rangle \dagger B \triangleq \langle\ \rangle$$
$$(\langle e \rangle \wedge t) \dagger B \triangleq \begin{cases} \langle e \rangle \wedge (t \dagger B) & \text{if } e \in B \\ t \dagger B & \text{otherwise} \end{cases} \quad (1)$$

Definition 2. All the trace of s user B can see is called the

projection of S on B, mark as $S⊙B$:

$$S⊙B \triangleq \{t\dagger B | t \in \tau S\} \tag{2}$$

And

$$\begin{aligned} &S⊙\alpha S = \tau S \\ &S⊙\{\ \} = \{\langle\ \rangle\} \end{aligned} \tag{3}$$

Definition 3. The deduction extent of S after user B's survey l is defined as:

$$infer\ S\ B\ l \triangleq \{t{:}\tau S | t\dagger B = l\} \qquad if\ B \subseteq \alpha S \wedge l \in S⊙B \tag{4}$$

All the t in τS that contents $t\ |B = l$ can reflect the extent of deduction to system S that B makes. And,

$$\begin{aligned} &infer\ S\ \{\alpha S\} l = \{l\} \\ &infer\ S\ \{\ \}\langle\ \rangle = \tau S \end{aligned} \tag{5}$$

If the user is αS, then the observing window is all the system alphabet, so the observing window and system behavior correspond, and the user deduce the behavior of system. The observing window is blank {} otherwise if the user cannot observe the system, and then the user cannot deduce anything.

For example, the alphabet A of system S = {a, b}, S first executes event a, then executes event b, finally ends. So: $S \triangleq a \to b \to STOP$. The trace is $\tau S = \{\langle\ \rangle, \langle a\rangle, \langle a,b\rangle\}$, the projection of a, b on S is $S⊙a = \{\langle\ \rangle, \langle a\rangle\}$, $S⊙b = \{\langle\ \rangle, \langle b\rangle\}$, then a, b can get deduction from each window is:

$$\begin{aligned} &infer\ S\ \{\alpha\}\langle\ \rangle = \{\langle\ \rangle\} \qquad &&infer\ S\ \{b\}\langle\ \rangle = \{\langle\ \rangle, \langle a\rangle\} \\ &infer\ S\ \{a\}\langle a\rangle = \{\langle a\rangle, \langle a,b\rangle\} \qquad &&infer\ S\ \{b\}\langle b\rangle = \{\langle a,b\rangle\} \end{aligned} \tag{6}$$

So user a in system S cannot deduce whether and when event b occurs from window {a}.

We give system Q as $A = \{a,b\}$ and the action of system Q as $Q \triangleq \mu X.((\alpha \to X)\Box(b \to STOP))$. The trace of Q is $\tau Q = \{a\}^* \bigcup t \wedge \langle b\rangle | t \in \{a\}^*$, the projection of a, b on R is $Q⊙a = \{a\}^*$, $Q⊙b = \{\langle\ \rangle, \langle b\rangle\}$, so a and b can get deduction from each window is:

$$\begin{aligned} &infer\ Q\ \{\alpha\} l = \{l, l \wedge \langle b\rangle\}\ \ for\ each\ l \in \{a\}^* \\ &infer\ Q\ \{b\}\langle\ \rangle = \{a\}^* \\ &infer\ Q\ \{b\}\langle b\rangle = \{t \wedge \langle b\rangle | t \in \{a\}^*\} \end{aligned} \tag{7}$$

So users in system Q cannot deduce how many times an event occurs by window {b}, but only knows event a will occur after b occurs once.

3.2 CSP Description of Non-Interference

Because process algebra Calculus of Communicating System (CSP) has completely formalized descriptive approach to what process may do and what process has already done, it is very easy to combine with non-deducible model, express security policy such as "system will never divulge information," and make real modeling and confirmation to security attributes of system by this formalized description. The object CSP focuses on the behavior model of a guest in the system, just CSP process. Each process is related to a component. The alphabet in CSP shows all the events completed by a process. The trace shows each event that the process has already done and can be recorded one by one.

The sets of all the events a process can provide at original state in certain environment is given by X, and the environment has the same alphabet is marked as P. Now put P in the environment. If P is deadlocked at the beginning of execution, X is a rejection set of P. This kind of rejection set is given by $refusals(P)$. To an uncertain process, at some point the process may refuse the execution of an event because of an uncertain choice. If a process cannot execute all the events it can execute, we call this process the certain process.

The rejection set of a process is given by $SF[\![P]\!]$, which is defined as

$$SF[\![P]\!] = \{(s,X) | s \in traces(P) \wedge P/s \downarrow \wedge X \in refusals(P/s)\} \tag{8}$$

where P/s is P after event in the execution trace s.

P executes all the event sequences recorded by trace s, and then refuses to do more things. We define it as an impasse (s,X), and use CSP to describe the stable failures model.

Theorem 1: If $\forall a,a' \in traces\ (S)$ makes $a \approx_L a'$
$initials\ (S/a) \bigcap L = initials\ (S/a') \bigcap L$
$refusals\ (S/a)\dagger L = refusals\ (S/a')\dagger L$
Then $\forall b, b \in traces\ (S)$, satisfies
$b \approx_L b'{:}SF[\![S/b]\!]\dagger L = SF[\![S/b']\!]\dagger L'$.

$s\dagger A$ is the set of trace s limited in event set A, just the set of trace without all the events that do not belong to A.

Theorem 1 shows that, if the traces that contain (1) and (2) are of equal value, S can still receive or reject the same event, and then s contents the attribute of noninterference in L.

Proof: The way to prove they are of equal values to prove the two impasses belong to each other.

Use any $f \in SF[\![P/b]\!]\dagger L$ to prove $f \in SF[\![P/b']\!]\dagger L$. Mark as $f = (c,X)$

From the projection of impasse, we can know that there exists $f' = (c',X')$, which makes

$$\begin{aligned} &c = c'\dagger L,\ X = X' \bigcap L,\ f' \in SF[\![S/b]\!] \\ &f \in SF[\![P/b]\!]\dagger L' \end{aligned} \tag{9}$$

We first investigate the rejection set X'. Because $(c',X') \in SF[\![S/b]\!]$, so $X' \in refusals\ (S/b(b \bigcap c'))$. The two sides are projected, then $X' \bigcap L \in refusals\ (S/b(b \bigcap c'))\dagger L$, or $X \bigcap refusals\ (S/b(b \bigcap c'))\dagger L$. From the hypothesis $b \approx_L b'$, we

can know that $c \approx_L c'$. The projections of their sequence on L are still equal, that is $(b \bigcap c') \approx_L (b' \bigcap c)$. Our hypothesis is $refusals\ (S/(b' \bigcap c)) = refusals\ (S/(b \bigcap c'))$.

Then, we investigate trace f'. We want to prove $c \in traces\ (S/b') \dagger L'$. The trace c' is given by $c' = < e_1, e_2, ..., e_n >$. From the definition, we know that $e_i \in initials\ (S/(b \bigcap < e_1, e_2, ..., e_n >))$. If $e_i \in L$, then $e_i \in initials\ (S/(b \bigcap < e_1, e_2, ..., e_{i-1} >))$.

By transforming to the presentation that can be simulated by CSP, we can simulate that it satisfies theorem 1, then it can be proved that the system is non-interference, and that the security approaching of the system is achieved.

4 Conclusion

In this paper, we focus on the characteristics of and problems with the cloud computing environment. We propose a theoretical model of innovative initiative security protection base of dual system. We also describe the base by formalized method and give the authentication method of security attribute.

References

[1] T. Schelling, "Models of segregation," *American Economic Review*, vol. 59, no. 2, pp. 488–493, May 1969.
[2] T. Schelling, "Dynamic models of segregation," *Journal of Mathematical Sociology*, vol. 1, no.2, pp. 143–186, 1971.
[3] M. Matuszewski, N. Beijar, J. Lehtinen, and T. Hyyrylainen, "Understanding attitudes towards mobile peer-to-peer content sharing services," in *PORTABLE' 07*, Orlando, FL, USA, pp. 1–5. doi: 10.1109/PORTABLE.2007.11.
[4] *Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations*, IETF Network Working Group RFC 2501, 1999.
[5] J. Li, C. Blake, D. De Couto, H. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *ACM MobiHoc 2001*, Long Beach, CA, USA, pp. 61–69. doi: 10.1145/381677.381684.
[6] X. Li, "Multicast capacity of wireless ad hoc networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 950–961, Jun. 2008. doi: 10.1109/TNET.2008.927256.
[7] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000. doi: 10.1109/18.825799.
[8] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *WMCSA 1999*, New Orleans, LA, USA, pp. 90–100. doi: 10.1109/MCSA.1999.749281.
[9] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, T. Imielinski and H. Korth, Eds., New York: Kulwer Academic Publishing, 1996, pp. 153–181.
[10] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multihop wireless networks," in *ACM SIGCOMM 2004*, Portland, OR, USA, pp. 133–144. doi: 10.1145/1015467.1015483.
[11] D. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high throughput path metric for multihop wireless routing," in *ACM MobiCom 2003*, San Diego, CA, USA, pp. 134–46. doi: 10.1145/938985.939000.

Manuscript received: 2014-03-03

Biographies

Yasha Chen (yashachen@gmail.com) has a PhD degree in computer software from Beijing University of Technology. She is currently working at the Institute of North Electronic Equipment. Her research interests include information security and network security.

Jianpeng Zhao (JianpengZhao@gmail.com) has a PhD degree in computer software from Beijing University of Posts and Telecommunications. He is currently working at the Institute of North Electronic Equipment. His research interests include information security and network security.

Junmao Zhu (JunmaoZhu@gmail.com) has a PhD degree in computer software from Beijing University of Posts and Telecommunications. He is currently working at the Institute of North Electronic Equipment. His research interests include information security and network security.

Fei Yan (FeiYan@gmail.com) has a PhD degree in computer software from Wuhan University. He is currently working at Wuhan University. His research interests include information security and network security.

New Member of *ZTE Communications* Editorial Board

Xiaodong Wang (S'98–M'98–SM'04–F'08) received the PhD degree in Electrical Engineering from Princeton University. He is a professor of Electrical Engineering at Columbia University in New York. Dr. Wang's research interests fall in the general areas of computing, signal processing and communications, and he has published extensively in these areas. Among his publications is a book entitled "Wireless Communication Systems: Advanced Techniques for Signal Reception", published by Prentice Hall in 2003. His current research interests include wireless communications, statistical signal processing, and genomic signal processing. Dr. Wang received the 1999 NSF CAREER Award, the 2001 IEEE Communications Society and Information Theory Society Joint Paper Award, and the 2011 IEEE Communication Society Award for Outstanding Paper on New Communication Topics. He has served as an Associate Editor for the *IEEE Transactions on Communications*, the *IEEE Transactions on Wireless Communications*, the *IEEE Transactions on Signal Processing*, and the *IEEE Transactions on Information Theory*. He is a Fellow of the IEEE and listed as an ISI Highly-Cited Author.

ZTE Communications Guidelines for Authors

• Remit of Journal

ZTE Communications publishes original theoretical papers, research findings, and surveys on a broad range of communications topics, including communications and information system design, optical fiber and electro–optical engineering, microwave technology, radio wave propagation, antenna engineering, electromagnetics, signal and image processing, and power engineering. The journal is designed to be an integrated forum for university academics and industry researchers from around the world.

• Manuscript Preparation

Manuscripts must be typed in English and submitted electronically in MS Word (or compatible) format. The word length is approximately 4000 to 7000, and no more than 6 figures or tables should be included. Authors are requested to submit mathematical material and graphics in an editable format.

• Abstract and Keywords

Each manuscript must include an abstract of approximately 150 words written as a single paragraph. The abstract should not include mathematics or references and should not be repeated verbatim in the introduction. The abstract should be a self–contained overview of the aims, methods, experimental results, and significance of research outlined in the paper. Five carefully chosen keywords must be provided with the abstract.

• References

Manuscripts must be referenced at a level that conforms to international academic standards. All references must be numbered sequentially in–text and listed in corresponding order at the end of the paper. References that are not cited in–text should not be included in the reference list. References must be complete and formatted according to IEEE Editorial Style www.ieee.org/documents/stylemanual.pdf. A minimum of 10 references should be provided. Footnotes should be avoided or kept to a minimum.

• Copyright and Declaration

Authors are responsible for obtaining permission to reproduce any material for which they do not hold copyright. Permission to reproduce any part of this publication for commercial use must be obtained in advance from the editorial office of *ZTE Communications*. Authors agree that a) the manuscript is a product of research conducted by themselves and the stated co–authors, b) the manuscript has not been published elsewhere in its submitted form, c) the manuscript is not currently being considered for publication elsewhere. If the paper is an adaptation of a speech or presentation, acknowledgement of this is required within the paper. The number of co–authors should not exceed five.

• Content and Structure

ZTE Communications seeks to publish original content that may build on existing literature in any field of communications. Authors should not dedicate a disproportionate amount of a paper to fundamental background, historical overviews, or chronologies that may be sufficiently dealt with by references. Authors are also requested to avoid the overuse of bullet points when structuring papers. The conclusion should include a commentary on the significance/future implications of the research as well as an overview of the material presented.

• Peer Review and Editing

All manuscripts will be subject to a two–stage anonymous peer review as well as copyediting, and formatting. Authors may be asked to revise parts of a manuscript prior to publication.

• Biographical Information

All authors are requested to provide a brief biography (approx. 150 words) that includes email address, educational background, career experience, research interests, awards, and publications.

• Acknowledgements and Funding

A manuscript based on funded research must clearly state the program name, funding body, and grant number. Individuals who contributed to the manuscript should be acknowledged in a brief statement.

• Address for Submission

magazine@zte.com.cn
12F Kaixuan Building, 329 Jinzhai Rd, Hefei 230061, P. R. China

ZTE COMMUNICATIONS

ZTE Communications has been indexed in the following databases:

- Cambridge Scientific Abstracts (CSA)
- China Science and Technology Journal Database
- Chinese Journal Fulltext Databases
- Index of Copernicus (IC)
- Inspec
- Norwegian Social Science Data Services (NSD)
- Ulrich's Periodicals Directory
- Wanfang Data—Digital Periodicals